Authorization

April 3, 2024

Disclaimer

- The information in this presentation represents a snapshot of work in progress within the DMTF.
- This information is subject to change without notice. The standard specifications remain the normative reference for all information.
- For additional information, see the DMTF website.
- This information is a summary of the information that will appear in the specifications. See the specifications for further details.

Assumptions

- This presentation makes the following assumptions
  - The endpoints in discussion communicate using SPDM (DSP0274) and SPDM Secured Messages (DSP0277)
    - Communication can use any transport that supports the above commands
  - To bootstrap Authorization, there needs to be a provisioning step for initial credential
  - Definition of Policy profiles is out of scope for the Authorization specification

Background

- Many features or functionality coming into SPDM 1.2 and later need authorization.
  - Multiple standards bodies are attempting to address authorization.
- Goal
  - Address authorization uniformly across PMCI, DMTF alliance partners and industry.
  - Make it easily leveraged by other standards

Authorization

- Definition:
  - Determining if the requesting entity has the appropriate privileges to perform protected actions. If yes, to allow them to perform those protected actions.
- Scope:
  - Provide a general mechanism for any use case (e.g., SPDM, PLDM, other present and future PMCI WG use cases, alliance partners, industry) to perform authorization.
    - Examples:
      - PLDM FW Update
      - PLDM Type 2 (ex. Configuring event receiver, configuring sensor thresholds)
      - PLDM Type 6 (get vs patch/post i.e RDE)
      - SPDM Set Certs (and other future “set” commands).

Authorization Use Cases (non-exhaustive)

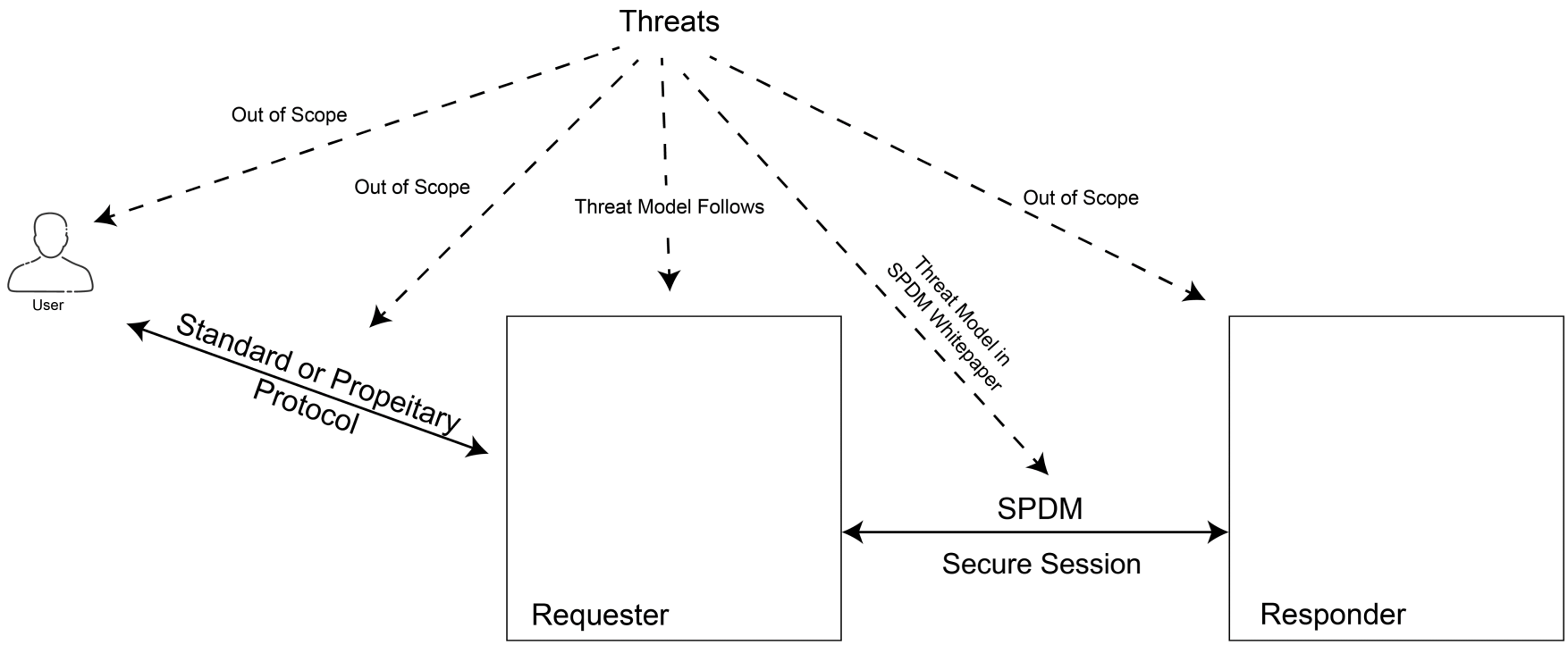
- Ownership Use case
  - Cert slot 0 owned and authorized by manufacturer, slot 1 by platform vendor
- Authorization of firmware update or destructive erase operations
  - Policy can restrict use of certain commands to a list of assigned credentials
- Sophisticated Datacenter Use Case
  - Authorize security and management actions (ex. FW update, crypto alg, Cert store changes) in an environment with multiple privilege domains (sec admin, sys admin etc)
- Service/Equipment/Cloud Provider Use Case
  - RMA Authorization to clear credentials/data, Reprovision
  - Authorize Device Ownership Transfer
- SPDM Endpoint Transfer of Ownership
  - Plug in cards – change security policy based on destination, clean credentials/data

