SPDM 1.1: Session Key Exchange Protocols

July 2020

www.dmtf.org

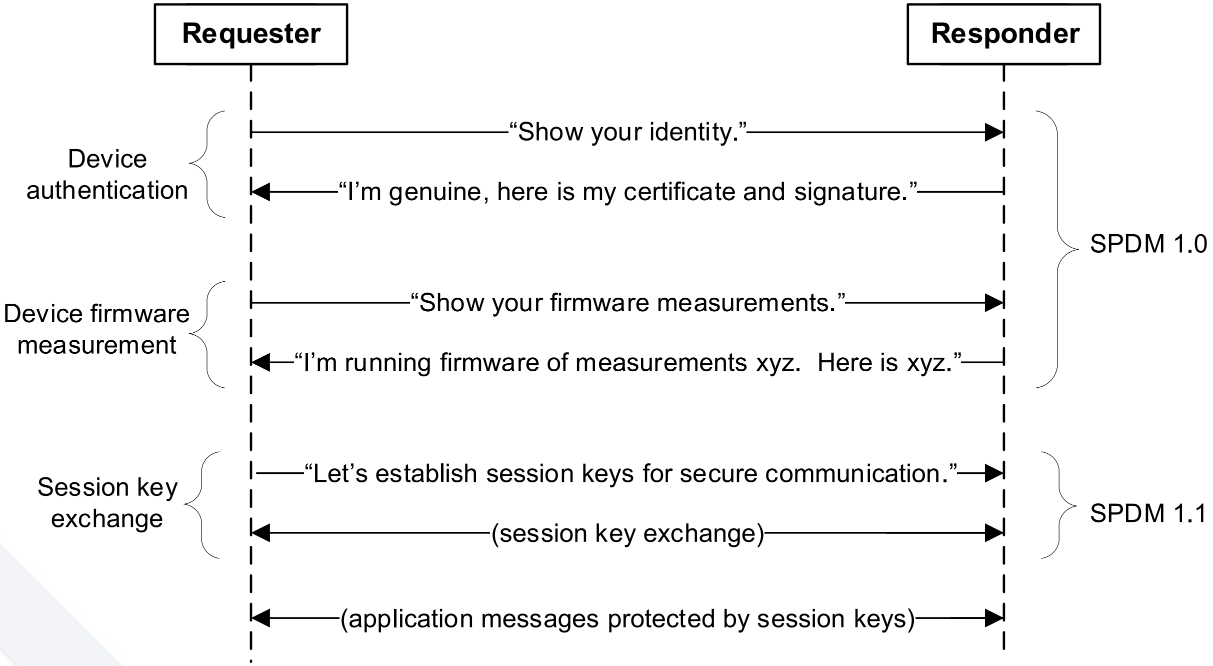
Disclaimer

- 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.

SPDM 1.1 Feature Additions

- Sessions
  - Key exchange
    - Exchange keys to enable encryption by the transport
    - SIGMA and Pre-shared key options
    - Suitable for adoption by many industry transport layers
  - Key confirmation
  - Key update
  - Key schedule
- Mutual authentication
- Derivation of additional keys

Session Key Exchange


Requester
Responder
Device authentication
"Show your identity."
"I'm genuine, here is my certificate and signature."
SPDM 1.0
Device firmware measurement
"Show your firmware measurements."
"I'm running firmware of measurements xyz. Here is xyz."
Session key exchange
"Let's establish session keys for secure communication."
(session key exchange)
SPDM 1.1
(application messages protected by session keys)

