Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 30, 2015

L. Wang
S. Hares
N. Wu
Huawei

September 26, 2014

Yang Data model for I2RS interface to the OSPF protocol
draft-wang-i2rs-ospf-dm-00

Abstract

OSPF (OSPFv2 and OSPFv3) is widely deployed link-state protocol in routing networks. During the past decades, it has been operated and maintained through typical CLI, SNMP and NETCONF. With the expansion and complication of modern networks, the necessity for rapid and dynamic control has been increased. The I2RS is a standard-based interface which provides a programmatic way to achieve this goal.

This document specifies an OSPF yang data model for the I2RS interface to OSPF. This model is based on the the I2RS OSPF informational model (draft-ietf-wu-ospf-info-model-00) which satisfies the requirements suggested by the I2RS use case requirements for the IGPs. This yang data model can be used by I2RS client-agent protocol to program OSPF routing entities.

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

Copyright Notice

Table of Contents

1. Introduction

As one of well-known link-state protocols, OSPF[RFC2328] has been widely used in the routing of intra domain networks. During the past decades, it has been deployed with the help of typical interfaces such as CLI, SNMP and NETCONF. As modern networks grow in scale and complexity, the necessity for rapid and dynamic control has been increased. The I2RS[I-D.ietf-i2rs-architecture] is a standard-based interface which provides a programmatic way to achieve this goal.

This document specifies an yang data model for I2RS interface to the OSPF protocol based on the I2RS information model specified in draft-ietf-wu-ospf-info-model-00.

In order to support large intra-domain, OSPF has been organized hierarchically into areas. The topology of one area is hidden from the rest of networks, which is beneficial from the reduction of routing traffic. Based on flooding mechanism, each routing-system in one OSPF area will maintain the identical database from which a pair-wise shortest tree is calculated in the distributed manner. As one client of RIB, OSPF SHOULD populate its routing information into RIB as stated in [I-D.ietf-i2rs-rib-info-model]

1.1. Yang Tree Diagrams

The Yang Tree diagrams used in this draft utilized a simple graphical representation of the data model. The meaning of the symbols are as follows:

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

- o Abbreviations before data node names: "rw" mean configuration (read-write) and "ro" state diagrams.

- o Symbols after data node names: "?" means an optinal node, "!" means a presence container, and "*" denotes a list and leaf-list.

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

- o Ellipis (". . . ") stand for the contents of subtress that are not shown.

Future yang symbols may be added to indicate the object relationship, ephemeral state, and other I2RS specific relationships in yang 1.1

2. OSPF data

This section describes the data involved in the OSPF information model in detail. Please note OSPF in this document means both OSPFv2 and OSPFv3[RFC5340]protocol unless specified. OSPF data includes information related to OSPF instance, OSPF area, OSPF multi-topology, OSPF interfaces, OSPF adjacencies and OSPF routes. A high-level architecture of the OSPF contents is shown as below.

```
                          OSPF routing-protocol
                                  |0..N
                                  |
                              OSPF instance
                                  |1..N
                                  |
                               Multi-topology
                                  |
                                  |
         +----------------------------------------+---------------+
         |0..N                                    |               |0..N
         |                                        |               |
        Area                                    MT-RIB          Policy
         |                                        |0..N
         |                                        |
     +-------+---------------+                   Route
     |       |1..1           |0..N                |
     |       |               |                    |
     TE     LSDB           Interface         +-------+------+
             |0..N           |               |       |1..N  |0..N
             |               |               |       |      |
            LSA     +---+--------+          Prefix Nexthop Backup
             |      |   |        |0..N                     nexthop
             |      |   |        |
             |     TE  Link-LSA NBR-list
             |
     +-------+-------+-------+------+-----------+-----------+
     |0..N   |0..N   |0..N   |0..N  |0..N       |0..N       |0..1
     |       |       |       |      |           |           |
   ADJ-list Intra-  Inter-  ASBR   ASE-prefix NSSA-prefix TE-router-ID
             area    area
             prefix  prefix
             list
```

