Mobile Ad hoc Networks Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 10, 2012

S. Ratliff
B. Berry
G. Harrison
D. Satterwhite
Cisco Systems
S. Jury
NetApp

February 6, 2012

Dynamic Link Exchange Protocol (DLEP)
draft-ietf-manet-dlep-02

Abstract

When routing devices rely on modems to effect communications over wireless links, they need timely and accurate knowledge of the characteristics of the link (speed, state, etc.) in order to make forwarding decisions. In mobile or other environments where these characteristics change frequently, manual configurations or the inference of state through routing or transport protocols does not allow the router to make the best decisions. A bidirectional, event-driven communication channel between the router and the modem is necessary.

Table of Contents

1. Introduction

There exist today a collection of modem devices that control links of variable bandwidth and quality. Examples of these types of links include line-of-sight (LOS) radios, satellite terminals, and cable/DSL modems. Fluctuations in speed and quality of these links can occur due to configuration (in the case of cable/DSL modems), or on a moment-to-moment basis, due to physical phenomena like multipath interference, obstructions, rain fade, etc. It is also quite possible that link quality and bandwidth varies with respect to individual neighbors on a link, and with the type of traffic being sent. As an example, consider the case of an 802.11g access point, serving 2 associated laptop computers. In this environment, the answer to the question "What is the bandwidth on the 802.11g link?" is "It depends on which associated laptop we're talking about, and on what kind of traffic is being sent." While the first laptop, being physically close to the access point, may have a bandwidth of 54Mbps for unicast traffic, the other laptop, being relatively far away, or obstructed by some object, can simultaneously have a bandwidth of only 32Mbps for unicast. However, for multicast traffic sent from the access point, all traffic is sent at the base transmission rate (which is configurable, but depending on the model of the access point, is usually 24Mbps or less).

In addition to utilizing variable bandwidth links, mobile networks are challenged by the notion that link connectivity will come and go over time. Effectively utilizing a relatively short-lived connection is problematic in IP routed networks, as routing protocols tend to rely on independent timers at OSI Layer 3 to maintain network convergence (e.g. HELLO messages and/or recognition of DEAD routing adjacencies). These short-lived connections can be better utilized with an event-driven paradigm, where acquisition of a new neighbor (or loss of an existing one) is somehow signaled, as opposed to a timer-driven paradigm.

Another complicating factor for mobile networks are the different methods of physically connecting the modem devices to the router. Modems can be deployed as an interface card in a router's chassis, or as a standalone device connected to the router via Ethernet, USB, or even a serial link. In the case of Ethernet or

serial attachment, with existing protocols and techniques, routing software cannot be aware of convergence events occurring on the radio link (e.g. acquisition or loss of a potential routing neighbor), nor can the router be aware of the actual capacity of the link. This lack of awareness, along with the variability in bandwidth, leads to a situation where quality of service (QoS) profiles are extremely difficult to establish and properly maintain. This is especially true of demand-based access schemes such as Demand Assigned Multiple Access (DAMA) implementations used on some satellite systems. With a DAMA-based system, additional bandwidth may be available, but will not be used unless the network devices emit traffic at rate higher than the currently established rate. Increasing the traffic rate does not guarantee additional bandwidth will be allocated; rather, it may result in data loss and additional retransmissions on the link.

In attempting to address the challenges listed above, the authors have developed the Data Link Exchange Protocol, or DLEP. The DLEP protocol runs between a router and its attached modem devices, allowing the modem to communicate link characteristics as they change, and convergence events (acquisition and loss of potential routing neighbors). The following diagrams are used to illustrate the scope of DLEP sessions.

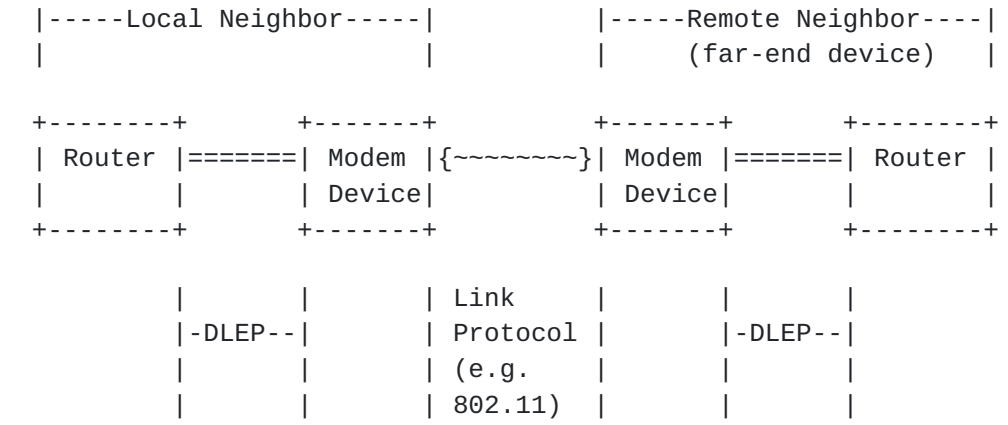


Figure 1: DLEP Network

In Figure 1, when a local client (Modem device) detects the presence of a remote neighbor, it sends an indication to its local router via the DLEP session. Upon receipt of the indication, the local router would take appropriate action (e.g. initiation of discovery or HELLO protocols) to converge the network. After notification of the new neighbor, the modem device utilizes the DLEP session to report the characteristics of the link (bandwidth, latency, etc) to the router on an as-needed basis.

DLEP is independent of the underlying link type and topology. Figure 2 shows how DLEP can support a configuration whereby routers are connected with different link types and with different network configurations. In this setup, the routers are connected

with two different devices (Modem device A and Modem device B). Modem A is connected via a point-to-point link, whereas Modem B is connected via a shared medium. In both cases, the DLEP session is used to report the characteristics of the link (bandwidth, latency, etc.) to network neighbors on an as-needed basis. The modem is also able to use the DLEP session to notify the router when the remote neighbor is lost, shortening the time required to re-converge the network.

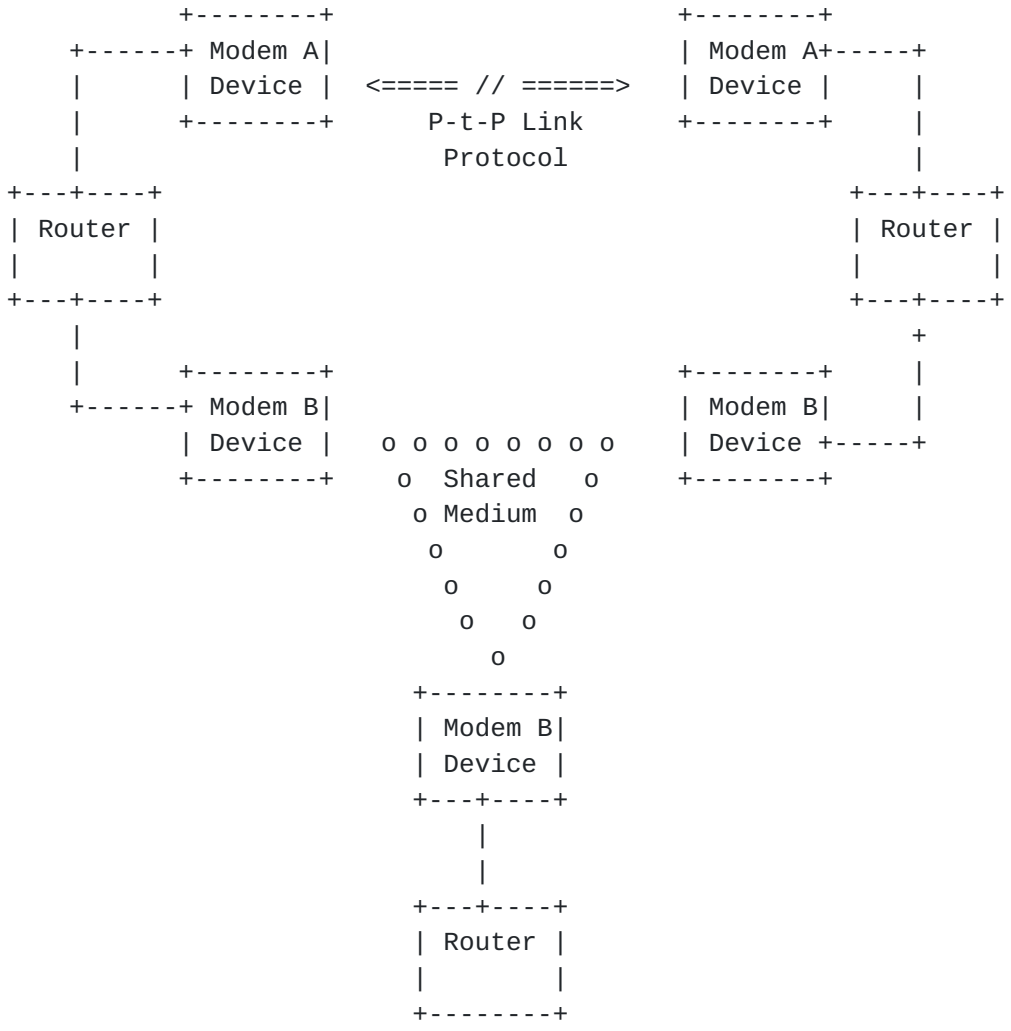


Figure 2: DLEP Network with Multiple Modem Devices

DLEP exists as a collection of type-length-value (TLV) based messages using [RFC5444] formatting. The protocol can be used for both Ethernet attached modems (utilizing, for example, a UDP socket for transport of the RFC 5444 packets), or in environments where the modem is an interface card in a chassis (via a message passing scheme). DLEP utilizes a session paradigm between the modem device and its associated router. If multiple modem devices are attached to a router (as in FIgure 2), a separate DLEP session MUST exist for each

modem. If a modem device supports multiple connections to a router (via multiple logical or physical interfaces), or supports connections to multiple routers, a separate DLEP session MUST exist for each connection.

1.1 Requirements

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14, RFC 2119 [RFC2119].

2. Assumptions

In order to implement discovery in the DLEP protocol (thereby avoiding some configuration), we have defined a first-speaker and a passive-listener scheme. Borrowing from existing terminology, this document refers to the first-speaker as the 'client', and the passive listener as the 'server', even though there is no client/server relationship in the classic sense. In a typical deployment, a router would appear as the DLEP 'server', and an attached modem device would act as the 'client' (e.g. the initiator for discovery).

DLEP assumes that participating clients appear to the server as a transparent bridge - specifically, the assumption is that the destination MAC address for data traffic in any frame emitted by the server should be the MAC address of the next-hop router or end-device, and not the MAC address of any of the intervening clients.

DLEP assumes that security on the session (e.g. authentication of session partners, encryption of traffic, or both) is dealt with by the underlying transport mechanism for the RFC 5444 packets (e.g. by using a transport such as DTLS [DTLS]).

DLEP utilizes a session-oriented paradigm. There are two classes of sessions - the first is identified as a 'peer session'. The peer session exists between a DLEP server and a DLEP client. All DLEP messages between client and server are transmitted within the context of the peer session.

The other type of DLEP session is referred to as a 'neighbor session'. Neighbor sessions can be instantiated by either the DLEP server or client, and represent an identifiable destination (i.e. an address) within the network. Examples of a destination would be a unicast address (for either a next-hop router, or for an end-station), or a multicast address. A DLEP neighbor session MUST exist for every destination that exists in the network.

The optional [RFC5444] message header Sequence Number MUST be included in all DLEP packets. Sequence Numbers start at 1 and are incremented by one for each original and retransmitted message. The unsigned 16-bit Sequence Number rolls over at 65535 to 1. A Sequence Number of 0 is not valid. Sequence Numbers are unique within the context of a DLEP session. Sequence numbers are used in

DLEP to correlate a response to a request.

3. Credits

DLEP includes an OPTIONAL credit-windowing scheme analogous to the one documented in [RFC5578]. In this scheme, traffic between the DLEP client and the DLEP server is treated as two unidirectional windows. This document identifies these windows as the "Client Receive Window", or CRW, and the "Server Receive Window", or SRW.

If credits are used, they MUST be granted by the receiver on a given window - that is, on the "Client Receive Window" (CRW), the DLEP client is responsible for granting credits to the server, allowing it (the server) to send data to the client. Likewise, the DLEP server is responsible for granting credits on the SRW, which allows the client to send data to the server.

DLEP expresses all credit data in number of octets. The total number of credits on a window, and the increment to add to a grant, are always expressed as a 64-bit unsigned quantity.

If used, credits are managed on a neighbor session basis; that is, separate credit counts are maintained for each neighbor session requiring the service. Credits do not apply to DLEP peer sessions.

4. Metrics

DLEP includes the ability for the client and server to communicate metrics that reflect the characteristics (e.g. bandwidth, latency) of the variable-quality link in use. As mentioned in the introduction section of this document, metrics have to be used within a context - for example, metrics to a unicast address in the network. DLEP allows for metrics to be sent within two contexts - neighbor session context (those for a given destination within the network), and peer session context (those that apply to all destinations accessed via the DLEP client). Metrics supplied on DLEP Peer messages are, by definition, in the context of a peer session; metrics supplied on Neighbor messages are, by definition, used in the context of a neighbor session.

Supplying metrics in a peer session context gives clients the ability to supply default metrics on a 'device-wide' basis. It is left to implementations to choose sensible default values based on their specific characteristics. Additionally, the metrics (either at a peer or neighbor session context) MAY be used to report non-changing, or static, metrics. Clients having static link metric characteristics SHOULD report metrics only once for a given neighbor session (or peer session, if all connections via the client are of this static nature).

The approach of allowing for different contexts for metric data

increases both the flexibility and the complexity of using metric data. This document details the mechanism whereby the data is transmitted, however, the specific algorithms for utilizing the dual-context metrics is out of scope and not addressed by this document.

5. Extensions to DLEP

While this draft represents the best efforts of the co-authors, and the working group, to be functionally complete, it is recognized that extensions to DLEP will in all likelihood be necessary as more link types are utilized. To allow for future innovation, the draft allocates numbering space for experimental orders and sub-TLVs. DLEP implementations MUST be capable of parsing and acting on the orders and sub-TLVs as documented in this specification. DLEP orders/sub-TLVs in the experimental numbering range SHOULD be silently dropped by an implementation if they are not understood. The intent of the experimental numbering space is to allow for further development of DLEP protocol features and function. If subsequent development yields new features with sufficient applicability, those features should be either included in an update of this specification, or documented in a standalone specification.

6. Normal Session Flow

A session between a client and a server is established by exchanging the "Peer Discovery" and "Peer Offer" messages described below.

The flows described in this document create a state-full protocol between client and server. Both client and server initialize in a "discovery" state, and the client issues a "Peer Discovery" message. When the server receives a Peer Discovery, it responds with a "Peer Offer" message, and enters an "in session" state with the client. Receipt of the Peer Offer at the client causes it (the client) to transition into the "in session" state.

Once that exchange has successfully occurred, messages transferred in the context of the peer session will consist of

- Periodic 'Heartbeat' messages, intended to keep the peer session alive, and to verify bidirectional connectivity, and/or
- Peer Update messages, indicating some change in status that one of the peers needs to communicate to the other.

