Evaluating an LLDPv2 proposal against LSVR Topology Discovery Requirements

Paul Congdon

Paul Bottroff

Paul Unbehagen

Randy Bush

($P^3 + R = Y$)

Background

- What is LSVR?
  - IETF Working Group doing Link State Vector Routing, combining link-state and path-vector routing mechanisms (i.e. BGP-SPF) for massive scale data centers.
  - https://datatracker.ietf.org/wg/lsvr/about/
- LSVR needs a topology discovery and liveness protocol for BGP-SPF with ability to exchange more data than can fit into a single PDU
- LSVR recently adopted LSoE as a Working Group document to define such a protocol
  - https://datatracker.ietf.org/doc/draft-ietf-lsvr-lsoe/
- This presentation attempts to summarize the requirements document and information exchanged by LSoE to see if the proposed LLDPv2 could support the needs.
  - https://tools.ietf.org/html/draft-ymbk-lsvr-discovery-req-01
  - https://mentor.ieee.org/802.1/dcn/18/1-18-0071-02-ICne-ieee802-ietf-workshop-network-discovery.pdf

Brief Review of LLDPv2 Proposal

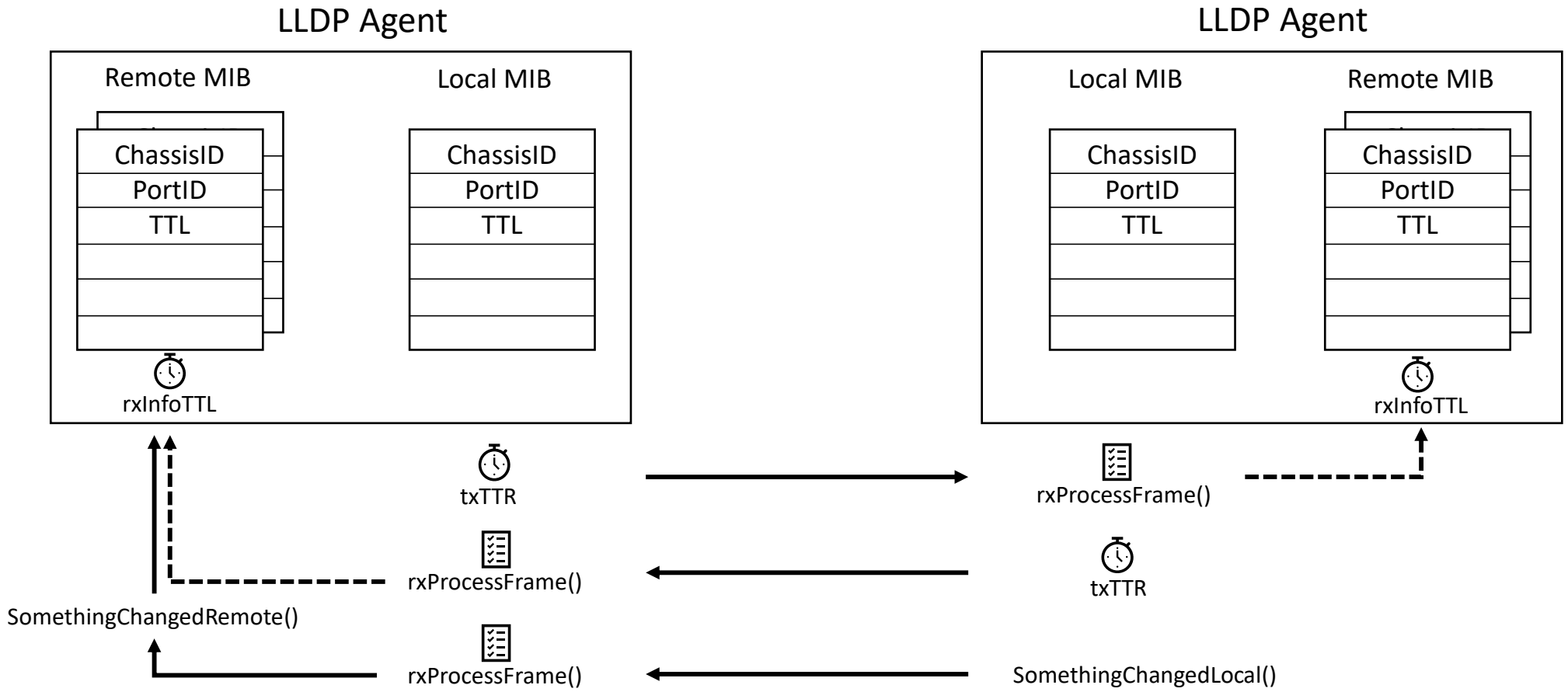
- Initially presented on 1/7/2019 at TSN call: http://www.ieee802.org/1/files/public/docs2019/new-congdon-lldpv2-consideration-0119-v01.pdf

