Security Protocol and Data Model (SPDM) Architecture

Version 0.9 Release

Work in Progress by PMCI Security Task Force

Last Updated: 04/03/2019

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.

Acknowledgement

- Some of the content in this presentation is derived from USB Authentication specification 1.0 at https://www.usb.org/sites/default/files/documents/usb_authentication_20180904.zip

Feedback

- Industry feedback on this proposal is encouraged
  - https://www.dmtf.org/standards/feedback/

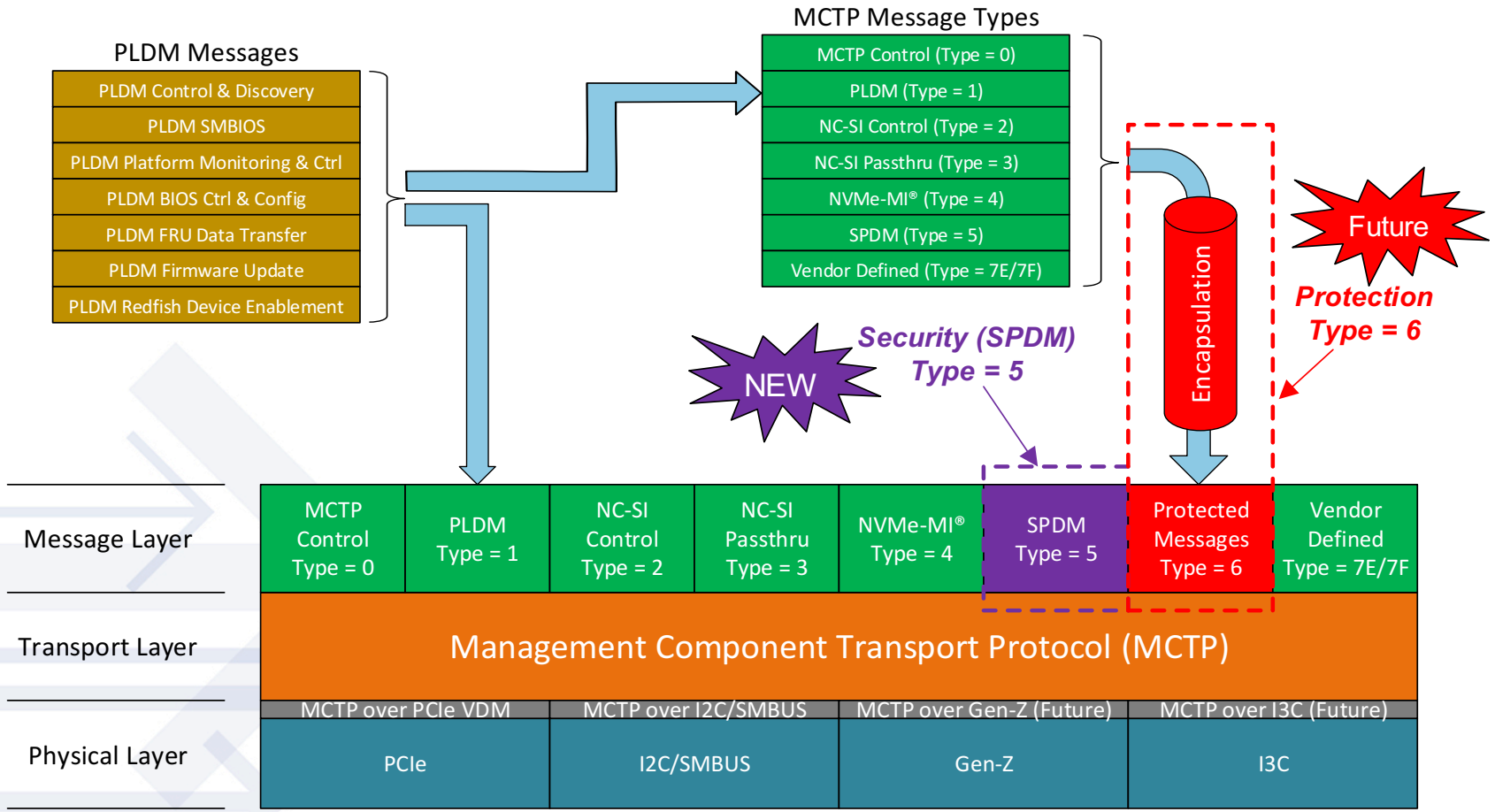
Guiding Principles

- Use MCTP message type 5 for all authentication commands including the future ones used for setting up secure sessions
- Use MCTP message type 6 for secured transport of encapsulated MCTP messages as appropriate (Future Version)
- Derived from USB Authentication
- Fields are defined to be little endian unless otherwise noted