In addition to the messages above, the peers will transmit DLEP messages concerning destinations in the network. These messages trigger creation/maintenance/termination of 'neighbor sessions'. For example, a peer will inform its DLEP partner of the presence of a new destination via the "Neighbor Up" message. Receipt of a Neighbor Up causes the receiving peer to allocate the necessary resources, creating a neighbor session, and transition to an "in session" state on the newly created neighbor session. The in-session state persists until notification of neighbor loss is received, or by optional timeout due to inactivity.

The loss of a destination is communicated via the "Neighbor Down"

message, and changes in status to the destination (e.g. varying link quality, or addressing changes) are communicated via a "Neighbor Update" message.

Again, metrics can be expressed within the context of a neighbor session via the Neighbor Update message, or within the context of

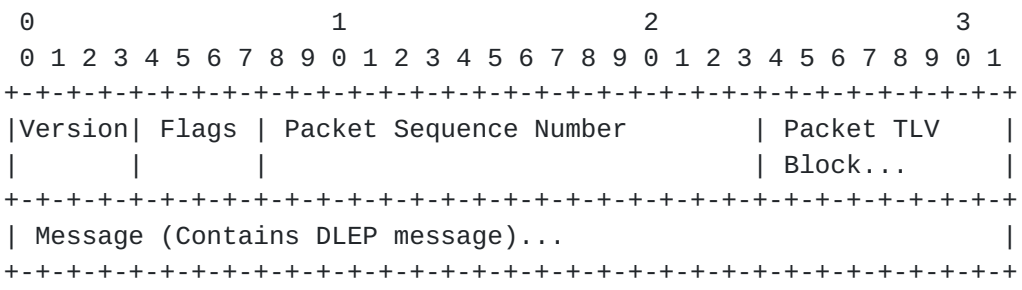
a peer session (reflecting the link as a whole) via the Peer Update message. In cases where metrics are provided on the peer session, the receiving peer MUST propagate the metrics to all neighbor sessions accessed via the peer. A DLEP peer MAY send metrics both in a peer session context (via the Peer Update message) and a neighbor session context (via Neighbor Update) at any time. The heuristics for applying received peer session and neighbor session metrics is left to implementations.

In addition to receiving metrics about the link, DLEP provides for the ability for a server to request a different amount of bandwidth, or latency, from the client via the Link Characteristics Message. This allows the server to deal with requisite increases (or decreases) of allocated bandwidth/latency in demand-based schemes in a more deterministic manner.

7. Generic DLEP Packet Definition

The Generic DLEP Packet Definition follows the format for packets defined in [RFC5444].

The Generic DLEP Packet Definition contains the following fields:



Version - Version of RFC 5444 specification on which the packet/messages/TLVs are constructed.

Flags - 4 bit field. All bits MUST be ignored by DLEP implementations.

Packet Sequence Number - If present, the packet sequence number is parsed and ignored. DLEP does NOT use or generate packet sequence numbers.

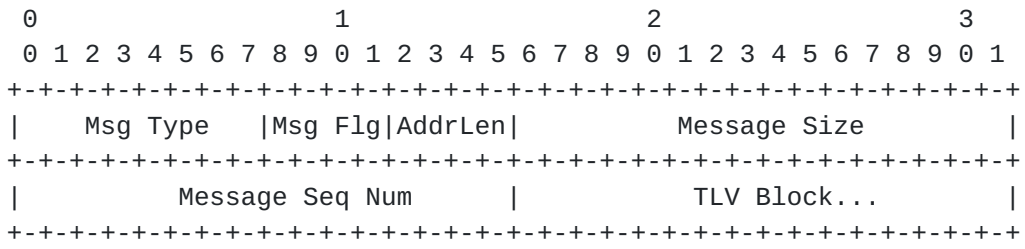
Packet TLV block - A TLV block which contains packet level TLV information. DLEP implementations MUST NOT use this TLV block.

Message - The packet MAY contain zero or more messages, however, DLEP messages are

encoded within an RFC 5444 Message
TLV Block.

8. Message Header Format

DLEP utilizes the following format for the RFC 5444 message header



Message Type - An 8-bit field which specifies the type of the message. For DLEP, this field contains DLEP_MESSAGE (value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to '0'.

Message Address Length - A 4-bit unsigned integer field encoding the length of all addresses included in this message. DLEP implementations do not use this field; contents SHOULD be ignored.

Message Size - A 16-bit unsigned integer field which specifies the number of octets that make up the message including the message header.

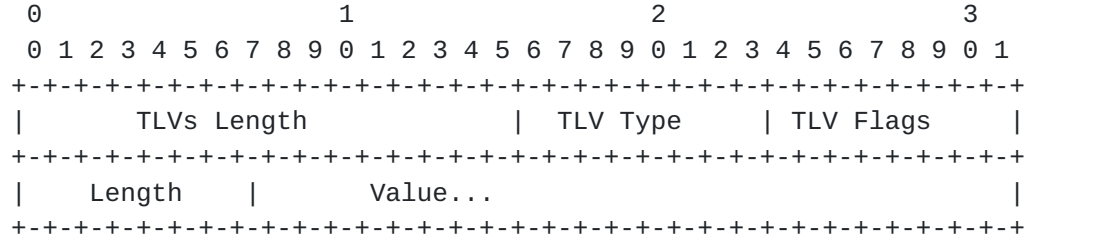
Message Sequence Number - A 16-bit unsigned integer field that contains a sequence number, generated by the originator of the message. Sequence numbers range from 1 to 65535. Sequence numbers roll over at 65535 to 1; 0 is invalid.

TLV Block - TLV Block included in the message.

9. Message TLV Block Format

The DLEP protocol is organized as a set of orders, each with a collection of Sub-TLVs. The Sub-TLVs carry information needed to process and/or establish context (e.g. the MAC address of a far-end router), and the 'tlv-type' field in the message TLV block carries the DLEP order itself. The DLEP orders are enumerated in section 11.1 of this document, and the messages created using these orders are documented in sections 12 through 27.

DLEP uses the following settings for an RFC 5444 Message TLV block:



TLVs Length - A 16-bit unsigned integer field that contains the total number of octets in all of the immediately following TLV elements (tlvs-length not included).

TLV Type - An 8-bit unsigned integer field specifying the type of the TLV. DLEP uses this field to specify the DLEP order. Valid DLEP orders are defined in section 11.1 of this document.

TLV Flags - An 8-bit flags bit field. Bit 3 (thasvalue) MUST be set; all other bits are not used and MUST be set to '0'.

Length - Length of the 'Value' field of the TLV

Value - A field of length <Length> which contains data specific to a particular TLV type. In the DLEP case, this field will consist of a collection of DLEP sub-TLVs appropriate for the DLEP action specified in the TLV type field.

10. DLEP sub-TLVs

DLEP protocol messages are transported in an RFC 5444 message TLV. All DLEP messages use the RFC 5444 DLEP_MESSAGE value (TBD). The protocol messages consist of a DLEP order, encoded in the 'tlv-type' field in the message TLV block, with the 'value' field of the TLV block containing a collection (1 or more) DLEP sub-TLVs.

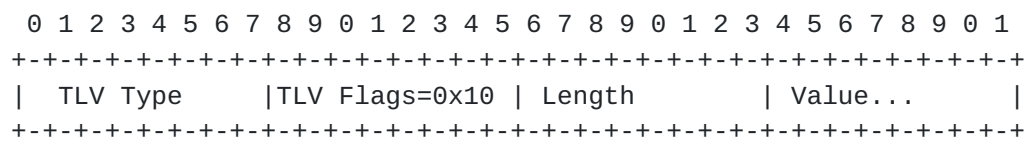
The format of DLEP Sub-TLVs is consistent with RFC 5444 in that the Sub-TLVs contain a flag field in addition to the type, length, and value fields. Valid DLEP Sub-TLVs are:

TLV Value	TLV Description
TBD	Identification sub-TLV

TBD	DLEP Version sub-TLV
TBD	Peer Type sub-TLV
TBD	MAC Address sub-TLV
TBD	IPv4 Address sub-TLV

TBD	IPv6 Address sub-TLV
TBD	Maximum Data Rate (MDR) sub-TLV
TBD	Current Data Rate (CDR) sub-TLV
TBD	Latency sub-TLV
TBD	Resources sub-TLV
TBD	Expected Forwarding Time (ETX) sub-TLV
TBD	Relative Link Quality (RLQ) sub-TLV
TBD	Status sub-TLV
TBD	Heartbeat Interval sub-TLV
TBD	Heartbeat Threshold sub-TLV
TBD	Neighbor down ACK timer sub-TLV
TBD	Link Characteristics ACK timer sub-TLV
TBD	Credit Window Status sub-TLV
TBD	Credit Grant sub-TLV
TBD	Credit Request sub-TLV

DLEP sub-TLVs contain the following fields:



TLV Type - An 8-bit unsigned integer field specifying the type of the sub-TLV.

TLV Flags - An 8-bit flags bit field. Bit 3 (thasvalue) MUST be set, all other bits are not used and MUST be set to '0'.

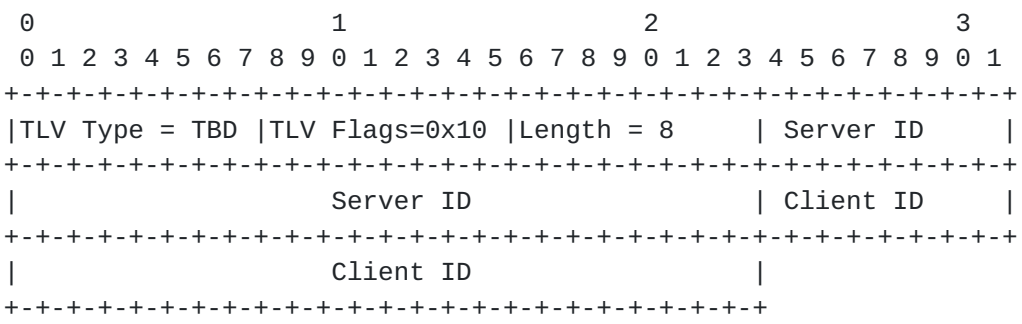
Length - An 8-bit length of the value field of the sub-TLV

Value - A field of length <Length> which contains data specific to a particular sub-TLV.

10.1 Identification Sub-TLV

This Sub-TLV MUST exist in the TLV Block for all DLEP messages, and MUST be the first Sub-TLV of the message. Further, there MUST be ONLY one Identification Sub-TLV in an RFC 5444 message TLV block. The Identification sub-TLV contains client and server identification information used to establish the proper context for processing DLEP protocol messages.

The Identification sub-TLV contains the following fields:



TLV Type - Value TBD

TLV Flags - 0x10, Bit 3 (thasvalue) is set, all other bits are unused and MUST be set to '0'.

Length - 8

Server ID - Indicates the Server ID of the DLEP session.

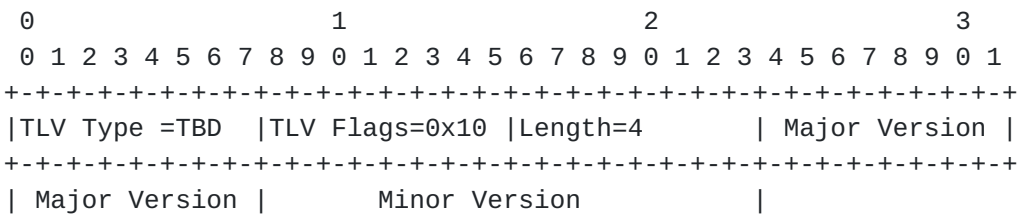
Client ID - indicates the Client ID of the DLEP session.

When the client initiates discovery (via the Peer Discovery message), it MUST set the Client ID to a 32-bit quantity that will be used to uniquely identify this session from the client-side. The client MUST set the Server ID to '0'. When responding to the Peer Discovery message, the server MUST echo the Client ID, and MUST supply its own unique 32-bit quantity to identify the session from the server's perspective. After the Peer Discovery/Peer Offer exchange, both the Client ID and the Server ID MUST be set to the values obtained from the Peer DIscovery/Peer Offer sequence.

10.2 DLEP Version Sub-TLV

The DLEP Version Sub-TLV is an OPTIONAL TLV in both the Peer Discovery and Peer Offer messages. The Version Sub-TLV is used to indicate the client or server version of the protocol. The client and server MAY use this information to decide if the peer is running at a supported level.

The DLEP Version Sub-TLV contains the following fields:



TLV Type - TBD

TLV Flags - 0x10, Bit 3 (thasvalue) is set, all other bits are not used and MUST be set to '0'.

Length - Length is 4

Major Version - Major version of the client or router protocol.

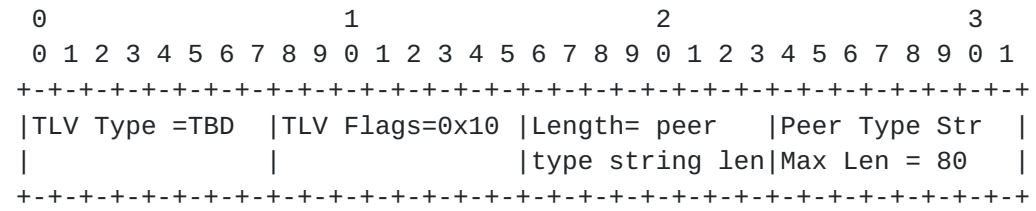
Minor Version - Minor version of the client or router protocol.

Support of this draft is indicated by setting the Major Version to '1', and the Minor Version to '2' (e.g. Version 1.2).

10.3 Peer Type Sub-TLV

The Peer Type Sub-TLV is used by the server and client to give additional information as to its type. It is an OPTIONAL sub-TLV in both the Peer Discovery Message and the Peer Offer message. The peer type is a string and is envisioned to be used for informational purposes (e.g. as output in a display command).

The Peer Type sub-TLV contains the following fields:



TLV Type - TBD

TLV Flags - 0x10, Bit 3 (thasvalue) is set, all other bits are not used and MUST be set to '0'.

Length - Length of peer type string (80 bytes maximum).

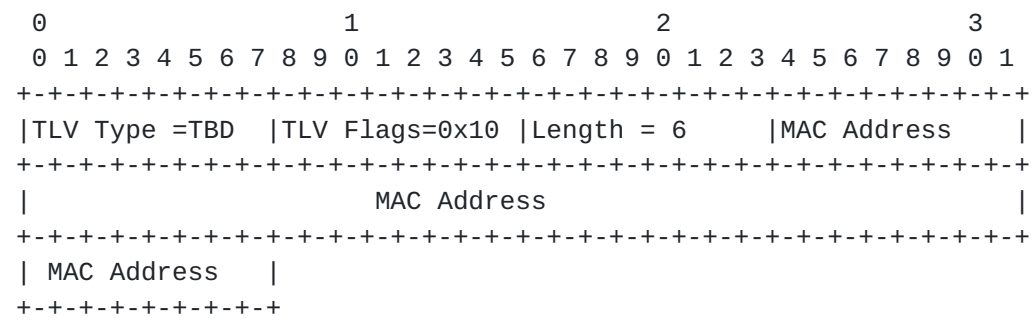
Peer Type String - Non-Null terminated peer type string, maximum length of 80 bytes. For example, a satellite modem might set this variable to 'Satellite terminal'.

