Building a Product with OP-TEE

Possible pitfalls while deploying OP-TEE in production

Rouven Czerwinski– r.czerwinski@pengutronix.de

About me

Rouven Czerwinski

Pengutronix e.K.

Emantor

rcz@pengutronix.de

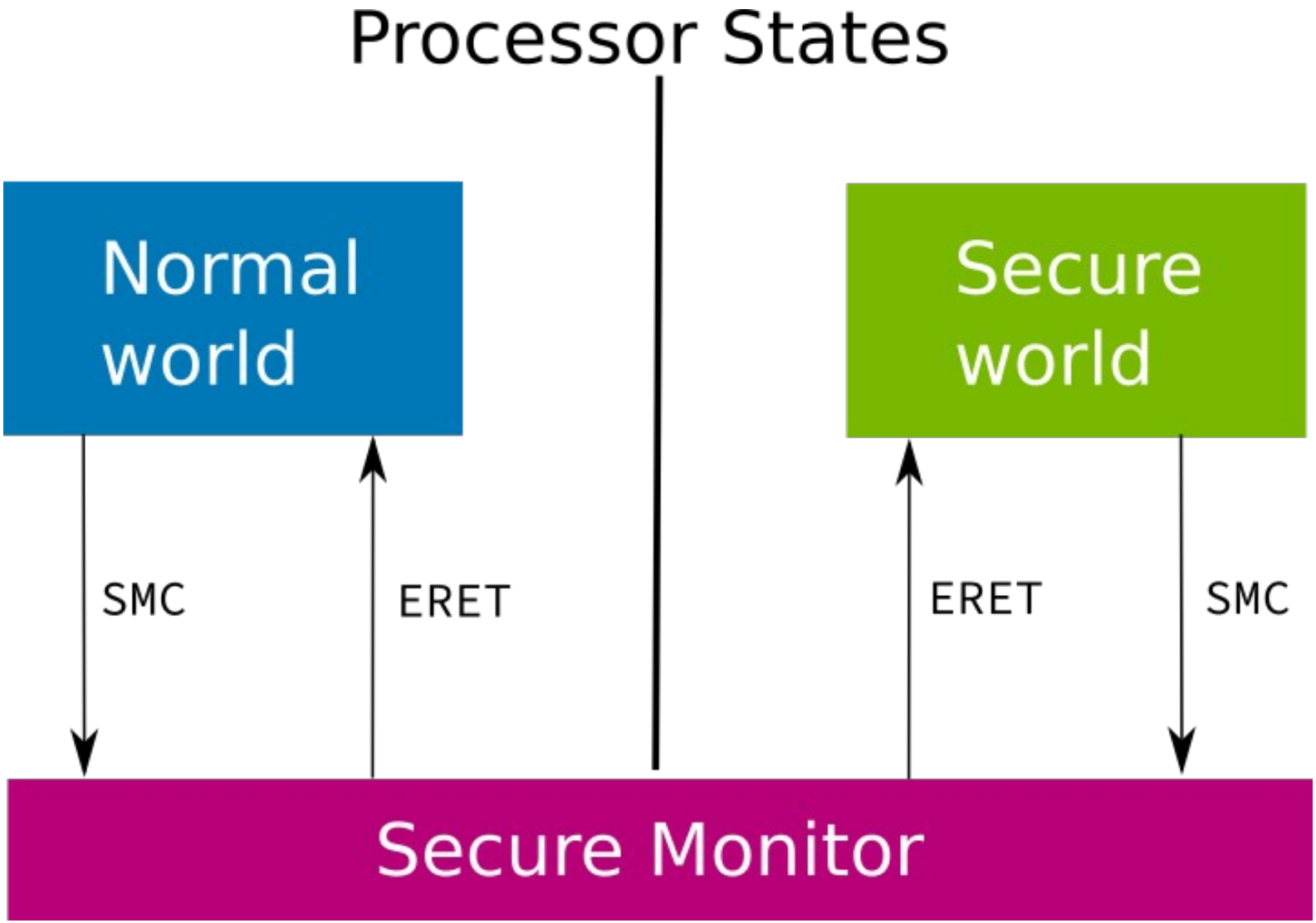
- OP-TEE
- System Integration
- Testing

Table of Contents

Short overview:

