PSA Crypto and the headers dilemma

Antonio de Angelis, Open-Source Software group

20.04.23

PSA Crypto and the headers dilemma

- Problem scenario: an SDK can contain different implementations of the PSA Crypto APIs
  - e.g. nRF Connect, but others for sure have similar requirements
    - mbed TLS include path: /root/mbedtls/include
      - It has both include/mbedtls and include/psa
    - TF-M include path: /root/trusted-firmware-m/interface/include
      - It has include/psa
- PSA Crypto spec: an application just needs to perform #include “psa/crypto.h”
  - All the other headers are private and implementation-specific
  - Important for TF-M and mbed TLS because TF-M has caller isolation hence structs are different
- How to survive this so far:
  - Both include paths can be on $PATH, if the building environment can enforce the ordering consistently (if an application wants to use TF-M’s PSA Crypto, it always needs to appear first)
  - Example: an NS app in Zephyr that wants to use mbed TLS for the TLS stack, configures MBEDTLS_USE_PSA_CRYPTO and at the same time wants to use PSA Crypto APIs from TF-M to provide S/NS isolation etc

PSA Crypto and the headers dilemma: real world feedback

- Enforcing the ordering is not always reliable (build systems can get complex fast)
  - It would be better if one of the two paths for PSA selectively added:
    - This is currently not possible because both psa and mbedtls include directories share the same root in root/mbedtls/include.
    - Eventually might get there when the PSA Crypto repo is separated from mbed TLS and dependencies resolved
- The interface psa/crypto.h defines an API that is standardized
  - structs are IMPDEF, and they are in crypto_struct.h (private for the implementation)
  - Types are standard, and they are in crypto_types.h, but…
    - typedef uint32_t psa_key_id_t;
    - #if !defined(MBEDTLS_PSA_CRYPTO_KEY_ID_ENCODES_OWNER)
    - typedef psa_key_id_t mbedtls_svc_key_id_t;
    - #else /* MBEDTLS_PSA_CRYPTO_KEY_ID_ENCODES_OWNER */
    - typedef struct {…} mbedtls_svc_key_id_t;
    - #endif /* !MBEDTLS_PSA_CRYPTO_KEY_ID_ENCODES_OWNER */
  - mbedtls_svc_key_id_t is used in psa/crypto.h of mbed TLS, meaning that TF-M PSA Crypto headers need to carry a private type of mbed TLS in order for TLS code that uses PSA Crypto to work with TF-M
- Can be patched in existing deployments as long as mbedtls_svc_key_id_t is not different from a uint32_t.

PSA Crypto and the headers dilemma: actions so far

- Concentrate implementation specific behaviour into crypto_struct and crypto_platform
  - Try to align as much as possible from the other headers between Mbed TLS and TF-M
  - At build time, crypto_struct and crypto_platform can be set through compiler defines instead of relying on what is found on the INCLUDE_PATH
- Consistent use of mbedtls_svc_key_id_t as the key ID type
  - It's an alias of psa_key_id_t as specified by the standard towards the clients of the API
  - It remains a complex structure when PSA Crypto is used as the backend library of client-server architecture (i.e. in the TF-M Crypto service), but that is hidden from applications
  - This has been chosen to avoid disruption in the existing integrations of Mbed TLS
- ABI compatibility remains out of the scope for now but will likely be investigated in the future