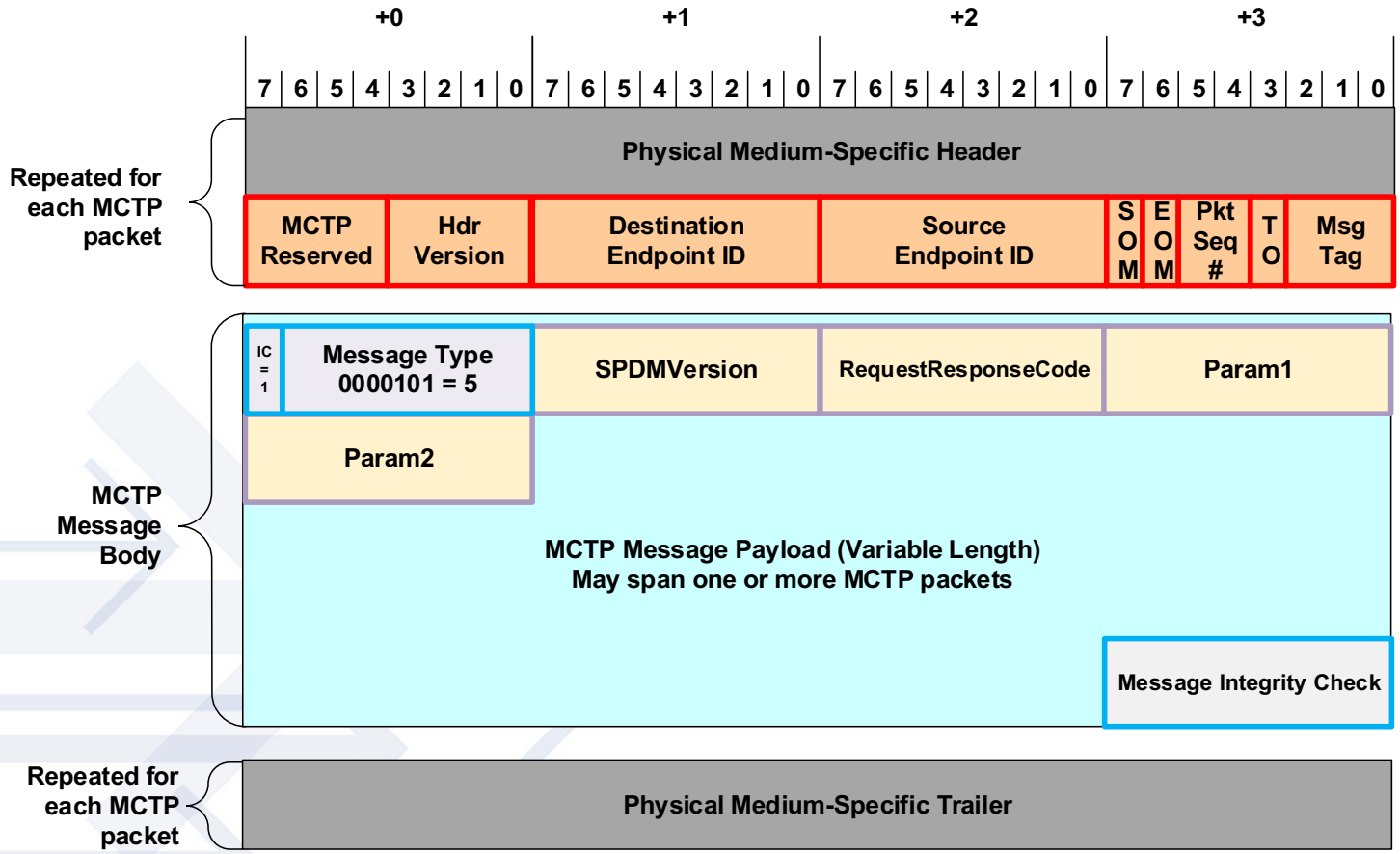
Specifications in Development

- DSP0274
  - Security Protocol and Data Model (SPDM) Specification
    - This specification will contain message exchange, sequence diagrams, message formats, and other relevant semantics for authentication, firmware measurement, and certificate management
    - Versioning scheme is WIP
- DSP0275
  - SPDM over MCTP Binding Specification
    - This specification will contain the mapping of SPDM to MCTP message type 5