High Level Architectural Components

- Authorization Flow
  - Use SPDM Sessions between Requester/Responder pair (simplifies supported options, baseline security)
  - Specify how to authorize generic messages
- Credential and Policy Management
  - Types of Credentials
    - Asymmetric Key Pair (Focus of initial release)
    - Symmetric Key (postponed)
      - Password/Passphrases (no usernames)
    - Username/Passwords (postponed)
  - Credential and Credential Policy
    - Standardize provisioning of credentials and associating them with their authorization policy
    - Authorization policy itself should be specified by the protocol leveraging this authorization specification

Authorization Threat Model Overview

- 2 Use Cases – the User is the authorized actor
  - SPDM Requester represents user
  - SPDM Requester is a proxy for user



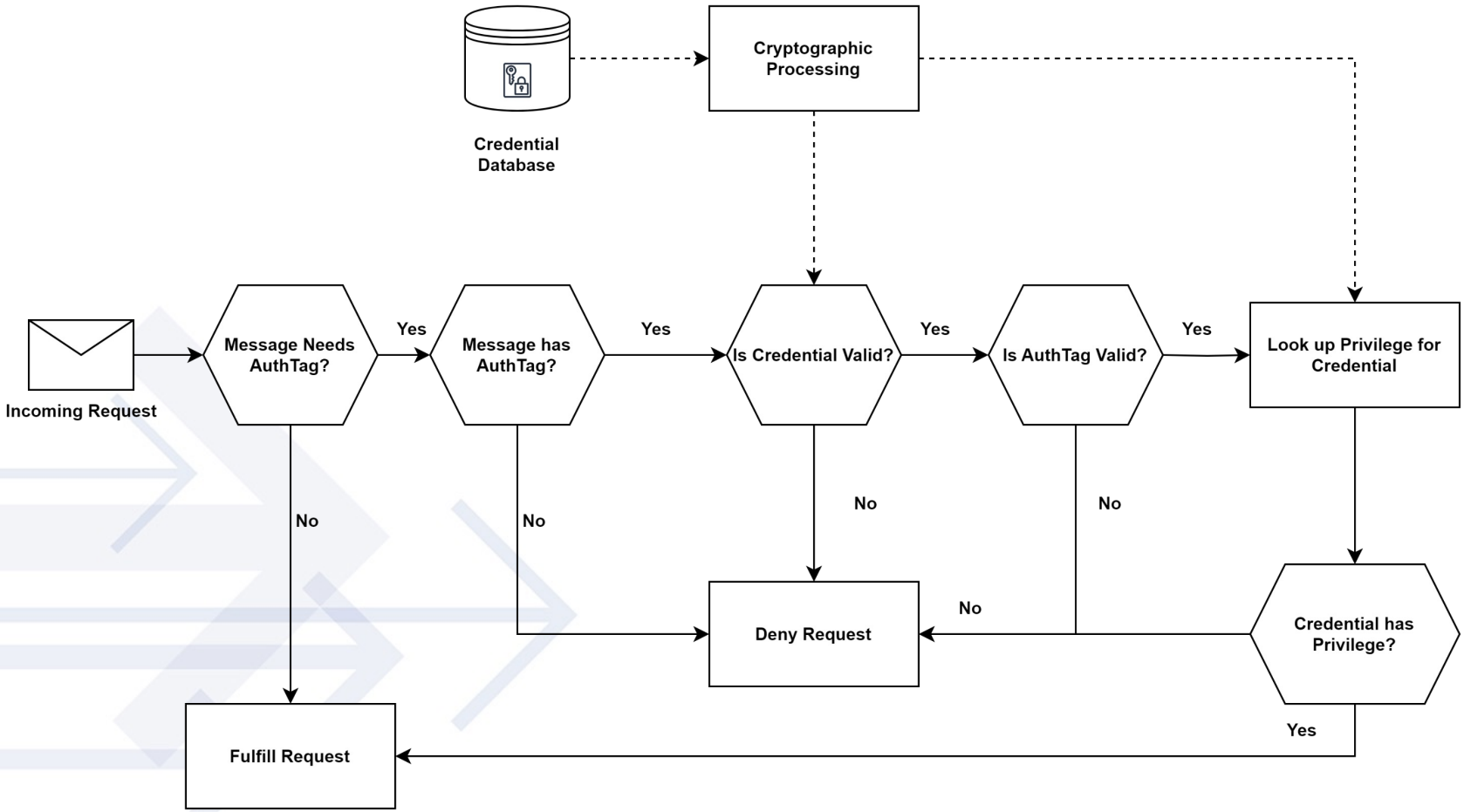
Threat Model – SPDM Requester represents user

