Trusted Firmware-M Profile Medium

David Hu

Agenda

- Refresh memory
  - TF-M Profiles
  - TF-M Profile Small
- TF-M Profile Medium design
  - Feature list
  - Details
- Profile Medium implementation proposal
- Current status

Refreshing memory

- TF-M Profiles
  - Challenges
    - Dramatic variation in device capabilities and usage scenarios
    - Diverse requirements on levels of security
  - Predefined lists of base profiles
    - Profile *Small*, Profile *Medium*, Profile *Large*
    - Target towards typical use cases with different hardware constraints
    - Alignment with PSA specifications and certification requirements

Refreshing memory

- TF-M Profile Small
  - Usage scenarios
    - Ultra-constrained resource devices
    - Simple service model and applications
    - Connection with Edge Gateway and IoT Cloud Services with symmetric cryptography
  - Feature
    - Smallest footprint
    - Lightweight framework
    - Symmetric cipher suite
    - Internal Trusted Storage only by default
  - Already supported in TF-M
  - Design document
    - Link

TF-M Profile Medium Design

- Usage scenario
  - Resource-constrained devices
    - More capable devices compared to Profile *Small* targets
  - Connect devices to IoT Cloud Services *directly* with *asymmetric* cipher support
  - Secure world and normal world are managed by different participants respectively

TF-M Profile Medium Design (cont'd)

- Major feature List
  - Firmware Framework
    - Inter-Process Communication (IPC) model
    - Level 2 isolation
  - Internal Trusted Storage (ITS)
  - Crypto
    - Asymmetric cipher suite
  - Asymmetric key algorithm based Initial Attestation
  - Multiple image boot
  - Protected Storage (PS) if off-chip storage device is integrated

Design details

- Firmware Framework
  - Aim to support more complicated secure service model and additional protection to PSA RoT, compared to Profile *Small*
  - Require more resource and configurations than Profile *Small* does
    - Larger footprint
    - Longer latency
  - Level 2 isolation
    - PSA RoT is protected from access by the App RoT
  - IPC model
    - Support higher level of isolation

Design details (cont'd)

- Crypto
  - Asymmetric cipher suite `TLS_ECDHE_ECDSA_WITH_AES_128_CCM` *by default*
    - ECDHE_ECDSA as key exchange algorithm
    - AES-128-CCM as AEAD algorithm
      - AES-128-CCM with truncated authentication tag to save bandwidth in networking

  - Digital Signature
    - ECDSA with ECC curve `secp256r1` by default

  - It is recommended to share the same algorithm among multiple application/secure services
    - Digital signature: Networking, Initial Attestation
    - AEAD: Networking, PS service

  - Default cipher suite can be replaced according to
    - Actual use cases
    - Crypto HW features

Design details (cont’d)

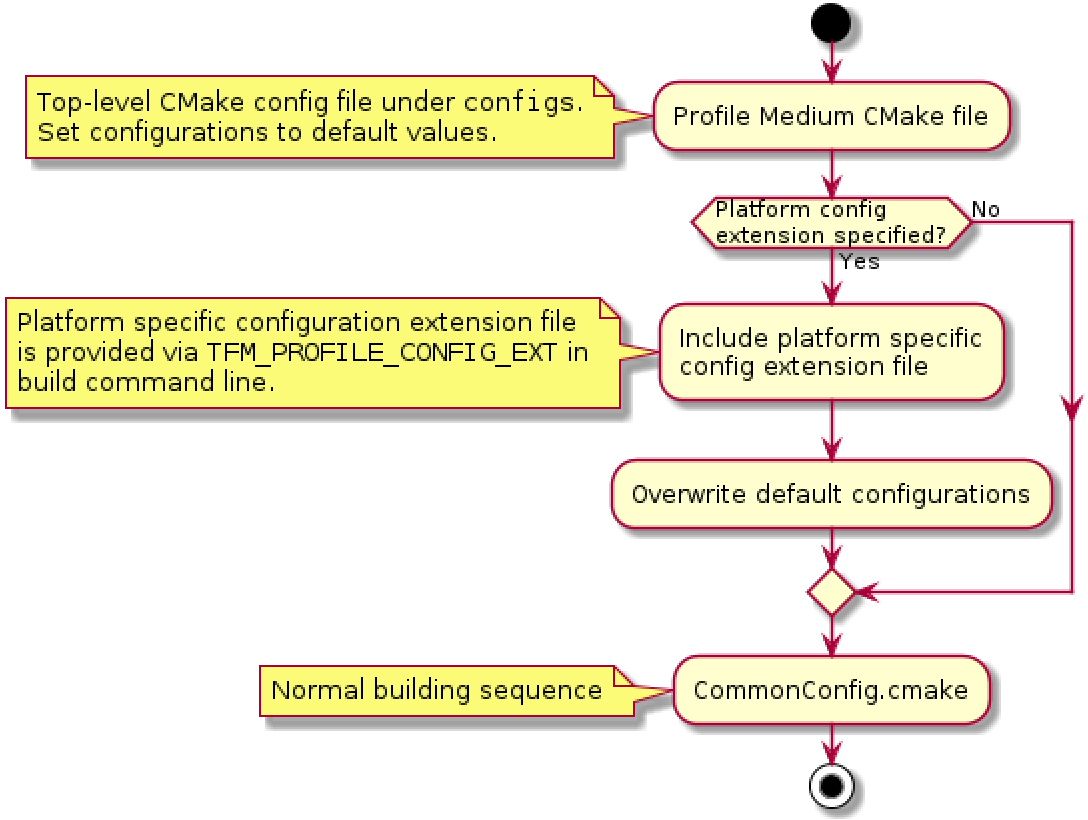
- BL2
  - Implementation defined and platform specific
  - Anti-rollback protection is still required
  - Multiple image boot is selected by default in TF-M MCUBoot
    - Secure and normal images can be signed independently with different keys and updated separately
    - Support multiple vendor scenarios, in which different participants create/update secure and normal images