- Why do we need to update LLDP?
  - LLDP is widely deployed in many environments
  - The number of TLVs sent in LLDPDUs continues to grow
    - New standards continue to defined new objects
    - A large number of Vendor Specific TLVs
  - Alternative protocols are being proposed to get around the single PDU size limit
  - Relying on Jumbo frames to support more TLVs is problematic in many environments
  - Summary: We need to be able to exchange more TLVs. - LSVR requirements are just one example use case where this is needed.

Objectives for a new version

- Support the ability to send more than 1 PDUs worth of TLVs
- Support the ability to communicate with an LLDPv1 implementation (only the first PDUs worth of TLVs).
- Ensure the integrity of the full set of TLVs is received by partner
  - NOTE: This can be useful in v1 implementations as well
- Consider if there are other optimizations to address
  - E.g. Less frequent updates
  - E.g. New reachability addresses (Nearest-station or Nearest-Router)
  - E.g. allow larger TLVs and/or the ability of the contents to span multiple extension PDUs

Current LLDP operation reminder

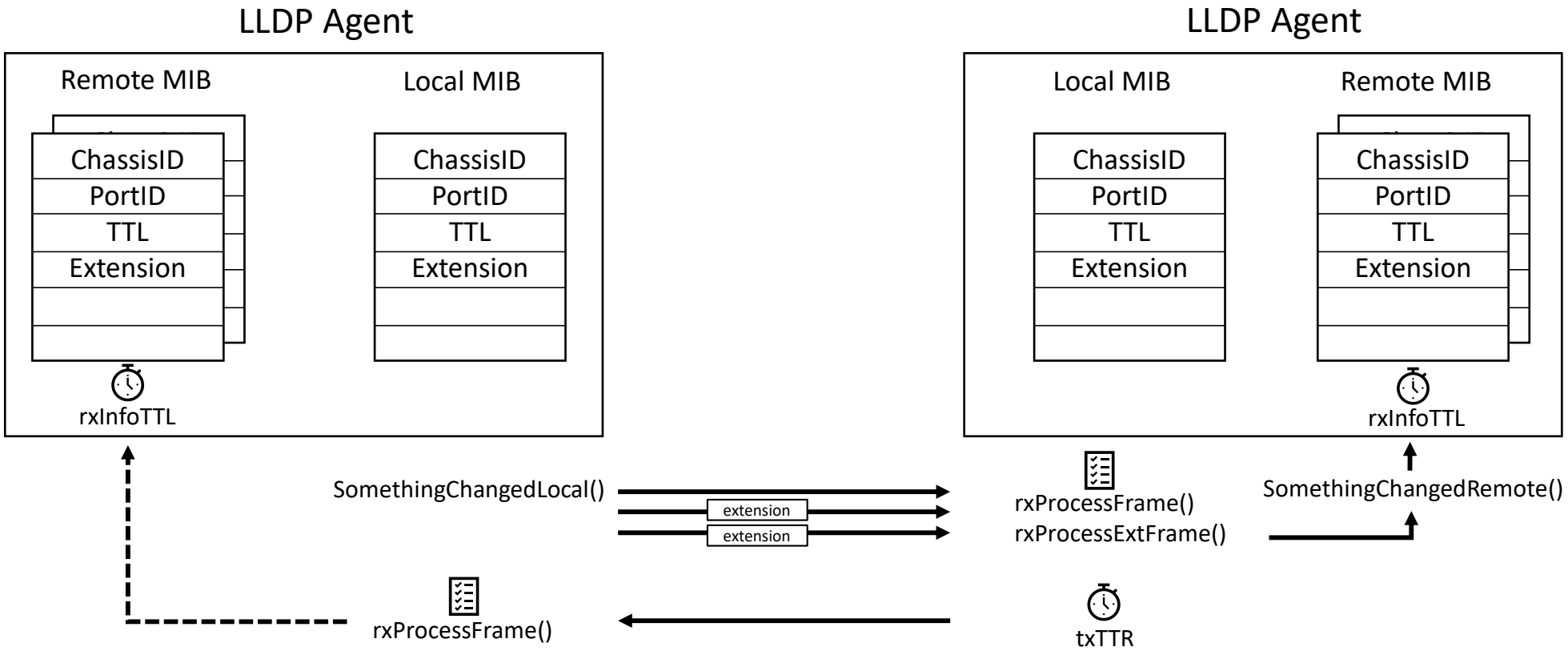


NOTE: Think of the Remote and Local MIBs as a database that must fit into a single PDU
Replace all values of the Remote MIB with contents of LLDPDU when something changes