Objective: Establish session keys that are known to only Requester and Responder

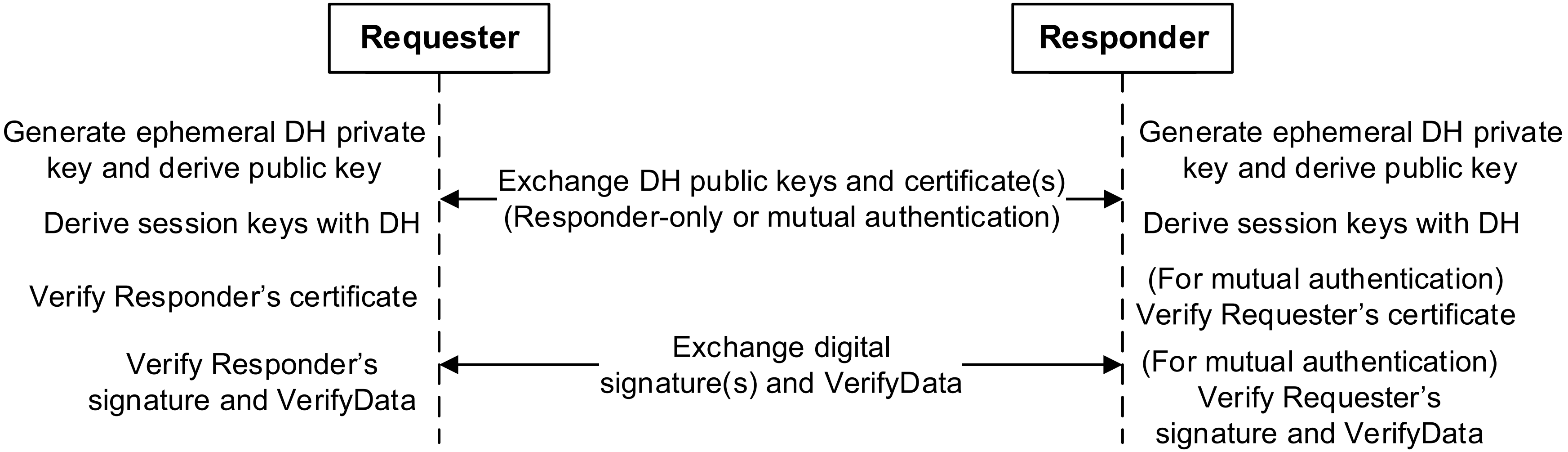
- Either endpoint may abort a session at any time.
- Authentication happens with session key exchange.
- Requester authenticates Responder. Optionally, Responder may authenticate Requester.

SPDM 1.1 specifies the following session key exchange schemes:

1. **SIGMA** option: based on ephemeral Diffie-Hellman and digital signatures.
2. **Pre-shared secret** option: based on a pre-shared secret known to both endpoints.

*SIGMA: http://webee.technion.ac.il/~hugo/sigma-pdf.pdf

SIGMA Option for Session Key Exchange



Requester

Responder

Generate ephemeral DH private key and derive public key

Derive session keys with DH

Verify Responder's certificate

Verify Responder's signature and VerifyData

Exchange DH public keys and certificate(s) (Responder-only or mutual authentication)

Exchange digital signature(s) and VerifyData

Generate ephemeral DH private key and derive public key

Derive session keys with DH

(For mutual authentication) Verify Requester's certificate

(For mutual authentication) Verify Requester's signature and VerifyData

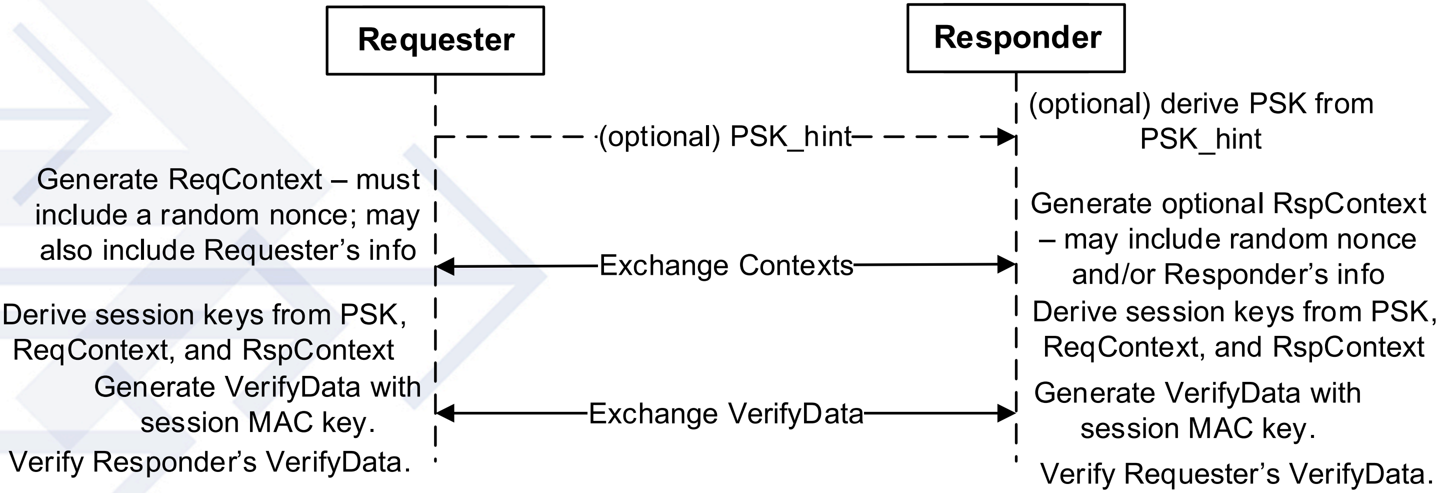
- Diagram above illustrates high-level sequence; arrows do not map to actual commands
- Based on SIGMA and TLS 1.3 handshake protocols
- Session key agreement uses Diffie-Hellman scheme (ECDHE or FFDHE)
- Features mutual or one-way (Responder to Requester) authentication
- Features forward secrecy
- Features session key confirmation through VerifyData exchanges
- Requester capabilities: RSA and/or ECC, HMAC, RNG
- Responder capabilities: RSA or ECC, HMAC, RNG (if mutual authentication or forward secrecy is required)
- Responder examples: graphics card, SSD, FPGA