- Proposal uses SPDM secured message session (DSP0277)
  - Provides end-to-end Confidentiality, Integrity and replay protection between requester and responder
  - No additional attacks are addressed by this specification

Threat Model - User Proxied Case

- To determine the level of protection needed to protect the integrity of privileged data across SPDM connection.
- Specifically, a user proxied use case is this:
  - The user has possession of the private key but not the Requester.
  - The Requester is acting as a “bridge” (or supposed to) between user and responder.
- Subject to attacks if SPDM Requester is not fully trusted
  - Requester (PITM) can save authorized (signed) commands and replay it since session is only between Requester and Responder.
  - Requester (PITM) can redirect requests to unintended Responder and responses can be dropped or redirected to unintended user.
  - Authorization must at least have end-to-end Integrity, Authenticity and Replay protection in the face of untrusted SPDM Requester

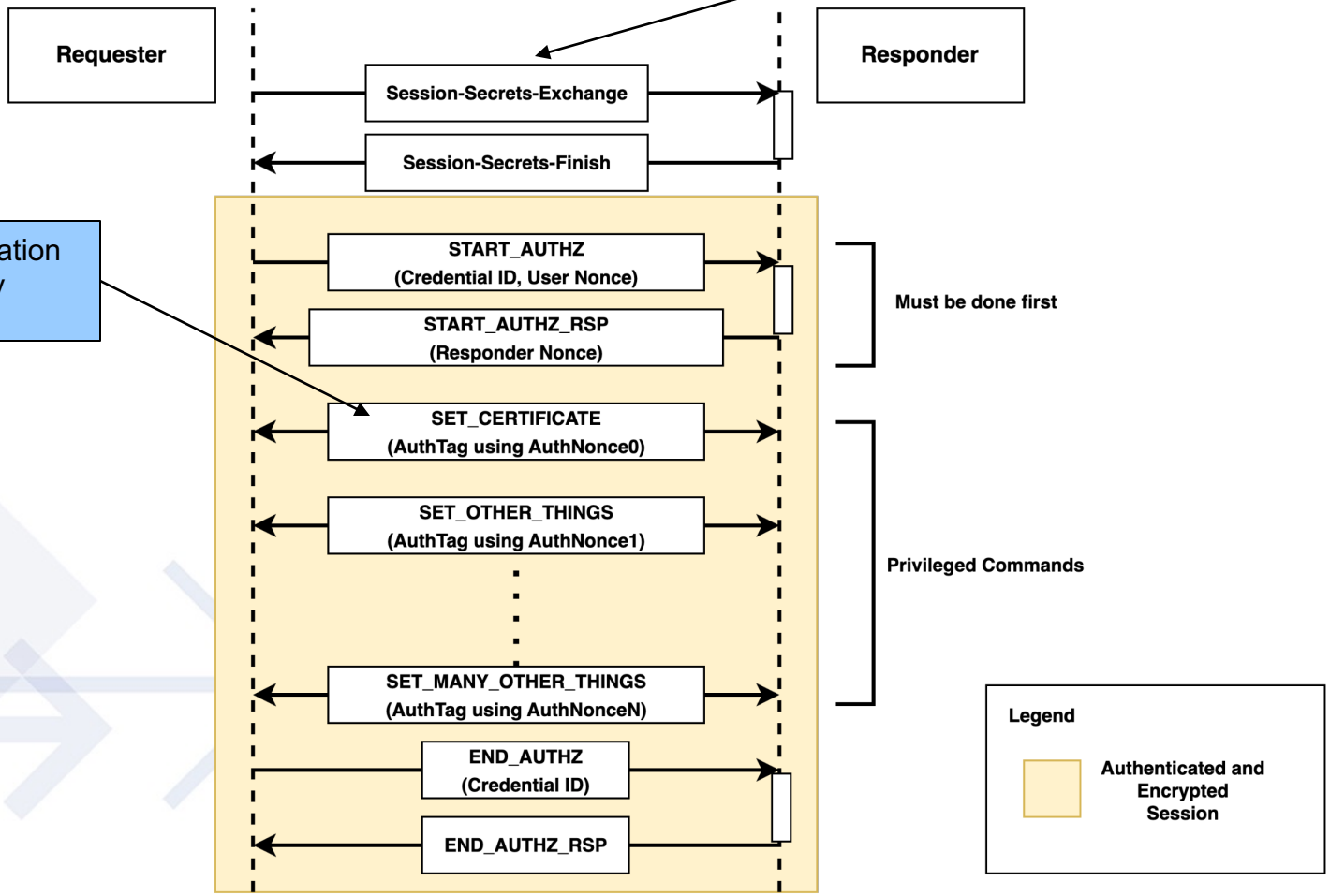
Example High Level Flow Diagram - Endpoint