- Introduction
- Motivation
- Problems
- Solutions
- Conclusion
- Outlook

TrustZone (32-bit)



Introduction

- Open Portable Trusted Execution Environment (OP-TEE)
- Open source (BSD-2 clause) implementation of the GP TEE specification using TrustZone
- Support for various ARM platforms (STM32, TI, Layerscape, broadcom,...)
- My focus is on i.MX6 platforms

Motivation

- Secure the OP-TEE and TAs for production use
- Ensure that upstream OP-TEE can be used securely on i.MX6
- Provide guidance which parts may be missing for other platforms (TI, STM, Layerscape,...)

Problem

- Which components do I need to secure OP-TEE?
- Which part of the configuration is already upstream?
- Which part needs to be managed by system integrator?

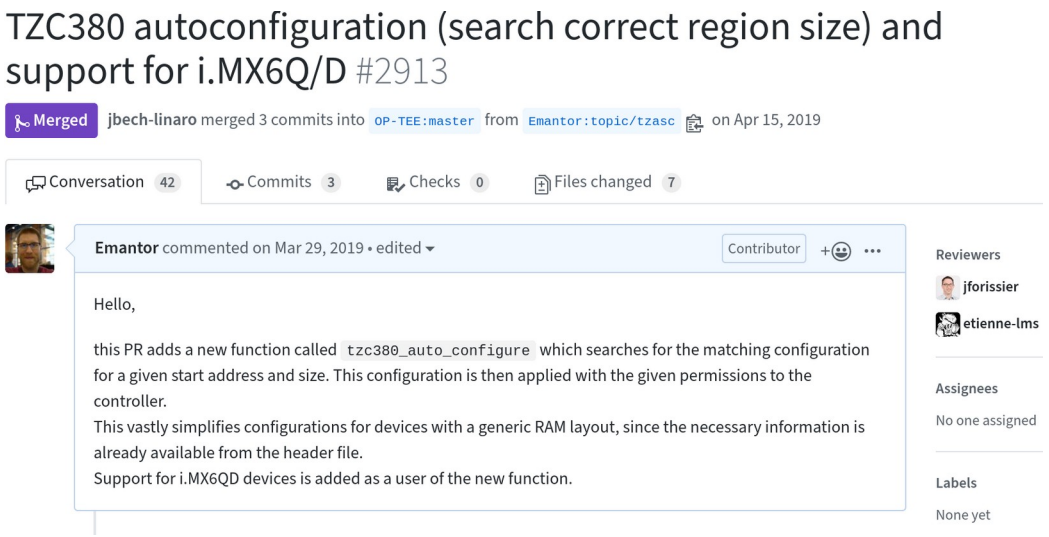
Securing upstream OP-TEE

- RAM protection/Pager
- Hardware Unique Key (HUK)
- RNG Seeding
- Peripheral Access Configuration
- Ensure trusted OP-TEE bootup
- Optional: storage rollback protection

RAM protection

- Configure the DDR firewall
- Protects part of RAM for secure world
- i.e. TZC380 with multiple regions
- For i.MX6:
  - TZC380 from ARM
  - Upstream driver already within OP-TEE

i.MX6 TZC380 autoconfiguration



TZC380 autoconfiguration (search correct region size) and support for i.MX6Q/D #2913

Merged jbech-linaro merged 3 commits into OP-TEE:master from Emantor:topic/tzasc on Apr 15, 2019

Conversation 42 Commits 3 Checks 0 Files changed 7

Emantor commented on Mar 29, 2019 • edited Contributor

Hello,

this PR adds a new function called `tzc380_auto_configure` which searches for the matching configuration for a given start address and size. This configuration is then applied with the given permissions to the controller.
This vastly simplifies configurations for devices with a generic RAM layout, since the necessary information is already available from the header file.
Support for i.MX6QD devices is added as a user of the new function.