Pre-Shared Key Option: Introduction

- Pre-shared key (PSK) is a secret known to both the Requester and the Responder, before the session key exchange flow is executed
- Provisioning of PSK is out of scope of SPDM 1.1.
  - Implementer’s policy is also out of scope of SPDM 1.1.
- Responder benefits: low cost (HMAC + unique device secret or secure storage for PSK)
- Responder examples: integrated webcam, integrated fingerprint scanner, devices soldered on board, CPU, GPU, NIC
- Requester capabilities: HMAC, RNG, secure storage

Pre-Shared Key Option for Session Key Exchange

- Diagram below illustrates high-level sequence; arrows do not map to actual commands.
- Some provisioning schemes require Requester to send PSK_hint to Responder during session key exchange flow, so the Responder can derive PSK. Content of PSK_hint depends on the underlying PSK provisioning scheme and is out of scope of SPDM.
- Requester context and Responder context are described in diagram below.
- Features session key confirmation through VerifyData exchanges
- Session keys are derived from PSK and contexts.



Requester

Responder

(optional) derive PSK from PSK_hint

(optional) PSK_hint

Generate ReqContext – must include a random nonce; may also include Requester's info

Generate optional RspContext – may include random nonce and/or Responder's info

Exchange Contexts

Derive session keys from PSK, ReqContext, and RspContext

Derive session keys from PSK, ReqContext, and RspContext

Generate VerifyData with session MAC key.

Generate VerifyData with session MAC key.