10.4 MAC Address Sub-TLV

The MAC address Sub-TLV MUST appear in all neighbor-oriented messages (e.g. Neighbor Up, Neighbor Up ACK, Neighbor Down, Neighbor Down ACK, Neighbor Update, Link Characteristics Request, and Link Characteristics ACK). The MAC Address sub-TLV contains the address of the far-end (neighbor) destination, and may be either a physical

or a virtual destination. Examples of a virtual destination would be a multicast MAC address, or the broadcast MAC (0xFFFFFFFFFFFF).

The MAC Address sub-TLV contains the following fields:



TLV Type - TBD

TLV Flags - 0x10, Bit 3 (thasvalue) is set, all other bits are not used and MUST be set to '0'.

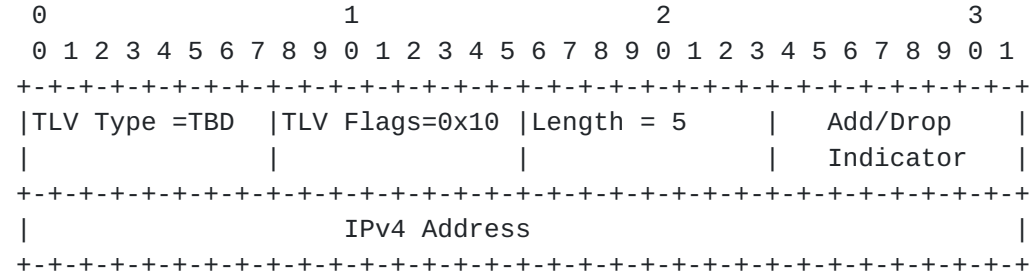
Length - 6

MAC Address - MAC Address of the destination (either physical or virtual).

10.5 IPv4 Address Sub-TLV

The IPv4 Address Sub-TLV MAY be used in Neighbor Up, Neighbor Update, and Peer Update Messages, if the client is aware of the Layer 3 address. When included in Neighbor messages, the IPv4 Address sub-TLV contains the IPv4 address of the far-end neighbor. In the Peer Update message, it contains the IPv4 address of the sending peer. In either case, the sub-TLV also contains an indication of whether this is a new or existing address, or is a deletion of a previously known address.

The IPv4 Address Sub-TLV contains the following fields:



TLV Type - TBD

TLV Flags - 0x10, Bit 3 (thasvalue) is set, all other bits are not used and MUST be set to '0'.

Length - 5

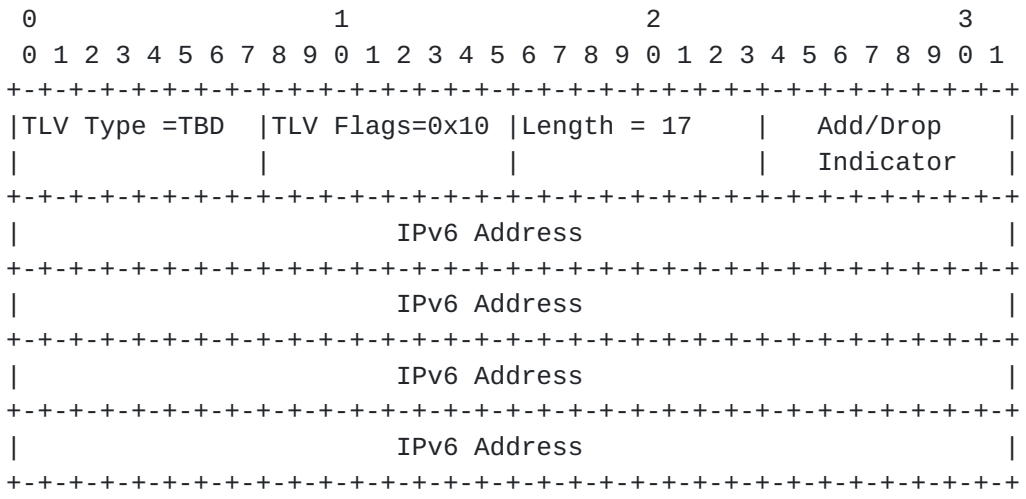
Add/Drop Indicator - Value indicating whether this is a new or existing address (0x01), or a withdrawal of an address (0x02).

IPv4 Address - IPv4 Address of the far-end neighbor or peer.

10.6 IPv6 Address Sub-TLV

The IPv6 Address Sub-TLV MAY be used in Neighbor Up, Neighbor Update, and Peer Update Messages, if the client is aware of the Layer 3 address. When included in Neighbor messages, the IPv6 Address sub-TLV contains the IPv6 address of the far-end neighbor. In the Peer Update, it contains the IPv6 address of the originating peer. In either case, the sub-TLV also contains an indication of whether this is a new or existing address, or is a deletion of a previously known address.

The IPv6 Address sub-TLV contains the following fields:



TLV Type - TBD

TLV Flags - 0x10, Bit 3 (thasvalue) is set, all other bits are not used and MUST be set to '0'.

Length - 17

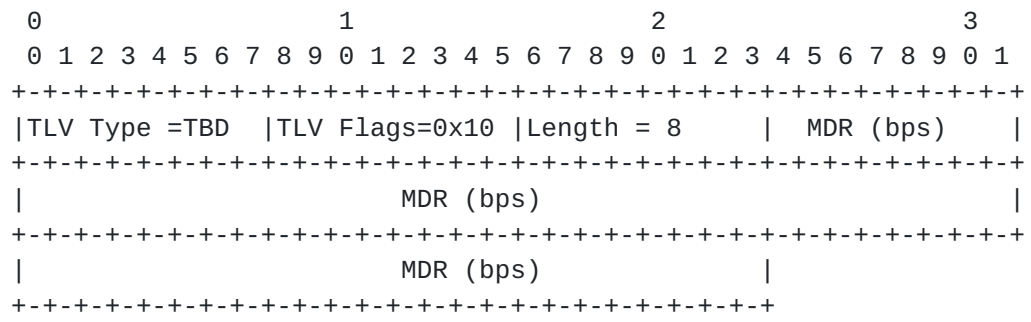
Add/Drop Indicator - Value indicating whether this is a new or existing address (0x01), or a withdrawal of an address (0x02).

IPv6 Address - IPv6 Address of the far-end neighbor or peer.

10.7 Maximum Data Rate Sub-TLV

The Maximum Data Rate (MDR) Sub-TLV is used in Neighbor Up, Neighbor Update, Peer Discovery, Peer Update, and Link Characteristics ACK Messages to indicate the maximum theoretical data rate, in bits per second, that can be achieved on the link. When metrics are reported via the messages listed above, the maximum data rate MUST be reported.

The Maximum Data Rate sub-TLV contains the following fields:



TLV Type - TBD

TLV Flags - 0x10, Bit 3 (thasvalue) is set, all other bits are not used and MUST be set to '0'.

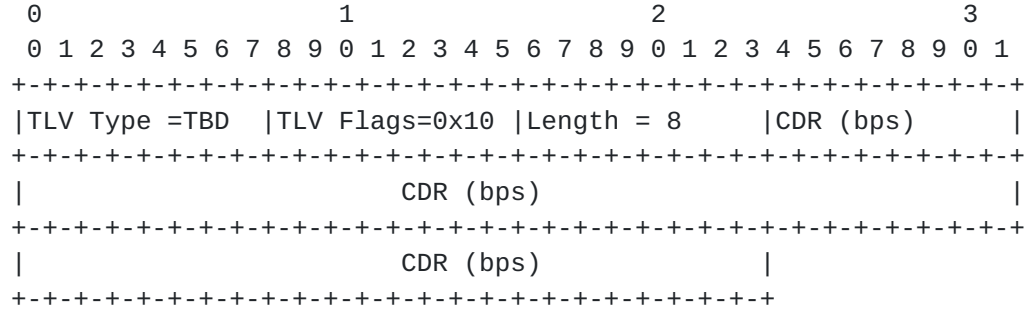
Length - 8

Maximum Data Rate - A 64-bit unsigned number, representing the maximum theoretical data rate, in bits per second (bps), that can be achieved on the link.

10.8 Current Data Rate Sub-TLV

The Current Data Rate (CDR) Sub-TLV is used in Neighbor Up, Neighbor Update, Peer Discovery, Peer Update, Link Characteristics Request, and Link Characteristics ACK messages to indicate the rate at which the link is currently operating, or in the case of the Link Characteristics Request, the desired data rate for the link.

The Current Data Rate sub-TLV contains the following fields:



TLV Type - TBD

TLV Flags - 0x10, Bit 3 (thasvalue) is set, all other bits are not used and MUST be set to '0'.

Length - 8

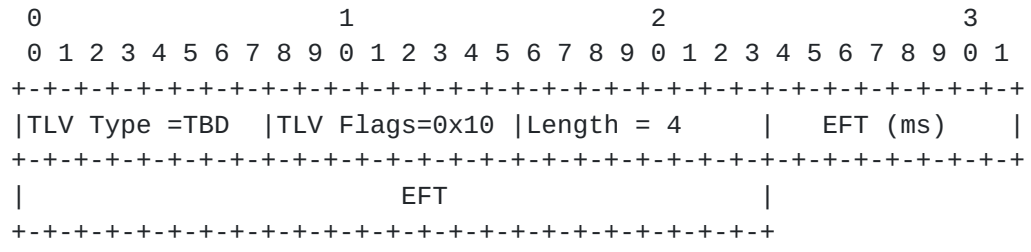
Current Data Rate - A 64-bit unsigned number, representing the current rate, in bits per second (bps), on the link. When reporting metrics (e.g, in Neighbor Up, Neighbor Down, Peer

Discovery, Peer Update, or Link Characteristics ACK), if there is no distinction between current and maximum data rates, current data rate SHOULD be set equal to the maximum data rate.

10.9 Expected Forwarding Time Sub-TLV

The Expected Forwarding Time (EFT) Sub-TLV is used in Neighbor Up, Neighbor Update, Peer Discovery, and Peer Update messages to indicate the typical latency between the arrival of a given packet at the transmitting device and the reception of the packet at the other end of the link. EFT combines transmission time, idle time, waiting time, freezing time, and queuing time to the degree that those values are meaningful to a given transmission medium.

The Expected Forwarding Time sub-TLV contains the following fields:



TLV Type - TBD

TLV Flags - 0x10, Bit 3 (thasvalue) is set, all other bits are not used and MUST be set to '0'.

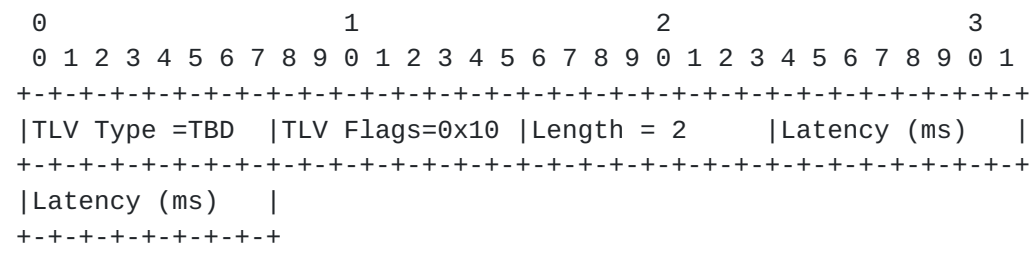
Length - 4

Current Data Rate - A 32-bit unsigned number, representing the expected forwarding time, in milliseconds, on the link.

10.10 Latency Sub-TLV

The Latency Sub-TLV is used in Neighbor Up, Neighbor Update, Peer Discovery, Peer Update, Link Characteristics Request, and Link Characteristics ACK messages to indicate the amount of latency on the link, or in the case of the Link Characteristics Request, to indicate the maximum latency required (e.g. a should-not-exeed value) on the link.

The Latency Sub-TLV contains the following fields:



TLV Type - TBD

TLV Flags - 0x10, Bit 3 (thasvalue) is set, all other bits are not used and MUST be set to '0'.

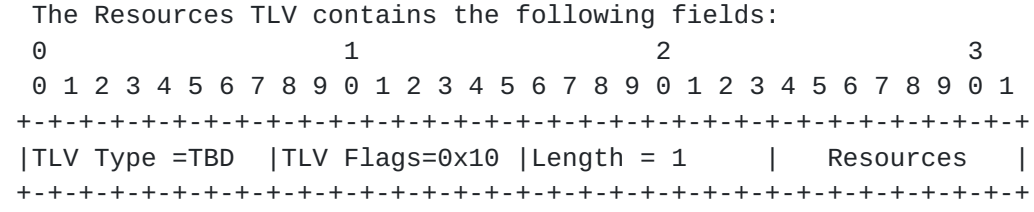
Length - 2

Latency - The transmission delay that a packet encounters as it is transmitted over the link. In Neighbor Up, Neighbor Update, and Link Characteristics ACK, this value is reported in absolute delay, in milliseconds. The calculation of latency is implementation dependent. For example, the latency may be a running average calculated from the internal queuing. If a device cannot calculate latency, it SHOULD be reported as 0. In the Link Characteristics Request Message, this value represents the maximum delay, in milliseconds, expected on the link.

10.11 Resources Sub-TLV

The Resources Sub-TLV is used in Neighbor Up, Neighbor Update, Peer Discovery, Peer Update, and Link Characteristics ACK messages to indicate a percentage (0-100) amount of resources (e.g. battery power) remaining on the originating peer.

The Resources TLV contains the following fields:



TLV Type - TBD

TLV Flags - 0x10, Bit 3 (thasvalue) is set, all other

bits are not used and MUST be set to '0'.

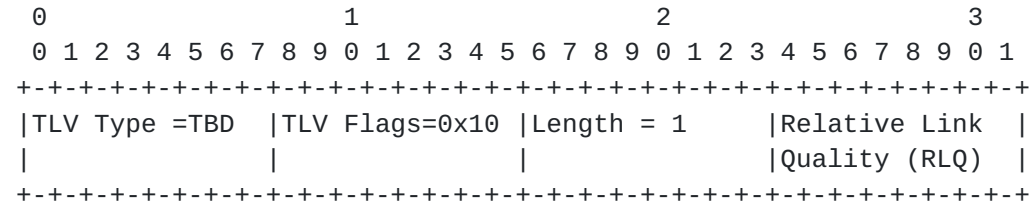
Length - 1

Resources - A percentage, 0-100, representing the amount of remaining resources, such as battery power. If resources cannot be calculated, a value of 100 SHOULD be reported.

10.12 Relative Link Quality Sub-TLV

The Relative Link Quality (RLQ) Sub-TLV is used in Neighbor Up, Neighbor Update, Peer Discovery, Peer Update, and Link Characteristics ACK messages to indicate the quality of the link as calculated by the originating peer.

The Relative Link Quality sub-TLV contains the following fields:



TLV Type - TBD

TLV Flags - 0x10, Bit 3 (thasvalue) is set, all other bits are not used and MUST be set to '0'.

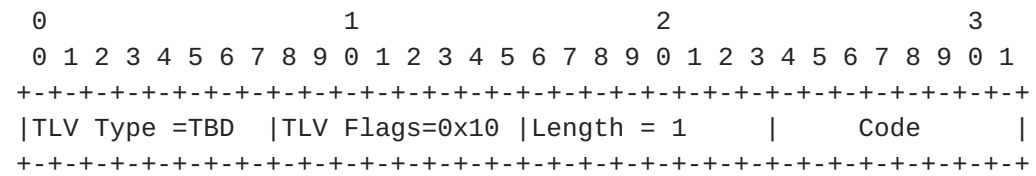
Length - 1

Relative Link Quality - A non-dimensional number, 0-100, representing relative link quality. A value of 100 represents a link of the highest quality. If the RLQ cannot be calculated, a value of 100 SHOULD be reported.

