Confidential A

MTK In-House TEE Overview & Customization Guide (Type B Customer)

v0.2

2014/2/23
MTK

Outline

- Revision
- Introduction
- Document Suit
- MTEE Architecture

Revision

Rev.	Date	Author	Description
0.1	2013/8/14	KL Huang	Initiative
0.2	2014/2/23	KL Huang	Support KK and AOSP

Introduction

- The document is applied for
  - MTK In-House Trusted Execution Environment (MTEE)
  - MTK SoCs and SW versions
    - MT8135
      - JB on Turnkey and AOSP
      - KK on Turnkey and AOSP
  - Type B Customer
    - Can get the encrypted and signed MTEE image
- The purpose of the document introduce the MTEE architecture and related information to customer
  - After reading this document, it is expected that customer will
    - Understand document suit of MTEE
    - Have background and understanding for MTEE architecture

Document Suit

- Documents related to MTEE includes
  - MTK In-House TEE - Overview & Customization Guide (Type B Customer)
    - Introduce MTEE architecture and customization

MTEE Architecture

- Terminology
  - Execution Environment (EE)
    - SW execution environment with HW resources (CPU, memory, peripheral devices, …) and SW infrastructure (OS, Core SW, …)
  - Rich Execution Environment (REE, Normal World)
    - Usually refer to normal OS (like Linux, Window, Unix, …) which all HW resources can be accessed in different permission
    - More open environment for SW to run
    - Not easy to apply security constraint on it
    - SW, executing in it, is easy to be hacked
  - Trusted Execution Environment (TEE, Secure World)
    - More secure execution environment
    - SW, executing in it, is separated from SW executing in REE in the following aspects.
      - CPU state/Memory space/HW Resources/Permission/SW infrastructure/…
    - Established by co-working with HW and SW

MTEE Architecture

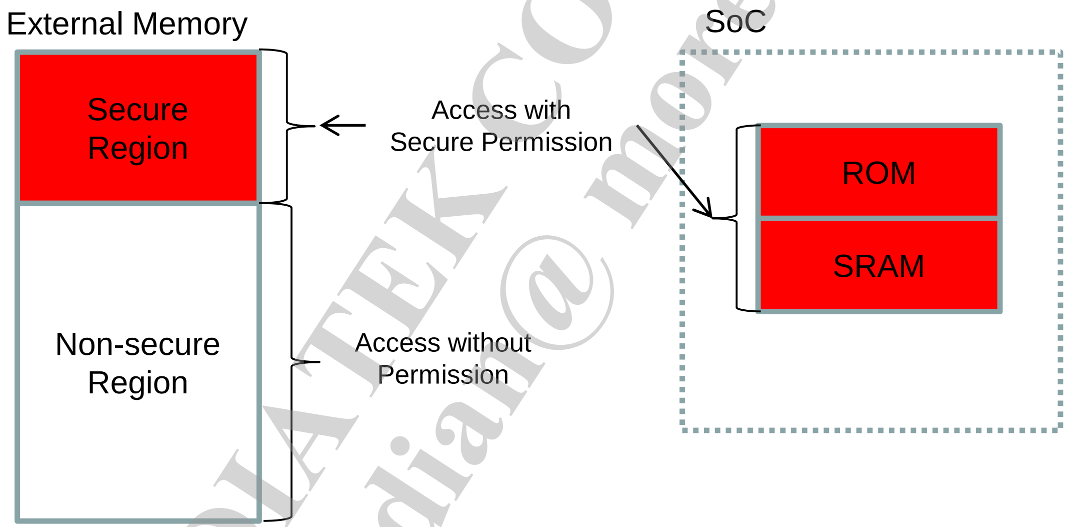
- Client Application (CA)
  - SW, in REE, uses REE-TEE mechanism to use the functionality of the SW in TEE or communicate with SW in TEE
    - CA could be in user space or kernel space of REE (such as Linux)
- Trusted Application (TA)
  - SW, in TEE, uses REE-TEE mechanism to use the functionality of the SW in REE or communicate with SW in REE
- MTK In-House TEE (MTEE)
  - Proprietary TEE implementation by MTK
  - Based on
    - ARM TrustZone Technology
    - MTK HW/SW implementation

MTEE Architecture

- SW Principle
  - Using ARM Instruction and ARM CPU state to separate the TEE/REE
  - Secure memory region is configured and can only be accessed in by secure access or SW in TEE
    - External memory (DRAM) or SRAM or ROM
    - Contains code/stack/heap/data in TEE
    - Refer to next slide
  - TEE SW environment is prepared before
    - CPU state is switched to non-secure state, and
    - REE SW environment is setup
- HW Principle
  - Register of some HWs related to security are designed tp
    - Be accessed by SW in TEE
    - Be configured to be accessed by SW in TEE
      - The configuration can only be done by SW in TEE
  - Memory can be separated into secure region and non-secure region
    - Configure HW related to memory protection
  - Some HWs direct memory access (DMA) can be configured to perform secure access
    - The configuration is done by SW in TEE

