Workgroup: Network Working Group
Internet-Draft: draft-ymbk-idr-l3nd-06
Published: 28 May 2023
Intended Status: Standards Track
Expires: 29 November 2023
Authors: R. Bush
Arrcus & Internet Initiative Japan
R. Housley
Vigil Security
R. Austein
Arrcus
S. Hares
Hickory Hill Consulting
K. Patel
Arrcus

Layer-3 Neighbor Discovery

Abstract

Data Centers where the topology is BGP-based need to discover neighbor IP addressing, IP Layer-3 BGP neighbors, etc. This Layer-3 Neighbor Discovery protocol identifies BGP neighbor candidates.

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

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 29 November 2023.

Copyright Notice

Table of Contents

1. Introduction

The Massive Data Center (MDC) environment presents unusual problems of scale, e.g. O(10,000) forwarding devices, while its homogeneity presents opportunities for simple approaches. Layer-3 Discovery and Liveness (L3DL), [I-D.ietf-lsvr-l3dl], provides neighbor discovery at Layer-2. This document (set) provides a similar solution at Layer-3, attempting to be as similar as reasonable to L3DL.

Some guiding principles when dealing with datacenters with tens of thousands of devices are

- *Predictable Reliability,
- *Security: Authorization and Integrity more than Confidentiality, and
- *Massive Scalability

Layer-3 Neighbor Discovery (L3ND) provides brutally simple mechanisms for neighboring devices to

- *Discover each other's IP Addresses,
- *Discover mutually supported layer-3 encapsulations, e.g. IPv4/ IPv6//MPLS,
- *Discover Layer-3 IP and/or MPLS addressing of interfaces of the encapsulations,
- *Provide authenticity, integrity, and verification of protocol messages, and
- *Accommodate scaling needed for EVPN etc.

L3ND is intended for use within single IP subnets (IP over Ethernet or other point-to-point or multi-point IP link) in order to exchange the data needed to bootstrap BGP-based peering, EVPN, etc.; especially in a datacenter Clos [Clos] environment. Once IP connectivity has been leveraged to discover layer-3 addressability and forwarding capabilities, normal IP forwarding and routing can take over.

L3ND might be more widely applicable to a range of routing and similar protocols which need Layer-3 neighbor discovery.

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:

Clos: A hierarchic subset of a crossbar switch topology commonly used in data centers [Clos].

Datagram: The L3ND content of a single Layer-3 UDP Datagram.

Encapsulation: Address Family Indicator and Subsequent Address Family Indicator (AFI/SAFI). I.e. classes of Layer-2.5 and Layer-3 addresses such as IPv4, IPv6, MPLS.

Link or Logical Link: A logical connection between two interfaces on two different devices. E.g. two VLANs between the same two ports are two links.

MDC: Massive Data Center, commonly composed of thousands of Top of Rack Switches (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 L3ND application layer message.

Session: An established, via exchange of OPEN PDUs, session between two L3ND capable IP interfaces on a link.

TOR Switch: Top Of Rack switch, aggregates the servers in a rack and connects to aggregation layers of the Clos tree, AKA the Clos spine.

3. Background

L3ND is primarily designed for a Clos type datacenter scale and topology, but can accommodate richer topologies which contain potential cycles.

While L3ND 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.

The number of addresses of one Encapsulation type on an interface link may be quite large given a TOR switch with tens of servers, each server having a few hundred micro-services, resulting in an inordinate number of addresses. And highly automated micro-service

migration can cause serious address prefix disaggregation, resulting in interfaces with thousands of disaggregated prefixes.

To meet such scaling needs, the L3ND protocol is session oriented and uses incremental announcement and withdrawal with session restart, a la BGP ([RFC4271]).

4. Inter-Link Protocol Overview

A device broadcasts a Layer-3 Multicast UDP datagram (HELLO) containing the port number that is willing to serve a TLS or raw TCP connection to support the data exchange of the rest of the protocol in a reliable and preferably authenticated manner.

Another device on the link then establishes a TLS or raw TCP session in which inter-device PDUs are used to exchange device and logical link identities and layer-2.5 (MPLS) and 3 identifiers (not payloads), e.g. more IP Addresses, loopback addresses, port identities, and Encapsulations.

To assure discovery of new devices coming up on a multi-link topology, devices on such a topology, and only on a multi-link topology, send periodic HELLOs forever, see Section 12.1.

Given the TLS/TCP session, OPEN PDUs (Section 8) are exchanged, the Encapsulations (Section 10) configured on an end point may be announced and modified. Note that these are only the encapsulation and addresses configured on the announcing interface; though a device's loopback and overlay interface(s) may also be announced.

4.1. L3ND Ladder Diagram

The HELLO, Section 6, is a priming message sent on all logical links configured for L3ND. It is a small L3ND Multicast UDP PDU with the simple goal of advertising a TLS/TCP service available on an advertised port on the sending IP interface.

The HELLO PDU is either IPv4 or IPv6, which selects the AFI to be used for the rest of the session(s) between end-points. Two endpoints MAY establish a link for each AFI.

An interface on the link receiving the HELLO PDU attempts to establish a TLS or raw TCP, as specified by the HELLO, session to the source IP address of the HELLO on the port advertised in the HELLO.

