IEEE P802.1ABdh Update to LSVP
(Note: P802.1ABdh == LLDPv2)

IETF-106

Singapore

Paul Congdon (Tallac Networks)

Paul Bottorff (Aruba)

November 19, 2019

Disclaimer

- This presentation should be considered as the personal view of the presenter not as a formal position, explanation, or interpretation of IEEE.

- Per IEEE-SA Standards Board Bylaws, December 2017
  - “At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that his or her views should be considered the personal views of that individual rather than the formal position of IEEE.”

Background - References

- Previous update at IETF-105 with previous background – July 2019
  - https://datatracker.ietf.org/meeting/105/materials/slides-105-lsvr-2-ieee-lldpv2-update
- IEEE 802 approval to start P802.1ABdh – Standard for Local and Metropolitan Area Networks - Station and Media Access Control Connectivity Discovery Amendment: Support for Multiframe Protocol Data Units – September 2019
  - https://standards.ieee.org/project/802_1ABdh.html
- Most recent technical proposal was presented in September 2019
  - http://www.ieee802.org/1/files/public/docs2019/dh-bottorff-alt-0919-v4.pdf
- Draft to define IETF TLVs for LSVR intended to be carried by LLDPv2
  - https://tools.ietf.org/html/draft-congdon-lsvr-lldp-tlvs-00
- Call for Participation press release by IEEE 802 - TBD

Technical changes since IETF-105 update

- Terminology and definitions
- Manifest definition changes to support larger databases
- Review of Shared Media worst case scenarios (see IEEE contribution)

