Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 3, 2019

R. Bush
Arrcus & IIJ
K. Patel
Arrcus
July 2, 2018

Link State Over Ethernet
draft-ymbk-lsvr-lsoe-01

Abstract

Used in a Massive Data Center (MDC), BGP-LS and BGP-SPF need link neighbor discovery, liveness, and addressability data. Link State Over Ethernet protocols provide link discovery, exchange AFI/SAFIs, and discover addresses over raw Ethernet. These data are pushed directly to BGP-LS/SPF, obviating the need for centralized controller architectures. This protocol is more widely applicable, and has been designed to support a wide range of routing and similar protocols which need link discovery and characterisation.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in [RFC2119] only when they appear in all upper case. They may also appear in lower or mixed case as English words, without normative meaning.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 3, 2019.

Table of Contents

1. Introduction

The Massive Data Center (MDC) environment presents unusual problems of scale, e.g. O(10,000) switches, 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 and similar environments. But it needs link state and addressing data from the network to build the routing topology. LLDP has scaling issues, e.g. in extending a PDU beyond 1,500 bytes.

Link State Over Ethernet (LSOE) provides brutally simple mechanisms for devices to

- Discover each other's MACs,
- Run MAC keep-alives for liveness assurance,
- Discover each other's unique IDs (ASN, RouterID, ...),
- Negotiate mutually supported AFI/SAFIs,
- Discover and maintain link IP/MPLS addresses,
- Enable layer three link liveness such as BFD, and finally
- Push these data up to BGP-SPF which computes the topology and builds routing and forwarding tables.

This protocol is more widely applicable than BGP-SPF, and has been designed to support a wide range of routing and similar protocols which need link discovery and characterisation.

2. Terminology

Even though it concentrates on the Ethernet layer, this document relies heavily on routing terminology. The following are some possibly confusing terms:

AFI/SAFI: Address Family Indicator and Subsequent Address Family Indicator. I.e. classes of addresses such as IPv4, IPv6, ...

ASN:	Autonomous System Number [RFC4271], a BGP identifier for an originator of routing, particularly BGP, announcements, see [RFC4271].
RouterID:	[RFC4271].
BGP-SPF	A hybrid protocol using BGP transport but Dijkstra SPF decision process. See [I-D.ietf-lsvr-bgp-spf].
Clos:	A hierarchic switch topology commonly used in data centers.
Frame	The payload of an Ethernet packet.
MAC:	Medium Access Control, essentially an Ethernet address, six octets.
MDC:	Massive Data Center, O(1,000) TORs or more.
PDU:	Protocol Data Unit, essentially an application layer message.
SPF:	Shortest Path First, an algorithm for finding the shortest paths between nodes in a graph.
TOR:	Top Of Rack switch, aggregates the servers in a rack and connects to the Clos spine.
ZTP:	Zero Touch Provisioning gives devices initial addresses, credentials, etc. on boot/restart.

3. Background

LSOE assumes a Clos-like topology, though the acyclic constraint is not necessary.

While LSOE is designed for the MDC, there are no inherent reasons it could not run on a WAN; though it is not clear that this would be useful. The authentication and authorisation needed to run safely on the WAN are not (yet) included in this protocol.

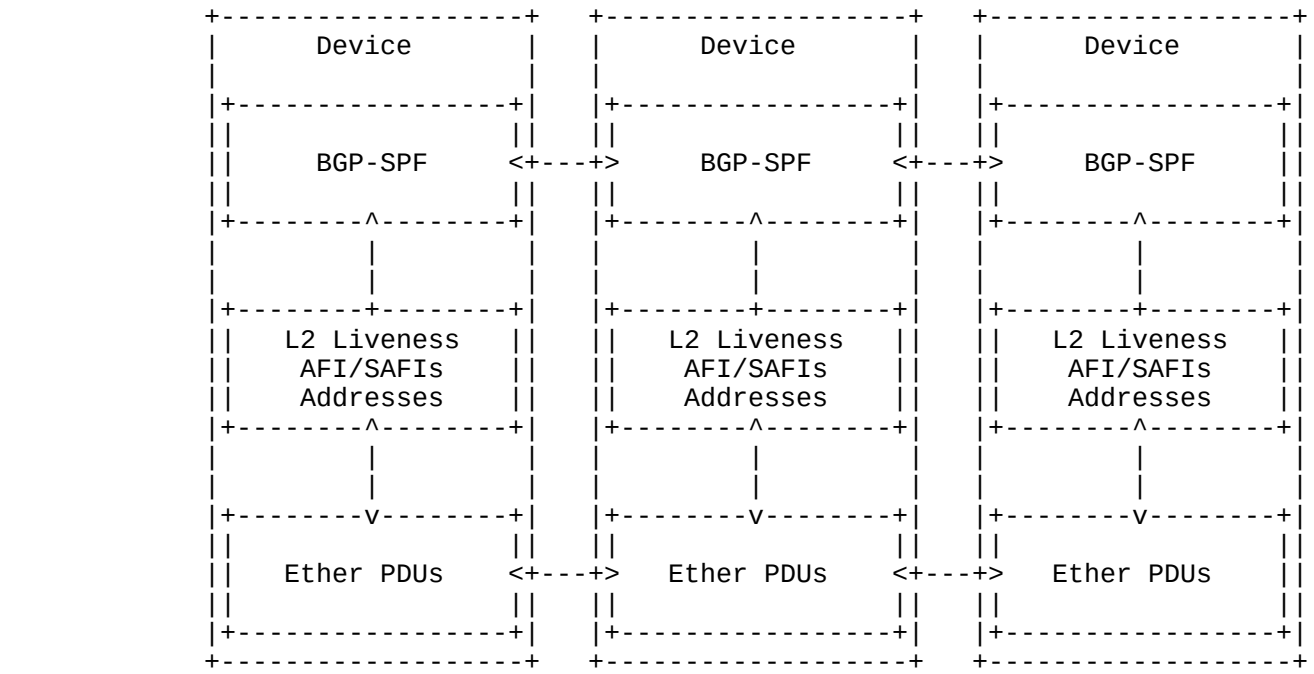
LLDP is not suitable because one can not extend a PDU beyond 1500 bytes without hitting an IPR barrier. It is also complex.