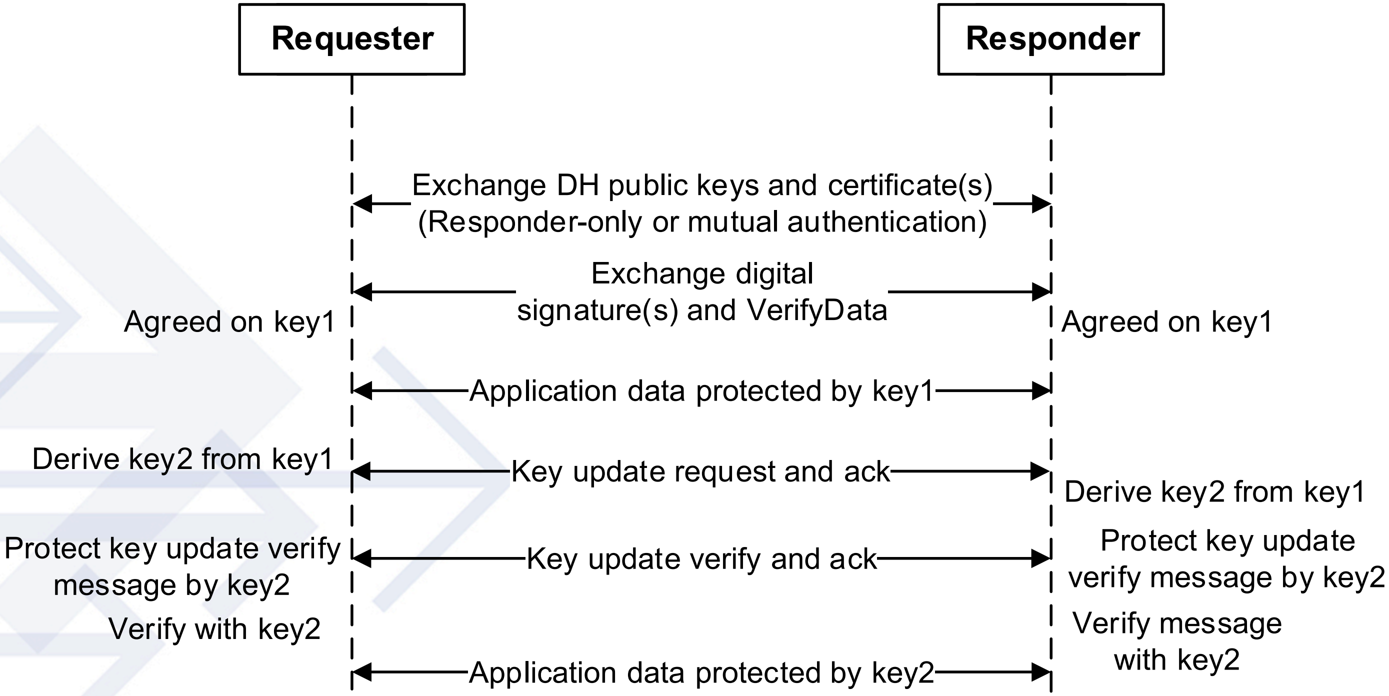
Exchange VerifyData

Verify Responder's VerifyData.

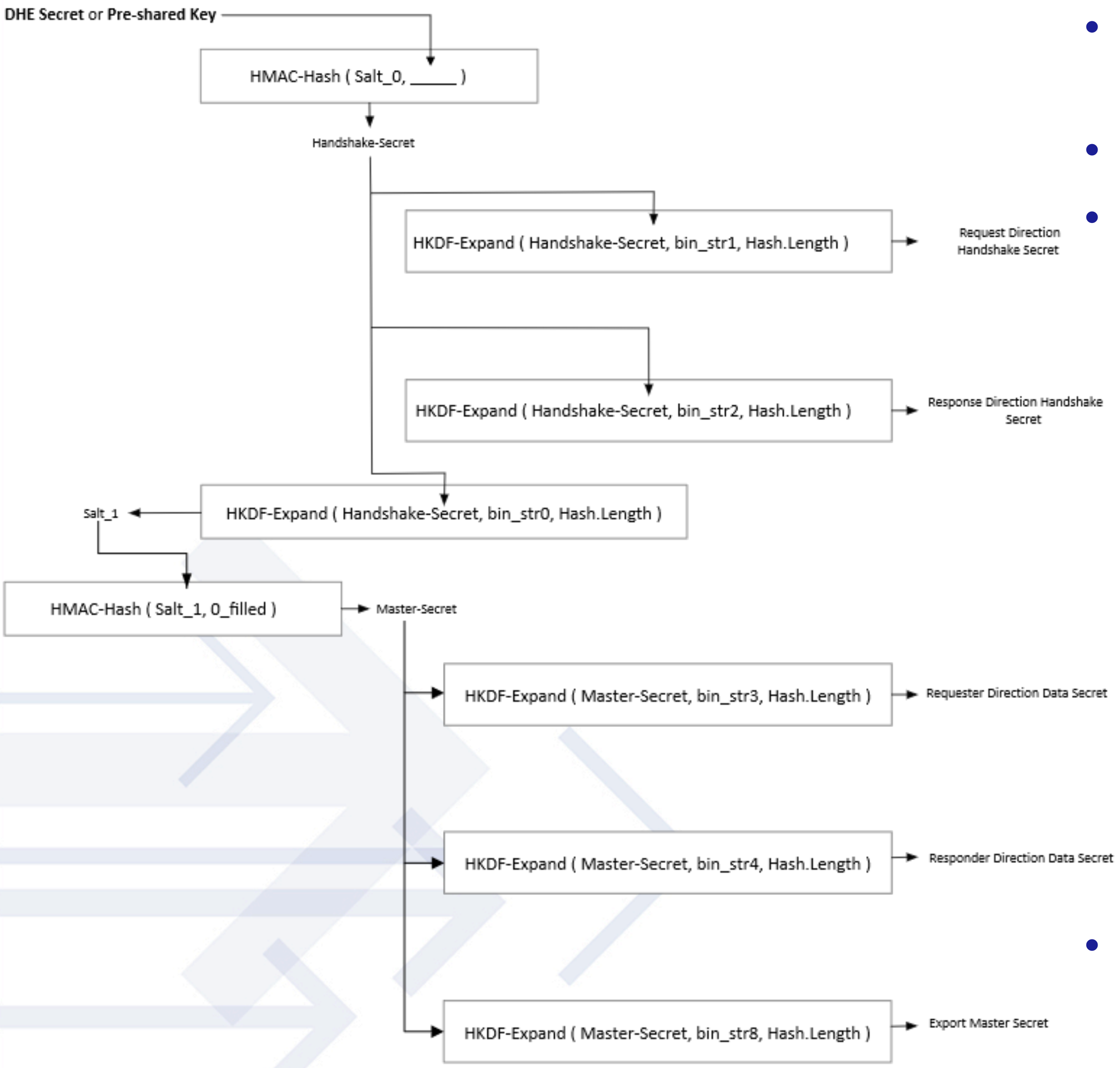
Verify Requester's VerifyData.