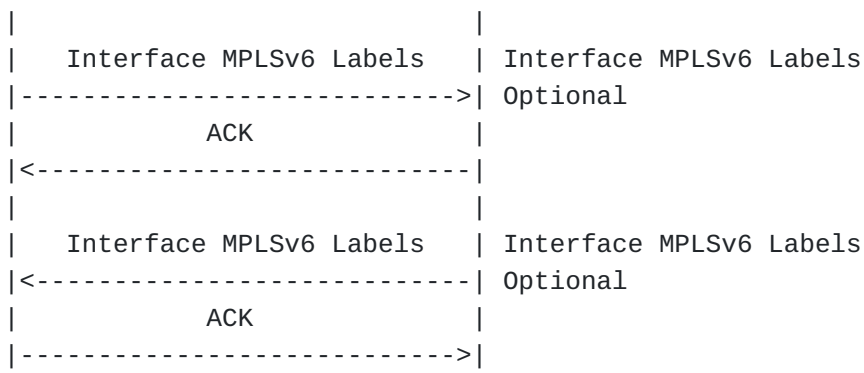
The OPEN, Section 8 PDUs, used to exchange details about the L3ND session, and the ACK/ERROR PDU, are mandatory; other PDUs are optional; though at least one encapsulation SHOULD be agreed at some point.

Like Multi-Protocol BGP, [RFC4760], an L3ND session running over one AFI MAY carry encapsulations etc. of different AFIs,

A L3DL extension, [I-D.ymbk-idr-l3nd-ulpc], describes the next upper layer L3DL protocol to exchange BGP parameter information.

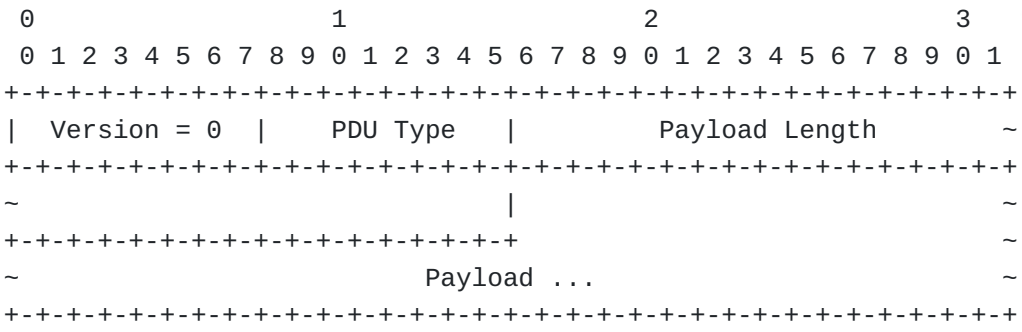
The following is a ladder-style diagram of the L3ND protocol exchanges:


HELLO
Logical Link Peer discovery
TCP OPEN
Mandatory
OPEN
IDs, security, etc.
ACK
OPEN
Mandatory
ACK
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

5. TLV PDUs

The basic L3ND application layer PDU is a typical TLV (Type Length Value) PDU. As it is transported over TCP, integrity is assured. When it is transported over TLS, authenticity is also provided.



The fields of the basic L3ND header are as follows:

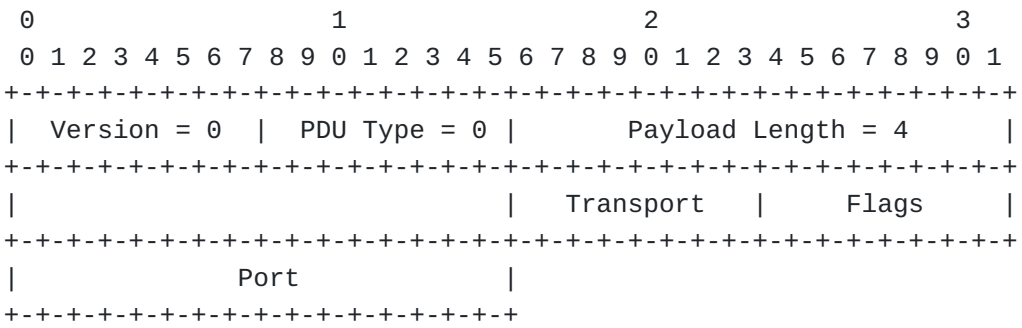
Version: An integer differentiating versions of the L3ND protocol. Currently only Version 0 MAY BE specified.

PDU Type: An integer differentiating PDU payload types. See Section 16.3.

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

Payload: The application layer content of the L3ND PDU.

6. HELLO



The Payload Length is 4 to cover the Transport, Flags, and Port fields.

The IPv4 UDP packets are sent to the IPv4 link local multicast address (TBD1) and the IPv6 UDP packets are sent to an IPv6 link Local multicast address (TBD2). See Section 12.1 for why multicast is used.

The HELLO PDU solicits a unicast TLS/TCP session open request of the same AFI from other devices on the link.

When a HELLO is received from a source IP address with which there is no established TLS/TCP L3ND session, if the receiver has the higher of the two IP addresses, it SHOULD respond by sending a TLS/TCP client open request, using the same AFI, to the source IP address of the HELLO to establish an L3ND TLS/TCP session.

All L3ND PDUs other than HELLO are sent via TLS/TCP, as the server's destination IP address is known after the HELLO.

When an interface is turned up on a device, it SHOULD issue a HELLO if it is configured to participate in L3ND sessions and repeat HELLOs at a configured interval, with a default of 60 seconds.

If the configured multicast destination address is one that is propagated by switches, the HELLO SHOULD be repeated at a configured interval, with a default of 60 seconds. This allows discovery by new devices which come up on the mesh. In this multi-link scenario, the operator should be aware of the trade-off between timer tuning and network noise and adjust the inter-HELLO timer accordingly.

By default, GTSM, [RFC5082], SHOULD be enabled to test that a received HELLO MUST be on the local link; thus leaving no middle on which a monkey in the middle might stand. It MAY be disabled by configuration. GTSM check failures SHOULD be logged, though with rate limiting to keep from overwhelming logs.

If more than one device responds, one adjacency is formed for each unique source IP address. L3ND treats each adjacency as a separate logical link.

To ameliorate possible load spikes during bootstrap or event recovery, there SHOULD be a jittered delay between receipt of a HELLO and TLS/TCP open. The default delay range SHOULD be zero to five seconds, and MUST be configurable.