10.13 Status Sub-TLV

The Status Sub-TLV is sent from either the client or server to indicate the success or failure of a given request

The Status Sub-TLV contains the following fields:



TLV Type - TBD

TLV Flags - 0x10, Bit 3 (thasvalue) is set, all other bits are not used and MUST be set to '0'.

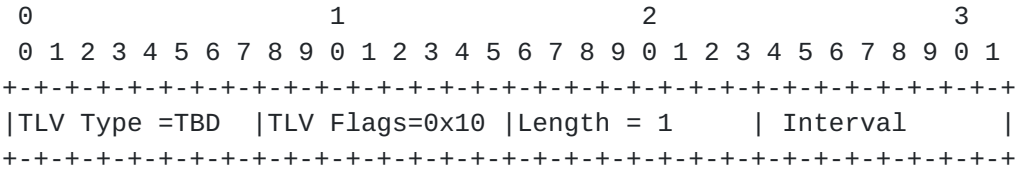
Length - 1

Termination Code - 0 = Success
Non-zero = Failure. Specific values of a non-zero termination code depend on the operation requested (e.g. Neighbor Up, Neighbor Down, etc).

10.14 Heartbeat Interval Sub-TLV

The Heartbeat Interval Sub-TLV MAY be sent from the client during Peer Discovery to indicate the desired Heartbeat timeout window. If included in the Peer Discovery, the server MUST either accept the timeout interval, or reject the Peer Discovery. Failing to include the Heartbeat Interval Sub-TLV in Peer Discovery indicates a desire to establish the peer-to-peer DLEP session without an activity timeout (e.g. an infinite timeout value).

The Heartbeat Interval Sub-TLV contains the following fields:



TLV Type - TBD

TLV Flags - 0x10, Bit 3 (thasvalue) is set, all other bits are not used and MUST be set to '0'.

Length - 1

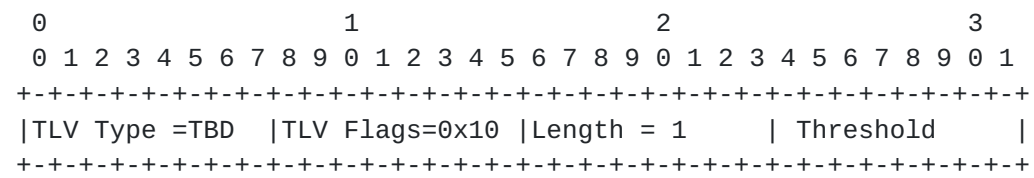
Interval - 0 = Do NOT use heartbeats on this peer-to-peer session. Non-zero = Interval, in seconds, for heartbeat messages.

10.15 Heartbeat Threshold Sub-TLV

The Heartbeat Threshold Sub-TLV MAY be sent from the client during Peer Discovery to indicate the desired number of windows, of time (Heartbeat Interval) seconds, to wait before either peer declares the peer session lost. In this case, the overall amount of time before a peer session is declared lost is expressed as (Interval * Threshold), where 'Interval' is the value in the Heartbeat Interval sub-TLV, documented above. If this sub-TLV is included by the client in the Peer Discovery, the client MUST also specify the Heartbeat Interval sub-TLV with a non-zero interval. If this sub-TLV is received during Peer Discovery, the server MUST

either accept the threshold, or reject the Peer Discovery. If the Heartbeat Interval Sub-TLV is included, but this Sub-TLV is omitted, then a threshold of '1' is assumed.

The Heartbeat Threshold Sub-TLV contains the following fields:



TLV Type - TBD

TLV Flags - 0x10, Bit 3 (thasvalue) is set, all other bits are not used and MUST be set to '0'.

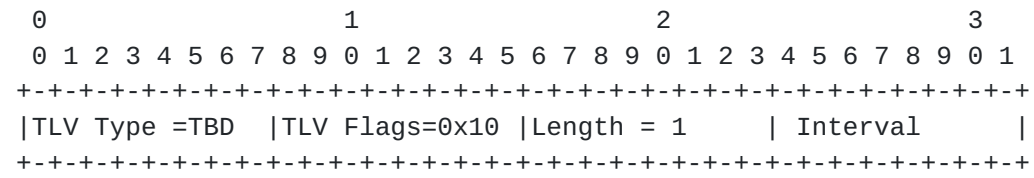
Length - 1

Threshold - 0 = Do NOT use heartbeats on this peer-to-peer session. Non-zero = Number of windows, of Heartbeat Interval seconds, to wait before declaring a peer-to-peer session to be lost.

10.16 Link Characteristics ACK Timer Sub-TLV

The Link Characteristic ACK Timer Sub-TLV MAY be sent from the client during Peer Discovery to indicate the desired number of seconds the server should wait for a response to a Link Characteristics Request. If this sub-TLV is received during Peer Discovery, the server MUST either accept the timeout value, or reject the Peer Discovery. If this Sub-TLV is omitted, implementations SHOULD choose a default value.

The Link Characteristics ACK Timer Sub-TLV contains the following fields:



TLV Type - TBD

TLV Flags - 0x10, Bit 3 (thasvalue) is set, all other bits are not used and MUST be set to '0'.

Length - 1

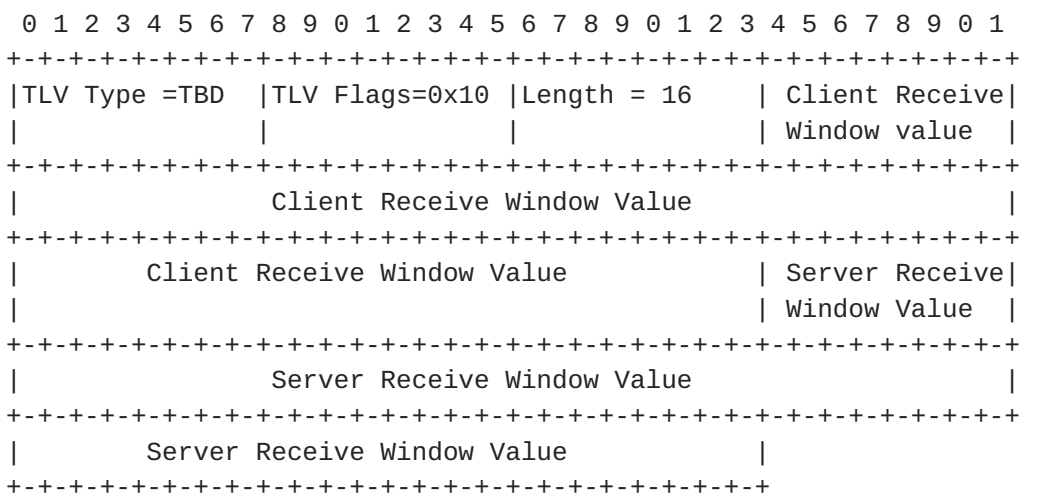
Interval - 0 = Do NOT use timeouts for Link Characteristics requests on this peer-to-peer session.

Non-zero = Interval, in seconds, to wait before considering a Link Characteristics Request has been lost.

10.17 Credit Window Status Sub-TLV

The Credit Window Status Sub-TLV MUST be sent by the DLEP peer originating a Neighbor Up message when use of credits is desired for a given session. In the Neighbor Up message, when credits are desired, the originating peer MUST set the value of the window it controls (e.g. the Client Receive Window, or Server Receive Window) to an initial, non-zero value. The peer receiving a Neighbor Up message with a Credit Window Status Sub-TLV MUST either reject the use of credits, via a Neighbor Up ACK response with the correct Status Sub-TLV, or set the initial value from the data contained in the Credit Window Status Sub-TLV. If the initialization completes successfully, the receiving peer MUST respond to the Neighbor Up message with a Neighbor Up ACK message that contains a Credit Window Status Sub-TLV, initializing its receive window.

The Credit Window Status Sub-TLV contains the following fields:



TLV Type - TBD

TLV Flags - 0x10, Bit 3 (thasvalue) is set, all other bits are not used and MUST be set to '0'.

Length - 16

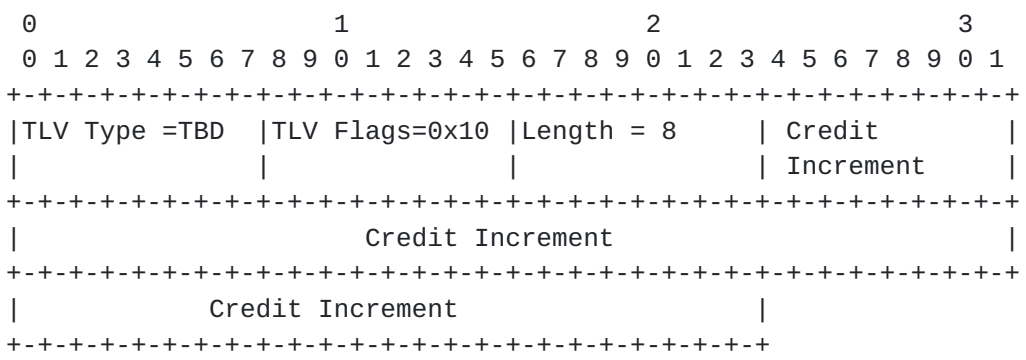
Client Receive Window value - A 64-bit unsigned number, indicating the current (or initial) number of credits available on the Client Receive Window.

Server Receive Window Value - A 64-bit unsigned number, indicating the current (or initial) number of credits available on the Server Receive Window.

10.18 Credit Grant Sub-TLV

The Credit Grant Request Sub-TLV MAY be sent from either DLEP peer to grant an increment to credits on a window. The Credit Grant Sub-TLV is sent as part of a Neighbor Update message. The value in a Credit Grant Sub-TLV represents an increment to be added to any existing credits available on the window. Upon successful receipt and processing of a Credit Grant Sub-TLV, the receiving peer SHOULD respond with a DLEP Neighbor Update message containing a Credit Window Status Sub-TLV to report the updated aggregate values for synchronization purposes.

The Credit Grant Sub-TLV contains the following fields:



TLV Type - TBD

TLV Flags - 0x10, Bit 3 (thasvalue) is set, all other bits are not used and MUST be set to '0'.

Length - 0

Reserved - A 64-bit unsigned number representing the additional credits to be assigned to the credit window. Since credits can only be granted by the receiver on a window, the applicable credit window (either the CRW or the SRW) is derived from the sender of the grant. The Credit Increment MUST NOT cause the window to overflow; if this condition occurs, implementations MUST set the credit window to the maximum value contained in a 64-bit quantity.

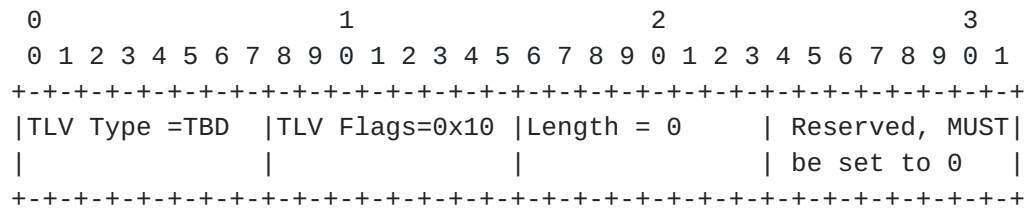
10.19 Credit Request Sub-TLV

The Credit Request Sub-TLV MAY be sent from either DLEP peer, via a Neighbor Update order, to indicate the desire for the partner to grant additional credits in order for data transfer to proceed on

the session. If the corresponding Neighbor Up message for this session did NOT contain a Credit Window Status Sub-TLV, indicating that credits are to be used on the session, then the Credit Request Sub-TLV MUST be rejected, by sending a Neighbor Update ACK containing a Status Sub-TLV, by the receiving peer. If credits are in use on

the session, then the receiving peer MAY respond with a DLEP Neighbor Update message containing a Credit Grant Sub-TLV with an increment of credits for the session.

The Credit Request Sub-TLV contains the following fields:



TLV Type - TBD

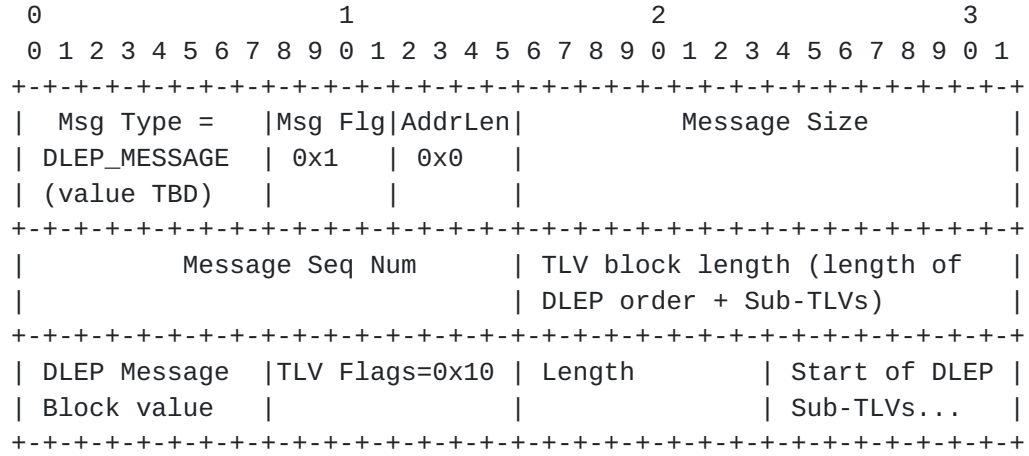
TLV Flags - 0x10, Bit 3 (thasvalue) is set, all other bits are not used and MUST be set to '0'.

Length - 0

Reserved - 0 = This field is currently unused and MUST be set to 0.

11. DLEP Protocol Messages

DLEP places no additional requirements on the RFC 5444 Packet formats, or the packet header. DLEP does require that the optional 'msg-seq-num' in the message header exist, and defines a set of values for the 'tlv-type' field in the RFC 5444 TLV block. Therefore, a DLEP message, starting from the RFC 5444 Message header, would appear as follows:



11.1 Message Block TLV Values

As mentioned above, all DLEP messages utilize a single RFC 5444 message type, the DLEP_MESSAGE (TBD). DLEP further identifies protocol messages by using the 'tlv-type' field in the RFC 5444 message TLV block. DLEP defines the following Message-Type-specific values for the tlv-type field:

TLV Value	TLV Description
TBD	Attached Peer Discovery
TBD	Detached Peer Discovery
TBD	Peer Offer
TBD	Peer Update
TBD	Peer Update ACK
TBD	Peer Termination
TBD	Peer Termination ACK
TBD	Neighbor Up
TBD	Neighbor Up ACK
TBD	Neighbor Down
TBD	Neighbor Down ACK
TBD	Neighbor Update
TBD	Neighbor Address Update
TBD	Neighbor Address Update ACK
TBD	Heartbeat
TBD	Link Characteristics Request
TBD	Link Characteristics ACK

In all of the diagrams following, the message layouts begin with the RFC 5444 message header.