PMCI MCTP Security Proposal – Diagram View



MCTP Message Type 5 (Security Commands) Format

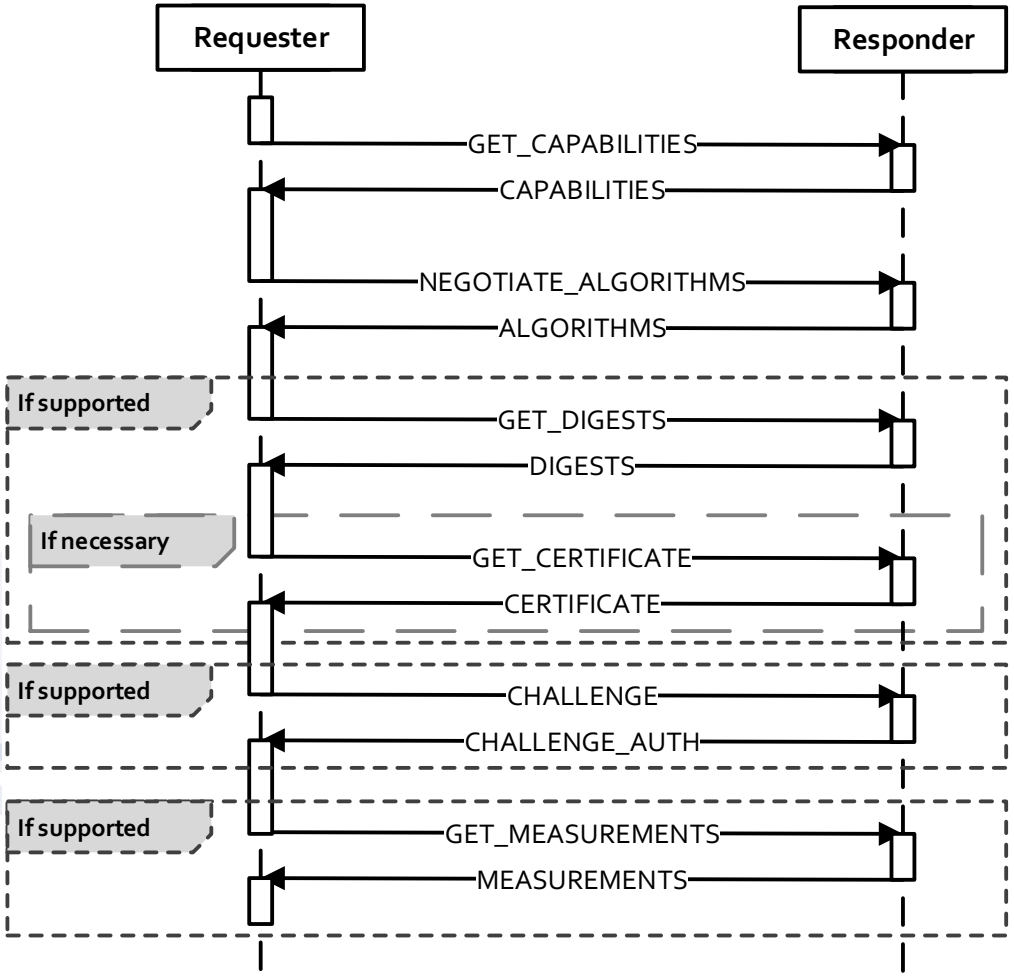


SPDM Specification Details

SPDM Common Format

Offset (byte)	Field Name	Size (bytes)	Definition
0	*SPDMVersion*	1	Reference to the version of the SPDM specification.
1	*RequestResponseCode*	1	Identifies type of request or type of response.
2	*Param1*	1	Present in all commands and responses. Value is command specific.
3	*Param2*	1	Present in all commands and responses. Value is command specific.