If a HELLO is received from an IP Address with which there is an established session for that AFI, the HELLO SHOULD be dropped.

A device with a TLS/TCP listener SHOULD log or otherwise report repeated failed inbound attempts.

6.1. Transport

The Transport signals the type of transport security for the session.

The actual transport options are actually pre-configured in the devices by provisioning, as most require certificates etc. It is best to think of this field as in-band signaling to conform the correctness of the pre-configurations. Any disagreements MUST BE

considered to indicate an error condition and brought to the attention of the operator.

The Transport field is an enumeration with the following values:

0: Raw TCP: TLS is not used.

1: TLS TOFU: TLS using a self-signed server certificate.

2: TLS CA-NoIP: TLS using a CA-Based server certificate, with no IP address extension.

3: TLS CA WithIP: TLS using a CA-Based server certificate, with the server's IP address in the subject alternative name extension (see [RFC5280] Section 4.2.1.6).

4-255: Reserved.

If server certificates are to be used, they may be locally generated and then signed by a CA or generated by the CA and loaded. See [RFC8635].

6.2. Flags

Though the Working Group scope for this protocol is within a data center, an issue was raised that, on an internet echange with route server(s), it would attempt to form adjacencies with all members of the exchange. Hence a Flag field is provided to indicate that a device does not intend to field a TLS/TCP server on the announcing interface, but does seek one or more from peers.

Currently, only one Flags field is defined

Bit 0: Client Only This interface does not provide a TLS/TCP server.

Bits 1-7: Reserved.

6.3. Port

The Port is the two octet TCP Port Number (default is TBD3) on which the HELLO sender SHOULD have a waiting TLS/TCP (as specified in Flags) server listening unless the Client Only Flag is set. Though the IANA assigned well-known port SHOULD be used, this field allows configuration of alternate ports.

7. TCP Set-Up

As it is assumed that the configured deployment of a data center would have compatible parameters on all devices, any disagreement

over TLS/TCP or trust anchors MUST be logged; with rate limiting of the logging.

By default, GTSM, [RFC5082], SHOULD be enabled to ensure that a SYN received in response to a HELLO is on the local link. It MAY be disabled by configuration. GTSM check failures SHOULD be logged; though with rate limiting to keep from overwhelming logs.

If the receiver of a HELLO agrees with the sender's choice of TLS/TCP and authentication, both sides have agreed on an AFI for the transport and on each other's IP address in that AFI. This is sufficient to open a TCP session between them, which will allow for reliable transport of very large data PDUs while obviating the need to invent complex transports.

The L3ND peer with the higher IP address MUST act as the TLS/TCP client and open the transport session (send SYN) toward the peer with the lower IP address.

The server, the sender of the HELLO from the lower IP address, listens on the advertised port for the TLS/TCP session open. The receiver of the acceptable HELLO, the TLS/TCP client, initiates a TLS or raw TCP session toward the sender of the HELLO, the TLS/TCP server, preferably TLS, as advertised. If TLS, the server has chosen and signaled either a self-signed certificate or one configured from the operational CA trusted by both parties, as negotiated in the HELLO exchange.

Once the TLS/TCP session is established, if its interface is configured as point to point, the client side SHOULD stop listening on any port for which it has sent a HELLO. The server, if configured as a point to point interface SHOULD stop sending HELLOs.

If the TLS/TCP open fails, then this SHOULD be logged and the parties MUST go back to the initial state and try HELLO. Logging SHOULD be rate limited.

Should an interface with an established TLS/TCP session be reconfigured changing the TLS/TCP parameters, the TLS/TCP session should be closed or torn down and both parties should return to the HELLO state.

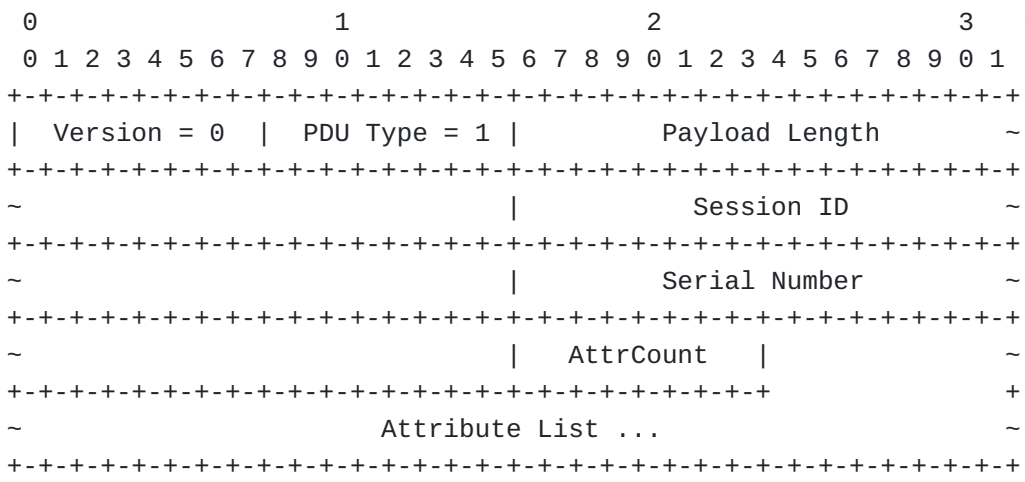
Should the TLS/TCP session terminate for any reason, the devices SHOULD restart/resume HELLOs. When the new TLS/TCP session is started, if possible the OPEN PDU SHOULD try to resume the lost logical session by using the same nonce and resuming from the last Serial Number.

Once the TLS/TCP session has been established, the two devices exchange L3ND PDUs, starting with OPENs.

8. OPEN

Each device has learned the other's IP Address from the HELLO exchange, see Section 6 and established a TLS/TCP session over a particular AFI.