12. Peer Discovery Messages

There are two different types of Peer Discovery Messages, Attached and Detached. Attached Peer Discovery Messages are sent by the client when it is directly attached to the server (e.g. the client exists as a card in the chassis, or it is connected via Ethernet with no intervening devices). The Detached Peer Discovery message, on the other hand, is sent by a "remote" client -- for example, a client at a satellite hub system might use a Detached Discovery Message in order to act as a proxy for remote ground terminals. To explain in another way, a detached client uses the variable link itself (the radio or satellite link) to establish a DLEP session with a remote server.

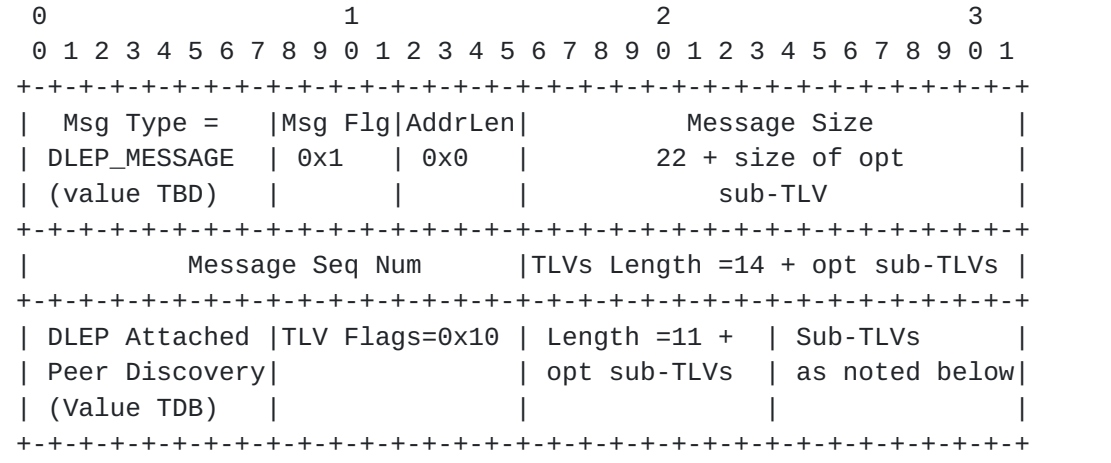
12.1 Attached Peer Discovery Message

The Attached Peer Discovery Message is sent by an attached client to a server to begin a new DLEP association. The Peer Offer message is required to complete the discovery process. The client MAY implement its own retry heuristics in the event it (the client) determines the Attached Peer Discovery Message has timed out. An Attached Peer Discovery Message received from a peer that is already in session MUST be processed as if a Peer Termination Message had

been received. An implementation MAY then process the received Attached Peer Discovery Message.

Note that metric Sub-TLVs MAY be supplied with the Peer Discovery order. If metric Sub-TLVs are supplied, they MUST be used as a default value for all neighbor sessions established via this peer.

The Attached Peer Discovery Message contains the following fields:



Message Type - DLEP_MESSAGE (value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). No other bits are used and MUST be set to '0'.

Message Address Length - 0x0

Message Size - 22 + size of optional sub-TLVs

Message Sequence Number - A 16-bit unsigned integer field containing a sequence number generated by the message originator.

TLV Block - TLVs Length: 14 + size of optional sub-TLVs.

Sub-TLVs:

Identification (MANDATORY)
Version (OPTIONAL)
Peer Type (OPTIONAL)
Heartbeat Interval (OPTIONAL)
Heartbeat Threshold (OPTIONAL)
Link Characteristics ACK Timer (OPTIONAL)
Maximum Data Rate (OPTIONAL)
Current Data Rate (OPTIONAL)
Latency (OPTIONAL)
Expected Forwarding Time (OPTIONAL)
Resources (OPTIONAL)
Relative Link Quality (OPTIONAL)

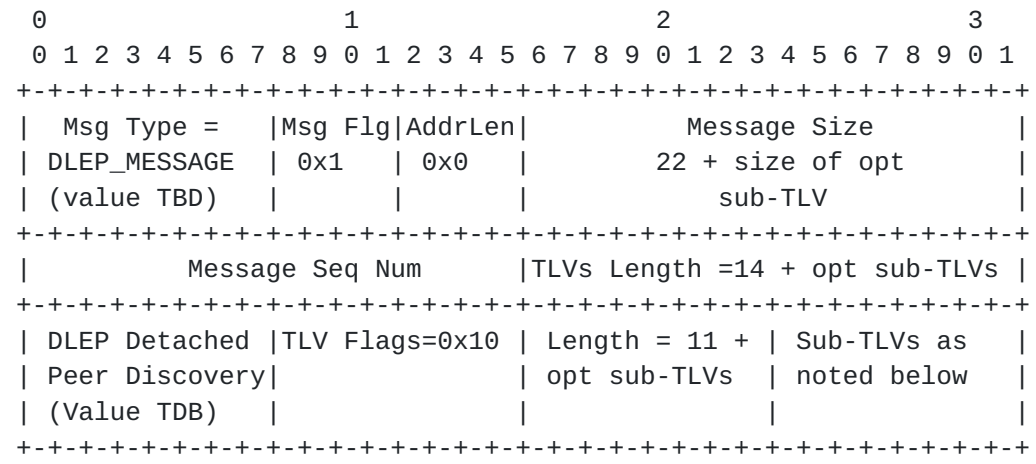
12.2 Detached Peer Discovery Message

The Detached Peer Discovery Message is sent by a detached client proxy to a server to begin a new DLEP session. The Peer Offer message is required to complete the discovery process. The client

MAY implement its own retry heuristics in the event it (the client) determines the Detached Peer Discovery Message has timed out. When a DLEP implementation responds to a Detached Discovery Message with a Peer Offer, the implementation MUST enter an "in session" state with the peer. Any subsequent discovery message received from the peer MUST be processed as if a Peer Termination Message had been received (e.g. the existing peer session MUST be terminated). An implementation MAY then process the received discovery message.

If metric sub-TLVs (e.g. Maximum Data Rate) are supplied with the Detached Peer Discovery message, these metrics MUST be used as the initial values for all far-end sessions (neighbors) established via the peer.

The Detached Peer Discovery Message contains the following fields:



Message Type - DLEP_MESSAGE (value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are not used and MUST be set to '0'.

Message Address Length - 0x0

Message Size - 22 + size of optional sub-TLVs

Message Sequence Number - A 16-bit unsigned integer field containing a sequence number, generated by the message originator.

TLV Block - TLVs Length: 14 + size of optional sub-TLVs.

Sub-TLVs

Identification (MANDATORY)
Version (OPTIONAL)
Peer Type (OPTIONAL)
Heartbeat Interval (OPTIONAL)

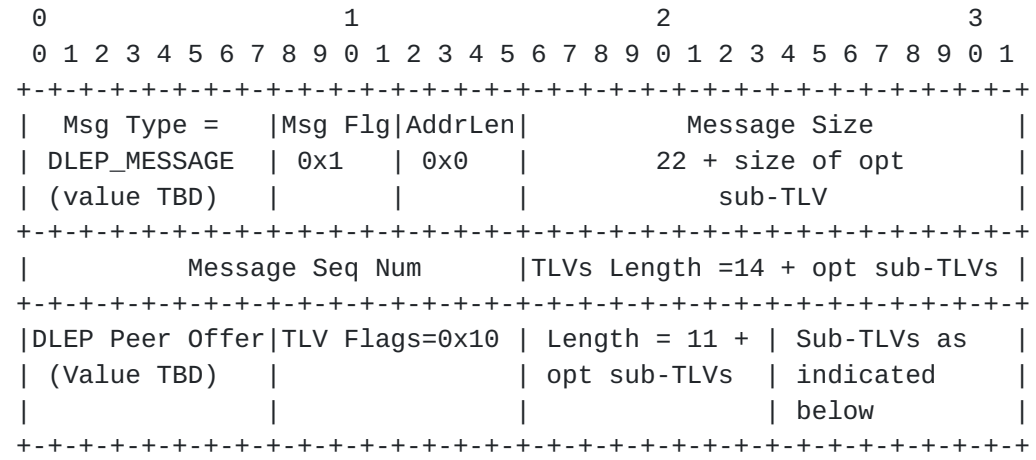
Heartbeat Threshold (OPTIONAL)
Link Char. ACK Timer (OPTIONAL)
Maximum Data Rate (OPTIONAL)
Current Data Rate (OPTIONAL)
Latency (OPTIONAL)
Expected Forwarding Time (OPTIONAL)
Resources (OPTIONAL)
Relative Link Quality (OPTIONAL)

As in the Attached Peer Discovery, the client MAY include metric sub-TLVs. If included, the router SHOULD use these values as defaults that will apply to all sessions established via this client.

13. Peer Offer Message

The Peer Offer Message is sent by a server to a client in response to a Peer Discovery Message. The Peer Offer Message is the response to either of the Peer Discovery messages (Attached or Detached), and completes the DLEP peer session establishment. Upon sending the Peer Offer Message, the server then enters an "in session" state with the client. From the client perspective, receipt and successful parsing of a Peer Offer order MUST cause the client to enter the "in session" state. Any subsequent Discovery messages sent or received on this session MUST be considered an error, and the session MUST be terminated as if a Peer Termination Message had been received.

The Peer Offer Message contains the following fields:



Message Type - DLEP_MESSAGE (Value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to '0'.

Message Address Length - 0x0

Message Size - 22 + size of optional sub-TLVs

Message Sequence Number - A 16-bit unsigned integer field containing a sequence number, generated by the message originator.

TLV Block - TLV Length: 14 + size of optional sub-TLVs

Sub TLVs

Identification (MANDATORY)
Version (OPTIONAL)
Peer Type (OPTIONAL)
IPv4 Address (OPTIONAL)
IPv6 Address (OPTIONAL)
Status (OPTIONAL)
Heartbeat Interval (OPTIONAL)
Heartbeat Threshold (OPTIONAL)
Link Characteristics ACK Timer (OPTIONAL)

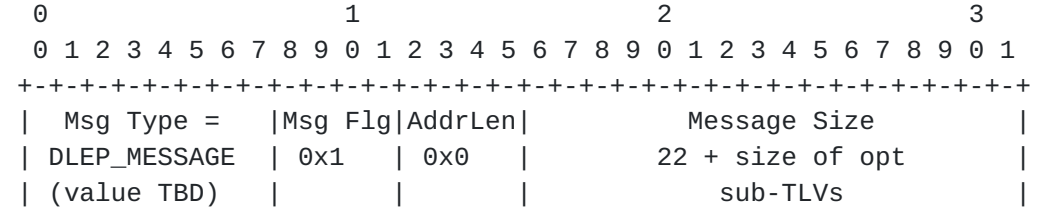
14. Peer Update Message

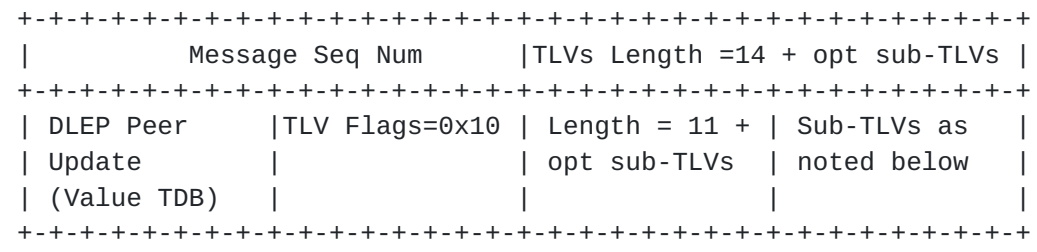
The Peer Update message is sent by a DLEP peer to indicate local Layer 3 address changes, or for metric changes on a device-wide basis. For example, addition of an IPv4 address to the server would prompt a Peer Update message to its attached DLEP clients. Also, a client that changes its Maximum Data Rate for all destinations MAY reflect that change via a Peer Update Message to its attached server.

With Layer 3 address changes, if the client is capable of understanding and forwarding this information, the address update would prompt any remote DLEP clients (DLEP clients that are on the far-end of the variable link) to issue a "Neighbor Update" message to their local servers with the new (or deleted) addresses. Clients that do not track Layer 3 addresses MUST silently parse and ignore the Peer Update Message. Clients that track Layer 3 addresses MUST acknowledge the Peer Update with a Peer Update ACK message. Servers receiving a Peer Update with metric changes MUST apply the new metric to all neighbor sessions established via the client. Peers MAY employ heuristics to retransmit Peer Update messages. The sending of Peer Update Messages for Layer 3 address changes SHOULD cease when a server implementation determines that a client does NOT support Layer 3 address tracking.

If metric Sub-TLVs are supplied with the Peer Update message (e.g. Maximum Data Rate), these metrics MUST be applied to all neighbor sessions accessible via the peer.

The Peer Update Message contains the following fields:




Message Seq Num
TLVs Length =14 + opt sub-TLVs
DLEP Peer Update (Value TDB)
TLV Flags=0x10
Length = 11 + opt sub-TLVs
Sub-TLVs as noted below

Message Type - DLEP_MESSAGE (Value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to '0'.

Message Address Length - 0x0

Message Size - 22 + optional Sub-TLVs

Message Sequence Number - A 16-bit unsigned integer containing a sequence number (generated by originator).

TLV Block - TLV Length: 14 + length of optional sub-TLVs.

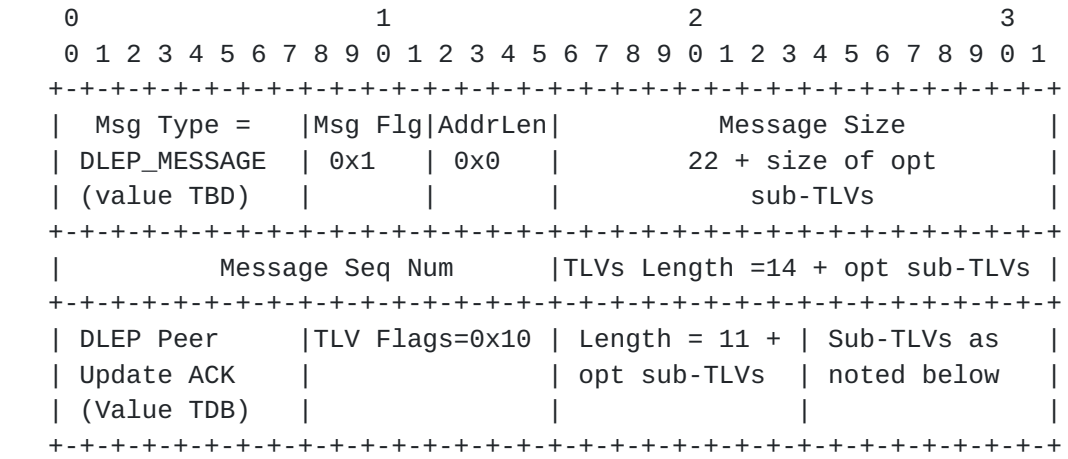
Sub TLVs

Identification (MANDATORY)
IPv4 Address (OPTIONAL)
IPv6 Address (OPTIONAL)
Maximum Data Rate (OPTIONAL)
Current Data Rate (OPTIONAL)
Latency (OPTIONAL)
Expected Forwarding Time (OPTIONAL)
Resources (OPTIONAL)
Relative Link Quality (OPTIONAL)

15. Peer Update ACK Message

A peer sends the Peer Update ACK Message to indicate whether a Peer Update Message was successfully processed.