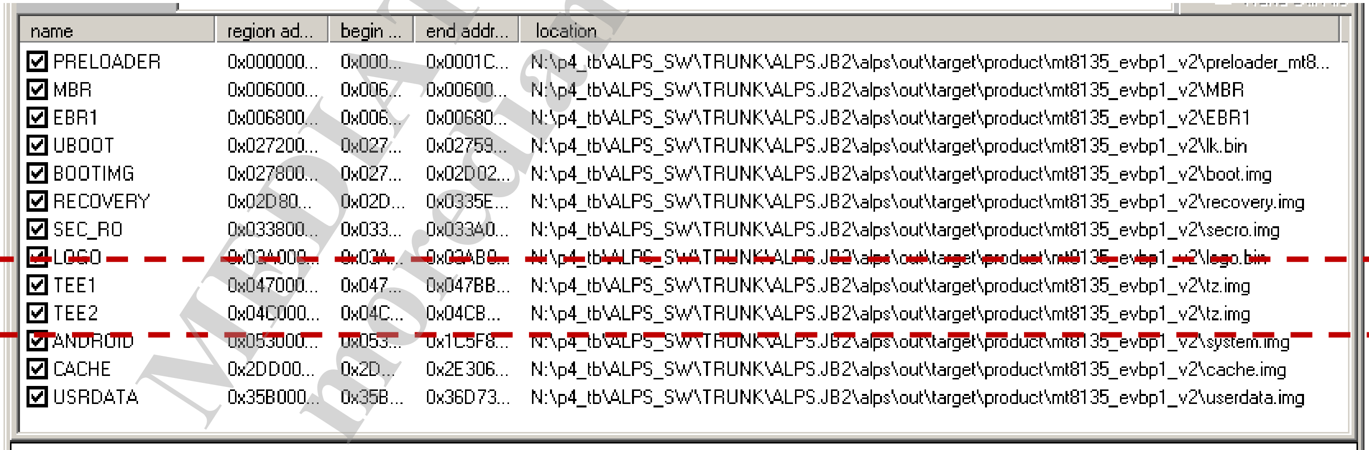
MTEE Architecture

- Memory layout with different access permission
  - The secure memory region is statically pre-configured and pre-configured in compiling time



MTEE Architecture

- Secure boot flow for MTEE images
  - TEE SW environment and SW is verified first and then loaded by Preloader
    - TEE images are signed and encrypted on the host PC
    - Target platform will use public key to verify the signature and then decrypt the image
  - OTA and Fastboot supports MTEE image upgrading for TEE1/TEE2 partition
  - Platform supports 2 partitions for MTEE image
    - MTEE1/MTEE2 partitions
      - Downloaded by SP Flash Tool (refer figure below)
    - Use tz.img for TEE1/TEE2 partitions



name	region ad...	begin ...	end addr...	location
☑ PRELOADER	0x000000...	0x000...	0x0001C...	N:\p4_tb\ALPS_SW\TRUNK\ALPS.JB2\alps\out\target\product\mt8135_evbp1_v2\preloader_mt8...
☑ MBR	0x006000...	0x006...	0x00600...	N:\p4_tb\ALPS_SW\TRUNK\ALPS.JB2\alps\out\target\product\mt8135_evbp1_v2\MBR
☑ EBR1	0x006800...	0x006...	0x00680...	N:\p4_tb\ALPS_SW\TRUNK\ALPS.JB2\alps\out\target\product\mt8135_evbp1_v2\EBR1
☑ UBOOT	0x027200...	0x027...	0x02759...	N:\p4_tb\ALPS_SW\TRUNK\ALPS.JB2\alps\out\target\product\mt8135_evbp1_v2\lk.bin
☑ BOOTIMG	0x027800...	0x027...	0x02D02...	N:\p4_tb\ALPS_SW\TRUNK\ALPS.JB2\alps\out\target\product\mt8135_evbp1_v2\boot.img
☑ RECOVERY	0x02D80...	0x02D...	0x0335E...	N:\p4_tb\ALPS_SW\TRUNK\ALPS.JB2\alps\out\target\product\mt8135_evbp1_v2\recovery.img
☑ SEC_RO	0x033800...	0x033...	0x033A0...	N:\p4_tb\ALPS_SW\TRUNK\ALPS.JB2\alps\out\target\product\mt8135_evbp1_v2\secro.img
☑ LOGO	0x03A000...	0x03A...	0x03AB0...	N:\p4_tb\ALPS_SW\TRUNK\ALPS.JB2\alps\out\target\product\mt8135_evbp1_v2\logo.bin
☑ TEE1	0x047000...	0x047...	0x047BB...	N:\p4_tb\ALPS_SW\TRUNK\ALPS.JB2\alps\out\target\product\mt8135_evbp1_v2\tz.img
☑ TEE2	0x04C000...	0x04C...	0x04CB...	N:\p4_tb\ALPS_SW\TRUNK\ALPS.JB2\alps\out\target\product\mt8135_evbp1_v2\tz.img
☑ ANDROID	0x053000...	0x053...	0x1C5F8...	N:\p4_tb\ALPS_SW\TRUNK\ALPS.JB2\alps\out\target\product\mt8135_evbp1_v2\system.img
☑ CACHE	0x2DD00...	0x2D...	0x2E306...	N:\p4_tb\ALPS_SW\TRUNK\ALPS.JB2\alps\out\target\product\mt8135_evbp1_v2\cache.img
☑ USRDATA	0x35B000...	0x35B...	0x36D73...	N:\p4_tb\ALPS_SW\TRUNK\ALPS.JB2\alps\out\target\product\mt8135_evbp1_v2\userdata.img

MEDIATEK

www.mediatek.com