The first PDU each sends MUST be an OPEN, and the other side MUST respond with an ACK PDU.



The four octet Payload Length is the number of octets in all fields of the PDU from the Session ID through the Serial Number.

The four octet Session ID is a nonce which uniquely identifies a session. It enables detection of a duplicate OPEN PDU. It SHOULD be either a random number or a high resolution timestamp. It is needed to prevent session closure due to a repeated OPEN caused by a race or a dropped or delayed ACK. It can be used to resume a dropped logical session.

The one octet AttrCount is the number of attributes in the Attribute List. A node may send zero or more attributes.

Attributes are single octets the semantics of which are operator-defined, 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.

Unlike L3DL [I-D.ietf-lsvr-l3dl], there are no verifiable keys in the PDUs. If the operator wants authentication, integrity, confidentiality, then TLS MUST have been requested by the HELLO and agreed by the TLS session open.

The Serial Number is a monotonically increasing four octet value representing the sender's state at the time of sending the last PDU.

It may be a non-negative integer, a timestamp, etc. If incrementing the Serial Number would cause it to be zero, it should be incremented again.

On session restart (new OPEN, same Session ID), a receiver MAY send the last received Serial Number to tell the sender to only send data with a Serial Number greater (in the [RFC1982] sense), or send a Serial Number of zero to request all data.

This allows a sender of an OPEN to tell the receiver that the sender would like to resume a logical session and that the receiver of the OPEN PDU only needs to send data starting with the PDU with the lowest Serial Number greater (in the [RFC1982] sense) than the one sent in the OPEN. If the sender is not trying to resume a dropped session, the Serial Number MUST be zero.

If the receiver of an OPEN PDU with a non-zero Serial Number can not resume from the requested point, it should return an ACK with an Error Code of 5, Session May Not Be Continued, EType of 1. The sender of the failing OPEN PDU SHOULD respond with an OPEN PDU with a Serial Number of zero.

If a sender of OPEN does not receive an ACK of the OPEN PDU in a configurable time (default 5 seconds), then they SHOULD close or otherwise terminate the TLS/TCP session and restart from the HELLO state.

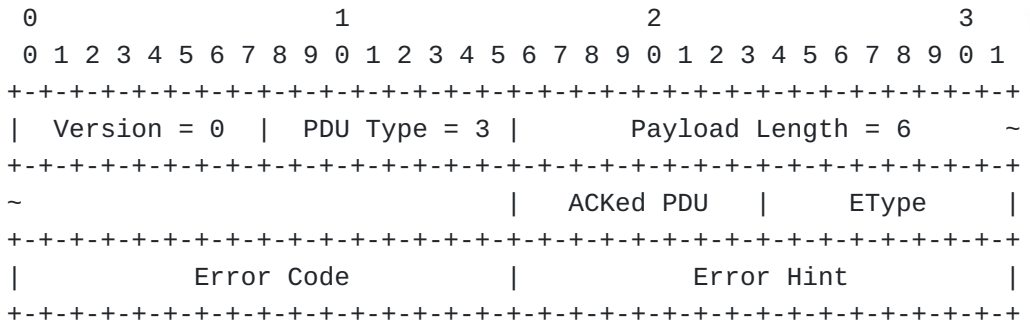
If an OPEN arrives at L3ND speaker A from B with which A believes it already has an L3ND session (i.e. OPENs have already been exchanged), and the Serial Number in B's OPEN PDU is non-zero, speaker A SHOULD establish a new sending session by sending an OPEN with the Serial Number being the same as that of A's last sent and ACKed PDU. A MUST resume sending encapsulations etc. subsequent to the requested Sequence Number. And B MUST retain all previously discovered encapsulation and other data received from A.

If an OPEN arrives at L3ND speaker A from B with which A believes it already has an L3ND session (i.e. OPENs have already been exchanged), and the Serial Number in B's OPEN is zero, then the A MUST assume that B's internal state has been reset. All Previously discovered encapsulation data MUST BE discarded; and A MUST respond with a new OPEN PDU with a Serial Number of zero.

TCP KeepAlives should be configured and tuned to meet local operational needs. Some defaults and recommendations are needed here.

9. ACK

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



The ACK PDU acknowledges receipt of an OPEN, Encapsulation, Vendor PDU, etc. and is used to return error codes if any.

The one octet ACKed PDU is the PDU Type of the PDU being acknowledged, e.g., OPEN, one of the Encapsulations, etc.

If there was an error processing the received PDU, then the one octet 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

The two octet Error Code, noting protocol failures, are listed in Section 16.5. Someone stuck in the 1990s might think the catenation of EType and Error Code as an echo of 0x1zzz, 0x2zzz, etc. They might be right; or not.

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

9.1. 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 five seconds) the TLS/TCP session should be closed or dropped, and both sides revert to HELLO state.

10. The Encapsulations

Once the devices know each other's IP Addresses, and have established a TLS/TCP session and have successfully exchanged OPENs, the L3ND session is considered established, and the devices SHOULD exchange Layer-3 interface encapsulations, Layer-3 addresses, and Layer-2.5 labels.

Encapsulation data for any AFI/SAFI may be exchanged over a TLS/TCP session irrespective of the AFI/SAFI of the session transport.

The Encapsulation types the peers exchange may be IPv4 (Section 10.3), IPv6 (Section 10.4), MPLS IPv4 (Section 10.6), MPLS IPv6 (Section 10.7), and/or possibly others not defined here.