UDP is unsuitable as it would require prior knowledge of IP level addressing, one of the key purposes of this discovery protocol.

LSOE assumes a new IEEE assigned EtherType (TBD).

4. Top Level Overview

- MAC Link State is exchanged over Ethernet
- AFI/SAFI data are exchanged and IP-Level Liveness Checks done
- BGP-SPF uses the data to discover and build the topology database



There are two sets of protocols:

- Ethernet to Ethernet protocols are used to exchange layer 2 data, i.e. MACs, and layer 2.5 and 3 data, i.e. ASNs, AFI/SAFIs, and interface addresses.

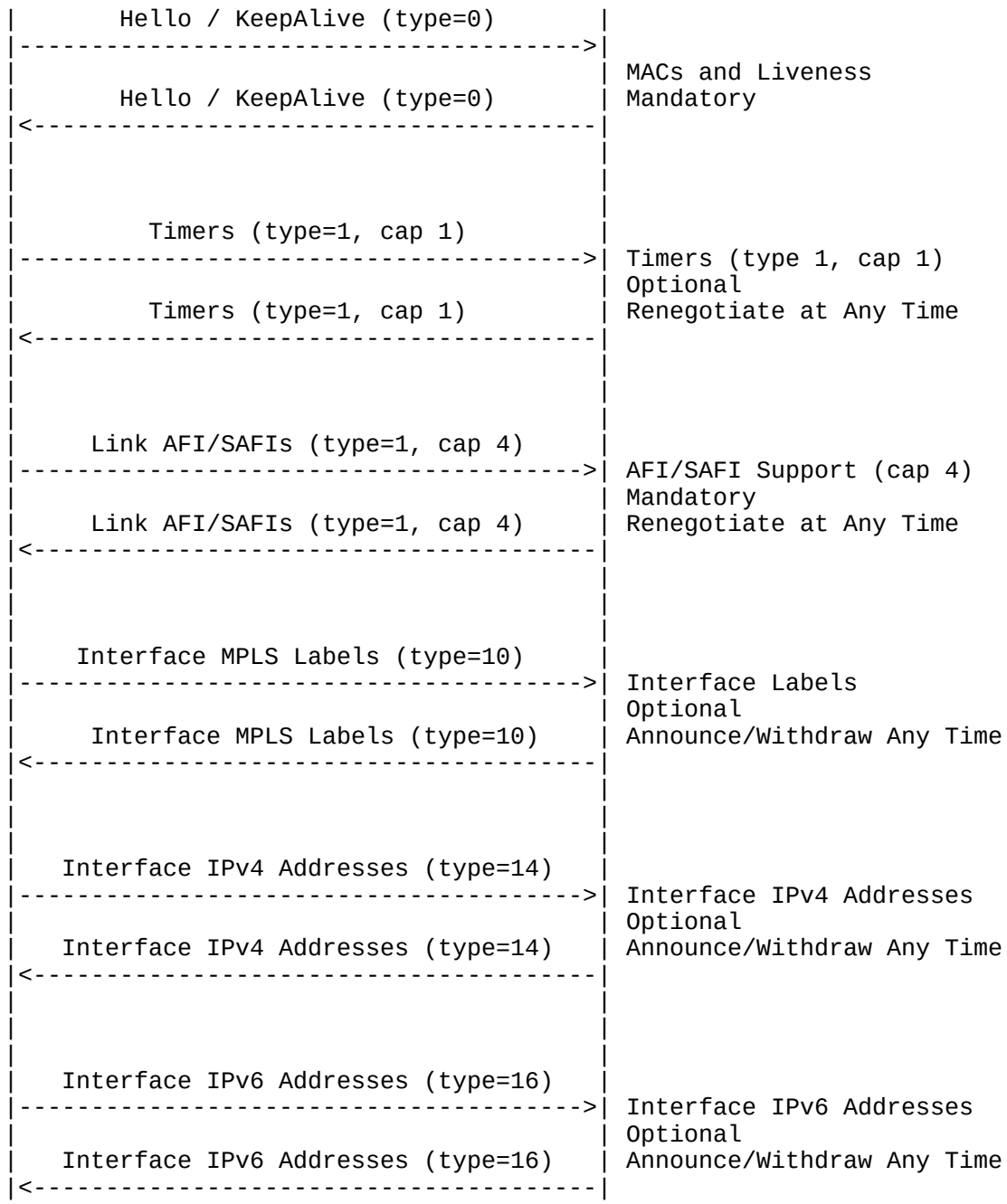
- A Link Layer to BGP protocol pushes these data up the stack to BGP-SPF, converting to the BGP-LS BGP-like data format.