Authorization Flow Sequence

Regular SPDM Key exchange (Asym, PSK etc) using endpoint key
Includes negotiating Secured Messages DSP0277 version

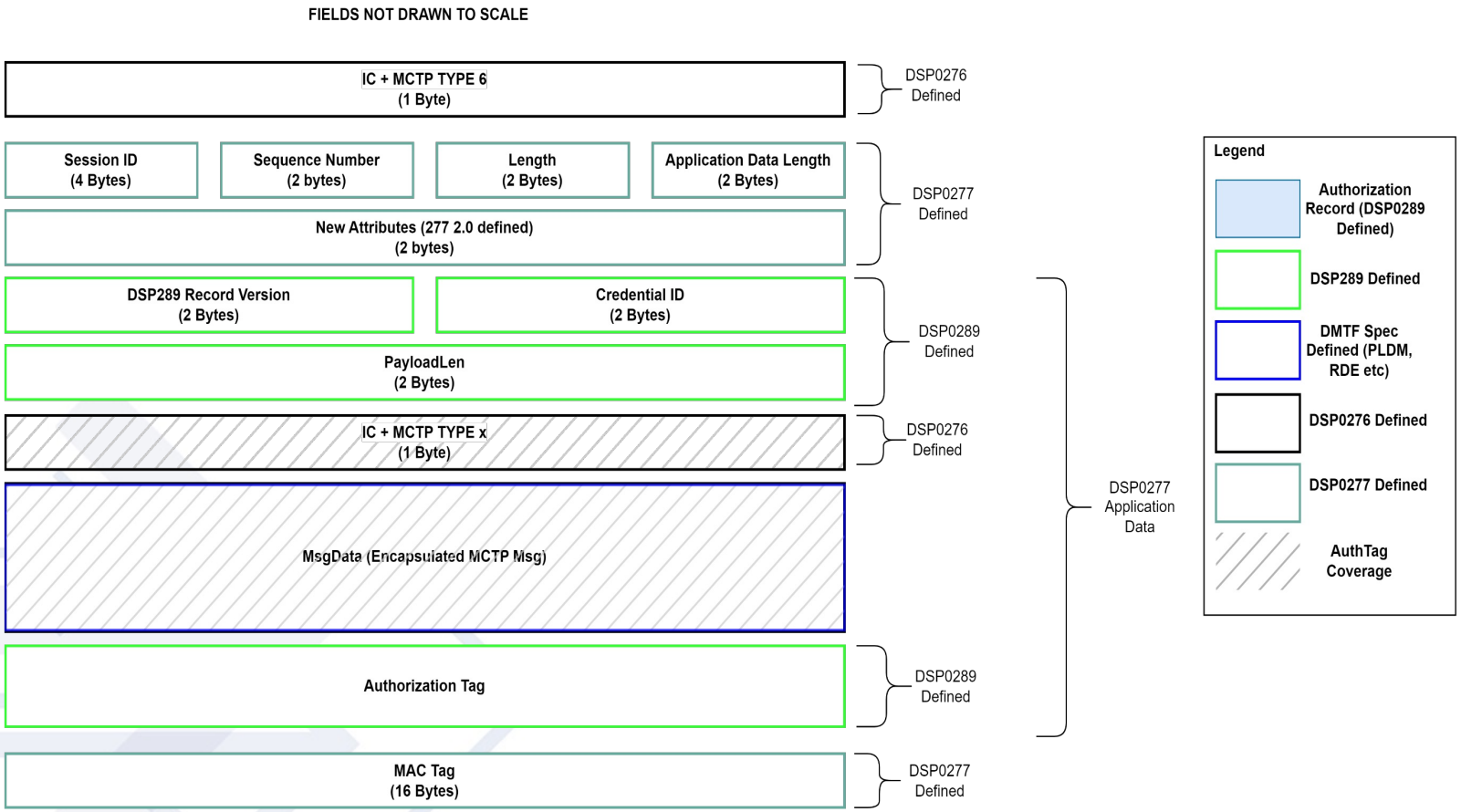
AuthTag uses authorization key (separate from key exchange keys)