Reviewers
jforissier
etienne-lms

Assignees
No one assigned

Labels
None yet

- TZC380 auto configuration upstream
- Correctly configures TZC380 for generic RAM devices with known memory layout

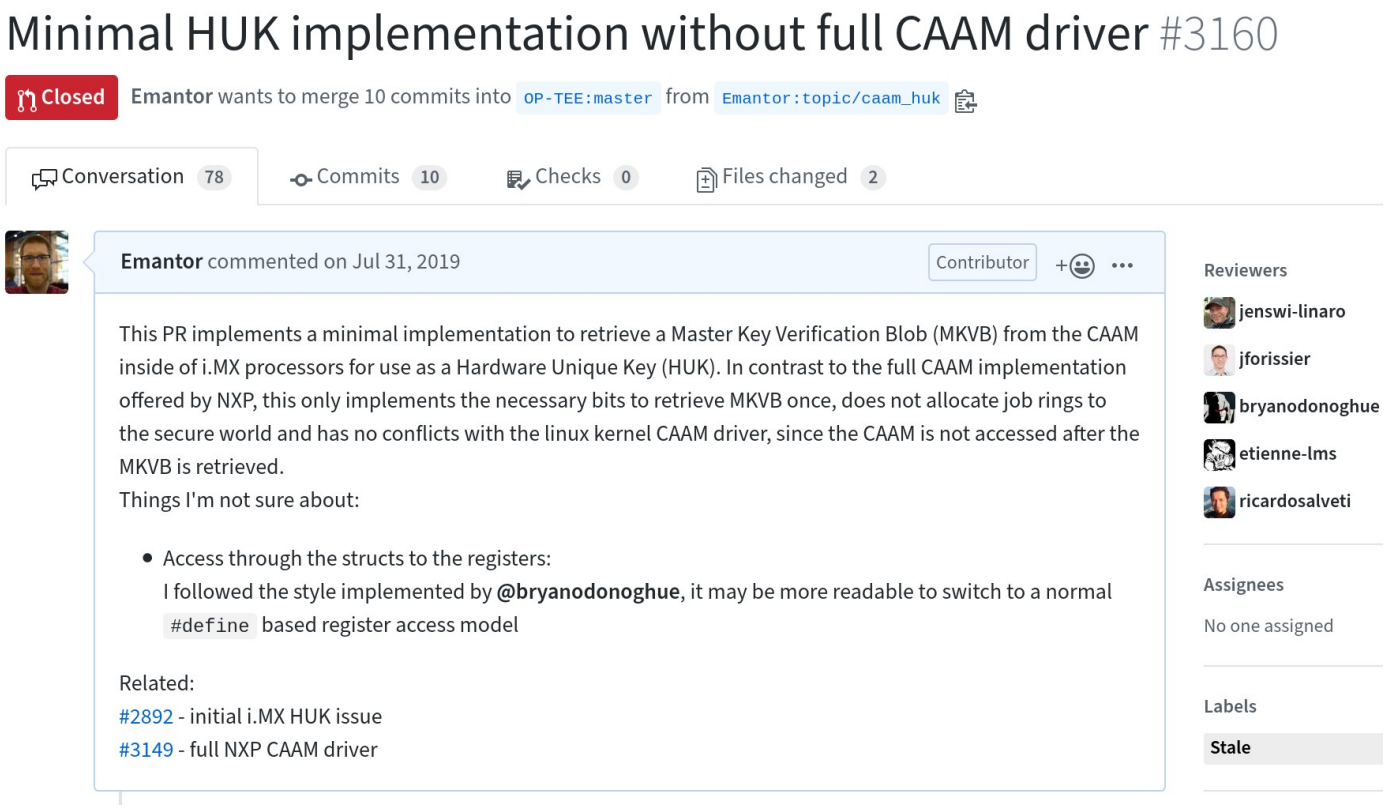
OP-TEE Pager

- Run small part of OP-TEE in SRAM
- Encrypt other memory pages live in DRAM
- Does not require a DDR firewall
- For i.MX6:
  - Chosen i.MX6UL may not have enough SRAM
  - Bigger variants may use SRAM for other use cases (IPU, GPU,...)