- And, of course, the BGP layer crosses all the devices, though it is not part of these LSOE protocols.

5. Ethernet to Ethernet Protocols

The basic Ethernet Framed protocols

5.1. Inter-Link Ether Protocol Overview



5.2. PDUs and Frames

This is all about inter-device Link State.

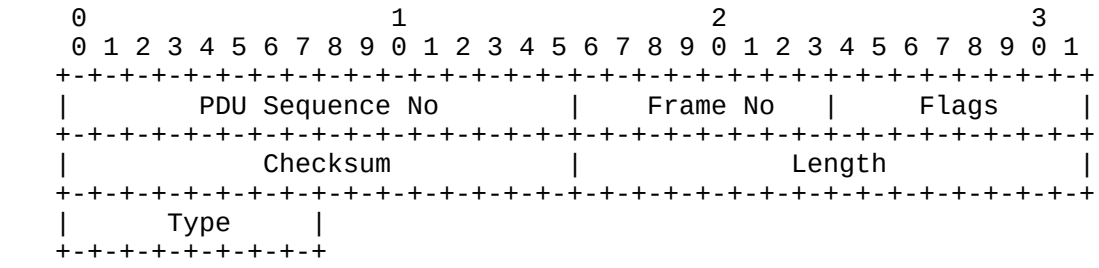
A PDU is one or more Ethernet Frames.

A Frame has a PDU Sequence Number and a Frame Number to allow assembly of out order frames.

Because BGP-SPF and Data Plane payloads are assumed to be IP over the same Ethernet, one needs to keep an eye on congestion.

5.2.1. Frame TLV

The basic Ethernet PDU is a typical TLV (Type Length Value) PDU, except it’s really LTV for the sake of alignment :)



The fields of the basic Ethernet PDU are as follows:

PDU Sequence No: Semi-unique identifier of a TLV PDU (e.g. the low order 16 bits of UNIX time)

Frame No: 0..255 Frame Sequence Number Within a multi-frame PDU

Flags: A bit field

- 0 - Sender has been restarted
- 1 - One of a multi-Frame sequence
- 2 - last of a multi-Frame sequence
- 3-7 - Reserved

Checksum: One’s complement over Frame, detect bit flips

Length: Total Bytes in PDU including all frames and fields

Type: An integer

- 0 - Hello / KeepAlive

1 - Capability
2-9 - Reserved
10 - AFI/SAFI ACK
11 - IPv4 Announce / Withdraw
12 - IPv6 Announce / Withdraw
13 - MPLS IPv4 Announce / Withdraw
14 - MPLS IPv6 Announce / Withdraw
15-255 Reserved

5.2.1.1. The Checksum

There is a reason conservative folk use a checksum in UDP. And when the operators stretch to jumbo frames ...

One’s complement is a bit silly, though trivial to implement and might be sufficient.