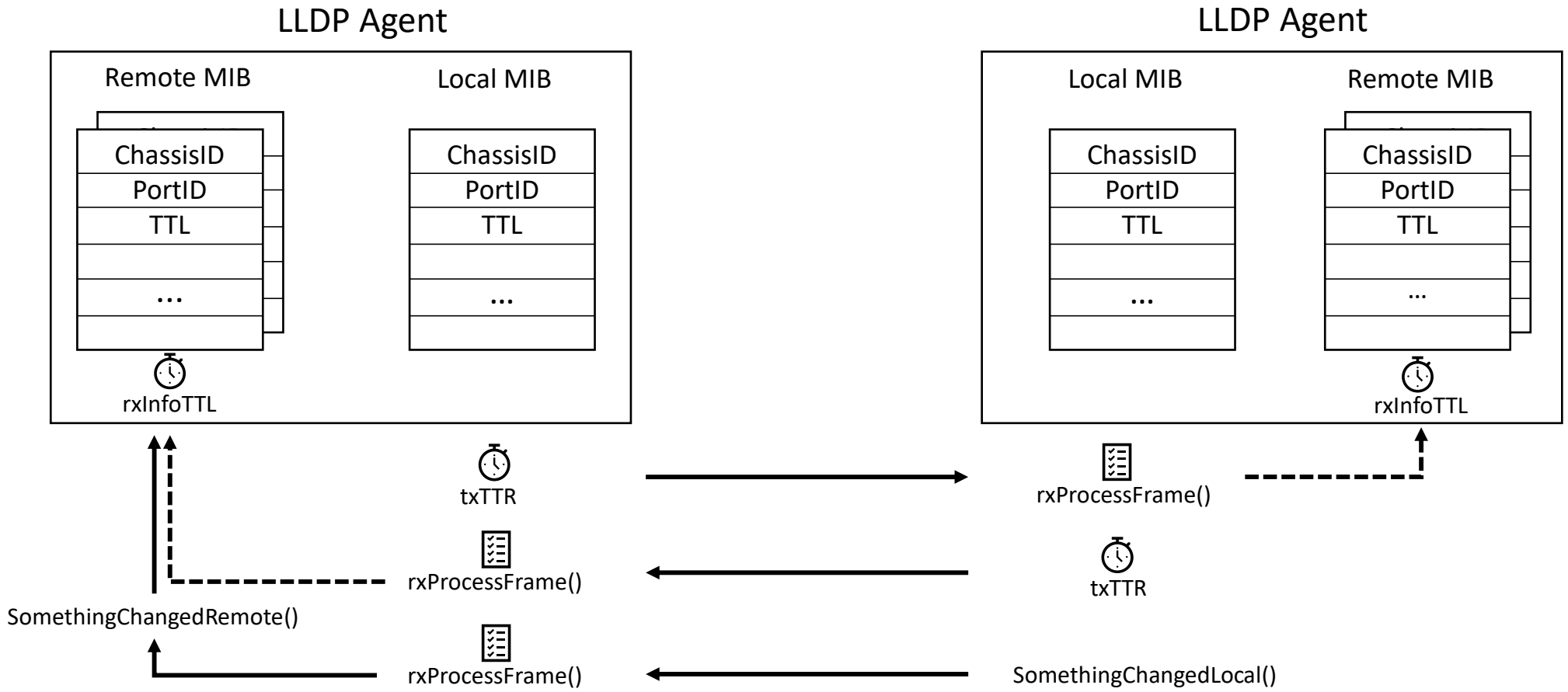
Objectives for New LLDPv2 Method

- Support LLDP databases larger than a single frame
  - Optimize LLDPv2 for databases around 100K bytes
  - For reference IETF currently believes database sizes around 64K bytes are sufficient
- Support the ability to limit the LLDP frame size to meet timing constraints imposed by some TSN applications
  - Do we need to split TLVs over multiple PDUs?
  - How big do these databases need to be?
- Support the ability to communicate with an LLDPv1 implementation
  - Only the LLDPv1 database would be exchanged between and LLDPv1 and LLDPv2 implementation
- Support shared media, optimize for point-to-point though allows shared
  - Duplicate MAC addressing should be handled by the extension protocol
- Ensure the integrity of the full set of TLVs received by partners
  - Do we also need to provide a means to authenticate the LLDP database? The IETF has this requirement.

Objectives for New LLDPv2 Method

- Support pacing of PDUs to receivers to prevent overloading low level network firmware
  - Historically OSPF and IS-IS have had problems from lack of flow and congestion management
- Reduce network traffic by reducing periodic transmission to the minimum
  - Only update the foundation LLDPv1 PDU periodically
  - Extension PDUs are only transmitted/updated on demand from receivers
  - Update extension PDUs only when they have changed
- Other optimizations and considerations which might be useful
  - Computational load requirements for LLDPv2 receivers to update and validate PDUs
  - Larger TLVs or is using multiple TLVs appears sufficient
  - TLVs spanning multiple extension database PDUs, is this required for TSN
  - Database authentication, is high want for IETF and other applications
    - Part of separate authentication extension
    - Key exchange requirements

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: Foundation PDU (F-PDU)

- The current LLDPv1 PDU with a Manifest TLV is the foundation PDU (F-PDU)
- The foundation PDU is exchanged using the existing LLDPv1 protocol without modifications
- All databases are created as LLDPv1 databases, no extension PDUs create new databases
- An extended LLDP database is composed of the foundation PDU and n-1 extension PDUs
- A manifest TLV placed in the LLDPv1 foundation PDU identifies all extension PDUs
- If no manifest TLV is present in the foundation PDU then no extension PDUs exist for the LLDP database
- The upper limit to the number of PDUs is determined by the LLDPv1 TLV size limit (512) and the format of the manifest TLV
  - Note: When we have a small max PDU size the manifest TLV size can be further limited resulting in limiting the database size
- The manifest TLV carries an identifier for each extension PDU
- Any change in an extension PDU is reflected as a change in the manifest TLV, therefore any change in an extension PDU will result in a change to the foundation PDU