High-level Authentication Sequence Diagram



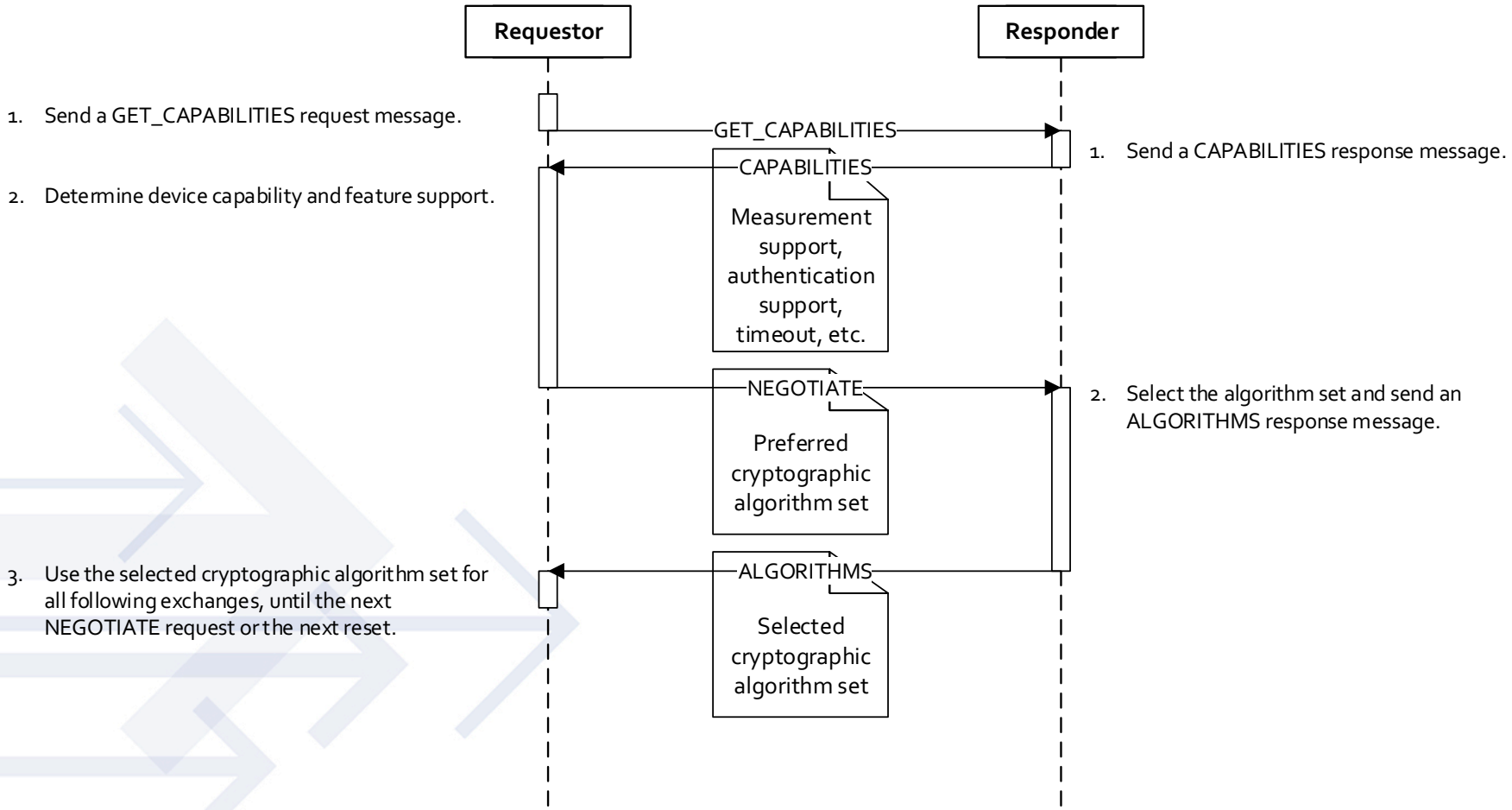
RequestResponseCode Field: Part 1

Value	Type	Required	Name	Description
80h	Request		Reserved	
81h	Request	Optional	GET_DIGESTS	Retrieve Cert chain digest
82h	Request	Optional	GET_CERTIFICATE	Retrieve segment of cert chain
83h	Request	Optional	CHALLENGE	Initiate authentication
84h – DFh	Request		Reserved	
E0h	Request	Optional	GET_MEASUREMENTS	Retrieve signed firmware measurement
E1h	Request	Yes	GET_CAPABILITIES	Retrieve capabilities
E2h	Request	Optional	SET_CERTIFICATE	Post 1.0: Install new cert chain (slots 1-7 only).
E3h	Request	Yes	NEGOTIATE_ALGORITHMS	Negotiate Cryptographic Algorithms
E4h – FEh	Request		Reserved	
FFh	N/A		Reserved	Do not use.

RequestResponseCode Field: Part 2

Value	Type	Name	Description
00h	Response	Reserved	
01h	Response	DIGESTS	Response to GET_DIGEST request.
02h	Response	CERTIFICATE	Response to GET_CERTIFICATE request.
03h	Response	CHALLENGE_AUTH	Response to CHALLENGE.
04h – 5Fh	Response	Reserved	
60h	Response	MEASUREMENTS	Response to GET_MEASUREMENTS request.
61h	Response	CAPABILITIES	Response to GET_CAPABILITIES request.
62h	Response	SET_CERT_RESPONSE	Post 1.0: Response to SET_CERTIFICATE request.
63h	Response	ALGORITHMS	Response to NEGOTIATE request
64h – 7Eh	Response	Reserved	
7Fh	Response	ERROR	Response to any unsuccessful request.

GET_CAPABILITIES Sequence Diagram



GET_CAPABILITIES Request

This request is used to discover endpoint protocol capabilities.

Offset	Field	Size	Value
0	*SPDMVersion*	1	V1.0 = 01h
1	*Request/Response Code*	1	E1h = GET_CAPABILITIES
2	*Reserved1*	1	Reserved
3	*Reserved2*	1	Reserved

Successful CAPABILITIES

Offset	Field	Size	Value
0	*SPDMVersion*	1	V1.0 = 01h
1	*Request/Response Code*	1	61h = CAPABILITIES
2	*Reserved1*	1	Reserved
3	*Reserved2*	1	Reserved
4	*DetailedVersion*	3	The remaining 3 bytes that are concatenated with Offset 0 to form the complete SPDM specification version. Offset 0 describes the major version.
7	*CT*	1	Timeout value associated with CHALLENGE and GET_MEASUREMENTS operations in µS, expressed in logarithmic (base 2) scale. This value is added to media specific timeout value when deriving Request-to-response timeout for CHALLENGE and GET_MEASUREMENTS requests.
8	*Flags*	4	Byte 0 - Bit 0 – Reserved for future version Bit 1 – Supports GET_DIGESTS, GET_CERTIFICATE and CHALLENGE requests Bit 2 – Supports SET_CERTIFICATE request Bit 3 – Support GET_MEASUREMENTS All other bits are reserved for future extension. Bytes 1-3 – Reserved
12	*Reserved3*	4	Reserved