The Peer Update ACK message contains the following fields:



Message Type - DLEP_MESSAGE (Value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to '0'.

Message Address Length - 0x0

Message Size - 22 + size of optional sub-TLVs.

Message Sequence Number - A 16-bit unsigned integer field containing the sequence number from the Neighbor Up Message that is being acknowledged.

TLV Block - TLV Length: 14 + optional sub-TLVs

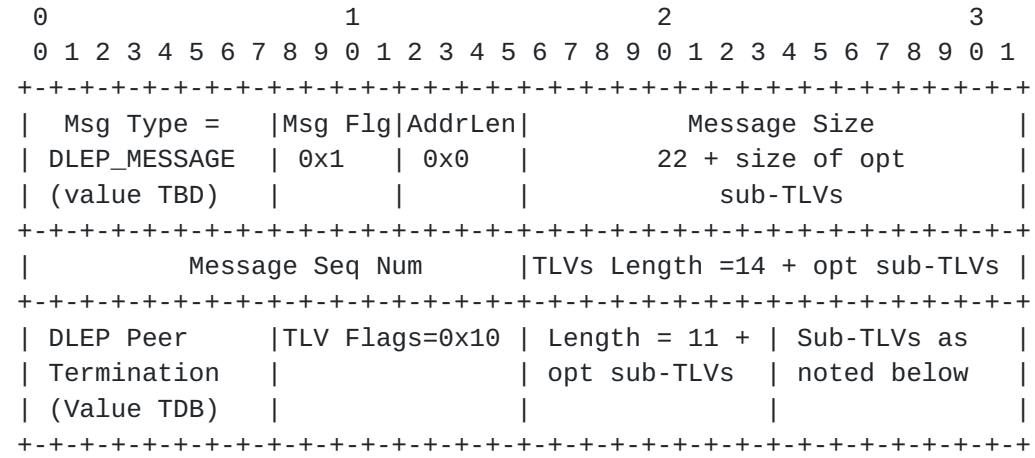
Sub TLVs

Identification (MANDATORY)
Status (OPTIONAL)

16. Peer Termination Message

The Peer Termination Message is sent by either the client or the server when a session needs to be terminated. Transmission of a Peer Termination ACK message is required to confirm the termination process. The sender of the Peer Termination message is free to define its heuristics in event of a timeout. The receiver of a Peer Termination Message MUST terminate all neighbor sessions and release associated resources. State machines are returned to the "discovery" state. No Neighbor Down messages are sent.

The Peer Termination Message contains the following fields:



Message Type - DLEP_MESSAGE (Value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to '0'.

Message Address Length - 0x0

Message Size - 22 + size of optional sub-TLVs.

Message Sequence Number - A 16-bit unsigned integer field containing a sequence number generated by the message originator.

TLV Block - TLV Length = 14 + optional sub-TLVs

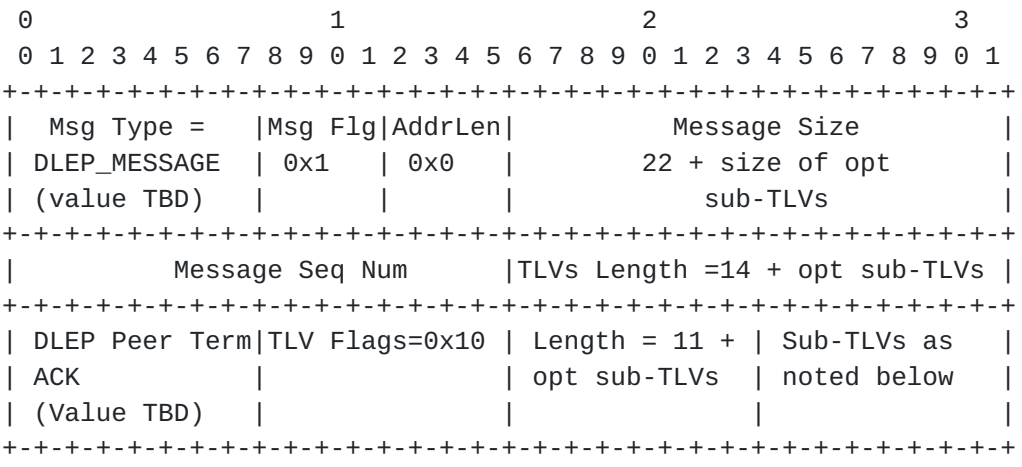
Sub TLVs

Identification (MANDATORY)
Status (OPTIONAL)

17. Peer Termination ACK Message

The Peer Termination Message ACK is sent by a DLEP peer in response to a received Peer Termination order.

The Peer Termination ACK Message contains the following fields:



Message Type - DLEP_MESSAGE (Value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to '0'.

Message Address Length - 0x0

Message Size - 22 + optional sub-TLVs.

Message Sequence Number - A 16-bit unsigned integer field containing the sequence number in the corresponding Peer Termination Message being acknowledged.

TLV Block - TLV Length = 14 + optional Sub-TLVs

Sub-TLVs

Identification (MANDATORY)
Status (OPTIONAL)

18. Neighbor Up Message

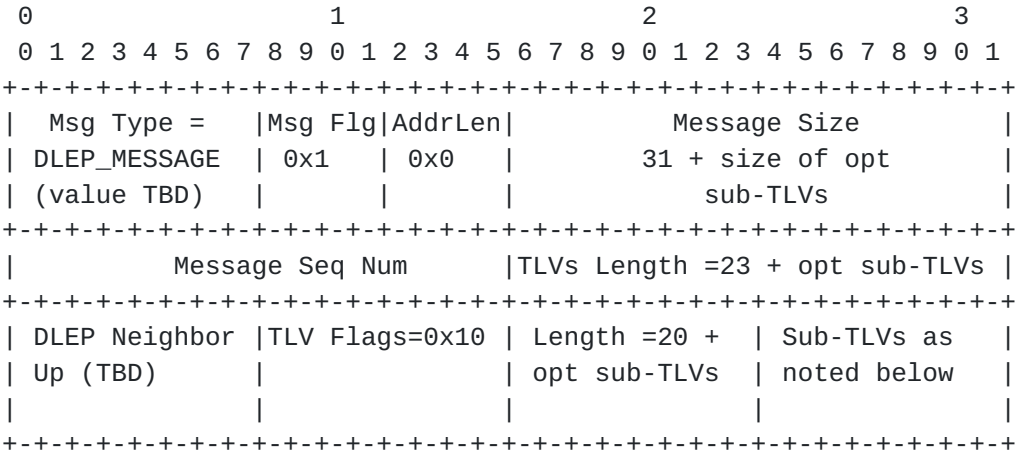
A peer sends the Neighbor Up message to report that a new potential routing neighbor, or a new destination within the network, has been detected. A Neighbor Up ACK Message is required

to confirm a received Neighbor Up. A Neighbor Up message can be sent by a client to signal that it (the client) has detected a new

neighbor, or by the server to indicate that new destinations (e.g. Multicast groups) exist within the network.

The sender of the Neighbor Up Message is free to define its retry heuristics in event of a timeout. When a Neighbor Up message is received and successfully parsed, the receiver should enter an "in session" state with regard to the far-end destination, and send an acknowledgement to the originating peer.

The Neighbor Up Message contains the following fields:



Message Type - DLEP_MESSAGE (Value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to '0'.

Message Address Length - 0x0

Message Size - 31 + optional Sub-TLVs

Message Sequence Number - A 16-bit unsigned integer field containing a sequence number generated by the message originator.

TLV Block - TLV Length: 23 + optional Sub-TLVs.

Sub-TLVs

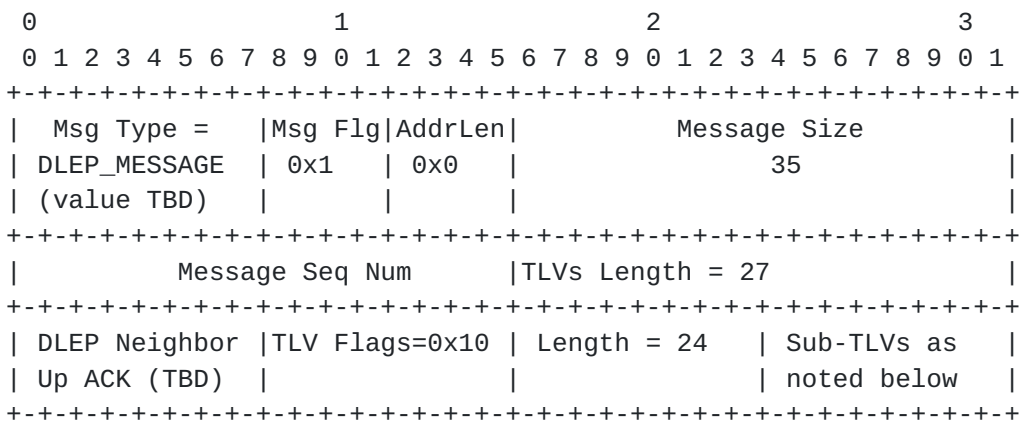
Identification (MANDATORY)
MAC Address (MANDATORY)
IPv4 Address (OPTIONAL)
IPv6 Address (OPTIONAL)
Maximum Data Rate (OPTIONAL)
Current Data Rate (OPTIONAL)
Latency (OPTIONAL)
Expected Forwarding Time (OPTIONAL)

Resources (OPTIONAL)
Relative Link Factor (OPTIONAL)
Credit Window Status (OPTIONAL)

19. Neighbor Up ACK Message

A peer sends the Neighbor Up ACK Message to indicate whether a Neighbor Up Message was successfully processed. When a peer receives a Neighbor Up ACK message containing a Status Sub-TLV with a status code of 0, the receiving peer should enter an "in session" state with respect to the far-end destination.

The Neighbor Up ACK message contains the following fields:



Message Type - DLEP_MESSAGE (Value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to '0'.

Message Address Length - 0x0

Message Size - 35

Message Sequence Number - A 16-bit unsigned integer field containing the sequence number from the Neighbor Down Message that is being acknowledged.

TLV Block - TLV Length: 27

Sub-TLVs - Identification (MANDATORY)
MAC Address Sub-TLV (MANDATORY)
Status Sub-TLV (MANDATORY)
Credit Window Status (OPTIONAL)

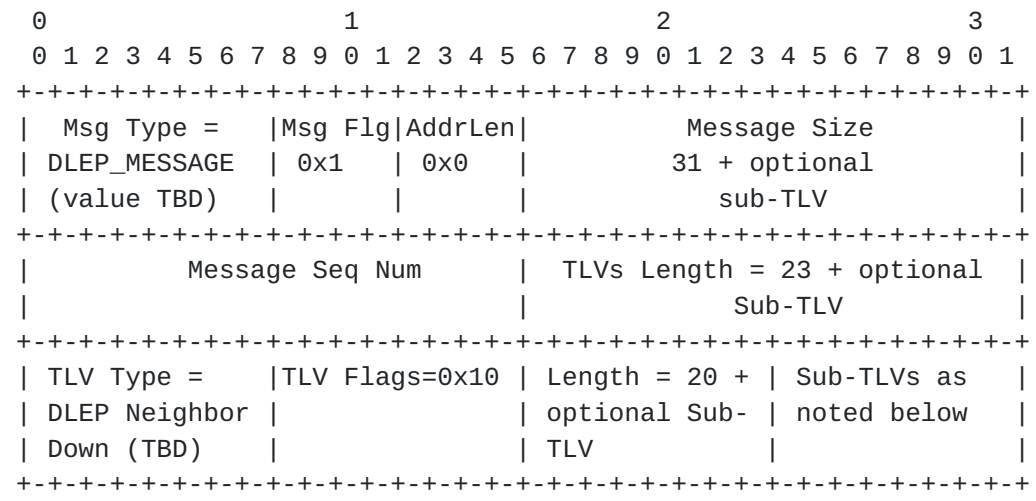
20. Neighbor Down Message

A DLEP peer sends the Neighbor Down message to report when a destination (a routing peer or a multicast group) is no longer reachable. The Neighbor Down message MUST contain a MAC Address TLV.

Any other TLVs present MAY be ignored. A Neighbor Down ACK Message is required to confirm the process. The sender of the Neighbor Down message is free to define its retry heuristics in event of a timeout. Upon successful receipt and parsing of a Neighbor Down message, the

receiving peer MUST remove all state information for the destination, and send a Neighbor Down ACK message to the originating peer.

The Neighbor Down Message contains the following fields:

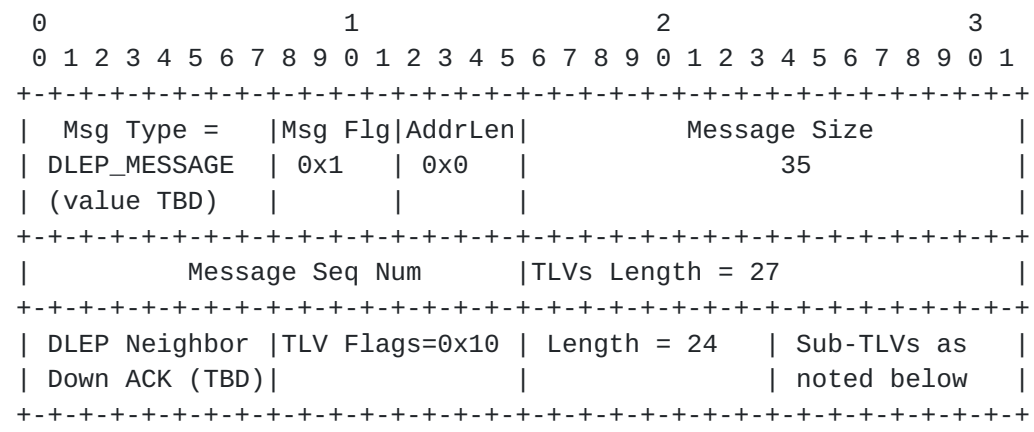


Message Type	- DLEP_MESSAGE (Value TBD)
Message Flags	- Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to '0'.
Message Address Length	- 0x0
Message Size	- 31 + optional TLVs
Message Sequence Number	- A 16-bit unsigned integer field containing a sequence number generated by the message originator.
TLV Block	- TLV Length: 23 + optional Sub-TLVs
Sub TLVs	Identification (MANDATORY) MAC Address (MANDATORY) Status (OPTIONAL)

21. Neighbor Down ACK Message

A peer sends the Neighbor Down ACK Message to indicate whether a received Neighbor Down Message was successfully processed. If successfully processed, the sending peer MUST remove all state information on the referenced neighbor session.

The Neighbor Down ACK message contains the following fields:



Message Type - DLEP_MESSAGE (Value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to '0'.

Message Address Length - 0x0

Message Size - 35

Message Sequence Number - A 16-bit unsigned integer field containing the sequence number from the Neighbor Down Message that is being acknowledged.