Implementation proposal

Build flow overview

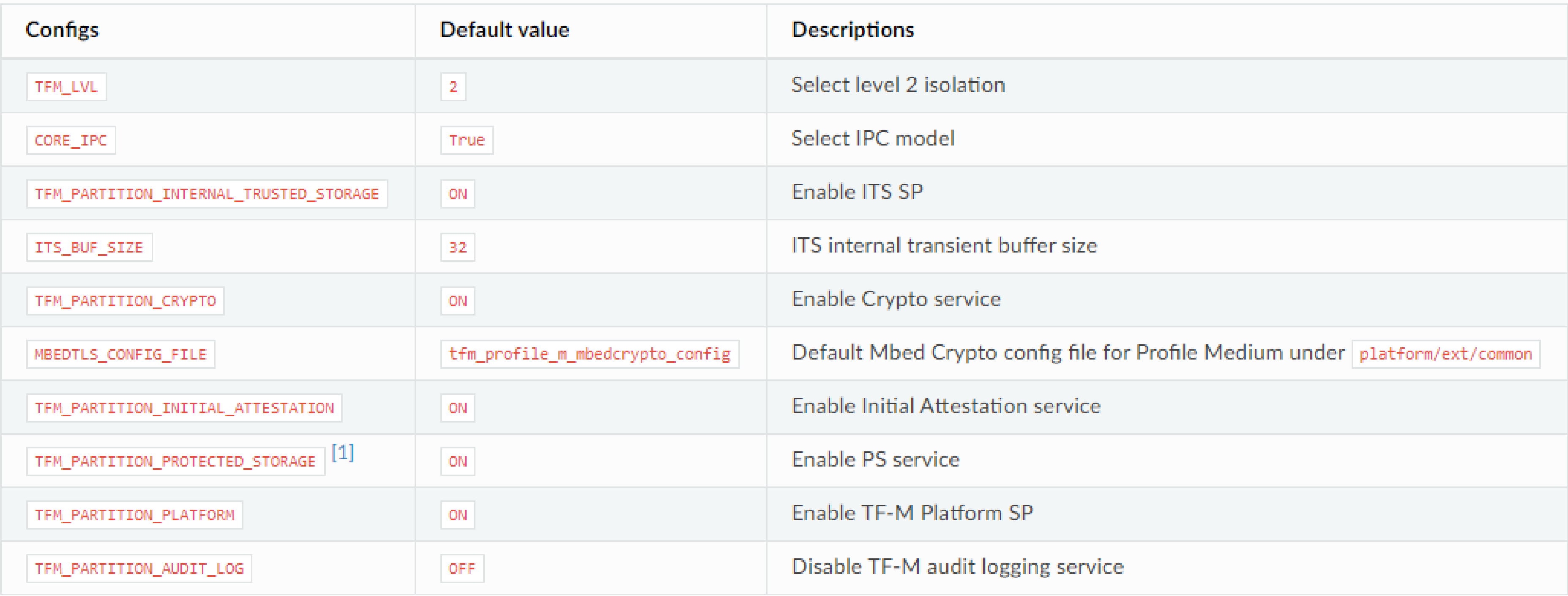
- Identical to that in Profile *Small*
- A top-level CMake config file collects all the config flags and set them to default values
  - `ConfigDefaultProfileM.cmake`
  - More convenient for partners to understand and overwrite default settings.
- A platform can overwrite default values in its config extension file via `TFM_PROFILE_CONFIG_EXT`