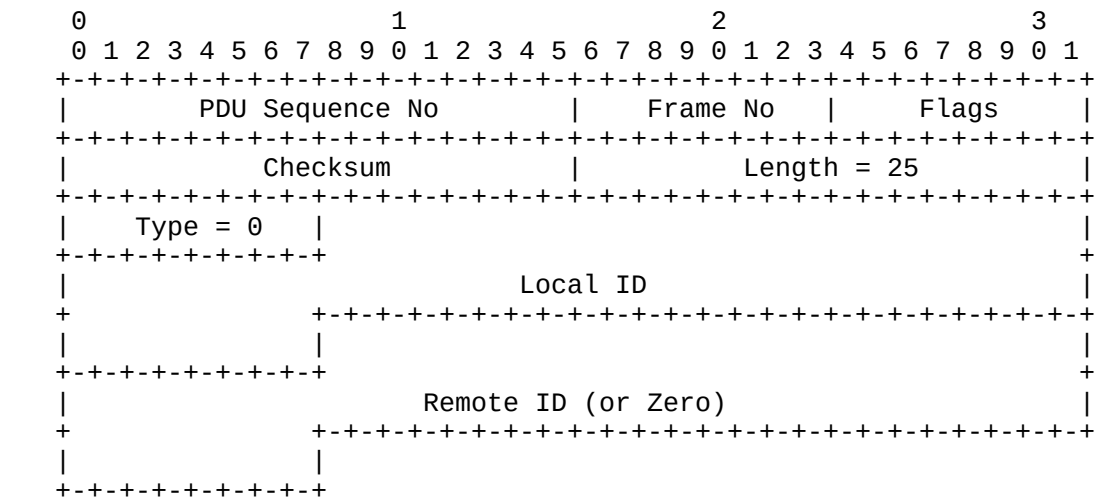
```
   Sum up either 16-bit shorts in a 32-bit int, or 32-bit ints in a
   64-bit long, then take the high-order section, shift it right,
   rotate, add it in, repeat until zero.  -- smb off the top of his head


   /* The F table from Skipjack, and it would work for the S-Box.
      There are other S-Box sources as well. -- Russ Housley */
   const BYTE 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
   };

   /* example C code, constant time even, thanks Rob Austein */

   uint16_t sbox_checksum(const *b, const size_t n)
   {
     uint32_t sum[2] = {0, 0};
     for (int i = 0; i < n; i++)
       sum[i & 1] += sbox[b[i]];
     uint32_t result = (sum[0] << 8) + sum[1];
     result = (result >> 16) + (result & 0xFFFF);
     result = (result >> 16) + (result & 0xFFFF);
     return (uint16_t) result;
   }
```

5.2.2. Link Hello / KeepAlive

The Hello and KeepAlive PDUs are one and the same.

Each device learns the other's MAC from its HELLO whining. I.e., all devices on a wire/interface know each others MACs and learn each other's IDs.

An ID can be an ASN with high order bits zero, a classic RouterID with high order bits zero, a catenation of the two, a 48-bit ISO System-ID, or any other identifier unique to a single device in the BGP-SPF routing space.



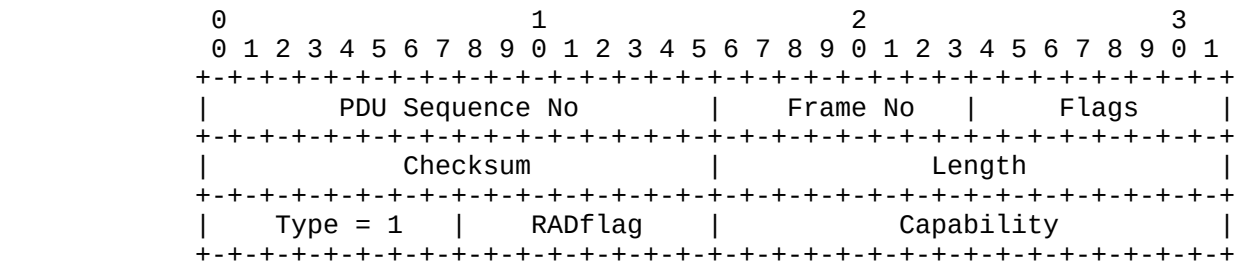
Once two devices know each other's MACs, Ethernet keep-alives may be started to ensure layer two liveness. The timing and acceptable drop of the keep-alives may be set with the Timer Negotiation capability exchange.

When the local sends a first Hello without knowing the remote device's ID, the Remote ID SHOULD be zero. The Local ID MUST never be zero.

5.2.3. Capability Exchange

Peers on the Ethernet exchange capabilities, such as timers, AFI/SAFIs supported, etc. There is a simple capability exchange.

By convention, the device with the lowest MAC sends first.

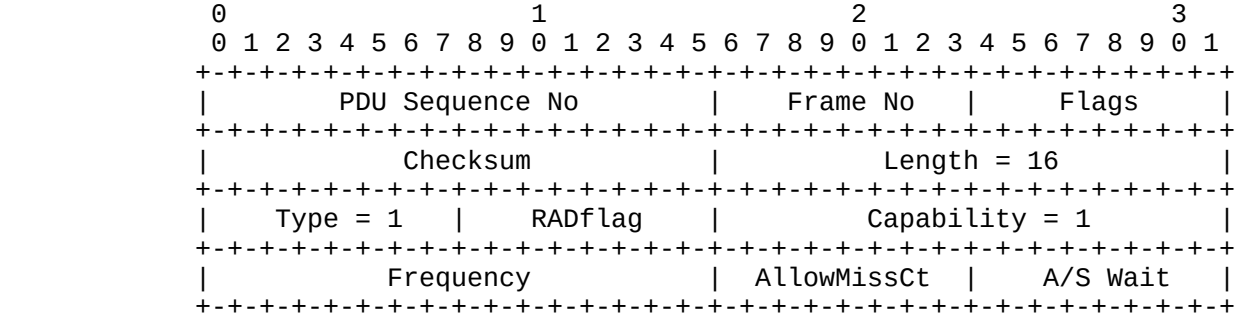


The RADflag is an integer field which signals the capability negotiation.

bit 0 - Request
bit 1 - Accept
bit 2 - Deny
bits 3-255 - Reserved

5.2.4. Timer Negotiation

Different operational scenarios may call for layer two and layer three timers which differ from the defaults. So there is a capability negotiation to modify these timers.