Key Update

- Either Requester or Responder may initialize a key update, updating the session keys to new values, for reasons such as counter overflow.
- No SIGMA or PSK. The new keys are derived from the current keys.
- The new keys are confirmed by both endpoints before used for protecting application data.


Requester
Responder
Exchange DH public keys and certificate(s)
(Responder-only or mutual authentication)
Exchange digital
signature(s) and VerifyData
Agreed on key1
Agreed on key1
Application data protected by key1
Derive key2 from key1
Key update request and ack
Derive key2 from key1
Protect key update verify
message by key2
Key update verify and ack
Protect key update
verify message by key2
Verify with key2
Verify message
with key2
Application data protected by key2

Key Schedule



DHE Secret or Pre-shared Key

HMAC-Hash (Salt_0, _____)

Handshake-Secret

HKDF-Expand (Handshake-Secret, bin_str1, Hash.Length)

Request Direction Handshake Secret

HKDF-Expand (Handshake-Secret, bin_str2, Hash.Length)

Response Direction Handshake Secret

Salt_1

HKDF-Expand (Handshake-Secret, bin_str0, Hash.Length)

HMAC-Hash (Salt_1, 0_filled)

Master-Secret

HKDF-Expand (Master-Secret, bin_str3, Hash.Length)

Requester Direction Data Secret

HKDF-Expand (Master-Secret, bin_str4, Hash.Length)

Responder Direction Data Secret

HKDF-Expand (Master-Secret, bin_str8, Hash.Length)

Export Master Secret

- Based on HMAC-Hash and HKDF-Expand
- Input: DHE secret or PSK
- Output:
  - **Handshake Secrets**: used to protect handshake messages
  - **Data Secrets**: used to protect application message
  - **Export Master Secret**: additional key derived for custom usages defined by vendor
- Different secrets for the two directions of communication, respectively

References

- PMCI Standards: where to find all the specs, white papers and presentations
  - https://www.dmtf.org/standards/pmci
- SPDM
  - DSP0274 (Security Protocol and Data Model (SPDM)): https://www.dmtf.org/dsp/DSP0274
  - DSP0275 (Security Protocol and Data Model (SPDM) over MCTP Binding Specification): https://www.dmtf.org/dsp/DSP0275
  - DSP0276 (Secured Messages using SPDM over MCTP Binding Specification): https://www.dmtf.org/dsp/DSP0276
  - DSP0277 (Secured Messages using SPDM Specification): https://www.dmtf.org/standards/pmci when released
  - DSP2058 (Security Protocol and Data Model (SPDM) Architecture White Paper): https://www.dmtf.org/standards/pmci when released

Backup