Figure 1: Architecture of OSPF information model

3. I2RS OSPF Data Model

```
          module: ospf-protocol
     +--rw ospf-v4ur-instance
     |  +--rw ospf-instance-name          string
     |  +--rw ospf-vpn-name?              string
     |  +--rw router-id                   inet:ip-address
     |  +--ro protocol-status             protocol-status-def
     |  +--ro ospf-type                   ospf-type-def
     |  +--ro version                     ospf-version-def
     |  +--ro ospf-process-create-mode    ospf-process-create-mode-def
     |  +--rw preference                  uint32
     |  +--rw hostname?                   string
     |  +--rw mt-list
     |     +--rw multi-topo* [mt-id]
     |        +--rw mt-id             uint16
     |        +--rw address-family    address-family-def
     |        +--rw mt-status?        enumeration
     |        +--rw policy-list* [policy-id]
     |        |  +--rw policy-id    string
     |        +--rw mt-rib
     |        |  +--rw route* [prefix]
     |        |     +--rw prefix              inet:ipv4-prefix
     |        |     +--rw nexthop-list
     |        |     |  +--rw nexthop* [ospf-nexthop]
     |        |     |     +--rw ospf-nexthop    inet:ipv4-prefix
     |        |     +--rw back-nexthop?       inet:ipv4-prefix
     |        |     +--rw metric?             uint32
     |        |     +--rw type?               ospf-route-type-def
     |        |     +--rw route-state-info
     |        |        +--rw metric?                 uint32
     |        |        +--rw route-current-state?   ospf-route-state-def
     |        |        +--rw route-previous-state?  ospf-route-state-def
     |        |        +--rw route-chg-reason?      route-chg-reason-def
     |        |        +--rw lsid?                  inet:ip-address
     |        |        +--rw lsa-type?              lsa-type-def
     |        |        +--rw advertiser?            inet:ip-address
```

```
      |        +--rw area-list
      |              +--rw area-id             uint16
      |              +--rw area-type?          area-type-def
      |              +--rw area-status?        area-status-def
      |              +--rw lsa-arrival-int?    uint32
      |              +--rw lsa-orig-int?       uint32
      |              +--rw router-number?      uint32
      |              +--rw area-auth
      |              |  +--rw (auth-mode-type)?
      |              |     +--:(mode-simple)
      |              |     |  +--rw simple-password?     string
      |              |     +--:(mode-md5)
      |              |     |  +--rw md5-password?        string
      |              |     +--:(mode-hmac-sha256)
      |              |     |  +--rw hmac-key-id?         uint32
      |              |     |  +--rw hmac-password?       string
      |              |     +--:(mode-keychain)
      |              |        +--rw keychain-key-id?     uint32
      |              |        +--rw keychain-password?   string
      |              |        +--rw keychain-mode?       enumeration
      |              |        +--rw keychain-periodic?   enumeration
      |              |        +--rw send_time?           uint32
      |              |        +--rw receive_tim?         uint32
```

```
   |              +--rw lsdb
   |              |  +--rw lsa*[lsa-v2-type link-state-id advertiser-id]
   |              |     +--rw lsa-age?            uint32
   |              |     +--rw lsa-options?        uint8
   |              |     +--rw lsa-v2-type         enumeration
   |              |     +--rw link-state-id       inet:ipv4-address
   |              |     +--rw advertiser-id       inet:ip-prefix
   |              |     +--rw seq-no?             uint32
   |              |     +--rw chksum?             uint32
   |              |     +--rw lsa-length?         uint32
   |              |     +--rw (ls-type)?
   |              |        +--:(ospf-v2-router-lsa)
   |              |        |  +--rw ospf-v2-router-lsa
   |              |        |     +--rw bit-flag   uint16
   |              |        |     +--rw link-num   uint16
   |              |        |     +--rw link-list* [link-id link-data]
   |              |        |        +--rw link-id   inet:ipv4-address
   |              |        |        +--rw link-data inet:ipv4-address
   |              |        |        +--rw link-type enumeration
   |              |        |        +--rw mt-num    uint16
   |              |        |        +--rw metric    uint16
   |              |        |        +--rw mt-metric* [mt-id]
   |              |        |           +--rw mt-id  uint16
   |              |        |           +--rw metric? uint16
   |              |        +--:(ospf-v2-network-lsa)
   |              |        |  +--rw ospf-v2-network-lsa
   |              |        |     +--rw network-mask  inet:ipv4-prefix
   |              |        |     +--rw attached-router* [router-id]
   |              |        |        +--rw router-id  inet:ipv4-address
   |              |        +--:(ospf-v2-summary-lsa)
   |              |        |  +--rw ospf-v2-summary-lsa
   |              |        |     +--rw network-mask  inet:ipv4-prefix
   |              |        |     +--rw mt-metric* [mt-id]
   |              |        |        +--rw mt-id     uint16
   |              |        |        +--rw metric?   uint16
   |              |        +--:(ospf-v2-as-external-lsa)
   |              |        |  +--rw ospf-v2-as-external-lsa
   |              |        |     +--rw network-mask  inet:ipv4-prefix
   |              |        |     +--rw mt-metric* [mt-id]
   |              |        |        +--rw e-bit?     uint8
   |              |        |        +--rw mt-id      uint8
   |              |        |        +--rw metric?    uint16
   |              |        |        +--rw forwarding-address?
   |              |        |               inet:ipv4-address
   |              |        |        +--rw external-route-tag?  uint32

   |              |        +--:(ospf-v2-nssa-external-lsa)
   |              |        |  +--rw ospf-v2-nssa-external-lsa
   |              |        |     +--rw network-mask    inet:ipv4-prefix
   |              |        |     +--rw mt-metric* [mt-id]
   |              |        |        +--rw e-bit?      uint8
   |              |        |        +--rw mt-id       uint8
   |              |        |        +--rw metric?     uint32
   |              |        |        +--rw forwarding-address?
   |              |        |                   inet:ipv4-address
   |              |        |        +--rw external-route-tag? uint32
   |              |        +--:(ospf-v2-te-router-lsa)
   |              |        |  +--rw ospf-v2-te-router-lsa
   |              |        |     +--rw type?        uint8
   |              |        |     +--rw length?      uint32
   |              |        |     +--rw router-id?   inet:ipv4-address
   |              |        +--:(ospf-te-link-lsa)
   |              |           +--rw ospf-te-link-lsa
   |              |              +--rw type?           uint8
   |              |              +--rw length?         uint32
   |              |              +--rw link-type-stlv
   |              |              |  +--rw type?        uint8
   |              |              |  +--rw length?      uint32
   |              |              |  +--rw link-type?   enumeration
   |              |              +--rw link-id-tlv-stlv
   |              |              |  +--rw type?      uint8
   |              |              |  +--rw length?    uint32
   |              |              |  +--rw link-id?   inet:ipv4-address
   |              |              +--rw local-address-stlv
   |              |              |  +--rw type?        uint8
   |              |              |  +--rw length?      uint32
   |              |              |  +--rw local-address-list*
   |              |                            [remote-address]
   |              |              |     +--rw remote-address
   |              |                            inet:ipv4-address
   |              |              +--rw remote-address-stlv
   |              |              |  +--rw type?         uint8
   |              |              |  +--rw length?       uint32
   |              |              |  +--rw remote-address-list*
   |              |                           [remote-address]
   |              |              |     +--rw remote-address
   |              |                           inet:ipv4-address
   |              |              +--rw te-metric-stlv
   |              |              |  +--rw type?     uint8
   |              |              |  +--rw length?   uint32
   |              |              |  +--rw value?    uint32
   |              |              +--rw maximum-bandwidth-stlv
   |              |              |  +--rw type?     uint8
   |              |              |  +--rw length?   uint32
   |              |              |  +--rw value?    uint32
   |              |              +--rw maximum-reservable-bandwidth-stlv
   |              |              |  +--rw type?     uint8
   |              |              |  +--rw length?   uint32
   |              |              |  +--rw value?    uint32
   |              |              +--rw unreserved-bandwidth-stlv
   |              |              |  +--rw type?     uint8
   |              |              |  +--rw length?   uint32
   |              |              |  +--rw value?    uint32
   |              |              +--rw administrative-group-stlv
   |              |                 +--rw type?     uint8
   |              |                 +--rw length?   uint32
   |              |                 +--rw value?    uint32

   |              +--rw interface-list
   |              |  +--rw interface* [interface-index]
   |              |     +--rw interface-index    uint64
   |              |     +--rw interface-name?    string
   |              |     +--rw interface-status?  interface-status-def
   |              |     +--rw interface-down-reason?
   |              |                    interface-down-reason-def
   |              |     +--rw interface-net-type? interface-net-type-def
   |              |     +--rw interface-role?     interface-role-def
   |              |     +--rw interface-te-info
   |              |     |  +--rw admin_group?     uint32
   |              |     |  +--rw max_bandwidth?   uint32
   |              |     |  +--rw max_rsv_bandwidth?  uint32
   |              |     |  +--rw unrsv_bandwidth?    uint32
   |              |     +--rw interface-auth
   |              |     |  +--rw (auth-mode-type)?
   |              |     |     +--:(mode-simple)
   |              |     |     |  +--rw simple-password?    string
   |              |     |     +--:(mode-md5)
   |              |     |     |  +--rw md5-password?       string
   |              |     |     +--:(mode-hmac-sha256)
   |              |     |     |  +--rw hmac-key-id?        uint32
   |              |     |     |  +--rw hmac-password?      string
   |              |     |     +--:(mode-keychain)
   |              |     |        +--rw keychain-key-id?    uint32
   |              |     |        +--rw keychain-password?  string
   |              |     |        +--rw keychain-mode?      enumeration
   |              |     |        +--rw keychain-periodic?  enumeration
   |              |     |        +--rw send_time?          uint32
   |              |     |        +--rw receive_tim?        uint32
   |              |     +--rw ip-address?       inet:ipv4-address
   |              |     +--rw nbr-list
   |              |        +--rw nbr* [router-id]
   |              |           +--rw router-id   inet:ip-address
   |              |           +--rw interface-index?  uint64
   |              |           +--rw interface-name?   string
   |              |           +--rw nbr-status?       nbr-status-def
   |              |           +--rw nbr-previous-status? nbr-status-def
   |              |           +--rw nbr-down-reason? nbr-down-reason-def
   |              |           +--rw nbr-address?     inet:ipv4-address
   |              |           +--rw ip-address?      inet:ipv4-address
   |              +--rw network-list* [network-prefix mask]
   |              |  +--rw network-prefix    inet:ipv4-prefix
   |              |  +--rw mask              inet:ipv4-prefix
   |              +--rw route-info-list* [route-info-index]
   |                 +--rw route-info-index    uint32
   |                 +--rw router-id           inet:ipv4-address
   |                 +--rw ip-address-list* [ip-address]
   |                    +--rw ip-address    inet:ipv4-address

   +--rw ospf-v6ur-instance
      +--rw ospf-instance-name          string
      +--rw ospf-vpn-name?              string
      +--rw router-id                   inet:ip-address
      +--ro protocol-status             protocol-status-def
      +--ro ospf-type                   ospf-type-def
      +--ro version                     ospf-version-def
      +--ro ospf-process-create-mode    ospf-process-create-mode-def
      +--rw preference                  uint32
      +--rw hostname?                   string
      +--rw mt-list
         +--rw multi-topo* [mt-id]
            +--rw mt-id             uint16
            +--rw address-family    address-family-def
            +--rw mt-status?        enumeration
            +--rw policy-list* [policy-id]
            |  +--rw policy-id    string
            +--rw mt-rib
            |  +--rw route* [prefix]
            |     +--rw prefix              inet:ipv6-prefix
            |     +--rw nexthop-list
            |     |  +--rw nexthop* [ospf-nexthop]
            |     |     +--rw ospf-nexthop  inet:ipv6-prefix
            |     +--rw back-nexthop?       inet:ipv6-prefix
            |     +--rw metric?             uint32
            |     +--rw type?               ospf-route-type-def
            |     +--rw route-state-info
            |        +--rw metric?               uint32
            |        +--rw route-current-state?  ospf-route-state-def
            |        +--rw route-previous-state? ospf-route-state-def
            |        +--rw route-chg-reason?     route-chg-reason-def
            |        +--rw lsid?                 inet:ip-address
            |        +--rw lsa-type?             lsa-type-def
            |        +--rw advertiser?           inet:ip-address
            +--rw area-list
               +--rw area* [area-id]
                  +--rw area-id            uint16
                  +--rw area-type?         area-type-def
                  +--rw area-status?       area-status-def
                  +--rw lsa-arrival-int?   uint32
                  +--rw lsa-orig-int?      uint32
                  +--rw router-number?     uint32
                  +--rw area-auth
                  |  +--rw (auth-mode-type)?
                  |     +--:(mode-simple)
                  |     |  +--rw simple-password?     string
                  |     +--:(mode-md5)
                  |     |  +--rw md5-password?        string
                  |     +--:(mode-hmac-sha256)
                  |     |  +--rw hmac-key-id?         uint32
                  |     |  +--rw hmac-password?       string
                  |     +--:(mode-keychain)
                  |        +--rw keychain-key-id?     uint32
                  |        +--rw keychain-password?   string
                  |        +--rw keychain-mode?       enumeration
                  |        +--rw keychain-periodic?   enumeration
                  |        +--rw send_time?           uint32
                  |        +--rw receive_tim?         uint32
                  +--rw lsdb
                  |  +--rw lsa* [lsa-v3-type link-state-id advertiser-id]
                  |     +--rw lsa-age?                uint32
                  |     +--rw lsa-v3-type             enumeration
                  |     +--rw link-state-id           uint32
                  |     +--rw advertiser-id           inet:ip-prefix
                  |     +--rw seq-no?                 uint32
                  |     +--rw chksum?                 uint32
                  |     +--rw lsa-length?             uint32
                  |     +--rw (ls-type)?
                  |        +--:(ospf-v3-router-lsa)
                  |        |  +--rw ospf-v3-router-lsa
                  |        |     +--rw option         uint16
                  |        |     +--rw link-list*
                  |        |  [link-type interface-id neighbor-interface-id]
                  |        |        +--rw link-type    enumeration
                  |        |        +--rw metric?      uint32
                  |        |        +--rw interface-id uint32
                  |        |        +--rw neighbor-interface-id uint32
                  |        |        +--rw neighbor-router-id?
                  |        |                     inet:ipv4-address
                  |        +--:(ospf-v3-network-lsa)
                  |        |  +--rw ospf-v3-network-lsa
                  |        |     +--rw option       uint32
                  |        |     +--rw link-list* [attached-router-id]
                  |        |        +--rw attached-router-id
                  |        |                      inet:ipv4-address
                  |        +--:(ospf-v3-inter-area-prefix-lsa)
                  |        |  +--rw ospf-v3-inter-area-prefix-lsa
                  |        |     +--rw metric?                uint32
                  |        |     +--rw prefix-length          uint8
                  |        |     +--rw prefix-options         uint8
                  |        |     +--rw address-prefix-list* [address-prefix]
                  |        |        +--rw address-prefix    inet:ipv6-prefix
                  |        +--:(ospf-v3-inter-area-router-lsa)
                  |        |  +--rw ospf-v3-inter-area-router-lsa
                  |        |     +--rw options                  uint8
                  |        |     +--rw metric?                  uint32
                  |        |     +--rw destination-router-id?
                  |        |                       inet:ipv4-address
                  |        +--:(ospf-v3-as-external-lsa)
                  |        |  +--rw ospf-v3-as-external-lsa
                  |        |     +--rw options                     uint16
                  |        |     +--rw metric                      uint16
                  |        |     +--rw prefix-length               uint8
                  |        |     +--rw prefix-options              uint8
                  |        |     +--rw referenced-ls-type          uint8
                  |        |     +--rw address-prefix-list* [address-prefix]
                  |        |     |  +--rw address-prefix    inet:ipv6-prefix
                  |        |     +--rw forwarding-address?  inet:ipv6-prefix
                  |        |     +--rw external-route-tag?  uint32
                  |        |     +--rw referenced-link-state-id?   uint32
                  |        +--:(ospf-v3-nssa-lsa)
                  |        |  +--rw ospf-v3-nssa-lsa
                  |        |     +--rw options                     uint16
                  |        |     +--rw metric                      uint16
                  |        |     +--rw prefixlength                uint8
                  |        |     +--rw prefixoptions               uint8
                  |        |     +--rw referenced-ls-type          uint8
                  |        |     +--rw address-prefix-list* [address-prefix]
                  |        |     |  +--rw address-prefix    inet:ipv6-prefix
                  |        |     +--rw forwarding-address?  inet:ipv6-prefix
                  |        |     +--rw external-route-tag?  uint32
                  |        |     +--rw referenced-link-state-id? uint32
                  |        +--:(ospf-v3-link-lsa)
                  |        |  +--rw ospf-v3-link-lsa
                  |        |     +--rw priority             uint8
                  |        |     +--rw options              uint32
                  |        |     +--rw link-local-interface-address?
                  |        |                             inet:ipv6-address
                  |        |     +--rw prefixes              uint32
                  |        |     +--rw address-prefix-list*
                  |        |                       [address-prefix-index]
                  |        |        +--rw address-prefix-index    uint32
                  |        |        +--rw prefix-length           uint8
                  |        |        +--rw prefix-options?         uint8
                  |        |        +--rw address-prefix* [address]
                  |        |           +--rw address    inet:ipv6-prefix
                  |        +--:(ospf-v3-intra-area-prefix-lsa)
                  |        |  +--rw ospf-v3-intra-area-prefix-lsa
                  |        |     +--rw prefixes                    uint32
                  |        |     +--rw referenced-ls-type          uint16
                  |        |     +--rw referenced-link-state-id    uint32
                  |        |     +--rw referenced-advertising-router
                  |        |                  inet:ipv4-address
                  |        |     +--rw address-prefix-list*
                  |        |                  [address-prefix-index]
                  |        |        +--rw address-prefix-index    uint32
                  |        |        +--rw prefix-length           uint8
                  |        |        +--rw prefix-options          uint8
                  |        |        +--rw address-prefix* [address]
                  |        |           +--rw address    inet:ipv6-prefix
                  |        +--:(ospf-v3-te-router-ipv6-address-lsa)
                  |        |  +--rw ospf-v3-te-router-ipv6-address
                  |        |     +--rw type         uint8
                  |        |     +--rw length       uint16
                  |        |     +--rw router-id    inet:ipv6-address
                  |        +--:(te-link-lsa)
                  |           +--rw ospf-te-link-lsa
                  |              +--rw type?           uint8
                  |              +--rw length?         uint32
                  |              +--rw link-type-stlv
                  |              |  +--rw type?        uint8
                  |              |  +--rw length?      uint32
                  |              |  +--rw link-type?   enumeration
                  |              +--rw link-id-tlv-stlv
                  |              |  +--rw type?      uint8
                  |              |  +--rw length?    uint32
                  |              |  +--rw link-id?   inet:ipv4-address
                  |              +--rw local-address-stlv
                  |              |  +--rw type?       uint8
                  |              |  +--rw length?     uint32
                  |              |  +--rw local-address-list*
                  |              |                 [remote-address]
                  |              |     +--rw remote-address
                  |              |               inet:ipv4-address
                  |              +--rw remote-address-stlv
                  |              |  +--rw type?              uint8
                  |              |  +--rw length?            uint32
                  |              |  +--rw remote-address-list*
                  |              |                [remote-address]
                  |              |     +--rw remote-address
                  |              |                inet:ipv4-address
                  |              +--rw te-metric-stlv
                  |              |  +--rw type?     uint8
                  |              |  +--rw length?   uint32
                  |              |  +--rw value?    uint32
                  |              +--rw maximum-bandwidth-stlv
                  |              |  +--rw type?     uint8
                  |              |  +--rw length?   uint32
                  |              |  +--rw value?    uint32
                  |              +--rw maximum-reservable-bandwidth-stlv
                  |              |  +--rw type?     uint8
                  |              |  +--rw length?   uint32
                  |              |  +--rw value?    uint32
                  |              +--rw unreserved-bandwidth-stlv
                  |              |  +--rw type?     uint8
                  |              |  +--rw length?   uint32
                  |              |  +--rw value?    uint32
                  |              +--rw administrative-group-stlv
                  |                 +--rw type?     uint8
                  |                 +--rw length?   uint32
                  |                 +--rw value?    uint32
                  +--rw interface-list
                  |  +--rw interface* [interface-index]
                  |     +--rw interface-index        uint64
                  |     +--rw interface-name?        string
                  |     +--rw interface-status?      interface-status-def
                  |     +--rw interface-down-reason?
                  |                  interface-down-reason-def
                  |     +--rw interface-net-type?    interface-net-type-def
                  |     +--rw interface-role?        interface-role-def
                  |     +--rw interface-te-info
                  |     |  +--rw admin_group?        uint32
                  |     |  +--rw max_bandwidth?      uint32
                  |     |  +--rw max_rsv_bandwidth?  uint32
                  |     |  +--rw unrsv_bandwidth?    uint32
                  |     +--rw interface-auth
                  |     |  +--rw (auth-mode-type)?
                  |     |     +--:(mode-simple)
                  |     |     |  +--rw simple-password?     string
                  |     |     +--:(mode-md5)
                  |     |     |  +--rw md5-password?        string
                  |     |     +--:(mode-hmac-sha256)
                  |     |     |  +--rw hmac-key-id?         uint32
                  |     |     |  +--rw hmac-password?       string
                  |     |     +--:(mode-keychain)
                  |     |        +--rw keychain-key-id?     uint32
                  |     |        +--rw keychain-password?   string
                  |     |        +--rw keychain-mode?       enumeration
                  |     |        +--rw keychain-periodic?   enumeration
                  |     |        +--rw send_time?           uint32
                  |     |        +--rw receive_tim?         uint32
                  |     +--rw ip-address               inet:ipv6-address
                  |     +--rw nbr-list
                  |        +--rw nbr* [router-id]
                  |           +--rw router-id          inet:ip-address
                  |           +--rw interface-index?   uint64
                  |           +--rw interface-name?    string
                  |           +--rw nbr-status?        nbr-status-def
                  |           +--rw nbr-previous-status? nbr-status-def
                  |           +--rw nbr-down-reason?   nbr-down-reason-def
                  |           +--rw nbr-address?       inet:ipv6-address
                  |           +--rw ip-address         inet:ipv6-address
                  +--rw network-list* [network-index]
                  |  +--rw network-index     uint32
                  |  +--rw network-prefix    inet:ipv4-prefix
                  |  +--rw mask              inet:ipv4-prefix
                  +--rw route-info-list* [route-info-index]
                     +--rw route-info-index    uint32
                     +--rw router-id           inet:ipv4-address
                     +--rw ip-address-list* [ip-address]
                        +--rw ip-address    inet:ipv4-address
```

Figure 2 top-level I2RS YANG model of OSPF

4. Relationship to other I2RS Data Models

(TBD)

5. OSPF Yang Data Model

```
module ospf-protocol  {

  namespace "urn:huawei:params:xml:ns:yang:rt:i2rs:i2rs-ospf";
    // replace with iana namespace when assigned
    prefix "i2rs-ospf";

  import ietf-inet-types {
    prefix inet;
    //rfc6991
  }

  organization "Huawei Technologies Co., Ltd.";
  contact
    "Email: wanglixing@huawei.com
```

```
     Email: shares@ndzh.com
     Email: eric.wu@huawei.com";

   revision "2014-08-22" {
     description "initial revision";
     reference "draft-wu-i2rs-ospf-info-model-00";
   }

   typedef address-family-def {
     description
       "tbd.";
     type enumeration {
       enum "v4ur";
       enum "v6ur";
       enum "v4mr";
       enum "v6mr";
     }
   }

   typedef ospf-type-def {
     type enumeration {
       enum "asbr";
       enum "abr";
     }
   }

   typedef ospf-route-type-def {
     description
       "The type of ospf route.";
     type enumeration {
       enum "ospf type 1";
       enum "ospf type 2";
       enum "ospf type 3";
       enum "ospf type 4";
       enum "ospf type 5";
       enum "ospf type 7";
     }
   }

   typedef lsa-type-def {
     description
       "The type of ospf lsa.";
     type enumeration {
       enum "route lsa";
       enum "network lsa";
       enum "summary3 lsa";
       enum "summary4 lsa";
       enum "ase lsa";
```

```
      enum "nssa lsa";
      enum "intter-area-prefix lsa";
      enum "inter-area-router lsa";
      enum "link lsa";
      enum "intra-area-prefix lsa";
      enum "te router-id lsa";
      enum "link-te lsa";
    }
  }

  typedef ospf-route-state-def {
    type enumeration {
      enum "active";
      enum "inactive";
      enum "primary";
      enum "backup";
    }
  }

  typedef route-chg-reason-def {
    description
      "The changing reason of ospf route .";
    type enumeration {
      enum "orig-adv";
      enum "orig-withdraw";
      enum "adj-down";
      enum "policy-deny";
    }
  }

  typedef area-status-def {
    type enumeration {
      enum "active";
      enum "reset";
      enum "shutdown";
    }
  }

  typedef area-type-def {
    type enumeration {
      enum "normal";
      enum "stub";
      enum "nssa";
    }
  }

  typedef lsdb-status-def {
    type enumeration {
      enum "normal";
      enum "overflow";
    }
  }

  typedef interface-net-type-def {
    type enumeration {
      enum "p2p";
      enum "brodcast";
      enum "nbma";
      enum "p2mp";
    }
  }

  typedef interface-status-def {
    type enumeration {
      enum "if-up";
      enum "if-down";
    }
  }

  typedef interface-down-reason-def {
    type enumeration {
      enum "phy-down";
      enum "admin-down";
      enum "ip-down";
      enum "i2rs-down";
    }
  }

  typedef nbr-status-def {
    type enumeration {
      enum "down";
      enum "attempt";
      enum "2-way";
      enum "exstat";
      enum "exchange";
      enum "loading";
      enum "full";
    }
  }
  typedef nbr-down-reason-def {
    type enumeration {
      enum "if-down";
      enum "bfd-down";
      enum "expiration";
      enum "cfd-chg";
      enum "i2rs-down";
    }
  }
  typedef interface-role-def {
    type enumeration {
      enum "dr";
      enum "bdr";
    }
  }

  typedef protocol-status-def {
    type enumeration {
      enum "active";
      enum "reset";
      enum "shutdown";
      enum "overload";
    }
  }


  typedef ospf-version-def {
    description
     "OSPF v2 is for IPV4, and ospf v3 is for IPV6.";
    type enumeration {
      enum "v2";
      enum "v3";
    }
  }

  typedef ospf-process-create-mode-def {
    type enumeration {
      enum "not-i2rs";
      enum "i2rsclient-create-ospf-instance";
      enum "i2rsagent-fails-ospf-instance-create";
      enum "i2rsagent-created-ospf-instance";
      enum "i2rsagent-ospf-instance-create";
      enum "i2rsagent-rejects-ospf-instance-create";
      enum "i2rsagent-attempts-ospf-instance-create";
    }
  }

  grouping ospf-instance-commom {
    description
      "the common structure of ospf process.";
    leaf ospf-instance-name {
      type string;
      mandatory true;
    }
    leaf ospf-vpn-name {
      type string;
      mandatory false;
    }

    leaf router-id {
      type inet:ip-address;
      mandatory true;

    }

    leaf protocol-status {
      type protocol-status-def;
      config "false";
      mandatory true;
    }

    leaf ospf-type {
      type ospf-type-def;
      config "false";
      mandatory true;
    }

    leaf version {
      type ospf-version-def;
      config "false";
      mandatory true;
    }

    leaf ospf-process-create-mode {
      type ospf-process-create-mode-def;
      config "false";
      mandatory true;
    }

   leaf preference {
     type uint32 {
       range "1..4294967295";
     }
     mandatory true;
    }

   leaf hostname {
     type string;
     mandatory false;
    }
  }
  grouping ospf-mt-commom {
    description
      "the common structure of ospf process.";
    leaf mt-id {
      type uint16;
    }
    leaf address-family {
      type address-family-def;
      mandatory true;
    }

    leaf mt-status {
      type enumeration {
    enum "active";
    enum "inactive";
      }
    }

    list policy-list {
      description
      "The policy of this MT.";
      key "policy-id";
      leaf policy-id {
         type string;
      }
    }
  }

  grouping auth-info {
    choice auth-mode-type {
      case mode-simple  {
        leaf simple-password {
          type string;
        }
      }
      case mode-md5  {
        leaf md5-password {
          type string;
        }
      }
      case mode-hmac-sha256  {
        leaf hmac-key-id {
          type uint32;
        }
        leaf hmac-password {
          type string;
        }
      }
      case mode-keychain  {
        leaf keychain-key-id {
          type uint32;
        }
        leaf keychain-password {
          type string;
        }
        leaf keychain-mode {
          type enumeration {
            enum "absolute";
            enum "periodic";
          }
        }

        leaf keychain-periodic {
          type enumeration {
            enum "daily";
            enum "weekly";
            enum "monthly";
            enum "yearly";
          }
        }
        leaf send_time {
          type uint32;
        }
        leaf receive_tim {
          type uint32;
        }
      }
    }
  }



  grouping ospf-area-commom {
    description
      "the area structure of ospf process.";
    leaf area-id {
      description  "Tbd.";
      type uint16;
    }

    leaf area-type {
      type area-type-def;
    }

    leaf area-status {
      type area-status-def;
    }

    leaf lsa-arrival-int {
      type uint32;
    }
    leaf lsa-orig-int {
      type uint32;
    }
    leaf router-number {
      type uint32;
    }
    container area-auth{
      uses auth-info;
    }
  }

  grouping ospf-route-commom {
    description
      "the common structure of ospf route.";
    leaf metric {
      type uint32;
     }

    leaf type {
      type ospf-route-type-def;
    }

    container route-state-info {
      leaf metric {
        type uint32;
      }

      leaf route-current-state {
         type ospf-route-state-def;
      }

      leaf route-previous-state {
        type ospf-route-state-def;
      }

      leaf route-chg-reason {
        type route-chg-reason-def;
      }

      leaf lsid {
        type inet:ip-address;
      }
      leaf lsa-type {
        type lsa-type-def;
      }

      leaf advertiser {
        type inet:ip-address;
      }
    }
  }

  grouping ospf-interface-commom {
    description
      "the area structure of ospf interface.";
    leaf interface-index {
      description  "Tbd.";
      type uint64;
    }

    leaf interface-name {
      description  "Tbd.";
      type string;
    }

    leaf interface-status {
      type interface-status-def;
    }

    leaf interface-down-reason {
      type interface-down-reason-def;
    }
    leaf interface-net-type {
      type interface-net-type-def;
    }

    leaf interface-role {
      type interface-role-def;
    }
    container interface-te-info {
      leaf admin_group {
        type uint32;
      }
      leaf max_bandwidth {
        type uint32;
      }
      leaf max_rsv_bandwidth {
        type uint32;
      }
      leaf unrsv_bandwidth {
        type uint32;
      }
    }
    container interface-auth{
      uses auth-info;
    }
  }

  grouping ospf-nbr-commom {
    description
      "the area structure of ospf nbr.";
    leaf router-id {
      type inet:ip-address;
    }


    leaf interface-index {
        description  "Tbd.";
        type uint64;
    }

    leaf interface-name {
       description  "Tbd.";
       type string;
    }

    leaf nbr-status {
      type nbr-status-def;
    }
    leaf nbr-previous-status {
      type nbr-status-def;
    }
    leaf nbr-down-reason {
      type nbr-down-reason-def;
    }
  }
  grouping ospf-v2-lsa-header-commom {
    description
      "the ospf v2 lsa header ";
    leaf lsa-age {
       type uint32;
    }
    leaf lsa-options {
       type uint8;
    }
    leaf lsa-v2-type {
      mandatory "true";
      type enumeration {
        enum router-lsa {
          value "1";
        }
        enum network-lsa {
          value "2";
        }
        enum summary-abr-lsa {
          value "3";
        }
        enum  summary-asbr-lsa {
          value "4";
        }
        enum ase-lsa {
          value "5";
        }
        enum nssa-lsa {
          value "7";
        }
        enum te-lsa {
          description "export-extcommunity and import-extcommunity:";
          value "10";
        }
      }
    }
    leaf link-state-id {
      type inet:ipv4-address;
      mandatory true;
    }
    leaf advertiser-id {
      type inet:ip-prefix;
      mandatory true;
    }
     leaf seq-no {
       type uint32;
    }

    leaf chksum {
      type uint32;
    }
    leaf lsa-length {
      type uint32;
    }
  }

  grouping ospf-v3-lsa-header-commom {
    description
      "the ospf v3 lsa header ";
    leaf lsa-age {
       type uint32;
    }
    leaf lsa-v3-type {
      mandatory "true";
      type enumeration {
        enum router-lsa {
          value "2001";
        }
        enum network-lsa {
          value "2002";
        }
        enum inter-area-prefix-lsa {
          value "2003";
        }
        enum  inter-area-router-lsa {
          value "2004";
        }
        enum as-external-lsas {
          value "4005";
        }
        enum nssa-lsa {
          value "2007";
        }
        enum link-lsa {
          value "0008";
        }
        enum intra-area-prefix-lsa {
          value "2009";
        }
        enum te-lsa {
          value "10";
          description "Te:";
        }
      }
    }

    leaf link-state-id {
      description "lsa type/scope unique identifier.";
      type uint32;
    }
    leaf advertiser-id {
      type inet:ip-prefix;
      mandatory true;
    }
     leaf seq-no {
       type uint32;
    }
```

```
    leaf chksum {
      type uint32;
    }
    leaf lsa-length {
      type uint32;
    }
  }
  grouping ospf-v2-router-lsa {
    container ospf-v2-router-lsa {
      leaf bit-flag {
        description  "bit V:When set, the router is
        an endpoint of one or more fully
        adjacent virtual links having the
        described area as Transit area
        (V is for virtual link endpoint).
        bit E:When set, the router is an AS boundary
         router (E is for external).
        bit B:When set, the router is an area
          border router (B is for border).";
        type uint16;
        mandatory true;
      }
      leaf link-num {
        description  "The number of router links
         described in this LSA.  This must be
         the total collection of router links
         (i.e., interfaces) to the area.";
        type uint16;
        mandatory true;
      }
      list link-list{
        key "link-id link-data";
        leaf link-id {
          description  "Identifies the object
            that this router link connects to.  Value
            depends on the link’s Type.  When
            connecting to an object that also
            originates an LSA (i.e., another router
            or a transit network) the Link ID is equal
            to the neighboring LSA’s Link
            State ID.  This provides the key
            for looking up the neighboring
            LSA in the link state database
            during the routing table calculation.";
          type inet:ipv4-address;
          mandatory true;
        }
        leaf link-data{
          type inet:ipv4-address;
        }

        leaf link-type {
          type enumeration {
            enum "p2p";
       enum "transit";
            enum "stub";
       enum "virtual";
          }
          mandatory true;
        }
        leaf mt-num {
          type uint16;
          mandatory true;
        }
        leaf metric {
          type uint16;
          mandatory true;
        }
        list mt-metric{
          key "mt-id";
          leaf mt-id {
            type uint16;
          }
          leaf metric {
            type uint16;
          }
        }
      }
    }
  }

  grouping ospf-v2-network-lsa  {
    container ospf-v2-network-lsa {
      leaf network-mask {
        description  "The ip address mask for the
        network.  for example, a class a
           network would have the mask 0xff000000.";
        type inet:ipv4-prefix;
        mandatory true;
      }
      list attached-router{
        description  "The router ids of each of the
        routers attached to the network.
          actually, only those routers that are fully
          adjacent to the designated router are listed.
          the designated router includes itself in this list. ";
        key "router-id";
        leaf router-id {
          type inet:ipv4-address;
        }
      }
    }
  }

  grouping ospf-v2-summary-lsa  {
    container ospf-v2-summary-lsa {
      leaf network-mask {
        description  "for type 3 summary-lsas, this
        indicates the destination network’s ip address
         mask.  for example, when advertising the
        location of a class a network the value 0xff000000 would be
        used.  this field is not meaningful and must be
        zero for type 4 summary-lsas.";
        type inet:ipv4-prefix;
        mandatory true;
      }

      list mt-metric{
        key "mt-id";
        leaf mt-id {
          type uint16;
        }
        leaf metric {
          type uint16;
        }
      }
    }
  }

  grouping ospf-v2-as-external-lsa  {
    container ospf-v2-as-external-lsa {
      leaf network-mask {
        description  "The ip address mask for the
          advertised destination.  for example,
          when advertising a class a network the
          mask 0xff000000 would be used.";
        type inet:ipv4-prefix;
        mandatory true;
      }

      list mt-metric{
        key "mt-id";
        leaf e-bit {
          description  "The type of external metric.
           if bit e is set, the metric specified is a type
           2 external metric.  this means the metric is
            considered larger than any link state path.
            if bit e is zero, the specified metric is a
            type 1 external metric.  this means
            that it is expressed in the same units as
            the link state metric
            (i.e., the same units as interface cost)..";
          type uint8;
        }
        leaf mt-id {
          type uint8;
        }
        leaf metric {
          type uint16;
        }
        leaf forwarding-address {
          description  "data traffic for the advertised
           destination will be forwarded to this address.
           if the forwarding address is set to 0.0.0.0,
           data traffic will be forwarded instead to the
           lsa’s originator (i.e., the responsible as
           boundary router).";
          type inet:ipv4-address;
        }
        leaf external-route-tag {
          description  "a 32-bit field attached to each external
            route.  this is not used by the ospf protocol itself.
            it may be used to communicate information between as
            boundary routers; the precise nature of
            such information is outside the scope of
            this specification.";
          type uint32;
        }
      }
    }
  }

  grouping ospf-v2-nssa-external-lsa  {
    container ospf-v2-nssa-external-lsa {
      leaf network-mask {
        description  "The ip address mask for the
        advertised destination.  for
          example, when advertising a class a
          network the mask 0xff000000
          would be used.";
        type inet:ipv4-prefix;
        mandatory true;
      }

      list mt-metric{
        key "mt-id";
        leaf e-bit {
          description  "The type of external metric.
            if bit e is set, the metric specified is a
            type 2 external metric.  this means the metric is
            considered larger than any link state path.
            If bit e is zero, the specified metric is a
            type 1 external metric.  This means
            that it is expressed in the same units as
            the link state metric
            (i.e., the same units as interface cost)..";
          type uint8;
        }
        leaf mt-id {
          type uint8;
        }
        leaf metric {
          type uint32;
        }
        leaf forwarding-address {
          description  "data traffic for the advertised
            destination will be forwarded to
            this address.  if the forwarding address is
            set to 0.0.0.0, data traffic will be forwarded
            instead to the lsa’s originator (i.e.,
            the responsible as boundary router).";
          type inet:ipv4-address;
        }
        leaf external-route-tag {
          description  "a 32-bit field attached to each
            external route.  this is not used by the ospf
            protocol itself.  it may be used to communicate
            information between as boundary routers;
            the precise nature of such information is outside
            the scope of this specification.";
          type uint32;
        }
      }
    }
  }

   grouping ospf-v2-te-router-lsa  {
    container ospf-v2-te-router-lsa {
      description  "The router address tlv specifies a
```

```
        stable ip address of the advertising router that
        is always reachable if there is any
        connectivity to it; this is typically implemented
        as a loopback address.  the key attribute is that
        the address does not become unusable if an interface
        is down.  in other protocols, this is known
        as the router id, but for obvious reasons this
        nomenclature is avoided here.  if a router advertises
        bgp routes with the bgp next hop attribute set to the
        bgp router id, then the router address
        should be the same as the bgp router id. ";
      leaf type {
        description  "The router address tlv is type 1,
          has a length of 4.";
        type uint8;
      }
      leaf length {
        description  "The router address tlv has a length of 4.";
        type uint32;
      }
      leaf router-id {
        description  "The value of router address tlv is the
          four octet ip address..";
        type inet:ipv4-address;
      }
    }
  }

   grouping ospf-te-link-lsa  {
    container ospf-te-link-lsa {
      description  "The link tlv describes a single link.
       It is constructed of a set of sub-tlvs.  There are no
       ordering requirements for the sub-tlvs.";
      leaf type {
        description  "The link tlv is type 2.";
        type uint8;
      }
      leaf length {
        description  "The length of the link tlv is variable.";
        type uint32;
      }
      container link-type-stlv {
        description  "The link type sub-tlv defines the
          type of the link.";
        leaf type {
          description  "The link type sub-tlv is tlv type 1.";
          type uint8;
        }
        leaf length {
          description  "The link type sub-tlv is one octet in length.";
          type uint32;
        }
        leaf link-type {
          description  ".      1 - point-to-point    2 - multi-access.";
          type enumeration {
            enum "point-to-point";
           enum "multi-access";
          }
        }
      }

      container link-id-tlv-stlv {
        description  "The link id sub-tlv identifies the
         other end of the link. The link id is identical to the
         contents of the link id field in the
          router lsa for these link types.";
        leaf type {
          description  "The link type sub-tlv is tlv type 2.";
          type uint8;
        }
        leaf length {
          description  "The link type sub-tlv is four octet in length.";
          type uint32;
        }
        leaf link-id {
          description  ".";
          type inet:ipv4-address;
        }
      }

      container local-address-stlv {
        description  "The local interface ip address sub-tlv
          specifies the ip address(es) of the interface corresponding
          to this link.  If there are multiple local addresses on
          the link, they are all listed in this sub-tlv.";
        leaf type {
          description  "The local interface ip address sub-tlv is tlv type 3
.";
          type uint8;
        }
        leaf length {
          description  "The local interface ip address sub-tlv is 4n
            octets in length, where n is the number of neighbor addresses.";
          type uint32;
        }
        list local-address-list {
          key "remote-address";
          leaf remote-address {
            type inet:ipv4-address;
          }
        }
      }

      container remote-address-stlv {
        description  "The remote interface ip address sub-tlv
          specifies the ip address(es) of the neighbor’s interface
          corresponding to this link. This and the
          local address are used to discern multiple parallel
          links between systems.  If the link type of the link
          is multi-access, the remote interface ip address is
          set to 0.0.0.0; alternatively, an
          implementation may choose not to send this sub-tlv.";
        leaf type {
          description  "The remote interface ip address sub-tlv is tlv type
4.";
          type uint8;
        }
        leaf length {
          description  "The remote interface ip address sub-tlv is 4n
            octets in length, where n is the number of neighbor addresses.";
          type uint32;
        }
        list remote-address-list {
          key "remote-address";
          leaf remote-address {
            type inet:ipv4-address;
          }
        }
      }

      container te-metric-stlv {
        description  "The traffic engineering metric sub-tlv
          specifies the link metric for traffic engineering purposes.
          This metric may be different than the
          standard ospf link metric. Typically, this metric
          is assigned by a network administrator..";
        leaf type {
          description  "The traffic engineering metric
           sub-tlv is tlv type 5.";
          type uint8;
        }
        leaf length {
          description  "The traffic engineering metric sub-tlv is
          four octets in length..";
          type uint32;
        }
        leaf value {
          type uint32;
        }
      }

      container maximum-bandwidth-stlv {
        description  "The maximum bandwidth sub-tlv specifies
          the maximum bandwidth that can be used on this link,
          in this direction (from the system originating the lsa
          to its neighbor), in ieee floating point format.
          This is the true link capacity.  The units are bytes
          per second. The maximum bandwidth sub-tlv is tlv type 6,
          and is four octets in length.";
        leaf type {
          description  "The maximum bandwidth sub-tlv is tlv type 6.";
          type uint8;
        }
        leaf length {
          description  "The maximum bandwidth sub-tlv is
            four octets in length.";
          type uint32;
        }
        leaf value {
          type uint32;
        }
      }

      container maximum-reservable-bandwidth-stlv {
        description  "The maximum reservable bandwidth
          sub-tlv specifies the maximum bandwidth that may
          be reserved on this link, in this direction, in
          ieee floating point format.  note that this may be
          greater than the maximum bandwidth (in which case
          the link may be oversubscribed).
          This should be user-configurable; The default value should
          be the maximum bandwidth.  the units are bytes per second.";
        leaf type {
          description  "The maximum reservable bandwidth sub-tlv
           is tlv type 7,.";
          type uint8;
        }
        leaf length {
          description  "The maximum reservable bandwidth sub-tlv is
          four octets  in length.";
          type uint32;
        }
        leaf value {
          type uint32;
        }
      }

      container unreserved-bandwidth-stlv {
        description  "The unreserved bandwidth sub-tlv specifies
          the amount of bandwidth not yet reserved at each of the
          eight priority levels in IEEE floating point format.
          The values correspond to the bandwidth that
          can be reserved with a setup priority of 0 through 7,
          arranged in increasing order with priority 0 occurring
          at the start of the sub-tlv, and priority 7 at the end
          of the sub-tlv. The initial values (before any bandwidth
          is reserved) are all set to the maximum reservable
          bandwidth.  each value will be less than or
          equal to the maximum reservable bandwidth.
          The units are bytes per second.";
        leaf type {
          description  "The unreserved bandwidth sub-tlv is
           tlv type 8.";
          type uint8;
        }
        leaf length {
          description  "The unreserved bandwidth sub-tlv is
           32 octets in length.";
          type uint32;
        }
        leaf value {
          type uint32;
        }
      }

      container administrative-group-stlv {
        description  "The administrative group sub-tlv contains
          a 4-octet bit mask assigned by the network administrator.
          Each set bit corresponds to one administrative group assigned
          to the interface.  a link may belong to multiple groups.
          by convention, the least significant bit is referred to
          as ’group 0’, and the most significant bit is referred
          to as ’group 31’.  The administrative group is also
          called resource class/color [5]..";

        leaf type {
          description  "The administrative group sub-tlv is tlv type 9.";
          type uint8;
        }
        leaf length {
          description  "The administrative group sub-tlv is
           four octet in length.";
          type uint32;
        }
        leaf value {
          type uint32;
        }
      }
    }
  }

  grouping ospf-v3-router-lsa  {
    container ospf-v3-router-lsa {
      description
        "router-lsas have ls type equal to 0x2001.
          Each router in an area originates one or more
          router-lsas.  the complete collection of
          router-lsas originated by the router describe
          the state and cost of the router’s interfaces
          to the area.";
      leaf option {
        description  " 0  |nt|x|v|e|b|  options  .";
        type uint16;
        mandatory true;
      }
      list link-list{
        key "link-type interface-id neighbor-interface-id";
        leaf link-type {
          type enumeration {
            enum "p2p";
         enum "transit";
            enum "reserved";
         enum "virtual";
          }
          mandatory true;
        }
        leaf metric {
          description  "The cost of using this router
            interface for outbound traffic.";
          type uint32;
         }
        leaf interface-id {
          description  "The interface id assigned to the
            interface being described.";
          type uint32;
         }

        leaf neighbor-interface-id{
          description  "The interface id the neighbor router
            has associated with the link, as advertised in the
            neighbor’s hello packets.  for transit (type
            2) links, the link’s designated router is the
            neighbor described. For other link types, the
            sole adjacent neighbor is described.";
          type uint32;
          }
        leaf neighbor-router-id{
          description  "The router id the of the neighbor router.
            For transit (type 2) links, the link’s designated
            router is the neighbor described. For other link types,
            the sole adjacent neighbor is described.";
          type inet:ipv4-address;
        }
      }
    }
  }

  grouping ospf-v3-network-lsa  {
    container ospf-v3-network-lsa {
      leaf option {
        description  "  0   |  options    .";
        type uint32;
        mandatory true;
      }
      list link-list{
        key "attached-router-id";
        leaf attached-router-id{
          description  "The router ids of each of the routers
           attached to the link.  Actually, only those routers
           that are fully adjacent to the designated router
           are listed.  the designated router includes
           itself in this list.";
          type inet:ipv4-address;
        }
      }
    }
  }

  grouping ospf-v3-inter-area-prefix-lsa  {
    container ospf-v3-inter-area-prefix-lsa {
      description " These lsas are the ipv6 equivalent of ospf
        for ipv4’s type 3 summary-lsas (see section 12.4.3 of
        [ospfv2]).  originated by area border routers, they
        describe routes to ipv6 address prefixes that belong
        to other areas. A separate inter-area-prefix-lsa is originated
        for each ipv6 address prefix. ";
      leaf metric {
        description  "The cost of  this rout.";
        type uint32;
      }
      leaf prefix-length {
        type uint8;
        mandatory true;
      }
      leaf prefix-options {
        type uint8;
        mandatory true;
      }
      list address-prefix-list{
        key "address-prefix";
        leaf address-prefix{
          type inet:ipv6-prefix;
        }
      }
    }
  }

  grouping ospf-v3-inter-area-router-lsa  {
    container ospf-v3-inter-area-router-lsa {
      description " inter-area-router-lsas have ls
       type equal to 0x2004.  these lsas are the ipv6
       equivalent of ospf for ipv4’s type 4 summary-lsas (see
       section 12.4.3 of [ospfv2]).  originated by
       area border routers, they describe routes
       to as boundary routers in other areas .";
      leaf options {
        type uint8;
        mandatory true;
      }
      leaf metric {
        description  "The cost of  this rout.";
        type uint32;
      }
      leaf destination-router-id {
        description  "The router id of the router being
        described by the lsa.";
        type inet:ipv4-address;
      }
    }
  }

  grouping ospf-v3-as-external-lsa  {
    container ospf-v3-as-external-lsa {
      description " As-external-lsas have ls type equal to 0x4005.
        These lsas are originated by as boundary routers and describe
        destinations external to the as.  Each lsa describes a route
        to a single ipv6 address prefix. .";
      leaf options {
        type uint16;
        mandatory true;
      }
      leaf metric {
        description  "The cost of  this rout.";
        type uint16;
        mandatory true;
      }
      leaf prefix-length {
        type uint8;
        mandatory true;
      }
      leaf prefix-options {
        type uint8;
        mandatory true;
      }
      leaf referenced-ls-type {
        type uint8;
        mandatory true;
      }
      list address-prefix-list{
        key "address-prefix";
        leaf address-prefix{
          type inet:ipv6-prefix;
        }
      }
      leaf forwarding-address {
        type inet:ipv6-prefix;
        mandatory false;
      }
      leaf  external-route-tag {
        type uint32;
        mandatory false;
      }
      leaf  referenced-link-state-id {
        type uint32;
        mandatory false;
      }
    }
  }

  grouping ospf-v3-nssa-lsa  {
    container ospf-v3-nssa-lsa {
      description " Nssa-lsas have ls type equal to 0x4005.
        These lsas are originated by as boundary routers and
        describe destinations external to the as. Each lsa
        describes a route to a single ipv6 address prefix. .";
      leaf options {
        type uint16;
        mandatory true;
      }
      leaf metric {
        type uint16;
        mandatory true;
      }
      leaf prefixlength {
        type uint8;
        mandatory true;
      }
      leaf prefixoptions {
        type uint8;
        mandatory true;
      }
      leaf referenced-ls-type {
        type uint8;
        mandatory true;
      }
      list address-prefix-list{
        key "address-prefix";
        leaf address-prefix{
          type inet:ipv6-prefix;
        }
      }
      leaf forwarding-address {
        type inet:ipv6-prefix;
        mandatory false;
      }
      leaf  external-route-tag {
        type uint32;
        mandatory false;
      }
      leaf  referenced-link-state-id {
        type uint32;
        mandatory false;
      }
    }
  }

  grouping ospf-v3-link-lsa  {
    container ospf-v3-link-lsa {
      description " Link-lsas have ls type equal to 0x0008.
        A router originates a separate link-lsa for each
        attached physical link. These lsas have
        link-local flooding scope; they are never flooded
```

```
          beyond the associated link.";

      leaf priority {
        description  " The router priority of the interface
           attaching the originating router to the link .";
        type uint8;
        mandatory true;
      }
      leaf options {
        description  "The set of options bits that the router
          would like set in the network-lsa that will be
          originated by the designated router on
          broadcast or nbma links  .";
        type uint32;
        mandatory true;
      }

      leaf link-local-interface-address {
        description  "The originating router’s link-local
          interface address on the link.";
        type inet:ipv6-address;
      }

      leaf prefixes {
        description  "The number of ipv6 address prefixes contained
            in the lsa.";
        type uint32;
        mandatory true;
      }

      list address-prefix-list{
        key "address-prefix-index";
        leaf address-prefix-index{
          type uint32;
          mandatory true;
        }
        leaf prefix-length{
          type uint8;
          mandatory true;
        }
        leaf prefix-options{
          type uint8;
        }
        list address-prefix{
          key "address";
          leaf address{
            type inet:ipv6-prefix;
          }
        }
      }

    }
  }

  grouping ospf-v3-intra-area-prefix-lsa  {
    container ospf-v3-intra-area-prefix-lsa {
      description " Intra-area-prefix-lsas have ls
         type equal to 0x2009.  a router uses
        intra-area-prefix-lsas to advertise one
        or more ipv6 address prefixes that are associated
        with a local router address,
        an attached stub network segment, or an attached
        transit network segment.  In ipv4,
        the first two were accomplished via the router’s
        router-lsa and the last via a network-lsa.
        In ospf for ipv6, all addressing information
        that was advertised in router-lsas and network-lsas
        has been removed and is now advertised in
        intra-area-prefix-lsas.";

      leaf prefixes {
        description  "The number of ipv6 address prefixes
          contained in the lsa.";
        type uint32;
        mandatory true;
      }
      leaf referenced-ls-type {
        description  " Referenced ls type, referenced link state id,
          and referenced advertising router identifies the router-lsa
          or network-lsa with which the ipv6
          address prefixes should be associated.  if referenced ls
          type is 0x2001, the prefixes are associated with a
          router-lsa, referenced link state id should be 0,
          and referenced advertising router
          should be the originating router’s router id.
          If referenced ls type is 0x2002, the prefixes
          are associated with a network-lsa, referenced link
          state id should be the interface id of the link’s
          designated router, and referenced advertising router
          should be the designated router’s router id.";
        type uint16;
        mandatory true;
      }
      leaf referenced-link-state-id {
        type uint32;
        mandatory true;
      }
      leaf referenced-advertising-router {
        type inet:ipv4-address;
        mandatory true;
      }

      list address-prefix-list{
        key "address-prefix-index";
        leaf address-prefix-index{
          type uint32;
        }
        leaf prefix-length{
          type uint8;
          mandatory true;
        }
        leaf prefix-options{
          type uint8;
          mandatory true;
        }
        list address-prefix{
          key "address";
          leaf address{
            type inet:ipv6-prefix;
          }
        }
      }
    }
  }
  grouping ospf-v3-te-router-ipv6-address  {
    container ospf-v3-te-router-ipv6-address {
      description  "The router ipv6 address tlv has
        type 3, length 16, and a value
        containing a 16-octet local ipv6 address.
        A link-local address must not be specified for this tlv.
        It must appear in exactly one traffic
        engineering lsa originated by an ospfv3 router supporting
        the te extensions.  the router ipv6 address tlv
        is a top-level tlv as defined in traffic engineering
        extensions to ospf ";
    leaf type {
        description  "The router address tlv is type 3, has a
        length of 16.";
        type uint8;
        mandatory true;
      }
      leaf length {
        description  "The router address tlv has a length of 4.";
        type uint16;
        mandatory true;
      }
      leaf router-id {
        description  "The value of router address tlv is the
          16 octet ip address..";
        type inet:ipv6-address;
        mandatory true;
      }
    }
  }

  container ospf-v4ur-instance {
    uses ospf-instance-commom;
    container mt-list {
      list multi-topo {
        key "mt-id";
        max-elements "unbounded";
        min-elements "1";
        uses ospf-mt-commom;
        container mt-rib {
          list route {
            key "prefix";
            max-elements "unbounded";
            min-elements "0";
            leaf prefix {
              type inet:ipv4-prefix;
              mandatory true;
            }
            container nexthop-list {
              list nexthop {
                key "ospf-nexthop";
                max-elements "unbounded";
                min-elements "0";
                leaf ospf-nexthop {
                type inet:ipv4-prefix;
                }
              }
            }
            leaf back-nexthop {
            type inet:ipv4-prefix;
          }
            uses ospf-route-commom;
          }
        }


        container area-list {
          list area {
```

```
          key "area-id";
          max-elements "unbounded";
          min-elements "1";
          uses ospf-area-commom;
          container lsdb {
            list lsa {
              key "lsa-v2-type link-state-id advertiser-id";
              max-elements "unbounded";
              min-elements "0";
              uses ospf-v2-lsa-header-commom;
              choice ls-type {
                case ospf-v2-router-lsa  {
                  uses ospf-v2-router-lsa;
                }

                case ospf-v2-network-lsa  {
                  uses ospf-v2-network-lsa ;
                }

                case ospf-v2-summary-lsa  {
                  uses ospf-v2-summary-lsa ;
                }

                case ospf-v2-as-external-lsa  {
                  uses ospf-v2-as-external-lsa ;
                }

                case ospf-v2-nssa-external-lsa  {
                  uses ospf-v2-nssa-external-lsa ;
                }

                case ospf-v2-te-router-lsa  {
                  uses ospf-v2-te-router-lsa ;
                }

                case ospf-te-link-lsa  {
                  uses ospf-te-link-lsa ;
                }
              }
            }
          }

          container interface-list {
            list interface {
              key "interface-index";
              max-elements "unbounded";
              min-elements "1";
              uses ospf-interface-commom;
```

```
                leaf ip-address {
                  type inet:ipv4-address;
                }
                container nbr-list {
                  list nbr {
                    key "router-id";
                    uses ospf-nbr-commom;
                    leaf nbr-address {
                      type inet:ipv4-address;
                     }
                    leaf ip-address {
                      type inet:ipv4-address;
                    }
                  }
                }
              }
            }

            list network-list {
              description " configure the ospf .";
              key "network-prefix mask";
              leaf network-prefix {
                type inet:ipv4-prefix;
                mandatory true;
              }
              leaf mask {
                type inet:ipv4-prefix;
                mandatory true;
              }
            }
            list route-info-list {
              description " collision detection .";
              key "route-info-index";
              leaf route-info-index {
                type uint32;
                mandatory true;
              }

              leaf router-id {
                type inet:ipv4-address;
                mandatory true;
              }
              list ip-address-list {
                description " collision detect .";
                key "ip-address";
                leaf ip-address {
                  type inet:ipv4-address;
                  mandatory true;
                }
              }
            }
          }
        }
      }
    }
  }


  container ospf-v6ur-instance {
    uses ospf-instance-commom;
    container mt-list {
      list multi-topo {
        key "mt-id";
        max-elements "unbounded";
        min-elements "1";
        uses ospf-mt-commom;
        container mt-rib {
          list route {
            key "prefix";
            max-elements "unbounded";
            min-elements "0";
            leaf prefix {
              type inet:ipv6-prefix;
              mandatory true;
            }
            container nexthop-list {
              list nexthop {
                key "ospf-nexthop";
                max-elements "unbounded";
                min-elements "0";
                leaf ospf-nexthop {
                  type inet:ipv6-prefix;
                }
              }
            }
            leaf back-nexthop {
              type inet:ipv6-prefix;
            }
            uses ospf-route-commom;
          }
        }

        container area-list {
          list area {
            key "area-id";
            max-elements "unbounded";
            min-elements "1";
            uses ospf-area-commom;
            container lsdb {
              list lsa {
                key "lsa-v3-type link-state-id advertiser-id";
                max-elements "unbounded";
                min-elements "0";
                uses ospf-v3-lsa-header-commom;
                choice ls-type {
                  case ospf-v3-router-lsa  {
                    uses ospf-v3-router-lsa ;
                  }

                  case ospf-v3-network-lsa  {
                    uses ospf-v3-network-lsa ;
                  }
                  case ospf-v3-inter-area-prefix-lsa  {
                    uses ospf-v3-inter-area-prefix-lsa ;
                  }

                  case ospf-v3-inter-area-router-lsa  {
                    uses ospf-v3-inter-area-router-lsa ;
                  }

                  case ospf-v3-as-external-lsa  {
                    uses ospf-v3-as-external-lsa ;
                  }

                  case ospf-v3-nssa-lsa  {
                    uses ospf-v3-nssa-lsa ;
                  }

                  case ospf-v3-link-lsa  {
                    uses ospf-v3-link-lsa ;
                  }

                  case ospf-v3-intra-area-prefix-lsa  {
                    uses ospf-v3-intra-area-prefix-lsa ;
                  }

                  case ospf-v3-te-router-ipv6-address-lsa  {
                    uses ospf-v3-te-router-ipv6-address ;
                  }

                  case te-link-lsa  {
                    uses ospf-te-link-lsa ;
                  }
                }
              }
            }

            container interface-list {
              list interface {
                key "interface-index";
                max-elements "unbounded";
                min-elements "1";
                uses ospf-interface-commom;
                leaf ip-address {
                  type inet:ipv6-address;
                  mandatory true;
                }
                container nbr-list {
                  list nbr {
                    key "router-id";
                    uses ospf-nbr-commom;
                    leaf nbr-address {
                      type inet:ipv6-address;
                    }
                    leaf ip-address {
                      type inet:ipv6-address;
                      mandatory true;
                    }
                  }
                }
              }
            }

            list network-list {
              description " Configure the ospf .";
              key "network-index";
              leaf network-index {
                type uint32;
                mandatory true;
              }
              leaf network-prefix {
                type inet:ipv4-prefix;
                mandatory true;
              }
              leaf mask {
                type inet:ipv4-prefix;
                mandatory true;
              }
            }
            list route-info-list {
              description " Collision detect .";
              key "route-info-index";
              leaf route-info-index {
                type uint32;
                mandatory true;
              }
              leaf router-id {
                type inet:ipv4-address;
                mandatory true;
              }
              list ip-address-list {
                description " Collision detect .";
                key "ip-address";
                leaf ip-address {
                  type inet:ipv4-address;
                  mandatory true;
                }
              }
            }
          }
        }
      }
    }
  }
}/*ospf model end */
```