Proposal

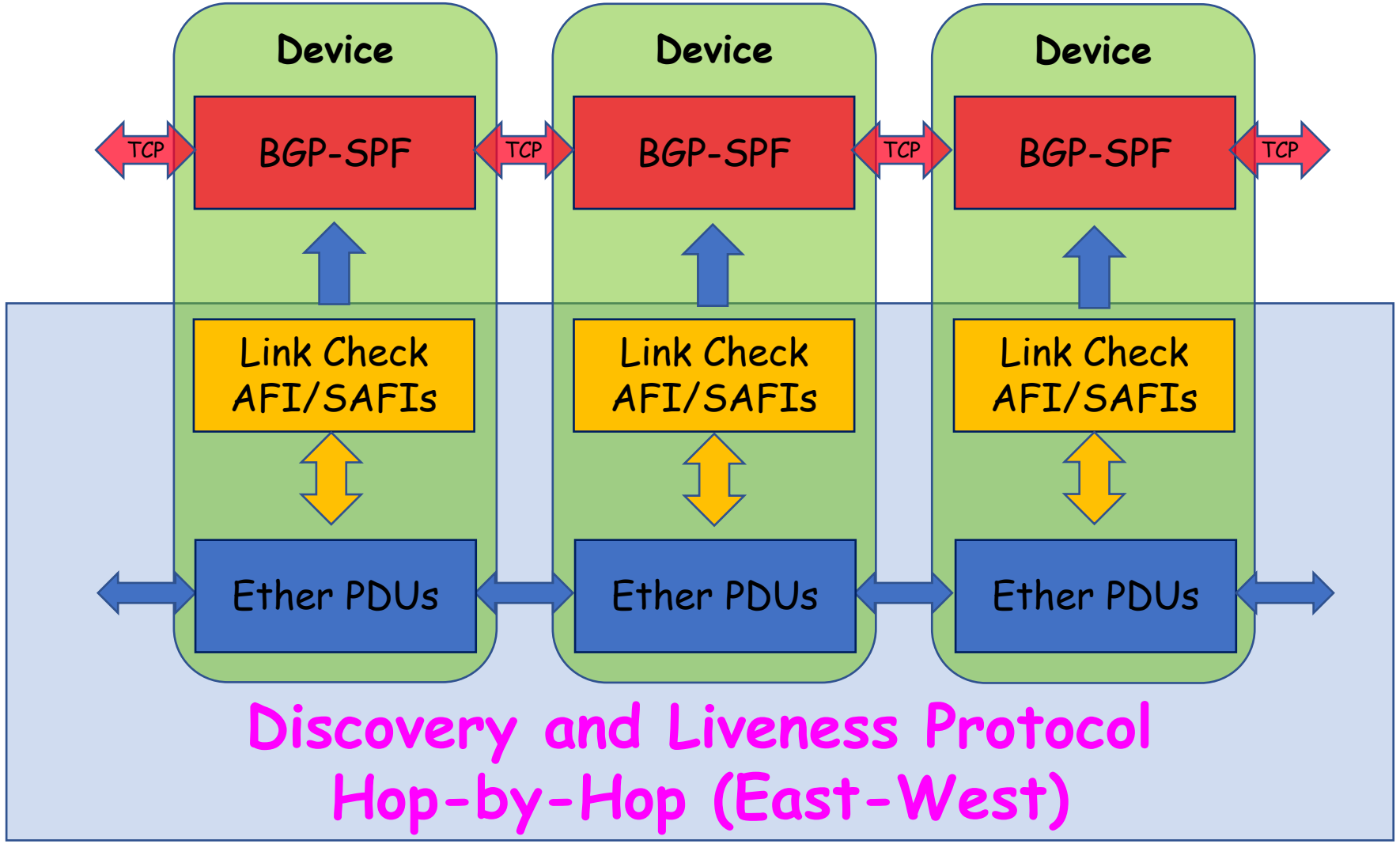
- Define a new mandatory (for v2 implementations) TLV that appears just after the current mandatory set of 3 TLVs.
  - ChassisID TLV + PortID TLV + TTL TLV + (new) ExtensionPDU TLV
- In the new TLV, define a vector that specifies:
  - The number of extension PDUs to be sent
  - An identity of each PDU (e.g. hash, checksum, version number or PDU number)
  - Acknowledges the receipt of partner extension PDUs
- The first v2 PDU looks like a standard v1 PDU with the extra ExtensionPDU TLV (i.e. will be received by v1 implementations).
- The new extension PDUs need to be ignored by v1 LLDP in a non-intrusive way. Options:
  - Force an error in the v2 PDUs – will cause error counters to increment
  - Use a new Ethertype for v2 extension PDUs - preferred
- The new PDUs need to have a mandatory format as well:
  - Includes at least the first two mandatory TLVs of a v1 PDU (ChassisID + PortID)
  - Includes new TLV that identifies the extension PDU.
- Optimizations:
  - There is no need to resend extension PDUs if nothing has changed, unless a previous extension PDU was not correctly received.
  - Only periodically send the 1st PDU.
  - TTL in 1st PDU relates to all extension PDUs.
- NOTE: The maximum size of a TLV defines the maximum number of extension PDUs that can be included. (depends on identity field)

Proposed LLDPv2 Operation



