libspdm
Past, Present, and Future

Jiewen Yao, Intel
Steven Bellock, NVIDIA

Disclaimer

- The information in this presentation represents a snapshot of work in progress within the DMTF SPDM WG.
- 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.

Agenda

- Past ←
- Present
- Future

Beginning of libspdm

- The SPDM 1.0 specification was published at the end of 2019.
- Jiewen started the openspdm project as an SPDM proof-of-concept in May 2020.
  - Conceived as a submodule of the EDK II project.
  - Archived at https://github.com/jyao1/openspdm
- To foster interoperability between SPDM endpoints Jiewen donated openspdm to the DMTF at the beginning of 2021.
- Project was renamed to libspdm and was officially sponsored by the DMTF as the SPDM reference implementation.
  - Includes the establishment of the SPDM Code Task Force.

Early Work

- Early work on libspdm focused on
  - Disentanglement from EDK II to be a standalone code base.
  - Increased test coverage and build stability.
  - Fixing bugs with respect to the specification.
  - Formal verification, security review, and security hardening.
- Releases
  - 1.0 – December 2021
  - 2.0 – Apr 2022
  - 3.0 – July 2023
- Approximately four releases a year based on a quarterly schedule.
- Contributor statistics
  - 52 contributors, 1705 pull requests, 2362 commits, 8899 unique cloners

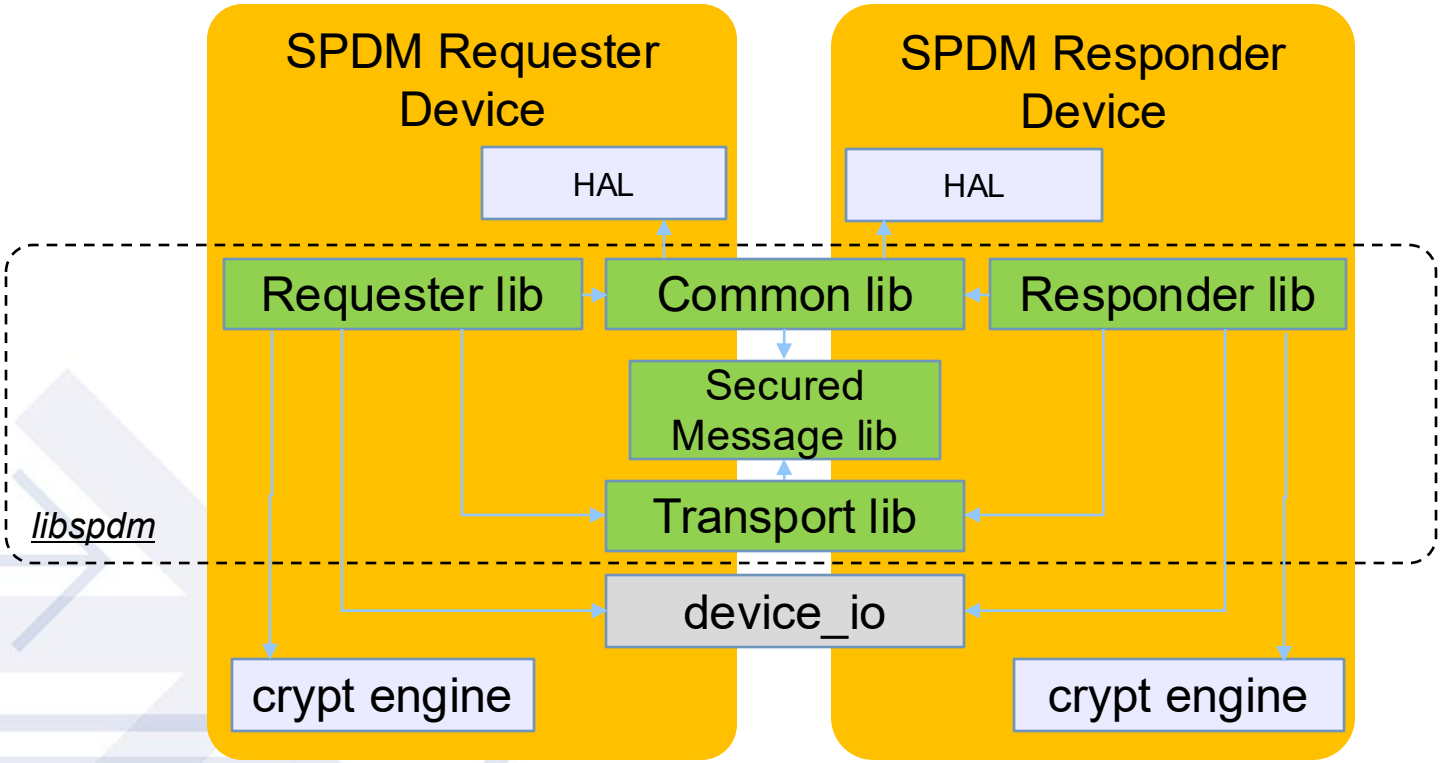
Agenda

- Past
- Present ←
- Future

Library Architecture

- Core libraries
  - Common (shared between Requester and Responder)
  - Responder
  - Requester
  - Secured Message (DSP0277)
  - Crypt - Includes bindings to OpenSSL and Mbed TLS.
  - Transport
    - MCTP (DSP0275/0276)
    - PCIe DOE
    - TCP (DSP0287)
    - Storage binding (DSP0286)
- Core libraries are written in C.

Library Architecture



Test and Verification

- libspdm is subjected to
  - Unit testing with weekly code coverage collection.
  - Random fuzz testing.
  - Static analysis - Coverity and CodeQL
  - Formal verification - CBMC.
  - Offensive security activities (Intel and NVIDIA)
  - Integration testing with SPDM emulator.
- Despite the above libspdm has had two CVEs.
  - DMTF-2023-0001
    - Bypass of mutual authentication.
  - DMTF-2023-0002
    - C undefined behavior.

Use of libspdm in Other Projects

- Other related DMTF projects
  - SPDM emulator - spdm-emu
  - SPDM message dump tool - spdm-dump
  - SPDM device validation tool - SPDM-Responder-Validator
- libspdm is leveraged by
  - TianoCore EDK II - Device Security
  - NVIDIA Linux kernel module driver - open-gpu-kernel-modules
  - QEMU – backend server
  - Arm Realm Management Monitor reference implementation - TF-RMM
  - TEE-IO device validation tool - tee-io-validator
  - Planned support in OpenBMC

SPDM Working Group and Specification

- Work on libspdm has led to clarifications and improvements in the SPDM specification.
  - Constant presence of nonce in measurement response.
  - Endianness of signatures and GCM sequence numbers.
  - Double hashing of messages in SPDM 1.0/1.1.
- Approximately 90 issues filed against the specification reference libspdm.
- Since GitHub hosts both the specification and the reference implementation it is easy to cross-reference issues and pull requests.
- SPDM Code Task Force also contributed to the formation of the Security Response Task Force to handle security issues across the entirety of the DMTF.

Agenda

- Past
- Present
- Future ←

Future of libspdm

- The SPDM Code Task force is working on **libspdm 4.0 release** to complete feature enablement of SPDM 1.3/1.4.
  - Multi-key support. (SPDM 1.3)
  - Post-quantum cryptography. (SPDM 1.4)
  - Architectural changes to reduce complexity and streamline integration.
- Future work will include implementation of the DMTF Authorization specification (DSP0289).