6. IANA Considerations

This draft registers a URI in the IETF XML registry [RFC3688]. Following the format in RFC3688, the following registration is requested:

URI: urn:huawei:params:xml:ns:yang:rt:i2rs:ospf-protocol";

Registrant Contact: The I2RS WG of IETF

XML: N/A, the request URI is in the XML namespace.

This document registres a Yang module in the Yang Module Names registry [RFC6020] with the following information:

name: IETF-i2rs-ospf-protocol

namespace: urn:ietf.params:xml:ns:yang:rt:i2rs:ospf

prefix:ospf-protocol

reference: RFC XXXX

7. Security Considerations

This document introduces no new security threat over the security threats posed by security requirements as stated in [I-D.ietf-i2rs-architecture]. (The authors would like feedback on the security issues.)

8. Acknowledgements

TBD

9. References

9.1. Informative References

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

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, April 1998.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, July 2008.

[RFC5511] Farrel, A., "Routing Backus-Naur Form (RBNF): A Syntax Used to Form Encoding Rules in Various Routing Protocol Specifications", RFC 5511, April 2009.

9.2. Normative References

[I-D.hares-i2rs-info-model-policy] Hares, S. and W. Wu, "An Information Model for Basic Network Policy", draft-hares-i2rs-info-model-policy-03 (work in progress), July 2014.

[I-D.hares-i2rs-usecase-reqs-summary] Hares, S., "Summary of I2RS Use Case Requirements", draft-hares-i2rs-usecase-reqs-summary-00 (work in progress), July 2014.

[I-D.ietf-i2rs-architecture] Atlas, A., Halpern, J., Hares, S., Ward, D., and T. Nadeau, "An Architecture for the Interface to the Routing System", draft-ietf-i2rs-architecture-05 (work in progress), July 2014.

[I-D.ietf-i2rs-rib-info-model]
Bahadur, N., Folkes, R., Kini, S., and J. Medved, "Routing Information Base Info Model", draft-ietf-i2rs-rib-info-model-03 (work in progress), May 2014.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

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

Authors' Addresses

Lixing Wang
Huawei
Huawei Bld., No.156 Beiqing Rd.
Beijing 10095
China

Email: wanglixing@huawei.com

Susan Hares
Huawei
7453 Hickory Hill
Saline, MI 48176
USA

Email: shares@ndzh.com

Nan Wu
Huawei
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: eric.wu@huawei.com