NEGOTIATE_ALGORITHMS Request

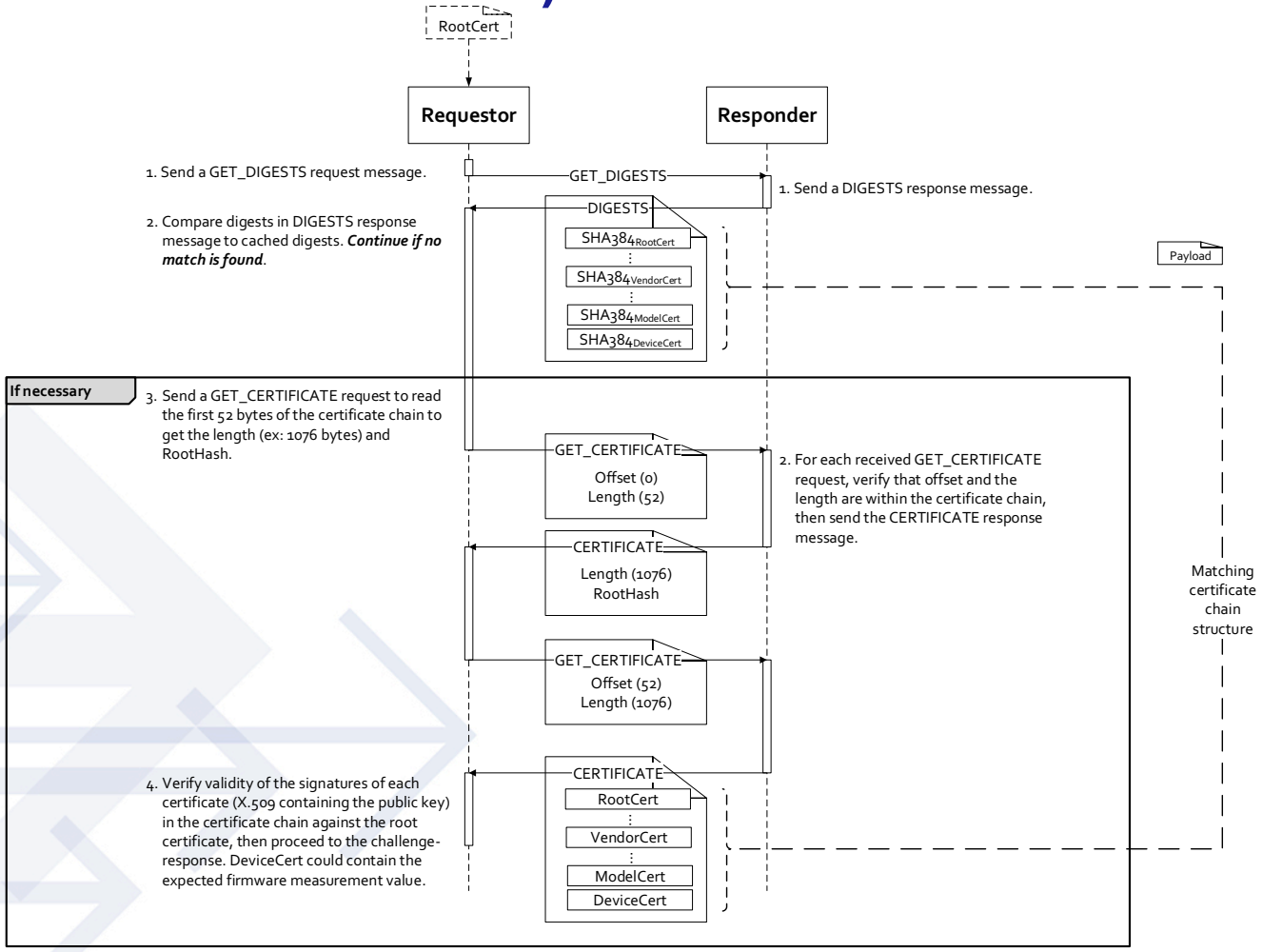
Offset	Field	Size	Value
0	*SPDMVersion*	1	V1.0
1	*Request/Response Code*	1	E3h = NEGOTIATE_ALGORITHMS
2	*Reserved1*	1	Reserved
3	*Reserved2*	1	Reserved
4	*Length*	2	Length of the request packet in bytes
6	*Measurement Specification*	1	Bit Mask – Bit position based on “Measurement Block Specification” Bit 7 Reserved for extension indication (to handle overflow of Bit Mask field in future versions).
7	*Reserved*	1	Reserved
8	*BaseAsymAlgo*	4	Bit vector listing PMCI enumerated asymmetric algorithms supported by requestor. Bit 0 – RSA 2048; Bit 1 – RSA 3072; Bit 2 – RSA 4096; Bit 3 – ECDSA secp256r1; Bit 4 – ECDSA secp384r1; Bit 5 – ECDSA secp521r1
12	*BaseHashAlgo*	4	Bit vector listing PMCI enumerated hashing algorithms supported by requestor. Bit 0 – SHA2-256 ; Bit 1 – SHA2-384; Bit 2 – SHA2-512; Bit 3 – SHA3-256 ; Bit 4 – SHA3-384; Bit 5 – SHA3-512
16	*Reserved*	8	Reserved
24	*ExtAsymCount*	1	Number of extended Asymmetric algorithms supported by requestor (=A)
25	*ExtHashCount*	1	Number of extended Hashing algorithms supported by requestor (=H)
26	*Reserved*	2	Reserved for future use
28	*ExtAsym*	3A	First byte is enumeration for the encoding for ExtAsym 0 – DMTF; 1 – TCG List of the extended asymmetric algorithms supported by requestor. At this time, DMTF has no algorithms defined.
28+3A	*ExtHash*	3H	First byte is enumeration for the encoding for ExtHash 0 – DMTF; 1 – TCG List of the extended Hashing algorithms supported by requestor. At this time, DMTF has no algorithms defined.
28+3A+3H	*Reserved*	-	Reserved for future expansion. Consult the Length field (offset 4) to determine the number of bytes in the request.