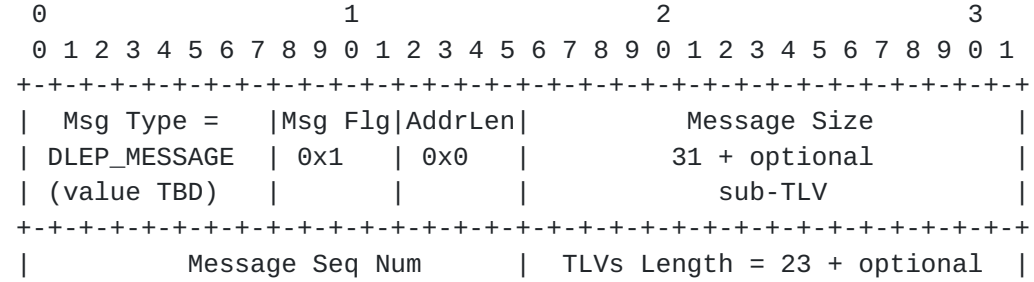
TLV Block - TLV Length: 27

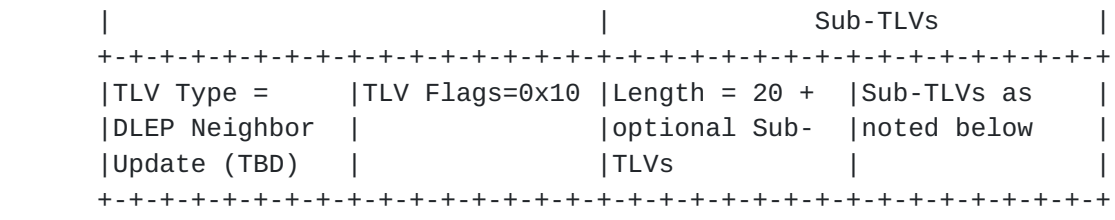
Sub-TLVs - Identification (MANDATORY)
MAC Address (MANDATORY)
Status (MANDATORY)

22. Neighbor Update Message

The client sends the Neighbor Update message when a change in link metric parameters is detected for a destination.

The Neighbor Update Message contains the following fields:



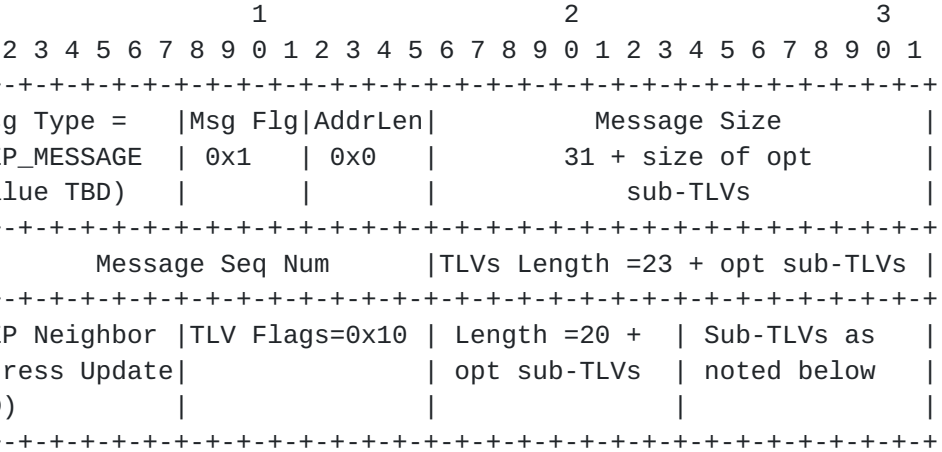

| | Sub-TLVs |
|TLV Type = DLEP Neighbor Update (TBD) |TLV Flags=0x10 |Length = 20 + optional Sub-TLVs |Sub-TLVs as noted below |

Message Type	- DLEP_MESSAGE (Value TBD)
Message Flags	- Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to '0'.
Message Address Length	- 0x0
Message Size	- 31 + optional TLVs
Message Sequence Number	- A 16-bit unsigned integer field containing a sequence number, generated by the message originator.
TLV Block	- TLVs Length - 23 + optional Sub-TLVs.
Sub TLVs	Identification (MANDATORY) MAC Address (MANDATORY) Maximum Data Rate (OPTIONAL) Current Data Rate (OPTIONAL) Latency (OPTIONAL) Resources (OPTIONAL) Relative Link Quality (OPTIONAL) Credit Window Status (OPTIONAL) Credit Grant (OPTIONAL) Credit Request (OPTIONAL)

23. Neighbor Address Update Message

The client sends the Neighbor Address Update message when a change in Layer 3 addressing is detected for a neighbor session.

The Neighbor Address Update Message contains the following fields:



Message Type	- DLEP_MESSAGE (Value TBD)
Message Flags	- Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to '0'.

Message Address Length - 0x0

Message Size - 31 + optional TLVs

Message Sequence Number - A 16-bit unsigned integer field containing a sequence number, generated by the message originator.

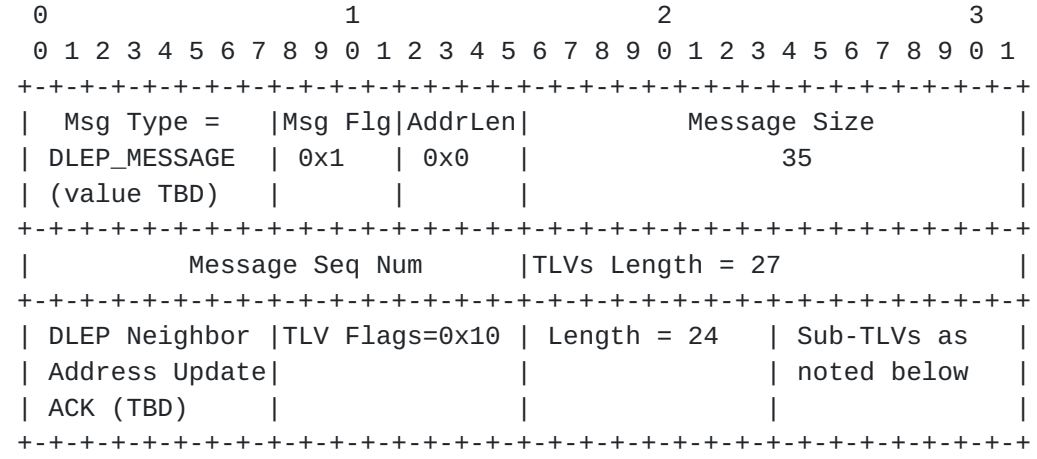
TLV Block - TLVs Length - 23 + optional Sub-TLVs.
Sub TLVs

Identification Sub-TLV (MANDATORY)
MAC Address Sub-TLV (MANDATORY)
IPv4 Address Sub-TLV (OPTIONAL)
IPv6 Address Sub-TLV (OPTIONAL)

24. Neighbor Address Update ACK Message

The server sends the Neighbor Address Update ACK Message to indicate whether a Neighbor Address Update Message was successfully processed.

The Neighbor Address Update ACK message contains the following fields:



Message Type - DLEP_MESSAGE (Value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to '0'.

Message Address Length - 0x0

Message Size - 35

Message Sequence Number - A 16-bit unsigned integer field containing

the sequence number from the Neighbor Down Message that is being acknowledged.

TLV Block - TLV Length: 27

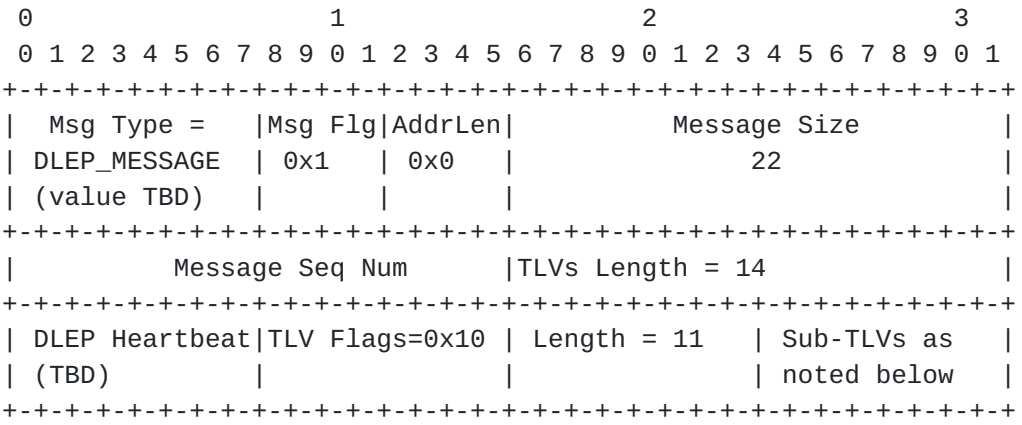
Sub TLVs

Identification Sub-TLV (MANDATORY)
MAC Address Sub-TLV (MANDATORY)
Status Sub-TLV (MANDATORY)

25. Heartbeat Message

A Heartbeat Message is sent by a peer every N seconds, where N is defined in the "Heartbeat Interval" field of the discovery message. The message is used by peers to detect when a DLEP session partner is no longer communicating. Peers SHOULD allow some integral number of heartbeat intervals (default 4) to expire with no traffic on the session before initiating DLEP session termination procedures.

The Heartbeat Message contains the following fields:



Message Type - DLEP_MESSAGE (Value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and SHOULD be set to '0'.

Message Address Length - 0x0

Message Size - 22

Message Sequence Number - A 16-bit unsigned integer field containing a sequence number generated by the message originator.

TLV Block - TLV Length = 14

Sub TLVs -

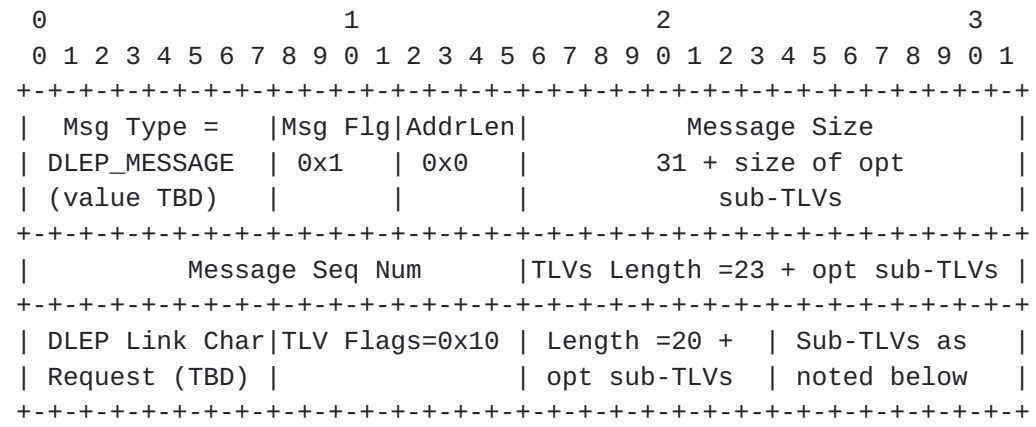
Identification Sub-TLV (MANDATORY)

26. Link Characteristics Request Message

The Link Characteristics Request Message is sent by the server to the client when the server detects that a different set of transmission characteristics is necessary (or desired) for the

type of traffic that is flowing on the link. It is important to note that the link can be a logical link for a multicast session where more than one remote neighbor participates. The request contains either a Current Data Rate (CDR) TLV to request a different amount of bandwidth than what is currently allocated, a Latency TLV to request that traffic delay on the link not exceed the specified value, or both. A Link Characteristics ACK Message is required to complete the request. Implementations are free to define their retry heuristics in event of a timeout. Issuing a Link Characteristics Request with ONLY the MAC Address TLV is a mechanism a peer MAY use to request metrics (via the Link Characteristics ACK) from its partner.

The Link Characteristics Request Message contains the following fields:



Message Type - DLEP_MESSAGE (Value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to '0'.

Message Address Length - 0x0

Message Size - 31 + length of optional (Current Data Rate and/or Latency) Sub-TLVs

Message Sequence Number - A 16-bit unsigned integer field containing a sequence number generated by the message originator.

TLV Block - Length: 23 + optional Sub-TLVs

Sub TLVs

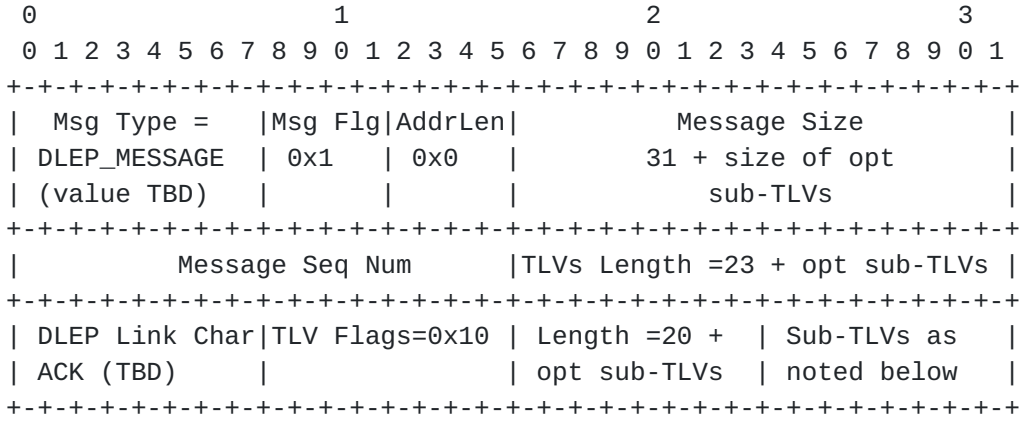
Identification Sub-TLV (MANDATORY)
MAC Address Sub-TLV (MANDATORY)
Current Data Rate Sub-TLV - if present,

this value represents the requested data rate in bits per second (bps). (OPTIONAL)
Latency TLV - if present, this value represents the maximum latency, in milliseconds, desired on the link. (OPTIONAL)

27. Link Characteristics ACK Message

The Link Characteristics ACK Message is sent by the client to the server letting the server know the success (or failure) of the requested change in link characteristics. The Link Characteristics ACK message SHOULD contain a complete set of metric TLVs. It MUST contain the same TLV types as the request. The values in the metric TLVs in the Link Characteristics ACK message MUST reflect the link characteristics after the request has been processed.

The Link Characteristics ACK Message contains the following fields:



Message Type - DLEP_MESSAGE (Value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to '0'.

Message Address Length - 0x0

Message Size - 31 + length of optional (Current Data Rate and/or Latency) TLVs

Message Sequence Number - A 16-bit unsigned integer field containing the sequence number that appeared on the corresponding Link Characteristics Request message.

TLV Block - TLVs Length = 23 + Optional TLVs

Sub TLVs

Identification Sub-TLV (MANDATORY)
MAC Address Sub-TLV (MANDATORY)
Maximum Data Rate Sub-TLV (OPTIONAL)

Current Data Rate Sub-TLV - if present, this value represents the NEW (or

unchanged, if the request is denied)
Current Data Rate in bits per second (bps).
(OPTIONAL)

Latency Sub-TLV - if present, this value represents the NEW maximum latency (or unchanged, if the request is denied), expressed in milliseconds, on the link. (OPTIONAL)

Resources Sub-TLV (OPTIONAL)

Relative Link Quality Sub-TLV (OPTIONAL)

28. Security Considerations

The protocol does not contain any mechanisms for security (e.g. authentication or encryption). The protocol assumes that any security would be implemented in the underlying transport (for example, by use of DTLS or some other mechanism), and is therefore outside the scope of this document.