Secured Message (DSP0277) Changes

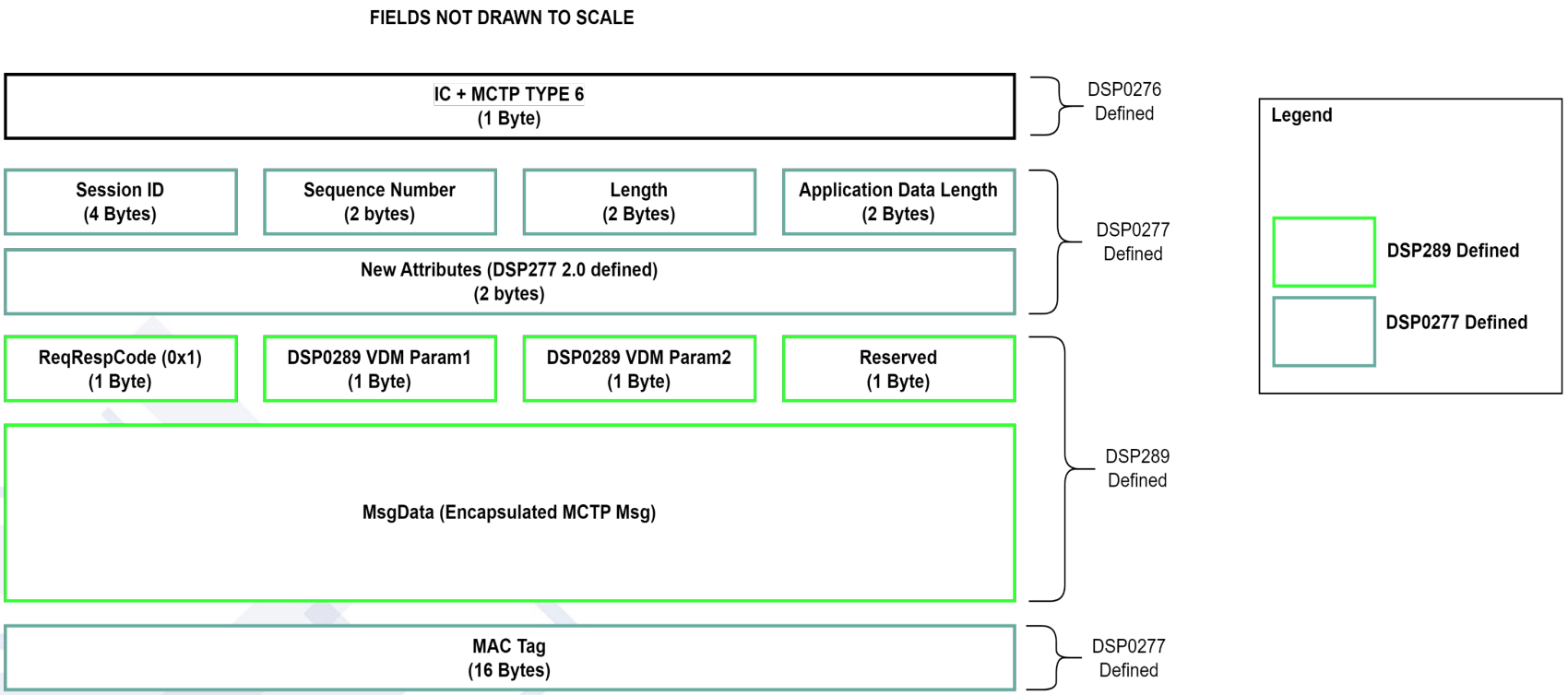
- DSP0277 v2.0 will add a field to indicate this is a message that contains authorization
  - MCTP2.0 requires adding a new 2 byte field for alignment
  - Authorization spec uses that field
  - Additional bytes make it incompatible with DSP0277 1.x
  - DSP0277 version is negotiated during key exchange
- Add “Authorization” to the title of DSP0277 and create new binding in place of DSP0276 to support DSP0277 2.0 to MCTP binding
- When the field is set, authorization tag is present and defined by DSP0289
- For any specification not using DSP0277, the method to indicate presence of authorization tag must be specified independently