Successful ALGORITHMS

Offset	Field	Size	Value
0	*SPDMVersion*	1	V1.0
1	*Request/Response Code*	1	63h = ALGORITHMS
2	*Reserved1*	1	Reserved
3	*Reserved2*	1	Reserved
4	*Length*	2	Length of the request packet in bytes
6	*Measurement Specification*	1	Bit Mask – Bit position based on “Measurement Block Specification” Bit 7 Reserved for extension indication (to handle overflow of Bit Mask field in future versions).
7	*Measurement Info Size*	1	Length in bytes of each measurement record (M)
8	*BaseAsymSel*	4	Bit vector listing PMCI enumerated asymmetric algorithms selected. No more than 1 bit can be set.
12	*BaseHashSel*	4	Bit vector listing PMCI enumerated hashing algorithms selected. No more than 1 bit can be set. This hash algorithm will also be used for measurement hashes.
16	*Reserved*	8	Reserved for future use
24	*ExtAsymSelCount*	1	The number of extended Asymmetric algorithms selected. Either 0 or 1. (=A)
25	*ExtHashSelCount*	1	The number of extended Hashing algorithms selected. Either 0 or 1. (=H)
26	*Reserved*	2	
28	*ExtAsymSel*	3A	First byte is enumeration for the encoding for ExtAsymSel 0 – DMTF; 1 – TCG List of the extended asymmetric algorithms selected
28+3A	*ExtHashSel*	3H	First byte is enumeration for the encoding for ExtHashSel 0 – DMTF; 1 – TCG List of the extended Hashing algorithms selected
28+3A+3H	*Reserved*	-	Reserved for future expansion. Consult the length field (offset 4) to determine the number of bytes in the response.

The responder shall respond showing no more than one chosen algorithm per method.

GET_DIGESTS / GET_CERTIFICATE Sequence Diagram (Single Certificate Chain)



GET_DIGESTS Request

This Request is used to retrieve Certificate Chain digests.

Offset	Field	Size	Value
0	*SPDMVersion*	1	V1.0 = 01h
1	*Request/Response Code*	1	81h = GET_DIGESTS
2	*Reserved1*	1	Reserved
3	*Reserved2*	1	Reserved

Successful DIGESTS

Offset	Field	Size	Value
0	*SPDMVersion*	1	V1.0 = 01h
1	*Request/Response Code*	1	01h = DIGESTS
2	*Capabilities*	1	Capabilities Field; shall be set to 01h for this specification. All other values reserved.
3	*SlotMask*	1	Slot mask. The bit in position K of this byte shall be set to 1b if and only if slot number K contains a Certificate Chain for the protocol version in the *SPDMVersion* field. (Bit 0 is the least significant bit of the byte.) The number of digests returned shall be equal to the number of bits set in this byte. The digests shall be returned in order of increasing slot number.
4	*Digest[0]*	H	H-byte digest of the first Certificate Chain. H is the size of the hashing algorithm output mutually agreed via NEGOTIATE_ALGORITHMS request. This field is big endian.
…	…	…	…
4 + (H * (n -1))	*Digest[n-1]*	H	H-byte digest of the last (n^{th}) Certificate Chain. H is the size of the hashing algorithm output mutually agreed via NEGOTIATE_ALGORITHMS request. This field is big endian.

GET_CERTIFICATE Request

This Request is used to retrieve Certificate Chains, one chunk at a time.

Offset	Field	Size	Value
0	*SPDMVersion*	1	V1.0 = 01h
1	*Request/Response Code*	1	82h = GET_CERTIFICATE
2	*SlotNum*	1	Slot number of the target Certificate Chain to read from. The value in this field shall be between 0 and 7 inclusive. Slot 0 is reserved for the Device certificate.
3	*Reserved2*	1	Reserved
4	*Offset*	2	Offset in bytes from the start of the Certificate Chain to where the read request begins.
6	*Length*	2	Length in bytes of the read request. Length is an unsigned 16-bit integer. If offset=0 & length=0xFFFF, the entire chain will be returned from the device. **If a device cannot return the entire chain it shall return** RequestedInfoTooLong **error code.**

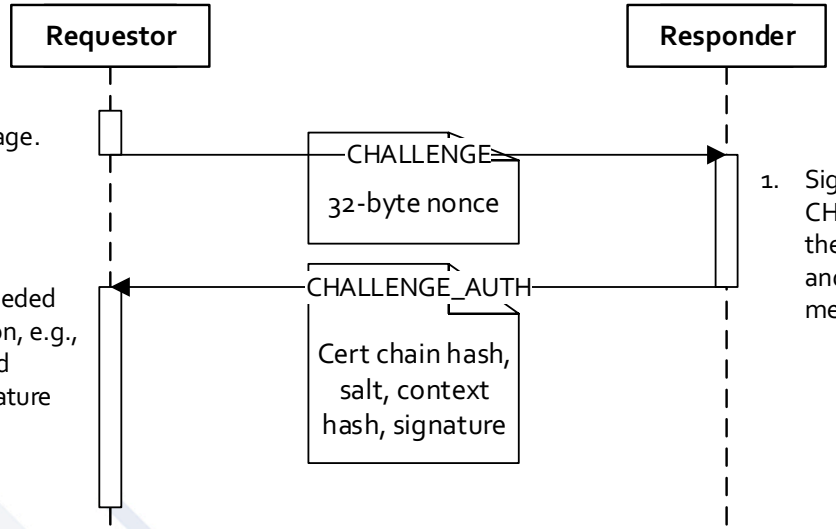
Successful CERTIFICATE

Offset	Field	Size	Value
0	*SPDMVersion*	1	V1.0 = 01h
1	*Request/Response Code*	1	02h = CERTIFICATE
2	*SlotNum*	1	Slot number of the Certificate Chain returned
3	*Reserved2*	1	Reserved
4	*CertChain*	*Length*	Data Requested contents of target Certificate Chain, formatted in DER. This field is big endian.

CHALLENGE Sequence Diagram

1. Send a CHALLENGE request message.

2. Collect the relevant information needed for verification signature verification, e.g., cert chain hash, and use the verified device public key to verify the signature field.



1. Signs the CHALLENGE request message + the CHALLENGE_AUTH response message (minus the signature field) using the device private key and send a CHALLENGE_AUTH response message.

CHALLENGE Request

This Request is used to authenticate an endpoint.

Offset	Field	Size	Value
0	*SPDMVersion*	1	V1.0 = 01h
1	*Request/Response Code*	1	83h = CHALLENGE
2	*SlotNum*	1	Slot number of the recipient's Certificate Chain that will be used for Authentication
3	*Reserved2*	1	Reserved
4	*Nonce*	H	Random H-byte nonce, a random value chosen by the Authentication Initiator. H is the size of the hashing algorithm output (per NIST SP800-90A) mutually agreed via ALGORITHMS BaseHashSel or ExtHashSel field.

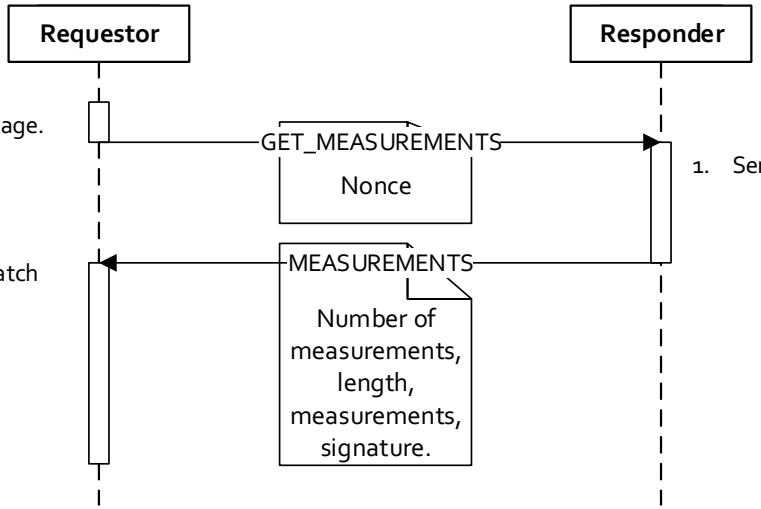
Successful CHALLENGE_AUTH

Offset	Field	Size	Value
0	*SPDMVersion*	1	V1.0 = 01h
1	*Request/Response Code*	1	03h = CHALLENGE_AUTH
2	*SlotNum*	1	Shall contain the Slot number in the *SlotNum* field of the corresponding CHALLENGE Request
3	*SlotMask*	1	Slot mask. The bit in position K of this byte shall be set to 1b if and only if slot number K contains a Certificate Chain for the protocol version in the *SPDMVersion* field. (Bit 0 is the least significant bit of the byte.)
4	*MinSPDMVersion*	1	Minimum SPDM version supported by this Device
5	*MaxSPDMVersion*	1	Maximum SPDM version supported by this Device
6	*Capabilities*	1	Set to 01h for this specification. All other values reserved
7	*Reserved*	1	Reserved
8	*CertChainHash*	H	Hash of the Certificate Chain used for Authentication. H is the size of the hashing algorithm output (per NIST SP800-90A) mutually agreed via ALGORITHMS BaseHashSel or ExtHashSel field. This field is big endian.
8+H	*Salt*	H	Value chosen by the Authentication Responder. *Note: the Salt shall be unique per response for the duration of a device reset cycle*
8+2H	*Context Hash*	H	Hash over device specific information. This field is big endian.
8+3H	*Signature*	S	Signature is the signed hash of the bytes (in order) from the CHALLENGE request *SPDMVersion* through the request Nonce and the CHALLENGE_AUTH response *SPDMVersion* through the response Context Hash. This is signed using the Device private key. S is the size of the asymmetric signing algorithm output mutually agreed via NEGOTIATE_ALGORITHMS request.

GET_MEASUREMENTS Sequence Diagram

1. Send a GET_MEASUREMENTS request message.

2. Verify signature and verify measurements match expected values.



1. Send a MEASUREMENTS response message.

GET_MEASUREMENTS Request

This Request is used to retrieve measurements of mutable firmware component(s) that the recipient endpoint is executing.