Hardware Unique Key (HUK)

- Used to derive other keys for OP-TEE
- Should be unique per device
- Should not be accessible from normal world
- For i.MX6:
  - Use CAAM Master Key Verification Blob (MKVB) and lockout generation afterwards

i.MX6 HUK generation



Minimal HUK implementation without full CAAM driver #3160

Closed **Emantor** wants to merge 10 commits into `OP-TEE:master` from `Emantor:topic/caam_huk`

Conversation 78 Commits 10 Checks 0 Files changed 2

Emantor commented on Jul 31, 2019 Contributor

This PR implements a minimal implementation to retrieve a Master Key Verification Blob (MKVB) from the CAAM inside of i.MX processors for use as a Hardware Unique Key (HUK). In contrast to the full CAAM implementation offered by NXP, this only implements the necessary bits to retrieve MKVB once, does not allocate job rings to the secure world and has no conflicts with the linux kernel CAAM driver, since the CAAM is not accessed after the MKVB is retrieved.
Things I'm not sure about:

- Access through the structs to the registers:
  I followed the style implemented by **@bryanodonoghue**, it may be more readable to switch to a normal `#define` based register access model

Related:
#2892 - initial i.MX HUK issue
#3149 - full NXP CAAM driver

Reviewers
jenswi-linaro
jforissier
bryanodonoghue
etienne-lms
ricardosalveti

Assignees
No one assigned

Labels
Stale

- Needs rebase on i.MX6/7 CAAM driver
- Will be done soon™

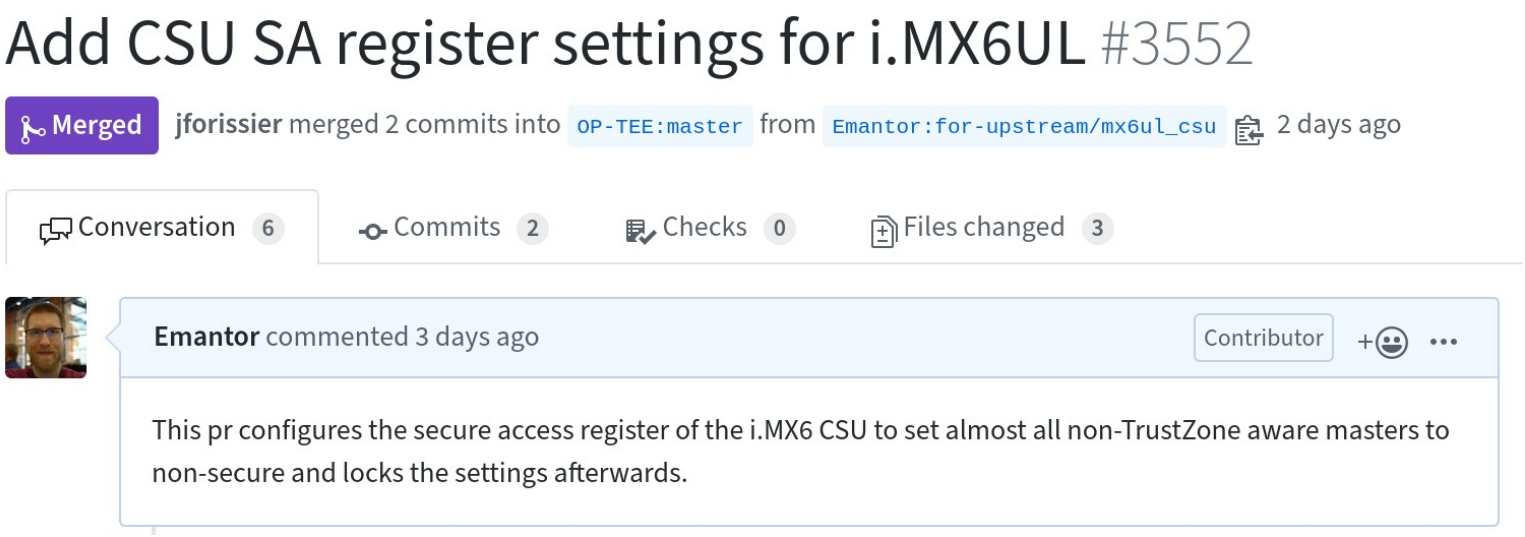
RNG seeding

- OP-TEE uses FORTUNA PRNG
- