Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: October 25, 2019

R. Bush
Arrcus & IIJ
R. Austein
K. Patel
Arrcus

April 23, 2019

Layer 3 Discovery and Liveness
draft-ietf-lsvr-l3dl-00

Abstract

In Massive Data Centers (MDCs), BGP-SPF and similar routing protocols are used to build topology and reachability databases. These protocols need to discover IP Layer 3 attributes of links, such as logical link IP encapsulation abilities, IP neighbor address discovery, and link liveness. The Layer 3 Discovery and Liveness protocol specified in this document collects these data, which are then disseminated using BGP-SPF and similar protocols.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The Massive Data Center (MDC) environment presents unusual problems of scale, e.g. O(10,000) devices, while its homogeneity presents opportunities for simple approaches. Approaches such as Jupiter Rising [JUPITER] use a central controller to deal with scaling, while BGP-SPF [I-D.ietf-lsvr-bgp-spf] provides massive scale-out without centralization using a tried and tested scalable distributed control plane, offering a scalable routing solution in Clos [Clos0][Clos1] and similar environments. But BGP-SPF and similar higher level device-spanning protocols, e.g. [I-D.malhotra-bess-evpn-lsoe], need logical link state and addressing data from the network to build the routing topology. They also need prompt but prudent reaction to (logical) link failure.

Layer 3 Discovery and Liveness (L3DL) provides brutally simple mechanisms for devices to

o Discover unique identities of devices/ports/... on a logical link,

o Run Layer 2 keep-alive messages for session continuity,

o Discover each other's unique endpoint identification,

o Discover mutually supported encapsulations, e.g. IP/MPLS,

o Discover Layer 3 IP and/or MPLS addressing of interfaces of the encapsulations,

o Enable layer 3 link liveness such as BFD, and finally

o Present these data, using a very restricted profile of a BGP-LS [RFC7752] API, to BGP-SPF which computes the topology and builds routing and forwarding tables.

This protocol may be more widely applicable to a range of routing and similar protocols which need layer 3 discovery and characterisation.

2. Terminology

Even though it concentrates on the inter-device layer, this document relies heavily on routing terminology. The following attempts to clarify the use of some possibly confusing terms:

ASN:	Autonomous System Number [RFC4271], a BGP identifier for an originator of Layer 3 routes, particularly BGP announcements.
BGP-LS:	A mechanism by which link-state and TE information can be collected from networks and shared with external components using the BGP routing protocol. See [RFC7752].
BGP-SPF	A hybrid protocol using BGP transport but a Dijkstra SPF decision process. See [I-D.ietf-lsvr-bgp-spf].
Clos:	A hierarchic subset of a crossbar switch topology commonly used in data centers.
Datagram:	The L3DL content of a single Layer 2 frame. A full L3DL PDU may be packaged in multiple Datagrams.
Encapsulation:	Address Family Indicator and Subsequent Address Family Indicator (AFI/SAFI). I.e. classes of layer 2.5 and 3 addresses such as IPv4, IPv6, MPLS, ...
Frame:	A Layer 2 packet.
Link or Logical Link:	A logical connection between two logical ports on two devices. E.g. two VLANs between the same two ports are two links.
LLEI:	Logical Link Endpoint Identifier, the unique identifier of one end of a logical link, see Section 9.
MAC Address:	48-bit Layer 2 addresses are assumed since they are used by all widely deployed Layer 2 network technologies of interest, especially Ethernet. See [IEEE.802_2001].
MDC:	Massive Data Center, commonly thousands of TORs.
MTU:	Maximum Transmission Unit, the size in octets of the largest packet that can be sent on a medium, see [RFC1122] 1.3.3.
PDU:	Protocol Data Unit, an L3DL application layer message. A PDU may need to be broken into multiple Datagrams to make it through MTU or other restrictions.
RouterID:	An 32-bit identifier unique in the current routing domain, see [RFC4271] updated by [RFC6286].
Session:	An established, via OPEN PDUs, session between two L3DL capable link end-points,
SPF:	Shortest Path First, an algorithm for finding the shortest paths between nodes in a graph; AKA Dijkstra's algorithm.
System Identifier:	An eight octet ISO System Identifier a la [RFC1629] System ID
TOR:	Top Of Rack switch, aggregates the servers in a rack and connects to aggregation layers of the Clos tree, AKA the Clos spine.

ZTP: Zero Touch Provisioning gives devices initial addresses, credentials, etc. on boot/restart.

3. Background

L3DL assumes a Clos type datacenter scale and topology, but can accommodate richer topologies which contain potential cycles.

While L3DL is designed for the MDC, there are no inherent reasons it could not run on a WAN. The authentication and authorization needed to run safely on a WAN need to be considered, and the appropriate level of security options chosen.

L3DL assumes a new IEEE assigned EtherType (TBD).

The number of addresses of the Encapsulations on a link may be fairly large given a TOR with more than 20 servers, each server possibly having on the order of a hundred micro-services resulting in an inordinate number of addresses. And security will further add to the length of PDUs. PDUs with lengths over 10,000 octets are likely or quite possible.

4. Top Level Overview

o Devices discover each other on logical links

o Logical Link Endpoint Identifiers are exchanged

o Layer 2 Liveness Checks may be started

o Encapsulation data are exchanged and IP-Level Liveness Checks enabled

o A BGP-like upper layer protocol is assumed to use these data to discover and build a topology database

```
+-------------------+   +-------------------+   +-------------------+
|      Device       |   |      Device       |   |      Device       |
|                   |   |                   |   |                   |
|+-----------------+|   |+-----------------+|   |+-----------------+|
||                 ||   ||                 ||   ||                 ||
||     BGP-SPF     <+---+>     BGP-SPF     <+---+>     BGP-SPF     ||
||                 ||   ||                 ||   ||                 ||
|+--------^--------+|   |+--------^--------+|   |+--------^--------+|
|         |         |   |         |         |   |         |         |
|         |         |   |         |         |   |         |         |
|+--------+--------+|   |+--------+--------+|   |+--------+--------+|
||  Encapsulations ||   ||  Encapsulations ||   ||  Encapsulations ||
||    Addresses    ||   ||    Addresses    ||   ||    Addresses    ||
||   L2 Liveness   ||   ||   L2 Liveness   ||   ||   L2 Liveness   ||
|+--------^--------+|   |+--------^--------+|   |+--------^--------+|
|         |         |   |         |         |   |         |         |
|         |         |   |         |         |   |         |         |
|+--------v--------+|   |+--------v--------+|   |+--------v--------+|
||                 ||   ||                 ||   ||                 ||
||Inter-Device PDUs<+---+>Inter-Device PDUs<+---+>Inter-Device PDUs||
||                 ||   ||                 ||   ||                 ||
|+-----------------+|   |+-----------------+|   |+-----------------+|
+-------------------+   +-------------------+   +-------------------+
```

There are two protocols, the inter-device per-link layer 3 discovery and the interface to the upper level BGP-like API:

- o Inter-device PDUs are used to exchange device and logical link identities and layer 2.5 and 3 identifiers (not payloads), e.g. device IDs, port identities, VLAN IDs, Encapsulations, and IP addresses.

- o A Link Layer to BGP API presents these data up the stack to a BGP protocol or an other device-spanning upper layer protocol, presenting them using the BGP-LS BGP-like data format.

The upper layer BGP family routing protocols cross all the devices, though they are not part of these L3DL protocols.

To simplify this document, Layer 2 framing is not shown. L3DL is about layer 3.

5. Inter-Link Protocol Overview

Two devices discover each other and their respective identities by sending multicast HELLO PDUs (Section 10). To allow discovery of new devices coming up on a multi-link topology, devices send periodic HELLOs forever, see Section 18.1.

Once a new device is recognized, both devices attempt to negotiate and establish peering by sending unicast OPEN PDUs (Section 11). In an established peering, Encapsulations (Section 13) may be announced and modified. When two devices on a link have compatible Encapsulations and addresses, i.e. the same AFI/SAFI and the same subnet, the link is announced via the BGP-LS API.

5.1. L3DL Ladder Diagram

The HELLO, Section 10, is a priming message. It is a small L3DL PDU encapsulated in an Ethernet multicast frame with the simple goal of discovering the identities of logical link endpoint(s) reachable from a Logical Link Endpoint, Section 9.

The HELLO and OPEN, Section 11, PDUs, which are used to discover and exchange detailed Logical Link Endpoint Identifiers, LLEIs, and the ACK/ERROR PDU, are mandatory; other PDUs are optional; though at least one encapsulation MUST be agreed at some point.

The following is a ladder-style sketch of the L3DL protocol exchanges:

```
|             HELLO            | Logical Link Peer discovery
|----------------------------->|
|             HELLO            | Mandatory
|<-----------------------------|
|                              |
|                              |
|             OPEN             | MACs, IDs, and Capabilities
|----------------------------->|
|             OPEN             | Mandatory
|<-----------------------------|
|                              |
|                              |
|   Interface IPv4 Addresses   | Interface IPv4 Addresses
|----------------------------->| Optional
|            ACK               |
|<-----------------------------|
|                              |
|   Interface IPv4 Addresses   |
|<-----------------------------|
|            ACK               |
|----------------------------->|
|                              |
|                              |
|   Interface IPv6 Addresses   | Interface IPv6 Addresses
|----------------------------->| Optional
|            ACK               |
```

```
   |<-----------------------------|
   |                              |
   |   Interface IPv6 Addresses   |
   |<-----------------------------|
   |            ACK               |
   |----------------------------->|
   |                              |
   |                              |
   |   Interface MPLSv4 Labels    | Interface MPLSv4 Labels
   |----------------------------->| Optional
   |            ACK               |
   |<-----------------------------|
   |                              |
   |   Interface MPLSv4 Labels    | Interface MPLSv4 Labels
   |<-----------------------------| Optional
   |            ACK               |
   |----------------------------->|
   |                              |
   |                              |
   |   Interface MPLSv6 Labels    | Interface MPLSv6 Labels
   |----------------------------->| Optional
   |            ACK               |
   |<-----------------------------|
   |                              |
   |   Interface MPLSv6 Labels    | Interface MPLSv6 Labels
   |<-----------------------------| Optional
   |            ACK               |
   |----------------------------->|
   |                              |
   |                              |
   |        L3DL KEEPALIVE        | Layer 2 Liveness
   |----------------------------->| Optional
   |        L3DL KEEPALIVE        |
   |<-----------------------------|
```

6. Transport Layer

L3DL PDUs are carried by a simple transport layer which allows long PDUs to occupy many Ethernet frames. The L3DL data in each frame is referred to as a Datagram.

The L3DL Transport Layer encapsulates each Datagram using a common transport header.

If a PDU does not fit in a single datagram, it is broken into multiple datagrams and reassembled by the receiver a la [RFC0791].

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |    Version    |L|Datagram Num.|        Datagram Length        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                           Checksum                            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The fields of the L3DL Transport Header are as follows:

Version: Version number of the protocol, currently 0. Values other than 0 are treated as errors.

L: A bit that set to one if this Datagram is the last Datagram of the PDU. For a PDU which fits in only one Datagram, it is set to one. Note that this is the inverse of the marking technique used by [RFC0791].

Datagram Number: 0..127, a monotonically increasing value, modulo 128, see [RFC1982] which starts at 0 for each PDU. Note that this does not limit an L3DL PDU to 128 frames.

Datagram Length: Total number of octets in the Datagram including all payloads and fields.

Checksum: A 32 bit hash over the Datagram to detect bit flips, see Section 7.

7. The Checksum

There is a reason conservative folk use a checksum in UDP. And as many operators stretch to jumbo frames (over 1,500 octets) longer checksums are the prudent approach.

For the purpose of computing a checksum, the checksum field itself is assumed to be zero.

The following code describes the suggested algorithm.

Sum up 32-bit unsigned ints in a 64-bit long, then take the high-order section, shift it right, rotate, add it in, repeat until zero.

```
   <CODE BEGINS>
   #include <stddef.h>
   #include <stdint.h>

   /* The F table from Skipjack, and it would work for the S-Box. */
   static const uint8_t sbox[256] = {
   0xa3,0xd7,0x09,0x83,0xf8,0x48,0xf6,0xf4,0xb3,0x21,0x15,0x78,
   0x99,0xb1,0xaf,0xf9,0xe7,0x2d,0x4d,0x8a,0xce,0x4c,0xca,0x2e,
   0x52,0x95,0xd9,0x1e,0x4e,0x38,0x44,0x28,0x0a,0xdf,0x02,0xa0,
   0x17,0xf1,0x60,0x68,0x12,0xb7,0x7a,0xc3,0xe9,0xfa,0x3d,0x53,
   0x96,0x84,0x6b,0xba,0xf2,0x63,0x9a,0x19,0x7c,0xae,0xe5,0xf5,
   0xf7,0x16,0x6a,0xa2,0x39,0xb6,0x7b,0x0f,0xc1,0x93,0x81,0x1b,
   0xee,0xb4,0x1a,0xea,0xd0,0x91,0x2f,0xb8,0x55,0xb9,0xda,0x85,
   0x3f,0x41,0xbf,0xe0,0x5a,0x58,0x80,0x5f,0x66,0x0b,0xd8,0x90,
   0x35,0xd5,0xc0,0xa7,0x33,0x06,0x65,0x69,0x45,0x00,0x94,0x56,
   0x6d,0x98,0x9b,0x76,0x97,0xfc,0xb2,0xc2,0xb0,0xfe,0xdb,0x20,
   0xe1,0xeb,0xd6,0xe4,0xdd,0x47,0x4a,0x1d,0x42,0xed,0x9e,0x6e,
   0x49,0x3c,0xcd,0x43,0x27,0xd2,0x07,0xd4,0xde,0xc7,0x67,0x18,
   0x89,0xcb,0x30,0x1f,0x8d,0xc6,0x8f,0xaa,0xc8,0x74,0xdc,0xc9,
   0x5d,0x5c,0x31,0xa4,0x70,0x88,0x61,0x2c,0x9f,0x0d,0x2b,0x87,
   0x50,0x82,0x54,0x64,0x26,0x7d,0x03,0x40,0x34,0x4b,0x1c,0x73,
   0xd1,0xc4,0xfd,0x3b,0xcc,0xfb,0x7f,0xab,0xe6,0x3e,0x5b,0xa5,
   0xad,0x04,0x23,0x9c,0x14,0x51,0x22,0xf0,0x29,0x79,0x71,0x7e,
   0xff,0x8c,0x0e,0xe2,0x0c,0xef,0xbc,0x72,0x75,0x6f,0x37,0xa1,
   0xec,0xd3,0x8e,0x62,0x8b,0x86,0x10,0xe8,0x08,0x77,0x11,0xbe,
   0x92,0x4f,0x24,0xc5,0x32,0x36,0x9d,0xcf,0xf3,0xa6,0xbb,0xac,
   0x5e,0x6c,0xa9,0x13,0x57,0x25,0xb5,0xe3,0xbd,0xa8,0x3a,0x01,
   0x05,0x59,0x2a,0x46
   };

   /* non-normative example C code, constant time even */

   uint32_t sbox_checksum_32(const uint8_t *b, const size_t n)
   {
     uint32_t sum[4] = {0, 0, 0, 0};
     uint64_t result = 0;
     for (size_t i = 0; i < n; i++)
       sum[i & 3] += sbox[*b++];
     for (int i = 0; i < sizeof(sum)/sizeof(*sum); i++)
       result = (result << 8) + sum[i];
     result = (result >> 32) + (result & 0xFFFFFFFF);
     result = (result >> 32) + (result & 0xFFFFFFFF);
     return (uint32_t) result;
   }
   <CODE ENDS>
```

8. TLV PDUs

The basic L3DL application layer PDU is a typical TLV (Type Length Value) PDU. It includes a signature to provide optional integrity and authentication. It may be broken into multiple Datagrams, see Section 6.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |      Type     |         Payload Length        |               ~
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+               ~
    ~                          Payload ...                          ~
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |    Sig Type   |        Signature Length       |               ~
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+               ~
    ~                           Signature                           ~
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The fields of the basic L3DL header are as follows:

Type: An integer differentiating PDU payload types

```
      0 - HELLO
      1 - OPEN
      2 - KEEPALIVE
      3 - ACK
      4 - IPv4 Announcement
      5 - IPv6 Announcement
      6 - MPLS IPv4 Announcement
      7 - MPLS IPv6 Announcement
      8-254 Reserved
      255 - VENDOR
```

Payload Length: Total number of octets in the Payload field.

Payload: The application layer content of the L3DL PDU.

Sig Type: The type of the Signature. Type 0, a null signature, is defined in this document.

Sig Type 0 indicates a null Signature. For very short PDUs, the underlying Datagram checksums may be sufficient for integrity, if not for authentication.

Other Sig Types may be defined in other documents.

Signature Length: The length of the Signature, possibly including padding, in octets. If Sig Type is 0, Signature Length must be 0.

Signature: The result of running the signature algorithm specified in Sig Type over all octets of the PDU except for the Signature itself.

9. **Logical Link Endpoint Identifier**

L3DL discovers neighbors on logical links and establishes sessions between the two ends of all consenting discovered logical links. A logical link is described by a pair of Logical Link Endpoint Identifiers, LLEIs.

An LLEI is a variable length descriptor which could be an ASN, a classic RouterID, a catenation of the two, an eight octet ISO System Identifier [RFC1629], or any other identifier unique to a single logical link endpoint in the topology.

An L3DL deployment will choose and define an LLEI which suits their needs, simple or complex. Two extremes are as follows:

A simplistic view of a link between two devices is two ports, identified by unique MAC addresses, carrying a layer 3 protocol conversation. In this case, the MAC addresses might suffice for the LLEIs.

Unfortunately, things can get more complex. Multiple VLANs can run between those two MAC addresses. In practice, many real devices use the same MAC address on multiple ports and/or sub-interfaces.

Therefore, in the general circumstance, a fully described LLEI might be as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                        System Identifier                      +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             ifIndex                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

System Identifier, a la [RFC1629], is an eight octet identifier unique in the entire operational space. Routers and switches usually have internal MAC Addresses which can be padded with high order zeros and used if no System ID exists on the device. If no unique

identifier is burned into a device, the local L3DL configuration SHOULD create and assign a unique one by configuration.

ifIndex is the SNMP identifier of the (sub-)interface, see [RFC1213]. This uniquely identifies the port.

For a layer 3 tagged sub-interface or a VLAN/SVI interface, Ifindex is that of the logical sub-interface, so no further disambiguation is needed.

L3DL PDUs learned over VLAN-ports may be interpreted by upper layer-3 routing protocols as being learned on the corresponding layer-3 SVI interface for the VLAN.

10. HELLO

The HELLO PDU is unique in that it is encapsulated in a multicast Ethernet frame. It solicits response(s) from other LLEI(s) on the link. See Section 18.1 for why multicast is used. The destination multicast MAC Addressees to be used MUST be one of the following, See Clause 9.2.2 of [IEEE802-2014]:

01-80-C2-00-00-0E: Nearest Bridge = Propagation constrained to a single physical link; stopped by all types of bridges (including MPRs (media converters)).

01-80-C2-00-00-03: Nearest non-TPMR Bridge = Propagation constrained by all bridges other than TPMRs; intended for use within provider bridged networks.

All other L3DL PDUs are encapsulated in unicast frames, as the peer's destination MAC address is known after the HELLO exchange.

When an interface is turned up on a device, it SHOULD issue a HELLO periodically. The interval is set by configuration with a default of 60 seconds.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Type = 0   |       Payload Length = 0      |  Sig Type = 0 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Signature Length = 0     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

If more than one device responds, one adjacency is formed for each unique source LLEI response. L3DL treats each adjacency as a separate logical link.

When a HELLO is received from a source LLEI with which there is no established L3DL adjacency, the receiver SHOULD respond with an OPEN PDU. The two devices establish an L3DL adjacency by exchanging OPEN PDUs.

The Payload Length is zero as there is no payload.

HELLO PDUs can not be signed as keying material has yet to be exchanged. Hence the signature MUST always be the null type.

11. OPEN

Each device has learned the other's MAC Address from the HELLO exchange, see Section 10. Therefore the OPEN and subsequent PDUs are unicast, as opposed to the HELLO's multicast frame.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Type = 1   |         Payload Length        |               ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Nonce                    |  LLEI Length  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                                                               ~
~                             My LLEI                           ~
~                                                               ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   AttrCount   |             Attribute List ...                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Auth Type   |           Key Length          |               ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+               ~
~                             Key ...                           ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Sig Type   |        Signature Length       |               ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+               ~
~                          Signature ...                        ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Payload Length is the number of octets in all fields of the PDU from the Nonce through the Key, not including the signature fields.

The Nonce enables detection of a duplicate OPEN PDU. It SHOULD be either a random number or the time of day. It is needed to prevent session closure due to a repeated OPEN caused by a race or a dropped or delayed ACK.

My LLEI is the sender's LLEI, see Section 9. LLEIs are big-endian.

AttrCount is the number of attributes in the Attribute List. Attributes are single octets whose semantics are user-defined.

A node may have zero or more user-defined attributes, e.g. spine, leaf, backbone, route reflector, arabica, ...

Attribute syntax and semantics are local to an operator or datacenter; hence there is no global registry. Nodes exchange their attributes only in the OPEN PDU.

Auth Type is the Signature algorithm suite, see Section 8.

Key Length is a 16-bit field denoting the length in octets of the Key, not including the Auth Type or the Key Lengths. If there is no Key, the Auth Type and key Length MUST both be zero.

The Key is specific to the operational environment. A failure to authenticate is a failure to start the L3DL session, an ERROR PDU is sent (Error Code 2), and HELLOs MUST be restarted.

The Signature fields are described in Section 8 and in an asymmetric key environment serve as a proof of possession of the signing auth data by the sender.

Once two logical link endpoints know each other, and have ACKed each other's OPEN PDUs, Layer 2 KEEPALIVEs (see Section 14) MAY be started to ensure Layer 2 liveness and keep the session semantics alive. The timing and acceptable drop of KEEPALIVE PDUs are discussed in Section 14.

If a sender of OPEN does not receive an ACK of the OPEN PDU Type, then they MUST resend the same OPEN PDU, with the same Nonce.

Resending an unacknowledged OPEN PDU, like other ACKed PDUs, SHOULD use exponential back-off, see [RFC1122].

If a properly authenticated OPEN arrives with a new Nonce from an LLEI with which the receiving logical link endpoint believes it already has an L3DL session (OPENs have already been exchanged), the receiver MUST assume that the sending LLEI or entire device has been reset. All discovered encapsulation data SHOULD be withdrawn via the BGP-LS API and the recipient MUST respond with a new OPEN. In this circumstance encapsulations SHOULD NOT be kept because, while the new OPEN is likely to be followed by new encapsulation PDUs of the same data, the old session might have an encapsulation type not in the new session.

12. ACK

The ACK PDU acknowledges receipt of a PDU and reports any error condition which might have been raised.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Type = 3   |       Payload Length = 5      |    PDU Type   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| EType |       Error Code      |           Error Hint          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Sig Type   |        Signature Length       |               ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+               ~
~                          Signature ...                        ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The ACK acknowledges receipt of an OPEN, Encapsulation, VENDOR PDU, etc.

The PDU Type is the Type of the PDU being acknowledged, e.g., OPEN or one of the Encapsulations.

If there was an error processing the received PDU, then the EType is non-zero. If the EType is zero, Error Code and Error Hint MUST also be zero.

A non-zero EType is the receiver's way of telling the PDU's sender that the receiver had problems processing the PDU. The Error Code and Error Hint will tell the sender more detail about the error.

The decimal value of EType gives a strong hint how the receiver sending the ACK believes things should proceed:

    0 - No Error, Error Code and Error Hint MUST be zero
    1 - Warning, something not too serious happened, continue
    2 - Session should not be continued, try to restart
    3 - Restart is hopeless, call the operator
    4-15 - Reserved

Someone stuck in the 1990s might think of the error codes as 0x1zzz, 0x2zzz, etc. They might be right. Or not.

The Error Code indicates the type of error.

The Error Hint is any additional data the sender of the error PDU thinks will help the recipient or the debugger with the particular error.

The Signature fields are described in Section 8.

12.1. Retransmission

If a PDU sender expects an ACK, e.g. for an OPEN, an Encapsulation, a VENDOR PDU, etc., and does not receive the ACK for a configurable time (default one second), and the interface is live at layer 2, the sender resends the PDU using exponential back-off, see [RFC1122]. This cycle MAY be repeated a configurable number of times (default three) before it is considered a failure. The session MAY BE considered closed in case of this ACK failure.

If the link is broken at layer 2, retransmission MAY BE retried when the link comes back up if data have not changed in the interim.

13. The Encapsulations

Once the devices know each other's LLEIs, know each other's upper layer identities, have means to ensure link state, etc., the L3DL session is considered established, and the devices SHOULD exchange L3 interface encapsulations, L3 addresses, and L2.5 labels.

The Encapsulation types the peers exchange may be IPv4 Announcement (Section 13.3), IPv6 Announcement (Section 13.4), MPLS IPv4 Announcement (Section 13.6), MPLS IPv6 Announcement (Section 13.7), and/or possibly others not defined here.

The sender of an Encapsulation PDU MUST NOT assume that the peer is capable of the same Encapsulation Type. An ACK (Section 12) merely acknowledges receipt. Only if both peers have sent the same Encapsulation Type is it safe to assume that they are compatible for that type.

A receiver of an encapsulation might recognize an addressing conflict, such as both ends of the link trying to use the same address. In this case, the receiver SHOULD respond with an ERROR (Error Code 1) instead of an ACK. As there may be other usable addresses or encapsulations, this error might log and continue, letting an upper layer topology builder deal with what works.

Further, to consider a logical link of a type to formally be established so that it may be pushed up to upper layer protocols, the addressing for the type must be compatible, e.g. on the same IP subnet.

13.1. The Encapsulation PDU Skeleton

The header for all encapsulation PDUs is as follows:

```
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      Type     |         Payload Length        |     Count     |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      ...      |             Encapsulation List...             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |    Sig Type   |        Signature Length       |               ~
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+               ~
   ~                          Signature ...                        ~
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The 16-bit Count is the number of Encapsulations in the Encapsulation list.

An Encapsulation PDU describes zero or more addresses of the encapsulation type.

An Encapsulation PDU of Type T replaces all previous encapsulations of Type T.

To remove all encapsulations of Type T, the sender uses a Count of zero.

If an LLEI has multiple addresses for an encapsulation type, one and only one address SHOULD be configured to be marked as primary, see Section 13.2.

Loopback addresses are generally not seen directly on an external interface. One or more loopback addresses MAY be exposed by configuration on one or more L3DL speaking external interfaces, e.g. for iBGP peering. They SHOULD be marked as such, see Section 13.2.

If there is exactly one non-loopback address for an encapsulation type on an interface, it SHOULD be marked as primary.

If a sender has multiple links on the same interface, separate data, ACKs, etc. must be kept for each peer.

Over time, multiple Encapsulation PDUs may be sent for an interface as configuration changes.

If the length of an Encapsulation PDU exceeds the Datagram size limit on media, the PDU is broken into multiple Datagrams. See Section 8.

The Signature fields are described in Section 8.

The Receiver MUST acknowledge the Encapsulation PDU with a Type=3, ACK PDU (Section 12) with the Encapsulation Type being that of the encapsulation being announced, see Section 12.

If the Sender does not receive an ACK in a configurable interval (default one second), and the interface is live at layer 2, they SHOULD retransmit. After a user configurable number of failures, the L3DL session should be considered dead and the OPEN process SHOULD be restarted.

If the link is broken at layer 2, retransmission MAY BE retried if data have not changed in the interim.

13.2. Prim/Loop Flags

```
 0               1               2               3    ...        7
+---------------+---------------+---------------+---------------+
|  Primary      |   Loopback    |  Reserved ... |               |
+---------------+---------------+---------------+---------------+
```

Each Encapsulation interface address MAY be marked as a primary address, and/or a loopback, in which case the respective bit is set to one.

Only one address MAY be marked as primary for an encapsulation type.

13.3. IPv4 Encapsulation

The IPv4 Encapsulation describes a device's ability to exchange IPv4 packets on one or more subnets. It does so by stating the interface's addresses and the corresponding prefix lengths.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Type = 4   |         Payload Length        |     Count     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      ...      | PrimLoop Flags|           IPv4 Address        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              ...              |   PrefixLen   |    more ...   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Sig Type   |        Signature Length       |               ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+               ~
~                          Signature ...                        ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The 16-bit Count is the number of IPv4 Encapsulations.

13.4. IPv6 Encapsulation

The IPv6 Encapsulation describes a logical link's ability to exchange IPv6 packets on one or more subnets. It does so by stating the interface's addresses and the corresponding prefix lengths.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Type = 5   |         Payload Length        |     Count     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      ...      | PrimLoop Flags|                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               +
|                                                               |
+                                                               +
|                                                               |
+                                                               +
|                          IPv6 Address                         |
+                               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               |   PrefixLen   |    more ...   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Sig Type   |        Signature Length       |               ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+               ~
~                          Signature ...                        ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The 16-bit Count is the number of IPv6 Encapsulations.

13.5. MPLS Label List

As an MPLS enabled interface may have a label stack, see [RFC3032], a variable length list of labels is needed.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Label Count  |                 Label                 | Exp |S|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Label                 | Exp |S|    more ...   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

A Label Count of zero is an implicit withdraw of all labels for that prefix on that interface.

13.6. MPLS IPv4 Encapsulation

The MPLS IPv4 Encapsulation describes a logical link's ability to exchange labeled IPv4 packets on one or more subnets. It does so by stating the interface's addresses the corresponding prefix lengths, and the corresponding labels.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |    Type = 6   |         Payload Length        |     Count     |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      ...      | PrimLoop Flags|      MPLS Label List ...      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |              ...              |          IPv4 Address         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |              ...              |   PrefixLen   |    more ...   |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |    Sig Type   |        Signature Length       |               ~
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+               ~
   ~                         Signature ...                         ~
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The 16-bit Count is the number of MPLSv6 Encapsulations.

13.7. MPLS IPv6 Encapsulation

The MPLS IPv6 Encapsulation describes a logical link's ability to exchange labeled IPv6 packets on one or more subnets. It does so by stating the interface's addresses, the corresponding prefix lengths, and the corresponding labels.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |    Type = 7   |         Payload Length        |     Count     |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      ...      | PrimLoop Flags|      MPLS Label List ...      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |              ...              |                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               +
   |                                                               |
   +                                                               +
   |                                                               |
   +                                                               +
   |                          IPv6 Address                         |
   +                               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                               |   Prefix Len  |    more ...   |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |    Sig Type   |        Signature Length       |               ~
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+               ~
   ~                         Signature ...                         ~
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The 16-bit Count is the number of MPLSv6 Encapsulations.

14. KEEPALIVE - Layer 2 Liveness

L3DL devices SHOULD beacon frequent Layer 2 KEEPALIVE PDUs to ensure session continuity. A receiver may choose to ignore KEEPALIVE PDUs.

An operational deployment MUST BE configured to use KEEPALIVEs or not, either globally, or down to per-link granularity. Disagreement MAY result in repeated session break and reestablishment.

KEEPALIVEs SHOULD be beaconed at a configured frequency. One per second is the default. Layer 3 liveness, such as BFD, may be more (or less) aggressive.

If a KEEPALIVE is not received from a peer with which a receiver has an open session for a configurable time (default 30 seconds), the link SHOULD BE presumed down. The devices MAY keep configuration state and restore it without retransmission if no data have changed. Otherwise, a new session SHOULD BE established and new Encapsulation PDUs exchanged.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |    Type = 2   |       Payload Length = 0      |  Sig Type = 0 |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      Signature Length = 0     |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

15. VENDOR - Vendor Extensions

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |   Type = 255  |         Payload Length        |      ...      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |               Enterprise Number               |    Ent Type   |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      Enterprise Data ...                      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |    Sig Type   |        Signature Length       |               ~
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+               ~
   ~                         Signature ...                         ~
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+--+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Vendors or enterprises may define TLVs beyond the scope of L3DL standards. This is done using a Private Enterprise Number [IANA-PEN] followed by Enterprise Data in a format defined for that Enterprise Number and Ent Type.

Ent Type allows a VENDOR PDU to be sub-typed in the event that the vendor/enterprise needs multiple PDU types.

As with Encapsulation PDUs, a receiver of a VENDOR PDU MUST respond with an ACK or an ERROR PDU. Similarly, a VENDOR PDU MUST only be sent over an open session.

16. Layers 2.5 and 3 Liveness

Layer 2 liveness may be continuously tested by KEEPALIVE PDUs, see Section 14. As layer 2.5 or layer 3 connectivity could still break, liveness above layer 2 MAY be frequently tested using BFD ([RFC5880]) or a similar technique.

This protocol assumes that one or more Encapsulation addresses will be used to ping, BFD, or whatever the operator configures.

17. The North/South Protocol

Thus far, a one-hop point-to-point logical link discovery protocol has been defined.

The devices know their unique LLEIs and know the unique peer LLEIs and Encapsulations on each logical link interface.

Full topology discovery is not appropriate at the L3DL layer, so Dijkstra a la IS-IS etc. is assumed to be done by higher level protocols such as BGP-SPF.

Therefore the LLEIs, link Encapsulations, and state changes are pushed North via a small subset of the BGP-LS API. The upper layer routing protocol(s), e.g. BGP-SPF, learn and maintain the topology, run Dijkstra, and build the routing database(s).

For example, if a neighbor's IPv4 Encapsulation address changes, the devices seeing the change push that change Northbound.

17.1. Use BGP-LS as Much as Possible

BGP-LS [RFC7752] defines BGP-like Datagrams describing logical link state (links, nodes, link prefixes, and many other things), and a new BGP path attribute providing Northbound transport, all of which can be ingested by upper layer protocols such as BGP-SPF; see Section 4 of [I-D.ietf-lsvr-bgp-spf].

For IPv4 links, TLVs 259 and 260 are used. For IPv6 links, TLVs 261 and 262. If there are multiple addresses on a link, multiple TLV pairs are pushed North, having the same ID pairs.

17.2. Extensions to BGP-LS

The Northbound protocol needs a few minor extensions to BGP-LS. Luckily, others have needed the same extensions.

Similarly to BGP-SPF, the BGP protocol is used in the Protocol-ID field specified in table 1 of [I-D.ietf-idr-bgpls-segment-routing-epe]. The local and remote node descriptors for all NLRI are the IDs described in Section 11. This is equivalent to an adjacency SID or a node SID if the address is a loopback address.

Label Sub-TLVs from [I-D.ietf-idr-bgp-ls-segment-routing-ext] Section 2.1.1, are used to associate one or more MPLS Labels with a link.

18. Discussion

This section explores some trade-offs taken and some considerations.

18.1. HELLO Discussion

A device with multiple Layer 2 interfaces, traditionally called a switch, may be used to forward frames and therefore packets from multiple devices to one logical interface (LLEI), I, on an L3DL speaking device. Interface I could discover a peer J across the switch. Later, a prospective peer K could come up across the switch. If I was not still sending and listening for HELLOs, the potential peering with K could not be discovered. Therefore, interfaces MUST continue to send HELLOs as long as they are turned up.

18.2. HELLO versus KEEPALIVE

Both HELLO and KEEPALIVE are periodic. KEEPALIVE might be eliminated in favor of keeping only HELLOs. But KEEPALIVEs are unicast, and thus less noisy on the network, especially if HELLO is configured to transit layer-2-only switches, see Section 18.1.

19. VLANs/SVIs/Sub-interfaces

One can think of the protocol as an instance (i.e. state machine) which runs on each logical link of a device.

As the upper routing layer must view VLAN topologies as separate graphs, L3DL treats VLAN ports as separate links.

L3DL PDUs learned over VLAN-ports may be interpreted by upper layer-3 routing protocols as being learned on the corresponding layer-3 SVI interface for the VLAN.

As Sub-Interfaces each have their own LLIEs, they act as separate interfaces, forming their own links.

20. Implementation Considerations

An implementation SHOULD provide the ability to configure a logical interface as L3DL speaking or not.

An implementation SHOULD provide the ability to configure whether HELLOs on an L3DL enabled interface send Nearest Bridge or Nearest non-TPMR Bridge multicast frames from that interface; see Section 10.

An implementation SHOULD provide the ability to distribute one or more loopback addresses or interfaces into L3DL on an external L3DL speaking interface.

An implementation SHOULD provide the ability to configure one of the addresses of an encapsulation as primary on an L3DL speaking interface. If there is only one address for a particular encapsulation, the implementation MAY mark it as primary by default.

21. Security Considerations

The protocol as it is MUST NOT be used outside a datacenter or similarly closed environment due to lack of formal definition of the authentication and authorization mechanism. Sufficient mechanisms may be described in separate documents.

Many MDC operators have a strange belief that physical walls and firewalls provide sufficient security. This is not credible. All MDC protocols need to be examined for exposure and attack surface. In the case of L3DL, Authentication and Integrity as provided in [draft-ymbk-l3dl-signing] is strongly recommended.

It is generally unwise to assume that on the wire Layer 2 is secure. Strange/unauthorized devices may plug into a port. Mis-wiring is very common in datacenter installations. A poisoned laptop might be plugged into a device's port, form malicious sessions, etc. to divert, intercept, or drop traffic.

Similarly, malicious nodes/devices could mis-announce addressing.

If OPENs are not being authenticated, an attacker could forge an OPEN for an existing session and cause the session to be reset.

For these reasons, the OPEN PDU's authentication data exchange SHOULD be used.

22. IANA Considerations

This document requests the IANA create a registry for L3DL PDU Type, which may range from 0 to 255. The name of the registry should be L3DL-PDU-Type. The policy for adding to the registry is RFC Required per [RFC5226], either standards track or experimental. The initial entries should be the following:

PDU Code	PDU Name
0	HELLO
1	OPEN
2	KEEPALIVE
3	ACK
4	IPv4 Announcement
5	IPv6 Announcement
6	MPLS IPv4 Announcement
7	MPLS IPv6 Announcement
8-254	Reserved
255	VENDOR

This document requests the IANA create a registry for L3DL Signature Type, AKA Sig Type, which may range from 0 to 255. The name of the registry should be L3DL-Signature-Type. The policy for adding to the registry is RFC Required per [RFC5226], either standards track or experimental. The initial entries should be the following:

Number	Name
0	Null
1-255	Reserved

This document requests the IANA create a registry for L3DL PL Flag Bits, which may range from 0 to 7. The name of the registry should be L3DL-PL-Flag-Bits. The policy for adding to the registry is RFC Required per [RFC5226], either standards track or experimental. The initial entries should be the following:

Bit	Bit Name
0	Primary
1	Loopback
2-7	Reserved

This document requests the IANA create a registry for L3DL Error Codes, a 16 bit integer. The name of the registry should be L3DL-Error-Codes. The policy for adding to the registry is RFC Required per [RFC5226], either standards track or experimental. The initial entries should be the following:

Error Code	Error Name
0	Reserved
1	Logical Link Addressing Conflict
2	Authorization Failure in OPEN
3	Signature Failure in PDU

23. IEEE Considerations

This document requires a new EtherType.

24. Acknowledgments

The authors thank Cristel Pelsser for multiple reviews, Jeff Haas for review and comments, Joe Clarke for a useful review, John Scudder for deeply serious review and comments, Larry Kreeger for a lot of layer 2 clue, Martijn Schmidt for his contribution, Neeraj Malhotra for review, Russ Housley for checksum discussion and sBox, and Steve Bellovin for checksum advice.

25. References

25.1. Normative References

[I-D.ietf-idr-bgp-ls-segment-routing-ext]
Previdi, S., Talaulikar, K., Filsfils, C., Gredler, H., and M. Chen, "BGP Link-State extensions for Segment Routing", draft-ietf-idr-bgp-ls-segment-routing-ext-12 (work in progress), March 2019.

[I-D.ietf-idr-bgpls-segment-routing-epe]
Previdi, S., Talaulikar, K., Filsfils, C., Patel, K., Ray, S., and J. Dong, "BGP-LS extensions for Segment Routing BGP Egress Peer Engineering", draft-ietf-idr-bgpls-segment-routing-epe-18 (work in progress), March 2019.

[I-D.ietf-lsvr-bgp-spf]
Patel, K., Lindem, A., Zandi, S., and W. Henderickx, "Shortest Path Routing Extensions for BGP Protocol", draft-ietf-lsvr-bgp-spf-04 (work in progress), December 2018.

[IANA-PEN]
"IANA Private Enterprise Numbers", <https://www.iana.org/assignments/enterprise-numbers/enterprise-numbers>.

[IEEE.802_2001]
IEEE, "IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture", IEEE 802-2001, DOI 10.1109/ieeestd.2002.93395, July 2002, <http://ieeexplore.ieee.org/servlet/opac?punumber=7732>.

[IEEE802-2014]
Institute of Electrical and Electronics Engineers, "Local and Metropolitan Area Networks: Overview and Architecture", IEEE Std 802-2014, 2014.

[RFC1213] McCloghrie, K. and M. Rose, "Management Information Base for Network Management of TCP/IP-based internets: MIB-II", STD 17, RFC 1213, DOI 10.17487/RFC1213, March 1991, <http://www.rfc-editor.org/info/rfc1213>.

[RFC1629] Colella, R., Callon, R., Gardner, E., and Y. Rekhter, "Guidelines for OSI NSAP Allocation in the Internet", RFC 1629, DOI 10.17487/RFC1629, May 1994, <http://www.rfc-editor.org/info/rfc1629>.

[RFC1982] Elz, R. and R. Bush, "Serial Number Arithmetic", RFC 1982, DOI 10.17487/RFC1982, August 1996, <http://www.rfc-editor.org/info/rfc1982>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3032] Rosen, E., Tappan, D., Fedorkow, G., Rekhter, Y., Farinacci, D., Li, T., and A. Conta, "MPLS Label Stack Encoding", RFC 3032, DOI 10.17487/RFC3032, January 2001, <http://www.rfc-editor.org/info/rfc3032>.

[RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <http://www.rfc-editor.org/info/rfc4271>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, DOI 10.17487/RFC5226, May 2008, <http://www.rfc-editor.org/info/rfc5226>.

[RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <http://www.rfc-editor.org/info/rfc5880>.

[RFC6286] Chen, E. and J. Yuan, "Autonomous-System-Wide Unique BGP Identifier for BGP-4", RFC 6286, DOI 10.17487/RFC6286, June 2011, <http://www.rfc-editor.org/info/rfc6286>.

[RFC7752] Gredler, H., Ed., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP", RFC 7752, DOI 10.17487/RFC7752, March 2016, <http://www.rfc-editor.org/info/rfc7752>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <http://www.rfc-editor.org/info/rfc8174>.

25.2. Informative References

[Clos0]   Clos, C., "A study of non-blocking switching networks [PAYWALLED]", Bell System Technical Journal 32 (2), pp 406-424, March 1953.

[Clos1]   "Clos Network", <https://en.wikipedia.org/wiki/Clos_network/>.

[I-D.malhotra-bess-evpn-lsoe]
Malhotra, N., Patel, K., and J. Rabadan, "LSoE-based PE-CE Control Plane for EVPN", draft-malhotra-bess-evpn-lsoe-00 (work in progress), March 2019.

[JUPITER] Singh, A., Germano, P., Kanagala, A., Liu, H., Provost, J., Simmons, J., Tanda, E., Wanderer, J., HAP.lzle, U., Stuart, S., Vahdat, A., Ong, J., Agarwal, A., Anderson, G., Armistead, A., Bannon, R., Boving, S., Desai, G., and B. Felderman, "Jupiter rising", Communications of the ACM Vol. 59, pp. 88-97, DOI 10.1145/2975159, August 2016.

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, DOI 10.17487/RFC0791, September 1981, <http://www.rfc-editor.org/info/rfc791>.

[RFC1122] Braden, R., Ed., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, DOI 10.17487/RFC1122, October 1989, <http://www.rfc-editor.org/info/rfc1122>.

Authors' Addresses

Randy Bush
Arrcus & IIJ
5147 Crystal Springs
Bainbridge Island, WA 98110
United States of America

Email: randy@psg.com

Rob Austein
Arrcus, Inc

Email: sra@hactrn.net

Keyur Patel
Arrcus
2077 Gateway Place, Suite #400
San Jose, CA 95119
United States of America

Email: keyur@arrcus.com