Measurements on their own are one of several methods to provide identity.

Signing shall use the device private key.

Offset	Field	Size	Value
0	*SPDMVersion*	1	V1.0 = 01h
1	*Request/Response Code*	1	E0h = GET_MEASUREMENTS
2	*CmdSpec1*	1	Request type: 0: Single or All Measurements 1: Measurement log 2: Post 1.0: Signed Measurement manifest Hash (Signed by Vendor key). 3: Post 1.0: Signed SRTM (Signed by Vendor Key). All other bits are reserved.
3	*CmdSpec2*	1	Measurement index. Value of 0xFF return all Measurements.
4	*Reserved*	2	Reserved to be compatible with the Cerberus definition.
6	*Nonce*	H	Random H-byte nonce chosen by the Authentication Initiator. H is the size of the hashing algorithm output mutually agreed via NEGOTIATE_ALGORITHMS request.

Successful MEASUREMENTS

Offset	Field	Size	Value
0	*SPDMVersion*	1	V1.0 = 01h
1	*Request/Response Code*	1	60h = MEASUREMENTS
2	*CmdSpec1*	1	When the requested Measurement index is 0, this parameter returns the total number of Measurement indices on the device; otherwise reserved.
3	*Reserved2*	1	Reserved
4	*Length*	2	Length in bytes
6	Salt	H	H arbitrary salt chosen by the Responder
6+H	*NumberofMeasurementBlocks (N)*	1	Number of Measurement blocks
6+H+1	*MeasurementBlocks*	*L*N*	Concatenation of all Measurement Blocks
6+H+1+ (*L*N*)	*Signature*	*S*	Signature of the GET_MEASUREMENTS Request and MEASUREMENTS messages, excluding the Signature field and signed using the Device Private Key. The size of the Signature field depends on the asymmetric signing algorithm that was mutually agreed upon via NEGOTIATE_ALGORITHMS.

Measurement Block

- Each Measurement block contains a 1-DWORD descriptor, followed by the cryptographic hash and optionally additional information
- Logical increment of the Measurement index implies bootstrapping of firmware stages
- When returning Measurement log, the requestor specifies the Measurement index that it needs the history for. Each event that caused changes in the Measurement hash is recorded in one Measurement block, distinguished by the step log field.

Offset	Field	Size	Value
0	Measurement index	1	0-255 (Dependent what specification ?)
1	Measurement type	1	0: immutable ROM (SRTM) 1: mutable firmware 2: HW configuration, e.g., straps, debug modes 3: FW configuration e.g. configurable FW policy All other bits are reserved.
2	Specification	1	0: DMTF (included in current specification development) Other values to be considered for post 1.0 development: 1: TPM 2.0 (Use index as PCR index) 2: Cerberus 3: OEM (Need OEM id): OEM_ID – Measurement hash Need feedback what the use models are.
3	Step log	1	Reserved. Considerations for post 1.0. When requesting for Measurement log, this field might be used to indicate the sequence of events/steps that cause changes to the Measurement hash.
4	Measurement Info	M	cryptographic hash and optionally additional information

ERROR

Offset	Field	Size	Value
0	*SPDMVersion*	1	Minimum Supported protocol version, V1.0 for now
1	*Request/Response Code*	1	7Fh = ERROR
2	*ErrorCode*	1	Error Code.
3	*ErrorData*	1	Error Data.

Error Code	Value	Description	Error Data
Reserved	00h	Reserved	Reserved
InvalidRequest	01h	One or more Request fields are invalid	00h
UnsupportedProtocol	02h	Requested Security Protocol Version is not supported	Maximum supported Security Protocol Version1
Busy	03h	Device cannot respond now, but will be able to respond in the future	00h
Unspecified	04h	Unspecified error occurred	00h
Uninitialized	05h	Command received without session initialization	00h
RequestedInfoTooLong	06h	The requested data cannot be sent in one response	
Reserved	07h-CFh	Reserved	Reserved
Reserved for other standards	D0h-EFh	Reserved	See reference specification.
Vendor Defined	F0h- FFh	Vendor defined	Vendor defined

Timeouts

Timing Specification	Symbol	Min	Max	Description
Number of request retries	AN1	2	See description	Total of three tries, minimum: the original try plus two retries. The maximum number of retries for a given request is limited by the requirement that all retries shall occur within MT4 from the corresponding media spec, max of the initial request.
Request-to-response time	AT1	-	MT1+CT	MT1 is the request-to-response timing defined in the media binding spec. CT is the allowance for crypto operations and reported via CAPABILITY response.
Time-out waiting for a response	AT2	AT1 max + MT2 min – MT1 max	MT4 min	MT2 min and MT1 max are defined in the appropriate media binding specification.

Note: All timeouts report the worst case value

Note: SPDM will have one set of timeouts related to crypto operations and the MCTP binding spec will have another set of timeouts basing their computations on the SPDM timeouts for their specific binding

Future Work

- Protection: Encryption / Integrity
- Measurement log
- Set certificate command
- Measurement manifest (Local attestation)

Feedback

- Industry feedback on this proposal is encouraged
  - https://www.dmtf.org/standards/feedback/