The meaning of the timer fields are as follows:

Frequency: Seconds/10 between KeepAlives (Default is 600)
AllowMissCt: Number of missed KeepAlives before declared down
A/S Wait AFI/SAFI ACK Timeout in Sec/10 (default 10)

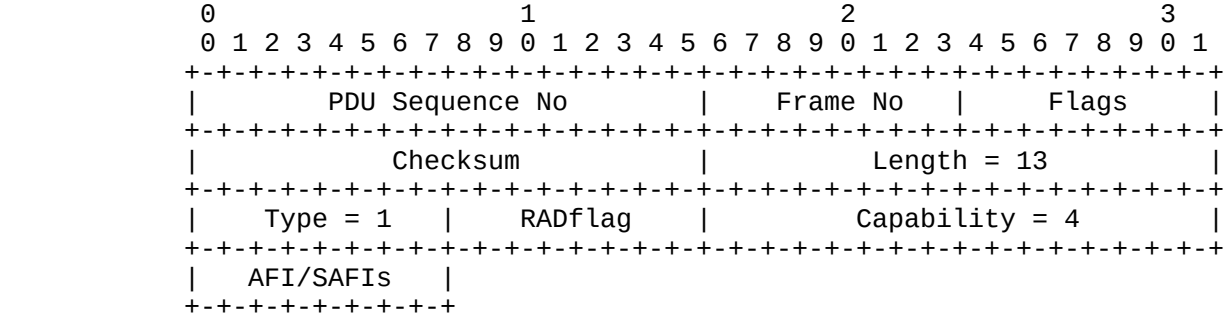
5.3. The AFI/SAFI Exchanges

The devices know each other’s MACs, have means to ensure link state, and know each other’s ASNs. Now they can negotiate which AFI/SAFIs are supported, and announce their interface addresses (and labels).

5.3.1. AFI/SAFI Capability Exchange

First they negotiate what AFI/SAFIs are supported on the link.

As before, the lowest MAC initiates the negotiation.

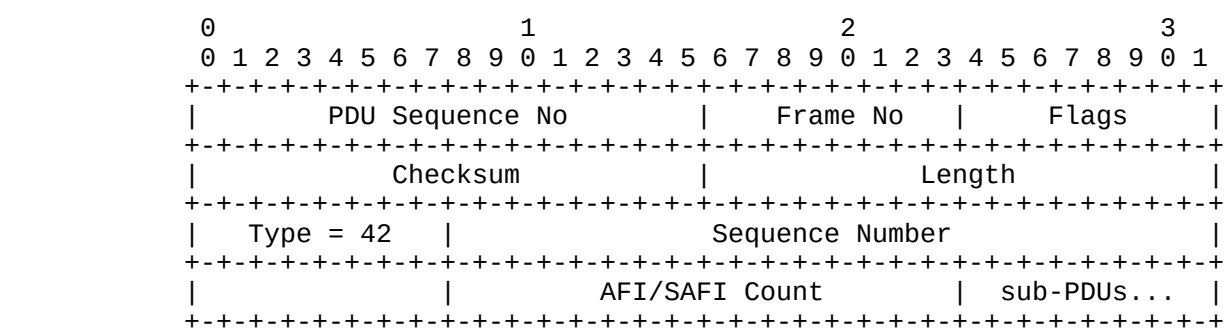


The AFI/SAFIs currently defined are as follows:

10 - IPv4
11 - IPv6
12 - MPLS IPv4
13 - MPLS IPv6
... - other tunnels (e.g. GRE)

5.3.2. The AFI/SAFI PDU Skeleton

Now both sides can exchange their actual interfaces addresses for all the negotiated AFI/SAFIs.



The AFI/SAFI Exchange is over an unreliable transport so there are Sequence Numbers and ACKs.

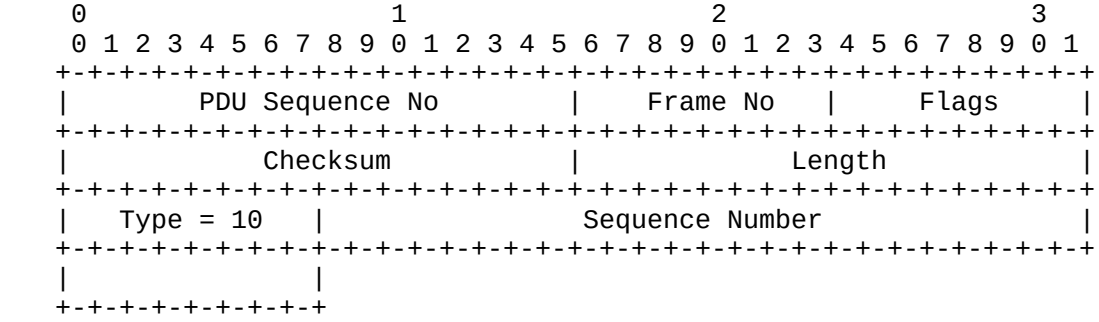
The Sequence Number is a point-to-point link announcement counter, incremented for each exchange in each direction on the link.

The Receiver will ACK it with a Type=10, see following PDU.

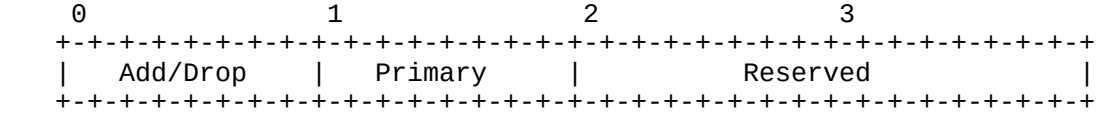
If the Sender does not receive an ACK in one second, they retransmit. Other delay timers may be negotiated using the Timing Capability.

If a sender has multiple links on the same interface, separate counters must be kept for each.

5.3.3. AFI/SAFI ACK



5.3.4. Add/Drop/Prim



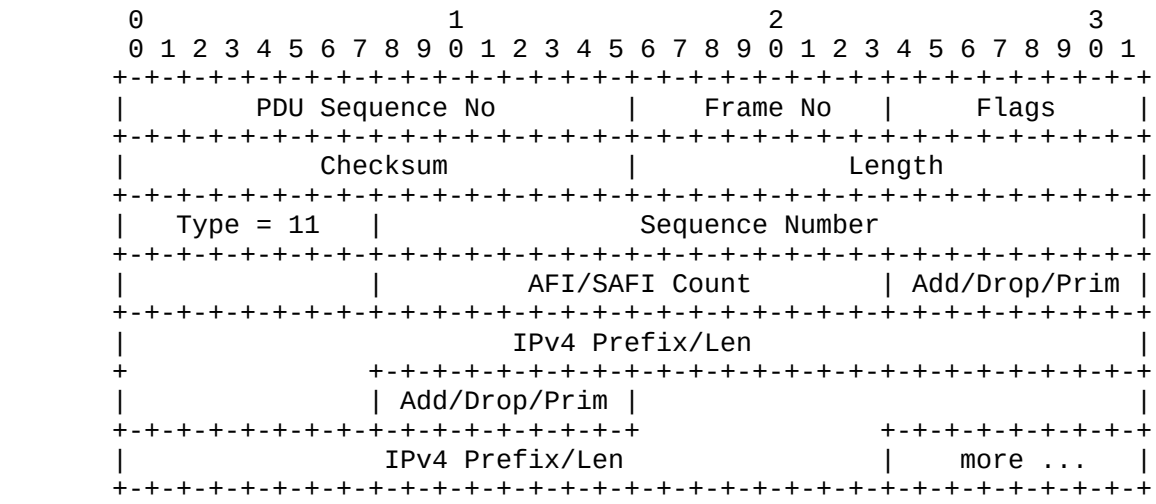
Each AFI/SAFI interface address may be announced (Add/Drop == 1), or withdrawn (Add/Drop == 0).

An interface may have multiple AFI/SAFIs.

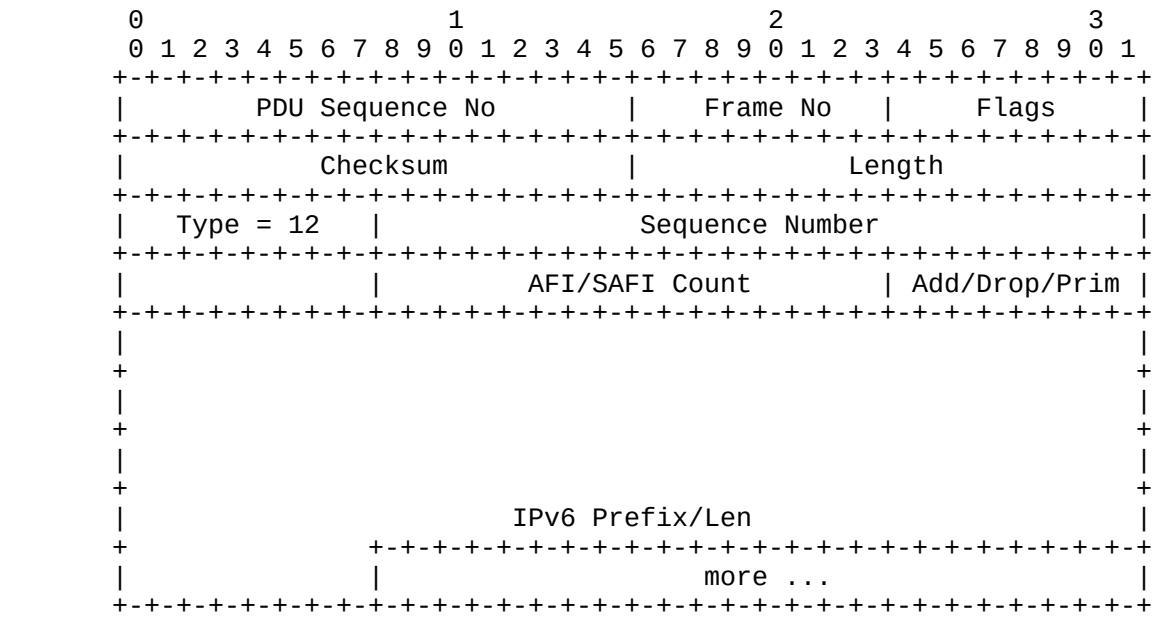
For each AFI/SAFI on an interface there might be multiple addresses.

One address per AFI/SAFI SHOULD be marked as primary (Primary == 1).

5.3.5. IPv4 Announce / Withdraw

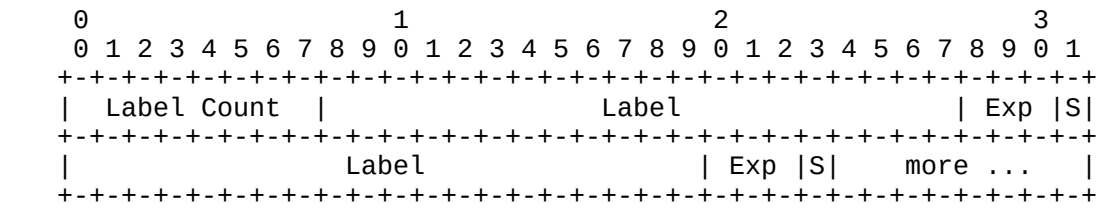


5.3.6. IPv6 Announce / Withdraw

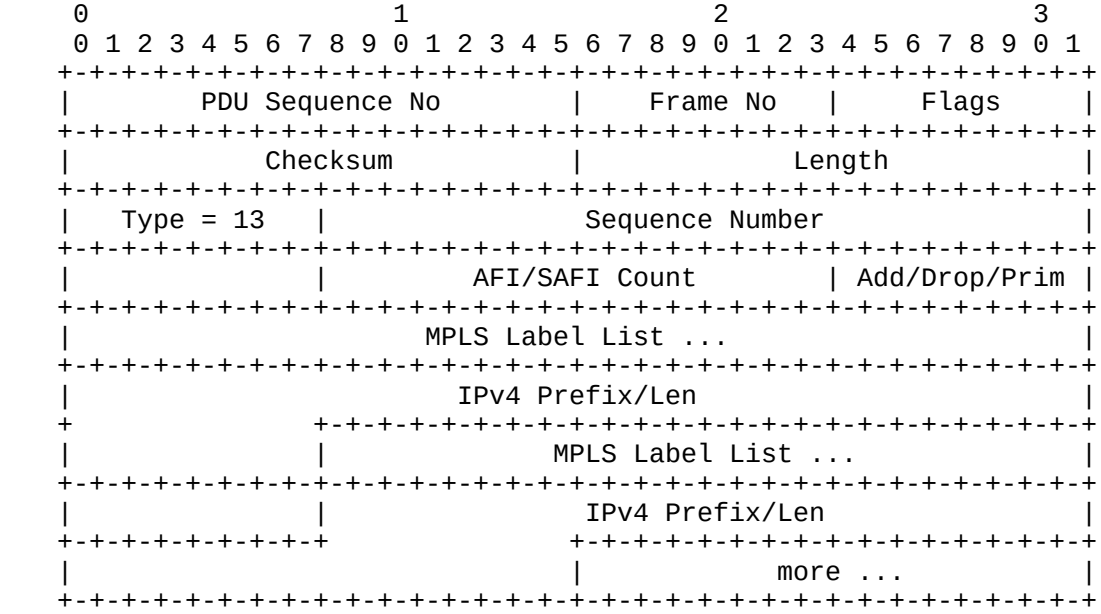


5.3.7. MPLS Label List

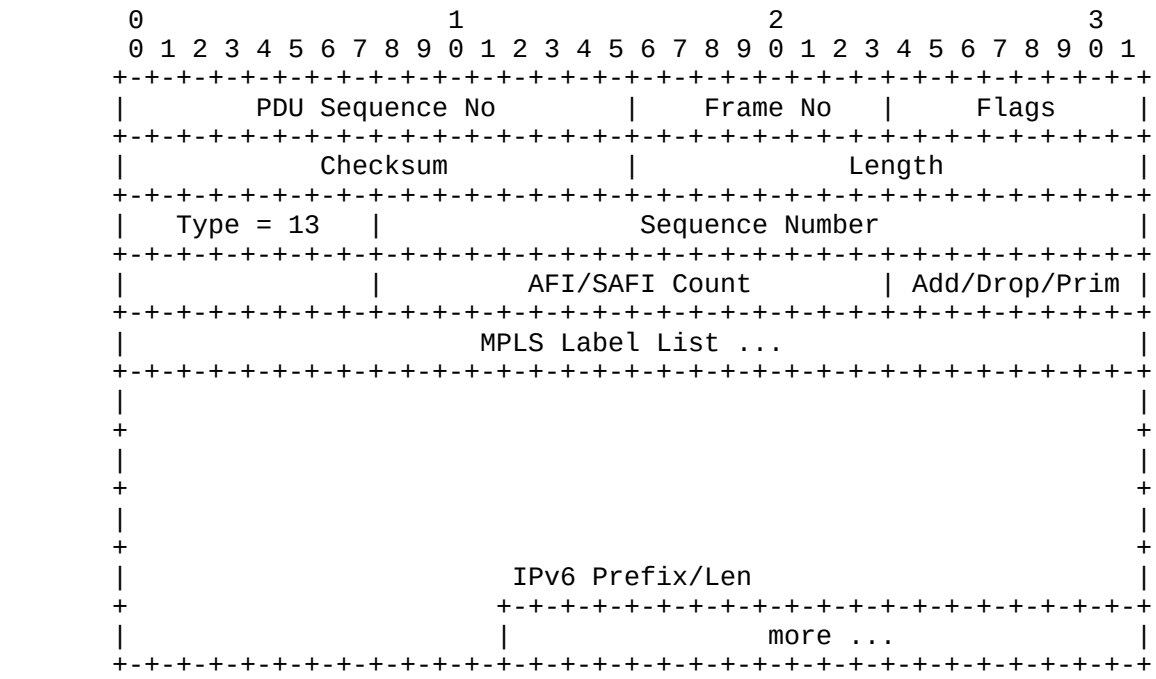
As an MPLS enabled interface may have a label stack, see [RFC3032], a variable length list of labels is needed.



5.3.8. MPLS IPv4 Announce / Withdraw



5.3.9. MPLS IPv6 Announce / Withdraw



6. Layer 2.5 and 3 Liveness

Ether liveness is continuously tested by Hello Keep-Alives, see Section 5.2.2. Now IP/Label liveness may be tested. As layer 2.5 or layer 3 connectivity could still break, liveness above layer 2 should be frequently tested.

Assume one or more AFI/SAFI addresses will be used to ping, BFD, or whatever the operator configures.

7. The North/South Protocol

Thus far, we have a one-hop point-to-point link discovery protocol.

We know what unique node identifiers (ASNs, RouterIDs, ...) and AFI/SAFIs are on each Link Interface.

At the Ethernet layer we do not want to do topology discovery and Dijkstra a la IS-IS.

So the node identifiers, link AFI/SAFIs, and state changes are pushed North to BGP-SPF which discovers and maintains the topology, runs Dijkstra, and builds the routing database.

For example, if a neighbor's MAC changes, the device seeing the change pushes that change Northbound.

7.1. Use BGP-LS as Much as Possible

BGP-LS [RFC7752] defines BGP-like PDUs describing 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 BGP-SPF; see Section 4 of [I-D.ietf-lsvr-bgp-spf].

For IPv4 links, TLVs 259 and 260 are used. And 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.

7.2. 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 ID's described in Section 5.2.2.

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.

8. Security Considerations

The protocol as is MUST NOT be used outside a datacenter or similarly closed environment due to lack of authentication and authorisation. These will be worked on in a later effort, likely using credentials configured using ZTP.