NOTE: Send primary LLDPDU and all extension PDU when something changes locally or retransmission is needed.
If extension data has NOT changed, no need to send anything other than the primary LDPDU

LSVR Discovery and Liveness Needs



Things LSVR needs to discovery and exchange

- From https://tools.ietf.org/html/draft-ymbk-lsvr-discovery-req-01
  - Node Identity
  - Link Identity
  - L2 Liveness
  - Encapsulations
  - Addresses
- Additional information exchanged by LSoE from https://datatracker.ietf.org/doc/draft-ietf-lsvr-lsoe/
  - Neighbor MAC address
  - Node Attributes
  - Authentication Data

Requirement – Node Identity

- Each node in the topology must have an identity/identifier which must be unique in the topology.
- The identity might be
  - an ASN with high order bits zero
  - a classic RouterID with high order bits zero
  - a catenation of the two
  - a 80-bit ISO System-ID
  - or any other identifier unique to a single device in the current routing space
- Exchanged in Open message of LSoE

Requirement – Link Identity

- A link is between two nodes. Each end of a link is a node/device interface. Each link in the topology must be uniquely identified and the identities of the nodes on the link must be identified.
- A link is identified when two peer devices have compatible Encapsulations and addresses, i.e. the same AFI/SAFI and the same subnet.
- LSoE link discovery is derived from Encapsulation and Address discovery in LSoE Address Announcement TLVs.