Authorization Record



- Used to transport message to be authorized

DSP0289 Authorization Command



- Used to send DSP0289 defined commands (Version, Capabilities etc)

Authorization Tag

- When the authorization field is set in DSP0277, the following are defined in DSP0289:
  - For requests that require authorization, we append an authorization tag to the end of the request.
  - Authorization tag is generated by the User using input from both endpoints, and verifiable by the Responder
  - The authorization tag will contain a credential ID and a signature using the secret key.
    - AuthNonce = Concat (User Nonce, Responder Nonce)
    - Hash = Concat (AuthNonce, all the fields in the request without AuthTag).
    - Hash algorithm and other related parameters are defined by the user's policy.
  - Authorization Tag is variable length.

Summary of Commands for Authorization

- New Commands:
  - START_AUTHZ, START_AUTHZ_RSP
    - To exchange nonces, Credential ID and indicate to responder to expect messages with AuthTags
  - END_AUTHZ, END_AUTHZ_RSP
    - Destroy/Free any state for a given Credential ID

What is a Credential?

Field Name	Description
Credential ID	A unique ID that identifies this credential. The responder determines the maximum number of credentials it wants to support. The responder can determine the values for each Credential ID.
Credential Type	The type of credential: - Asymmetric Key
Signing Algorithm	• Replication of the asymmetric algorithms in Negotiate ALGORITHMS request or response. • Only one algorithm can be selected at a time
Hashing Algorithm	• Replicate the hashing algorithm that is supported in Negotiate Algorithms. • Only one algorithm can be selected at a time
Credential Data	• For Asymmetric Algorithm, this will be the public key.

Credential Provisioning

- DSP0289 defines 8 persistent Credential Slots, minimum supported is one
  - The ability to clear slot contents will be included in v1.0
- Slot 0 – Only provisioned in a trusted environment (ex: Secure Manufacturing, secure onboarding)
  - Bootstrap authorization
- All other slots can be provisioned in
  - A trusted environment
  - Authorized via a selected credential(s) already provisioned
- All credentials associated with a policy
  - Defines what the credential can be used to authorize (ex: SET_CERTIFICATE, PLDM FW Activation etc)
- DSP0289 defines credential provisioning commands for credentials and policies

Credential Provisioning Commands

- New Privileged/Authorized Operation:
  - Set/Get Credential Info
    - Used to provision credential
    - Slot 0 provisioned in secure provisioning environment (bootstrap), optionally authorize for other slots
    - “Delete” Credential in slot X
      - Authorized by a selected provisioned credential
  - Set/Get Credential Policy
    - Associate what the credential can authorize (ex: SET_CERTIFICATE, provisioning/deletion of other credential slots)
      - A list of authorizable operations and policies needs to be provided
    - Per credential policy
  - Get Credential Mgmt Capabilities
    - Discover algorithms, number of credentials provisioned etc

Policy Details

- Partner organizations and vendors can create a policy binding specification
  - A Policy is a grouping of commands that can be assigned or not assigned to a Credential
  - Some commands may be excluded from Policy controls
    - For instance, read commands may be determined to be safe
- Example policies:
  - Authorization
    - Allows DSP0289 commands to manage Credentials and Policy assignments
    - Granted to Credential slot 0 by default
  - Firmware Update
    - Allows PLDM and private API commands to update firmware
- Mapping of commands and number of Policies available is managed by the Responder device

Call to Action

- Submit feedback to DMTF
  - We believe that the change to DSP0277 v2.0 does not affect the data plane, is this a valid assumption?
  - Do we need to add categories for policy groupings?
  - Does this meet your authorization needs?
  - Is requiring an SPDM secured session acceptable?