29. IANA Considerations

This section specifies requests to IANA.

29.1 TLV Registrations

This specification defines:

- One TLV types which must be allocated from the 0-223 range of the "Assigned Message TLV Types" repository of [RFC5444].
- A new repository for DLEP orders, with seventeen values currently assigned.
- A new repository for DLEP Sub-TLV assignments with nineteen values currently assigned.

29.2 Expert Review: Evaluation Guidelines

For the registries for TLV type extensions where an Expert Review is required, the designated expert SHOULD take the same general recommendations into consideration as are specified by [RFC5444].

29.3 Message TLV Type Registration

The Message TLV specified below must be allocated from the "Message TLV Types" namespace of [RFC5444].

- DLEP_MESSAGE

29.4 DLEP Order Registration

A new repository must be created with the values of the DLEP orders. Valid orders are:

- Attached Peer Discovery Message
- Detached Peer Discovery Message
- Peer Offer Message
- Peer Update Message
- Peer Update ACK Message
- Peer Termination Message
- Peer Termination ACK Message
- Neighbor Up Message
- Neighbor Up ACK Message
- Neighbor Down Message
- Neighbor Down ACK Message
- Neighbor Update Message
- Neighbor Address Update Message
- Neighbor Address Update ACK Message
- Heartbeat Message
- Link Characteristics Request Message
- Link Characteristics ACK Message

This registry should be created according to the guidelines for 'Message-Type-Specific TLV' registration as specified in section 6.2.1 of [RFC5444].

29.5 DLEP Sub-TLV Type Registrations

A new repository for DLEP Sub-TLVs must be created. Valid Sub-TLVs are:

- Identification Sub-TLV
- DLEP Version Sub-TLV
- Peer Type Sub-TLV
- MAC Address Sub-TLV
- IPv4 Address Sub-TLV
- IPv6 Address Sub-TLV
- Maximum Data Rate Sub-TLV
- Current Data Rate Sub-TLV
- Latency Sub-TLV
- Expected Forwarding Time Sub-TLV
- Resources Sub-TLV
- Relative Link Quality Sub-TLV
- Status Sub-TLV
- Heartbeat Interval Sub-TLV
- Heartbeat Threshold Sub-TLV
- Link Characteristics ACK Timer Sub-TLV
- Credit Window Status Sub-TLV

- Credit Grant Sub-TLV
- Credit Request Sub-TLV

It is also requested that the registry allocation contain space reserved for experimental sub-TLVs.

30. Appendix A.

Peer Level Message Flows

*Modem Device (Client) Restarts Discovery

```
   Server                    Client   Message Description
   ================================================================

   <-------Peer Discovery---------    Client initiates discovery


    ---------Peer Offer----------->   Server detects a problem, sends
      w/ Non-zero Status TLV          Peer Offer w/ Status TLV indicating
                                      the error.

                                      Client accepts failure, restarts
                                      discovery process.

   <-------Peer Discovery---------    Client initiates discovery


    ---------Peer Offer----------->   Server accepts, sends Peer Offer
         w/ Zero Status TLV           w/ Status TLV indicating success.

                                      Discovery completed.
```

*Modem Device Detects Peer Offer Timeout

```
   Server                    Client   Message Description
   ================================================================

   <-------Peer Discovery---------    Client initiates discovery,
                                      starts a guard timer.

                                      Client guard timer expires.
                                      Client restarts discovery process.

    <-------Peer Discovery---------   Client initiates discovery,
                                      starts a guard timer.

    ---------Peer Offer----------->   Server accepts, sends Peer Offer
         w/ Zero Status TLV           w/ Status TLV indicating success.

                                      Discovery completed.
```

*Server Peer Offer Lost

Server	Client	Message Description
<-------Peer Discovery---------		Client initiates discovery, starts a guard timer.
---------Peer Offer-------||		Server offers availability
		Client times out on Peer Offer, restarts discovery process.
<-------Peer Discovery---------		Client initiates discovery
---------Peer Offer---------->		Server detects subsequent discovery, internally terminates the previous, accepts the new association, sends Peer Offer w/ Status TLV indicating success.
		Discovery completed.

*Discovery Success

Server	Client	Message Description
<-------Peer Discovery---------		Client initiates discovery
---------Peer Offer---------->		Server offers availability
-------Peer Heartbeat--------->		
<-------Peer Heartbeat---------		
-------Peer Heartbeat--------->		
<==============================>		Neighbor Sessions
<-------Peer Heartbeat---------		
-------Peer Heartbeat--------->		
--------Peer Term Req--------->		Terminate Request
<--------Peer Term Res---------		Terminate Response

*Server Detects a Heartbeat timeout

```
   Server                    Client   Message Description
   ========================================================================

   <-------Peer Heartbeat---------

    -------Peer Heartbeat--------->

      ||---Peer Heartbeat---------

           ~ ~ ~ ~ ~ ~ ~

    -------Peer Heartbeat--------->

      ||---Peer Heartbeat---------
                                    Server Heartbeat Timer expires,
                                    detects missing heartbeats. Server
                                    takes down all neighbor sessions
                                    and terminates the Peer association.

    ------Peer Terminate --------->   Peer Terminate Request

                                    Client takes down all neighbor
                                    sessions, then acknowledges the
                                    Peer Terminate

   <----Peer Terminate ACK---------   Peer Terminate ACK
```

*Client Detects a Heartbeat timeout

```
   Server                    Client   Message Description
   ========================================================================

   <-------Peer Heartbeat---------

    -------Peer Heartbeat------||

   <-------Peer Heartbeat---------

           ~ ~ ~ ~ ~ ~ ~

    -------Peer Heartbeat------||

   <-------Peer Heartbeat---------
                                    Client Heartbeat Timer expires,
                                    detects missing heartbeats. Modem
```

takes down all neighbor sessions
and terminates the Peer association.

```
    <-------Peer Terminate--------    Peer Terminate Request

                                      Server takes down all neighbor
                                      sessions, then acknowledges the
                                      Peer Terminate

    ------Peer Terminate ACK----->    Peer Terminate ACK
```

*Peer Terminate (from Client) Lost

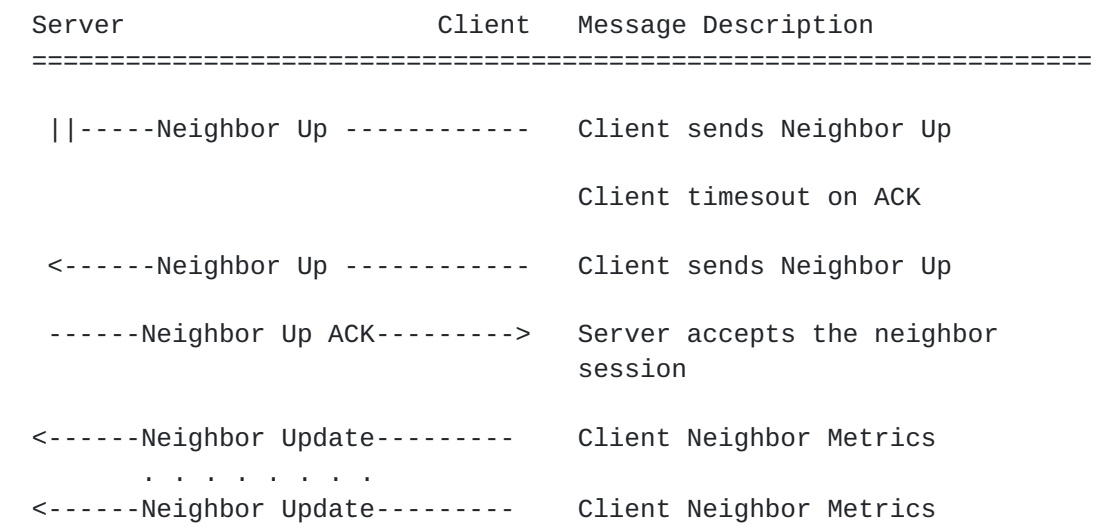
```
   Server                    Client   Message Description
   ================================================================

     ||------Peer Terminate--------   Client Peer Terminate Request

                                      Server Heartbeat times out,
                                      terminates association.

    --------Peer Terminate------->    Server Peer Terminate

    <-----Peer Terminate ACK------    Client sends Peer Terminate ACK
```

*Peer Terminate (from server) Lost

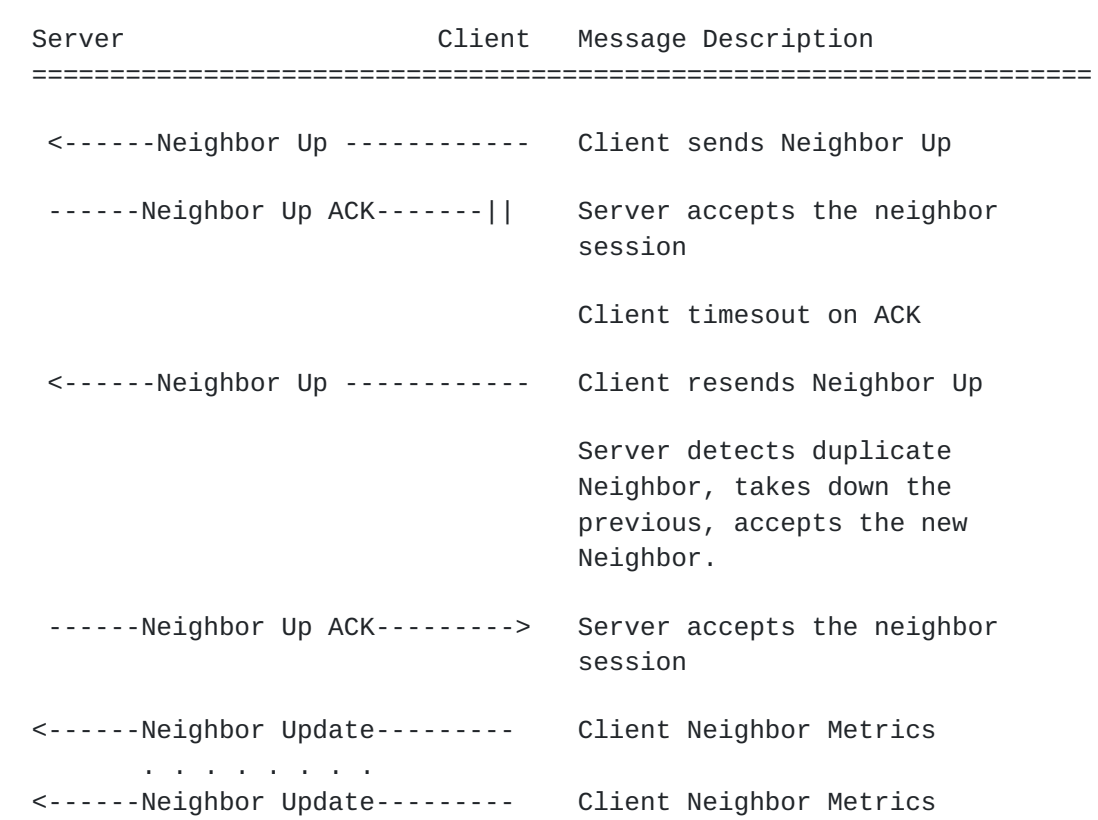
```
   Server                    Client   Message Description
   ================================================================

    -------Peer Terminate-------->    Server Peer Terminate Request

                                      Client HB times out,
                                      terminates association.

    <------Peer Terminate--------     Client Peer Terminate

    ------Peer Terminate ACK----->    Peer Terminate ACK
```

Neighbor Level Message Flows

*Client Neighbor Up Lost



*Server Detects Duplicate Neighbor Ups



*Neighbor Up, No Layer 3 Addresses

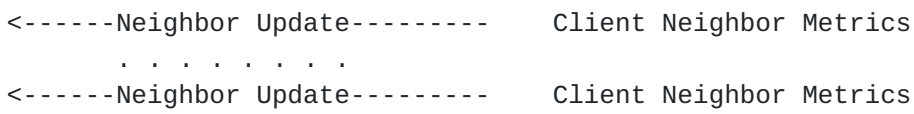
Server	Client	Message Description
<------Neighbor Up ------------		Client sends Neighbor Up
------Neighbor Up ACK--------->		Server accepts the neighbor session Server ARPs for IPv4 if defined. Server drives ND for IPv6 if defined.
<------Neighbor Update---------		Client Neighbor Metrics
. . . . .		
<------Neighbor Update---------		Client Neighbor Metrics

*Neighbor Up with IPv4, No IPv6

Server	Client	Message Description
<------Neighbor Up ------------		Client sends Neighbor Up with the IPv4 TLV
------Neighbor Up ACK--------->		Server accepts the neighbor session Server drives ND for IPv6 if defined.
<------Neighbor Update---------		Client Neighbor Metrics
. . . . .		
<------Neighbor Update---------		Client Neighbor Metrics

*Neighbor Up with IPv4 and IPv6

Server	Client	Message Description
<------Neighbor Up ------------		Client sends Neighbor Up with the IPv4 and IPv6 TLVs
------Neighbor Up ACK--------->		Server accepts the neighbor session


<------Neighbor Update--------- Client Neighbor Metrics
. . . . . .
<------Neighbor Update--------- Client Neighbor Metrics

*Neighbor Session Success

```
   Server                    Client   Message Description
   ========================================================================

    ---------Peer Offer----------->   Server offers availability

    -------Peer Heartbeat--------->


   <------Neighbor Up -----------      Client

    ------Neighbor Up ACK-------->     Server

   <------Neighbor Update---------     Client
          . . . . . . . .
   <------Neighbor Update---------     Client

                                       Client initiates the terminate

   <------Neighbor Down ----------     Client

    ------Neighbor Down ACK------->    Server

                                       or

                                       Server initiates the terminate

    ------Neighbor Down ---------->    Server

   <------Neighbor Down ACK-------     Client
```

Acknowledgements

The authors would like to acknowledge the influence and contributions of Chris Olsen, Teco Boot, Subir Das, Jaewon Kang, Vikram Kaul, Rick Taylor, and John Dowdell.

Normative References

[RFC5444] Clausen, T., Ed,. "Generalized Mobile Ad Hoc Network (MANET) Packet/Message Format", RFC 5444, Februar, 2009.

[RFC5578] Berry, B., Ed., "PPPoE with Credit Flow and Metrics", RFC 5578, February 2010.

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

Informative References

[DTLS] Rescorla, E., Ed,. "Datagram Transport Layer Security", RFC 4347, April 2006.

Author's Addresses

Stan Ratliff
Cisco
170 West Tasman Drive
San Jose, CA 95134
USA
EMail: sratliff@cisco.com

Bo Berry
Cisco
170 West Tasman Drive
San Jose, CA 95134
USA
EMail: boberry@cisco.com

Greg Harrison
Cisco
170 West Tasman Drive
San Jose, CA 95134
USA
EMail: greharri@cisco.com

Shawn Jury
NetApp
7301 Kit Creek Road, Building 2
Research Triangle Park, NC 27709
USA
Email: shawn.jury@netapp.com

Darryl Satterwhite
Cisco
170 West Tasman Drive
San Jose, CA 95134
USA
Email: dsatterw@cisco.com