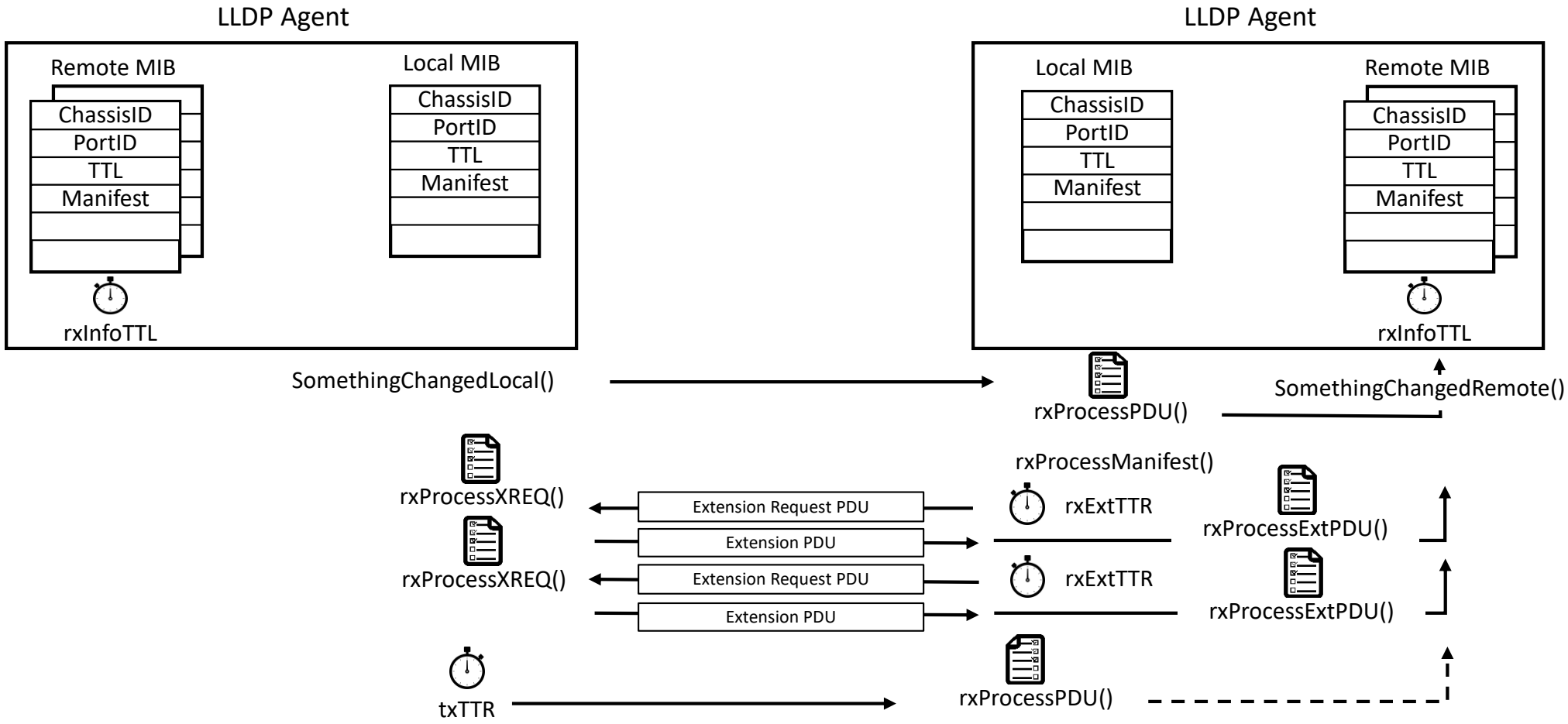
Proposal: Extension PDUs (X-PDUs)

- The extension LLDPDU will be ignored by LLDPv1
  - An alternate Ethertype is used for LLDPv2 PDUs to guarantee PDUs are never directed to LLDPv1
- Each extension PDU has three mandatory TLVs in the beginning of the PDU:
  - Each extension PDU contains the first two mandatory TLVs of a LLDPDUv1 (ChassisID + PortID)
  - Each extension PDU contains a new extension TLV that identifies the PDU
  - Before an extension PDU is added to a database it's {ChassisID, PortID, ExtensionID} must match the manifest TLV
- Each extension PDU is transmitted as a unicast in response to a receiver request
  - Extension PDUs are only transmitted in response to requests
  - The DA of an Extension PDU is the SA of the request
- The TTL in foundation PDU relates to all extension PDUs

Proposal: Extension Request PDU (XREQ-PDU)