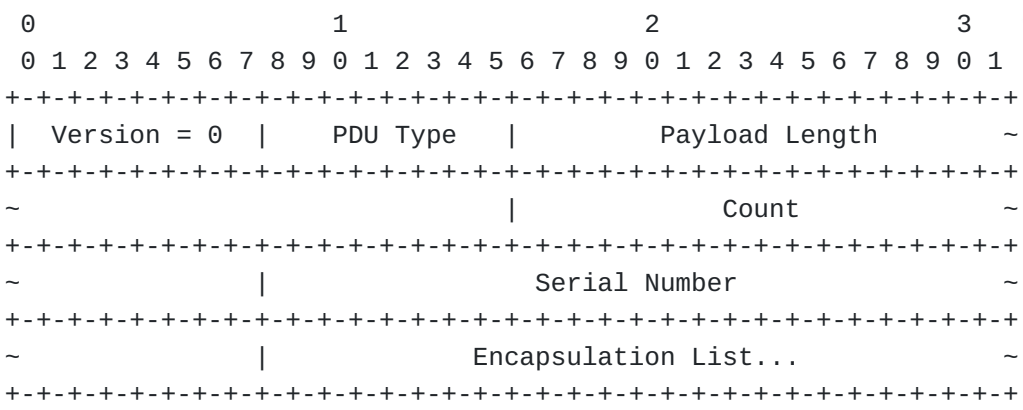
The sender of an Encapsulation PDU MUST NOT assume that the receiver is capable of the same Encapsulation Type. An ACK (Section 9) with EType of 0 merely acknowledges receipt. Only if both peers have sent the same Encapsulation Type is it safe for Layer-3 protocols to assume that they are compatible for that Encapsulation 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 MUST respond with an error (Error Code 2, Logical Link Addressing Conflict) 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 Encapsulation Type to formally be established so that it may be used by other protocols, the addressing for the type must be compatible, e.g. on the same IP subnet.

10.1. The Encapsulation PDU Skeleton

The header for all encapsulation PDUs is as follows:



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

The three octet Count is the number of Encapsulations in the Encapsulation List.

The Serial Number is a monotonically increasing four octet value representing the sender's state in time. It may be an integer, a timestamp, etc. On session restart (new OPEN), a receiver MAY send the last received Serial Number to request the sender to only send newer data.

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

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

The Receiver MUST acknowledge the Encapsulation PDU with an ACK PDU (Section 9) with the Type field being that of the Type of the Encapsulation PDU being announced, see Section 9.

If the Sender does not receive an ACK in a configurable interval (default five seconds), they SHOULD retransmit. After a user configurable number of failures (default three), the L3ND session should be considered dead, TLS/TCP torn down, and the HELLO process SHOULD be restarted.

If the link is broken below layer-3, retransmission MAY BE retried if data have not changed in the interim and the TLS/TCP session is still alive.

Should an Encapsulation in the Encapsulation List be syntactically invalid, e.g. an out of bounds prefix length, the entire Encapsulation PDU MUST be dropped and the sending party notified by an ACK PDU with an EType of 1 and an Error Code of 3, Encapsulation Error.

10.2. Encapsulaion Flags

The one octet Encapsulation Flags field is a sequence of one bit fields as follows:

Each encapsulation in an Encapsulation PDU of Type T may announce new and/or withdraw old encapsulations of Type T. It indicates this with the Ann/With Encapsulation Flag, Announce == 1, Withdraw == 0.

Announcing an encapsulation which already exists SHOULD raise an Announce/Withdraw Error (see Section 16.5); the EType SHOULD be 2, suggesting a session restart (see Section 9) so all encapsulations will be resent.

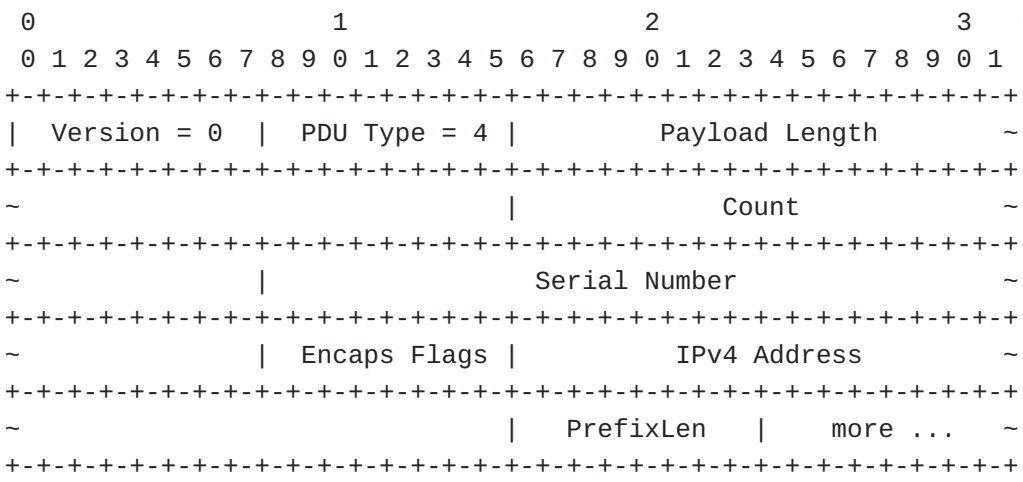
If an interface on a link has multiple addresses for an encapsulation type, one and only one address MAY be marked as primary (Primary Flag == 1) for that Encapsulation Type.

An Encapsulation interface address in an Encapsulation PDU MAY be marked as a loopback, in which case the Loopback bit is set. 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 L3ND speaking external interfaces, e.g. for iBGP peering. They SHOULD be marked as such, Loopback Flag == 1.

Each Encapsulation interface address in an Encapsulation PDU is that of the direct 'underlay interface (Under/Over == 1), or an 'overlay' address (Under/Over == 0), likely that of a VM or container guest bridged or configured on to the interface already having an underlay address.