Overall build flow


Profile Medium CMake file
Top-level CMake config file under configs. Set configurations to default values.
Platform config extension specified?
No
Yes
Include platform specific config extension file
Platform specific configuration extension file is provided via TFM_PROFILE_CONFIG_EXT in build command line.
Overwrite default configurations
CommonConfig.cmake
Normal building sequence

Implementation proposal (cont’d)

Major options configuration in Profile Medium top-level CMake file



Configs	Default value	Descriptions
`TFM_LVL`	`2`	Select level 2 isolation
`CORE_IPC`	`True`	Select IPC model
`TFM_PARTITION_INTERNAL_TRUSTED_STORAGE`	`ON`	Enable ITS SP
`ITS_BUF_SIZE`	`32`	ITS internal transient buffer size
`TFM_PARTITION_CRYPTO`	`ON`	Enable Crypto service
`MBEDTLS_CONFIG_FILE`	`tfm_profile_m_mbedcrypto_config`	Default Mbed Crypto config file for Profile Medium under `platform/ext/common`
`TFM_PARTITION_INITIAL_ATTESTATION`	`ON`	Enable Initial Attestation service
`TFM_PARTITION_PROTECTED_STORAGE` [1]	`ON`	Enable PS service
`TFM_PARTITION_PLATFORM`	`ON`	Enable TF-M Platform SP
`TFM_PARTITION_AUDIT_LOG`	`OFF`	Disable TF-M audit logging service

[1] PS service is enabled by default. Platforms without off-chip storage devices can turn off `TFM_PARTITION_PROTECTED_STORAGE` to disable PS service.

Implementation proposal (cont'd)

- Details
  - TF-M Crypto service
    - Mbed Crypto configurations
      - Default Mbed Crypto config file `tfm_profile_m_mbedcrypto_config.h`
      - Select CCM mode by default
        - Enable optimization to skip CCM decrypt part to decrease memory footprint
      - Default configs can be modified by platform specific Mbed Crypto configs
        - Replace the default `tfm_profile_m_mbedcrypto_config.h` with platform specific config file
        - Overwrite default configs via `MBEDTLS_USER_CONFIG_FILE`

Implementation proposal (cont'd)

- Details
  - TF-M PS service
    - Enabled by default in top-level CMake file
      - For test purpose
      - TF-M Platform secure partition is enabled by default to provide Non-Volatile Counters to PS service
        - Support anti-rollback protection in PS
    - Adjustment to enable selecting AEAD algorithm
      - Profile Medium explicitly selects AES-CCM by default
    - Platform without off-chip storage device can disable PS service by
      - Turning off `TFM_PARTITION_PROTECTED_STORAGE` in extension file via `TFM_PROFILE_CONFIG_EXT`
        - An example `profile_m_config_ext_ps_disabled.cmake` which disables PS service is provided
      - Hacking Profile Medium top-level CMake directly to turn off `TFM_PARTITION_PROTECTED_STORAGE`
        - In local development

Implementation proposal (cont'd)

- Enable Profile Medium on a platform
  - Add the platform into the support list in Profile Medium top-level CMake file
    - Default configuration: `ConfigDefaultProfileM.cmake`
    - Regression tests: `ConfigRegressionProfileM.cmake`
  - Overwrite the default settings in its configuration extension file if necessary
  - Build as usual, specifying the Profile Medium config
    - Build with default configs

Note: The following build commands are executed in current build system. Commands may vary when a new TF-M build system is deployed

```
cmake -G"Unix Makefiles" -DPROJ_CONFIG=`readlink -f ../configs/ConfigDefaultProfileM.cmake` \
                         -DTARGET_PLATFORM=${PLATFORM}     \
                         -DCMAKE_BUILD_TYPE=${BUILD_TYPE} \
                         -DCOMPILER=${COMPILER} ../
cmake --build ./ -- install
```

- Build with platform specific config extension

```
cmake -G"Unix Makefiles" -DPROJ_CONFIG=`readlink -f ../configs/ConfigDefaultProfileM.cmake` \
                         -DTARGET_PLATFORM=${PLATFORM}     \
                         -DCMAKE_BUILD_TYPE=${BUILD_TYPE} \
                         -DCOMPILER=${COMPILER} \
                         -DTFM_PROFILE_CONFIG_EXT=${PLATFORM_CONGI_EXT} ../
cmake --build ./ -- install
```

Current status

- Profile Medium design document under review
  - Link

- Profile Medium implementation under review
  - Patch set

Comments are welcome!

arm

Thank You

Danke

Merci

谢谢

ありがとう

Gracias

Kiitos

감사합니다

धन्यवाद

شكرًا

ধন্যবাদ

תודה