- The extension Request PDU will be ignored by LLDPv1
  - An alternate Ethertype is used for LLDPv2 PDUs to guarantee PDUs are never directed to LLDPv1
- Each extension request PDU has three mandatory TLVs in the beginning of the PDU:
  - Each contains the first two mandatory TLVs of a LLDPDU (ChassisID + PortID)
    - However the ChassisID and PortID are for the destination rather than the source
  - Each contains a new extension request TLV that identifies the PDUs a list of extension PDU to be transmitted
- An extension request (XREQ-PDU) is sent between peers to request transmission of an extension PDU
  - The LLDP extension protocol supports multiple peers on a shared media
  - Transmission of X-PDUs is only in response to an XREQ-PDU generated by the receiving system
  - A receivers requests X-PDU transmission when it determines the current X-PDU does not match the manifest TLV
  - Receivers can have only a single XREQ-PDU pending at a time
  - A single XREQ-PDU can request transmission of multiple X-PDUs
  - The receiver controls the transmission rate by controlling the number of X-PDUs requested and the timing between XREQ-PDUs
  - Receivers time out the requested X-PDU responses
  - Transmitters periodically send the foundation F-PDU which can update the manifest TLV in turn resulting XREQs for X-PDUs
- Each extension request PDU is transmitted as a unicast
  - The DA of an extension request PDU is the SA of the foundation PDU

LLDP Extension Operation Proposal: Receiver Pacing



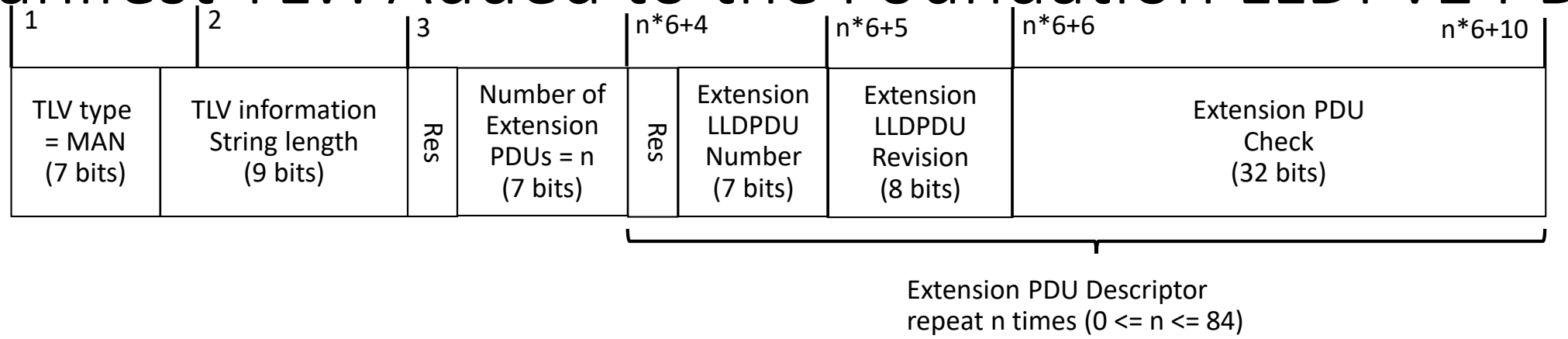
NOTE: Send LLDPDU as specified by LLDPv1 when something changes and periodically
Only send extension LLDPDU when explicitly requested by a XREQ
Only issue XREQ when manifest shows the local copy is out of date

LLDPv2 Project (P802.1ABdh) Next Steps

- Continued technical contributions
  - Would love to have an Open Source implementation for evaluation
- Review areas of change to IEEE Std IEEE 802.1AB-2016 (aka LLDP)
- Initial draft by an individual contributor
- Assign editor in 802.1 Working Group