10.3. IPv4 Encapsulation

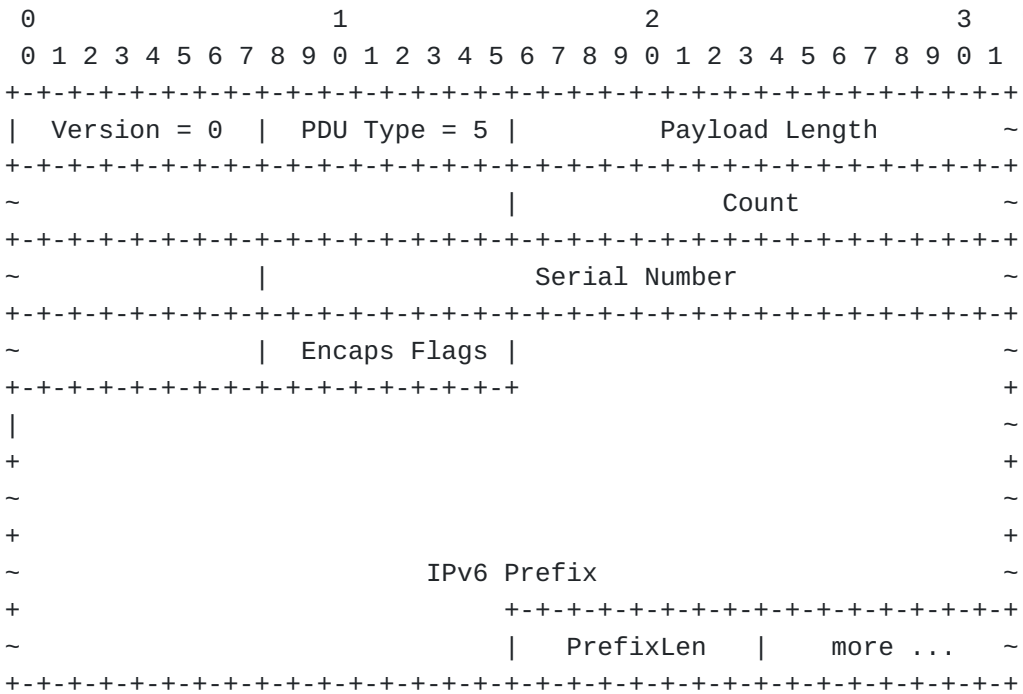
The IPv4 Encapsulation describes a device's ability to exchange IPv4 packets on one or more subnets. It does so by stating the interface's addresses and the corresponding prefix lengths.



The three octet Count is the sum of the number of IPv4 Encapsulations being announced and/or withdrawn.

10.4. IPv6 Encapsulation

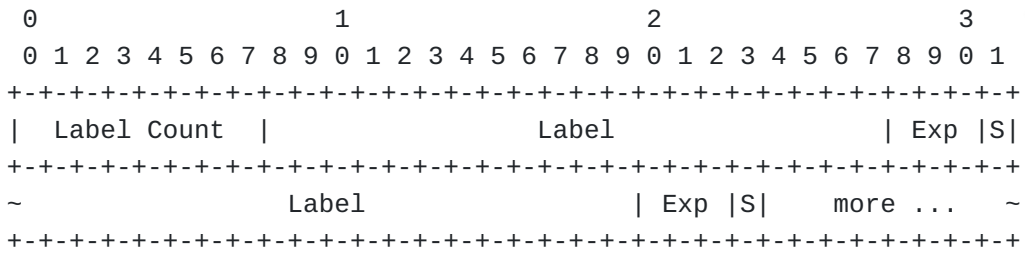
The IPv6 Encapsulation describes a link's ability to exchange IPv6 packets on one or more subnets. It does so by stating the interface's addresses and the corresponding prefix lengths.



The three octet Count is the sum of the number of IPv6 Encapsulations being announced and/or withdrawn.

10.5. MPLS Label List

As an MPLS enabled interface may have a label stack, see [RFC3032], a variable length list of labels is needed. These are the labels the sender will accept for the prefix to which the list is attached.

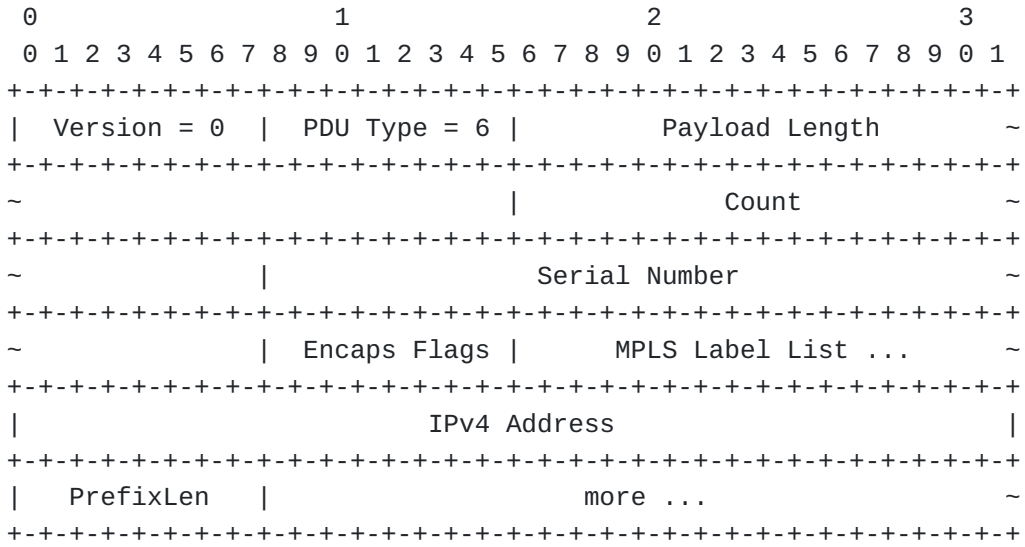


A one octet Label Count of zero is an implicit withdraw of all labels for that prefix on that interface.

The bottom of the stack flag, S, MUST be set on one and only one label. Should this not be the case, the receiver of the erroneous PDU MUST respond with an ACK PDU of EType 1 and Error Code 1, MPLS Error.

10.6. MPLS IPv4 Encapsulation

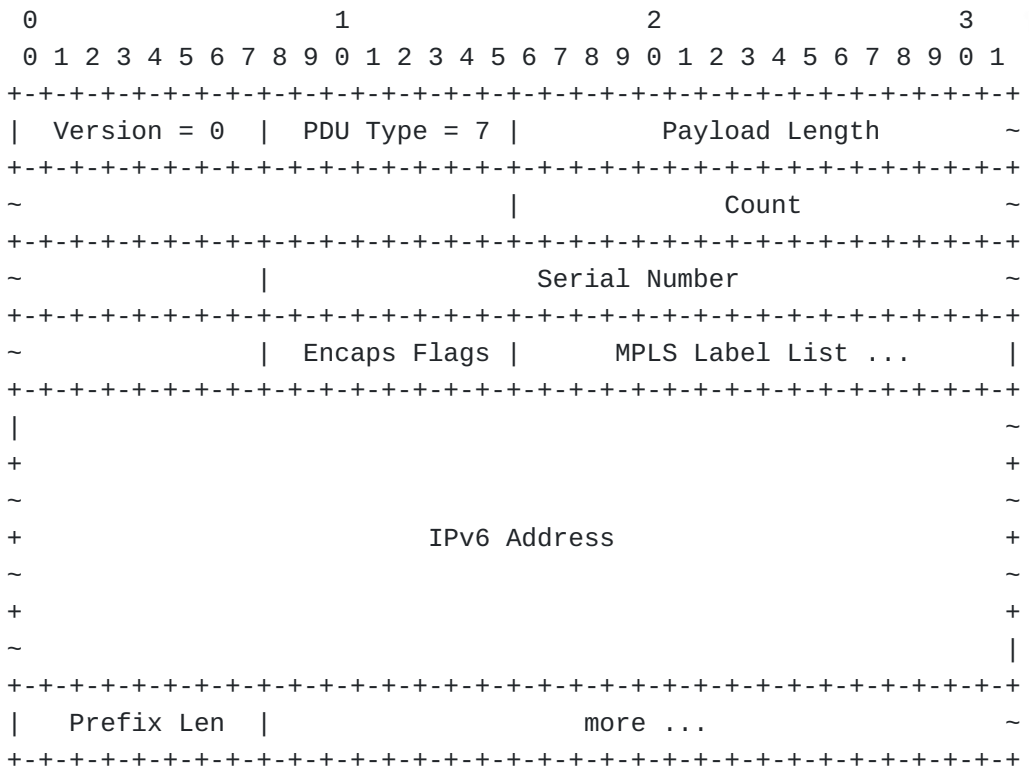
The MPLS IPv4 Encapsulation describes a logical link's ability to exchange labeled IPv4 packets on one or more subnets. It does so by stating the interface's addresses the corresponding prefix lengths, and the corresponding labels which will be accepted for each address.



The three octet Count is the sum of the number of MPLSv4 Encapsulation being announced and/or withdrawn.

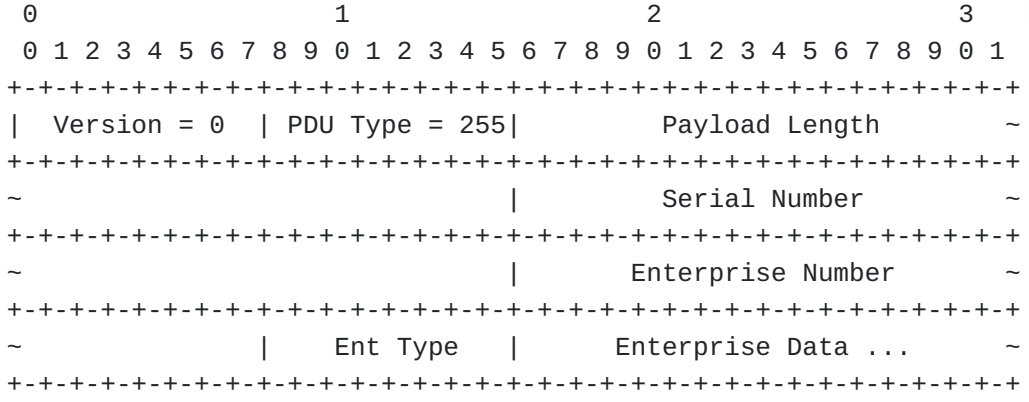
10.7. MPLS IPv6 Encapsulation

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 which will be accepted for each address.



The three octet Count is the sum of the number of MPLSv6 Encapsulations being announced and/or withdrawn.

11. VENDOR - Vendor Extensions



Vendors or enterprises may define TLVs beyond the scope of L3ND standards. This is done using a Private Enterprise Number [IANA-PEN]

followed by Enterprise Data in a format defined for that three octet Enterprise Number and one octet 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 PDU, possibly signalling an error. Similarly, a Vendor PDU MUST only be sent over an open session.

12. Discussion

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

12.1. HELLO Discussion

A device may send IP packets over a Layer-3 interface which transmits data over a single Layer-2 interface or multiple Layer-2 interfaces. Packets sourced by one Layer-3 IP interface over multiple Layer-2 should consider that a Layer-3 interface with multiple Layer-2 interfaces could have many devices which might come at various times, therefore the configured HELLO PDU retransmit time SHOULD be set to a non-zero value, and periodic HELLOs should continue. Packets transmitted on a single Layer-2 interface on a point-to-point (p2p) connection, MAY set the configuration value to zero, so when a TLS/TCP session is up, HELLOs are no longer desirable.