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.

On the wire Ethernet is assumed to be secure, though it could be tapped and data modified by an in-house on the wire attacker.

Malicious nodes/devices could mis-announce addressing, form malicious associations, etc.

9. IANA Considerations

This document requests the IANA create a registry for LSOE PDU Type, which may range from 0 to 255. The name of the registry should be LSOA-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 / KeepAlive
1	Capability
2-9	Reserved
10	AFI/SAFI ACK
11	IPv4 Announce / Withdraw
12	IPv6 Announce / Withdraw
13	MPLS IPv4 Announce / Withdraw
14	MPLS IPv6 Announce / Withdraw
15-255	Reserved

This document requests the IANA create a registry for LSOE AFI/SAFI Type, which may range from 0 to 255. The name of the registry should be LSOA-AFI/SAFI-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:

AFI/SAFI Type Code	AFI/SAFI Type Name
0-9	Reserved
10	IPv4
11	IPv6
12	MPLS IPv4
13	MPLS IPv6
14-255	Reserved

10. IEEE Considerations

This document needs a new EtherType.

11. Acknowledgments

The authors thank Cristel Pelsser for multiple reviews, Joe Clarke for a useful review, Martijn Schmidt for his contribution, Rob Austein for reviews and checksum code, Russ Housley for checksum discussion and sBox, and Steve Bellovin for checksum advice.

12. References

12.1. 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-08 (work in progress), May 2018.

[I-D.ietf-idr-bgpls-segment-routing-epe]
Previdi, S., 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-15 (work in progress), March 2018.

[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-01 (work in progress), May 2018.

[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>.

[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>.

12.2. Informative References

[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.

Authors' Addresses

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

Email: randy@psg.com

Keyur Patel
Arrcus
2077 Gateway Place, Suite #250
San Jose, CA 95119
United States of America

Email: keyur@arrcus.com