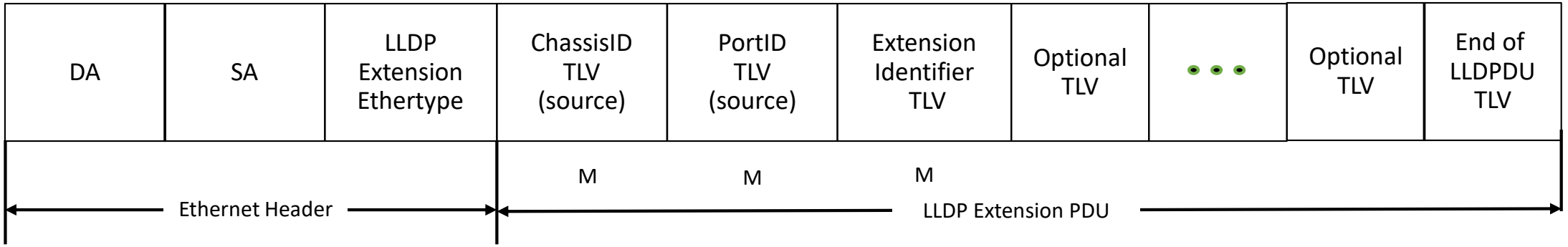
Backup - Details

Manifest TLV: Added to the Foundation LLDPv1 PDU



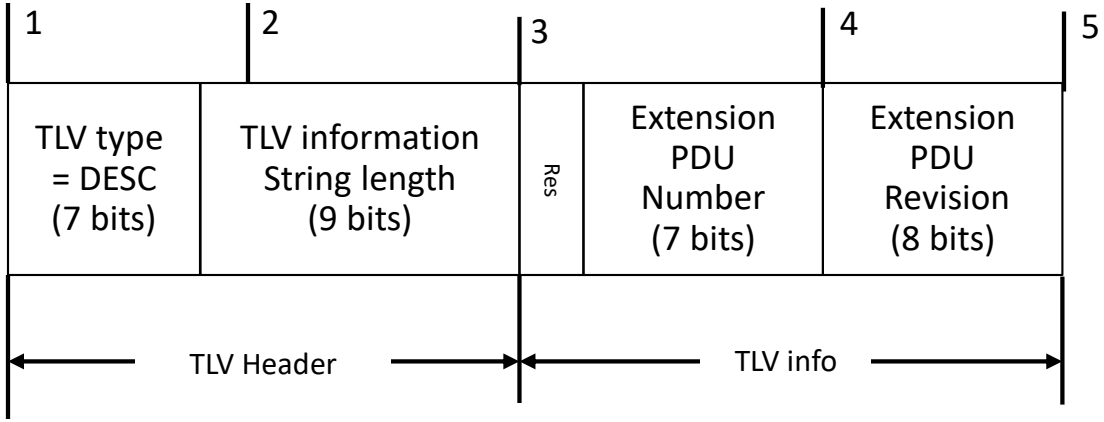
- Number of extension PDUs indicates the number of valid PDU descriptors in the manifest
  - Some implementations may fix the manifest TLV size however load it with a variable number of PDUs
  - If we don't need to hold the manifest TLV size constant, then the TLV length is sufficient to determine the number of manifest entries
- Each Extension PDU is identified by a:
  - Extension LLDPDU number, this number is included in the manifest to facilitate PDU deletion and insertion
  - Extension LLDPDU revision, updated modulo 256 on every change to the extension LLDPDU
  - Extension LLDPDU check: for example 32 bits of MD5

Format for LLDP Extension PDUs (X-PDU)



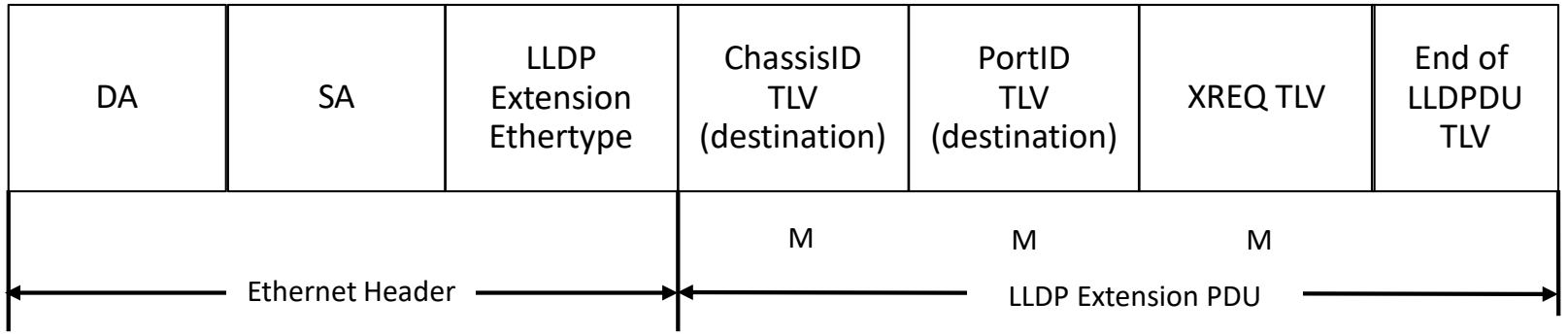
- LLDPv2 Ethertype
  - New LLDPv2 Ethertype for Extension PDUs prevents conflict with LLDPv1
  - Extension PDUs are identified by the presence of the Extension Desc TLV
  - Since extensions are not multicast and only delivered on request no new Ethertype is required, though one could be used if desired
- Chassis ID + Port ID are mandatory
  - The Chassid ID and Port ID of the PDU source
  - Note TTL from 1st PDU should apply and is not needed here
- Extension Identifier TLV is mandatory and must be the third TLV
  - Identifies this Extension PDU, the PDU revision

Extension PDU Identifier TLV (XID TLV):



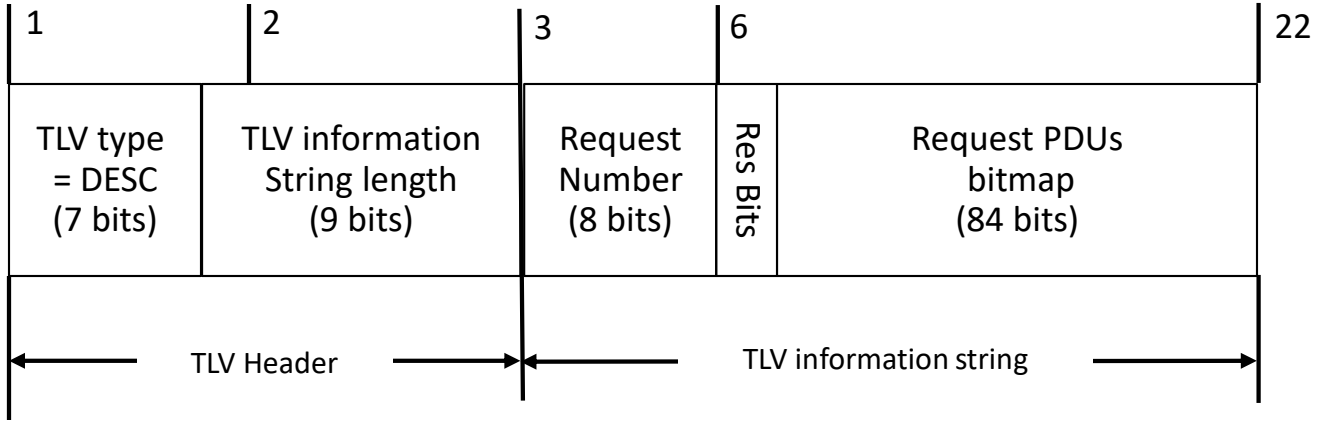
- Extension PDU Number is the designation number for this PDU
  - The PDU number is in the range from 1 – 84
  - Matched to the manifest extension PDU number
- Extension PDU revision number
  - Incremented modulo 256 whenever the extension LLDPDU is changed
  - Matched to the manifest to guarantee the extension LLDPDU is the one represented in the manifest
- Note the extension PDU check code is not carried in the Extension TLV and so must be calculated to match the manifest check code

Request For Extension PDUs (XREQ-PDU)



- LLDP Extension Ethertype
  - New LLDP Ethertype for Extension PDUs to prevent conflict with LLDPv1 implementations
- ChasssisID and PortID TLVs are mandatory in a Request for Extension PDU
  - ChassisID is the first and PortID is the second TLV in the PDU
  - Unlike a standard LLDPDU the ChassisID and PortID identify the destination not the source
- Extension Request TLV is mandatory in a Request for Extension PDU
  - The Extension Request TLV is the third TLV in the PDU
  - Request PDUs are identified by the presence of the Request for Extension TLV

Extension Request PDUs TLV (XREQ TLV)



- Extension Request PDUs
  - A given chassis/port may only have a single XREQ TLV pending at a time
  - Multiple XREQs PDUs may be used to pace the PDUs at the receiver by withholding XREQs
  - A single XREQ PDU may request multiple Extension PDUs if the receiver has sufficient buffer for them
  - The bit map is used to identify the list of Extension LLDPDUs by number
    - The index to the bit map identifies the Extension LLDPDU number
- Extension LLDPDUs are not multicast, instead they are unicast
  - The extension LLDPDUs are sent to the SA address within the foundation LLDPDU
  - On a shared media each individual LLDP Agent must provide independent requests for extension frames
  - This allows the individual receivers to pace PDUs at rates that match their ability to handle the reception