A device with multiple Layer-2 interfaces, traditionally called a switch, may be used to forward packets from multiple devices to one Layer-3 interface, I, on an L3ND 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, on multi-link interfaces, L3ND MUST continue to send HELLOs as long as they are turned up.

13. 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, L3ND treats VLAN ports as separate links.

As Sub-Interfaces each have their own layer-3 identities, they act as separate interfaces, forming their own links.

14. Implementation Considerations

An implementation SHOULD provide the ability to configure each logical interface as L3ND speaking or not.

An implementation SHOULD provide the ability to distribute one or more loopback addresses or interfaces into L3ND on an external L3ND speaking interface.

An implementation SHOULD provide the ability to distribute one or more overlay and/or underlay addresses or interfaces into L3ND on an external L3ND speaking interface.

An implementation SHOULD provide the ability to configure one of the addresses of an encapsulation as primary on an L3ND speaking interface. If there is only one address for a particular encapsulation, the implementation MAY mark it as primary by default.

An implementation MAY allow optional configuration which updates the local forwarding table with overlay and underlay data both learned from L3ND peers and configured locally.

15. Security Considerations

For TLS, versions greater than 1.1 MUST be used.

The protocol as is MUST NOT be used outside a datacenter or similarly closed environment without using TLS encapsulation which is based on a configured CA trust anchor.

Many datacenter operators have a strange belief that physical walls and firewalls provide sufficient security. This is not credible. All DC protocols need to be examined for exposure and attack surface. In the case of L3ND, authentication and integrity as provided by TLS validated to a configured shared CA trust anchor is strongly RECOMMENDED.

It is generally unwise to assume that on the wire Layer-3 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 OPEN PDUs are not over validated TLS, an attacker could forge an OPEN for an existing session and cause the session to be reset.

16. IANA Considerations

16.1. Link Local Layer-3 Addresses

IANA is requested to assignment one address (TBD1) for L3DL-L3-LL from the IPv4 Multicast Address Space Registry from the Local Network Control Block (224.0.0.0 - 224.0.0.255 (224.0.0/24)).

IANA is requested to assign one address (TBD2) for L3DL-L3-LL from the IPv6 Multicast Address Space Registry in the the IPv6 Link-Local Scope Multicast address (TBD:2).

16.2. Ports for TLS/TCP

This document requests the IANA to assign a well-known TCP Port Number (TBD3) to the Layer-3 Neighbor Discovery Protocol for the following, see Section 7:

l3nd-server

16.3. PDU Types

This document requests the IANA create a registry for L3ND PDU Type, which may range from 0 to 255. The name of the registry should be L3ND-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	reserved
2	OPEN
3	ACK
4	IPv4 Announcement
5	IPv6 Announcement
6	MPLS IPv4 Announcement
7	MPLS IPv6 Announcement
8-254	Reserved
255	Vendor

16.4. Flag Bits

This document requests the IANA create a registry for L3ND PL Flag Bits, which may range from 0 to 7. The name of the registry should be L3ND-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	Announce/Withdraw (ann == 0)
1	Primary
2	Underlay/Overlay (under == 0)
3	Loopback
4-7	Reserved

16.5. Error Codes

This document requests the IANA create a registry for L3ND Error Codes, a 16 bit integer. The name of the registry should be L3ND-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	No Error
1	MPLS Error
2	Logical Link Addressing Conflict
3	Encapsulation Error
4	Announce/Withdraw Error
5	Session May Not Be Continued

17. Acknowledgments

The authors thank Ben Maddison and Jeff Haas.

18. References

18.1. Normative References

[I-D.ietf-lsvr-l3dl] Bush, R., Austein, R., and K. Patel, "Layer-3 Discovery and Liveness", Work in Progress, Internet-Draft, draft-ietf-lsvr-l3dl-09, 2 May 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-lsvr-l3dl-09>.

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

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://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, <https://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, <https://www.rfc-editor.org/info/rfc4271>.

[RFC5082] Gill, V., Heasley, J., Meyer, D., Savola, P., Ed., and C. Pignataro, "The Generalized TTL Security Mechanism (GTSM)", RFC 5082, DOI 10.17487/RFC5082, October 2007, <https://www.rfc-editor.org/info/rfc5082>.

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

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/info/rfc5280>.

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

[RFC8635] Bush, R., Turner, S., and K. Patel, "Router Keying for BGPsec", RFC 8635, DOI 10.17487/RFC8635, August 2019, <https://www.rfc-editor.org/info/rfc8635>.

18.2. Informative References

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

[I-D.ymbk-idr-l3nd-ulpc] Bush, R. and K. Patel, "L3ND Upper-Layer Protocol Configuration", Work in Progress, Internet-Draft, draft-ymbk-idr-l3nd-ulpc-06, 1 October 2022, <https://datatracker.ietf.org/doc/html/draft-ymbk-idr-l3nd-ulpc-06>.

[RFC1122] Braden, R., Ed., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, DOI 10.17487/

RFC1122, October 1989, <https://www.rfc-editor.org/info/rfc1122>.

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

[RFC4760] Bates, T., Chandra, R., Katz, D., and Y. Rekhter, "Multiprotocol Extensions for BGP-4", RFC 4760, DOI 10.17487/RFC4760, January 2007, <https://www.rfc-editor.org/info/rfc4760>.

Authors' Addresses

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

Email: randy@psg.com

Russ Housley
Vigil Security, LLC
516 Dranesville Road
Herndon, VA 20170
United States of America

Email: housley@vigilsec.com

Rob Austein
Arrcus, Inc

Email: sra@hactrn.net

Susan Hares
Hickory Hill Consulting
7453 Hickory Hill
Saline, MI 48176
United States of America

Phone: +1-734-604-0332
Email: shares@ndzh.com

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

Email: keyur@arrcus.com