Requirement – L2 Liveness

- Because adjacencies and topology changes must be quickly detected, Layer-2 stability of each link should be monitored and reported.
  NOTE: this is in addition to L3 liveness via BFD
- LSoE maintains liveness via the periodic KEEPALIVE message. Default frequency is 1 second
- NOTE: May not be a good fit for LLDP

Requirement - Encapsulations

- The encapsulation(s) (IPv4, IPv6, ...) on each link must be known. One or more of the common AFI/SAFIs must be supported on each link, IPv4, IPv6, MPLS, etc. NOTE: It is assumed that the set of encapsulations is the same across the entire topology.
- LSoE exchanges Encapsulation Addresses in IPv4 Announcement, IPv6 Announcement, MPLS IPv4 Announcement and MPLS IPv6 Announcement TLVs

Requirement - Addresses

- The available addresses on the node interfaces for each encapsulation must be known. More than one address for an encapsulation must be supported.
- LSoE exchanges Encapsulation Addresses in IPv4 Announcement, IPv6 Announcement, MPLS IPv4 Announcement and MPLS IPv6 Announcement TLVs

Additional info exchanged by LSoE

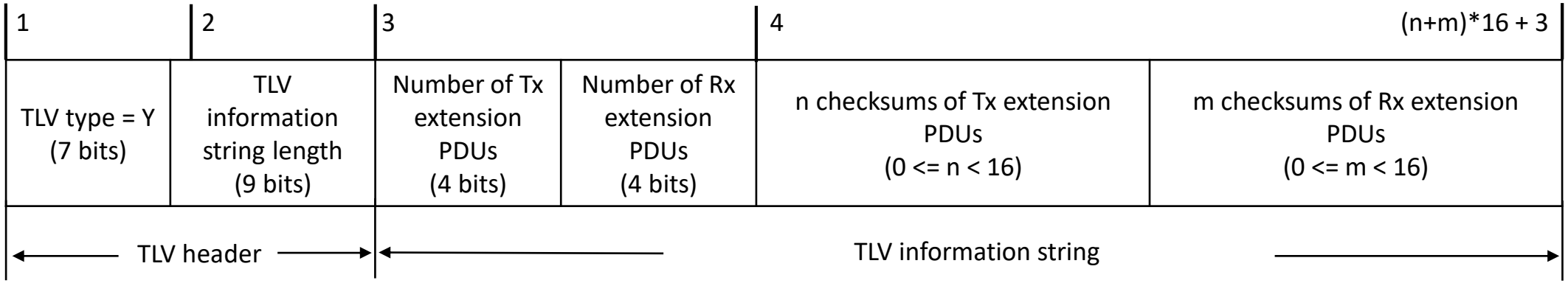
- Peer MAC address
  - Exchanged in LSoE Hello message
- Node Attributes
  - Byte array of locally defined attributes such as role or position in the topology (up to the Operator, i.e. no global registry).
  - Exchanged in LSoE Open message.
- Authentication Data
  - Work in progress
  - Currently defined as locally specific to Operator environment
  - Likely to evolve into some signed blob to authenticate peer.
  - NOTE: May be larger than what can fit in a single LLDP TLV

Summary of ability to use LLDP for LSoE exchange

Information Exchanged	Supportable by an LLDP TLV and Protocol	Comments
MyMac Address	☑	
Local ID / Remote ID	☑	
Attribute List	☑	
Authentication Data	?	May exceed current LLDP TLV length restrictions
Encapsulation & Addresses	☑	Must be split across multiple TLVs & PDUs (as with LSoE)
Keepalives	☒	Frequency may not be appropriate for existing LLDP
Acks	☑	Implicit part of LLDPv2 proposal

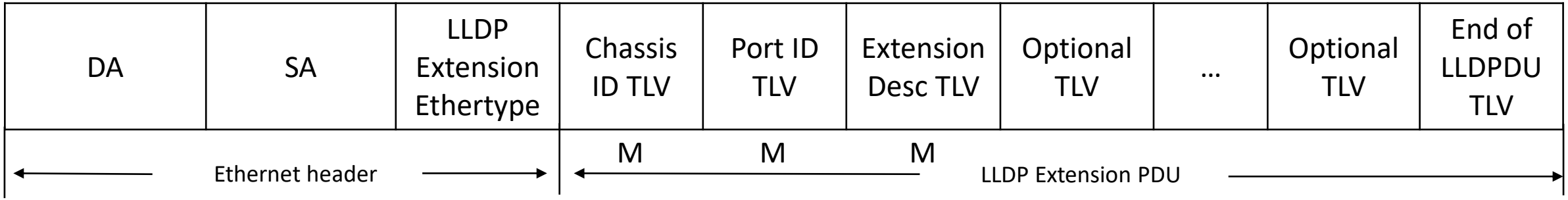
Backup

Example Extension TLV



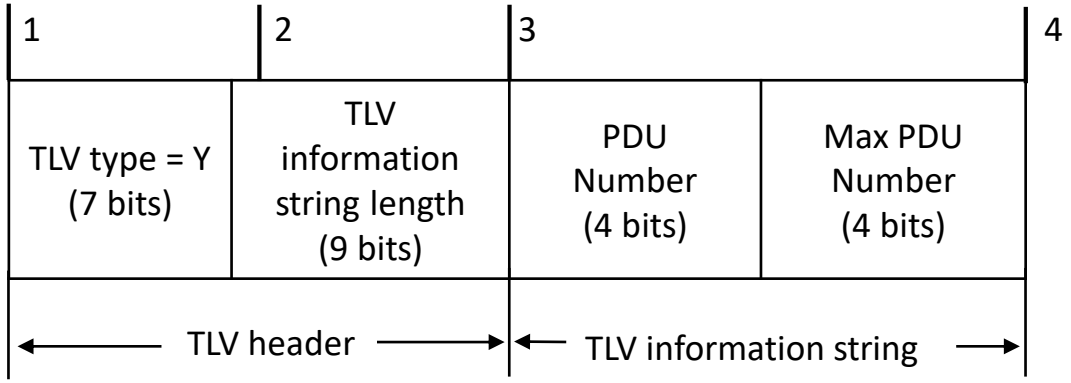
- TLV Type
  - probably use the next reserved type (i.e. 9)
- Number of Tx and Rx extension PDUs
  - If using Checksum of 16 bytes, can only pack 30 sums into a TLV
- Checksums
  - Should cover the entire extension LLDPDU

Example Extension PDU



- LLDP Extension Ethertype
  - New Ethertype allows LLDPv1 implementations to ignore these frames
- Chassis ID + Port ID are mandatory
  - Note TTL from 1st PDU should apply and is not needed here
- Extension Description TLV
  - Numbers the extension PDU in the sequence of all extension PDUs

Example Extension Description TLV



- TLV Type
  - Another new base TLV type (i.e. 10)
- PDU Number and Max PDU Number
  - For example PDU 1 of 5

Questions / comments / TBDs

- How to define the extension PDU TLV?
  - It needs to contain a vector of information for all extension PDUs
  - It needs to acknowledge received extension PDUs.
  - The smaller the identity field, the more extension PDUs that can be supported (e.g. CRC-16 or SHA-256 Hash?)
  - We could define two extension TLVs – one for Tx and one for Rx to support more extension PDUs
  - Should the SHA-256 Hash cover all PDUs or individual?
  - Should we allow a single TLV to span multiple PDUs (e.g. like IP fragments)?
- When to send the 1st PDU as an ACK of received extension PDUs?
  - Need a final bit in the extension PDUs or a PDU number scheme?
  - Define another End of LLDPDU TLV?
- Retransmission strategy? SACK or just retransmit the entire sequence?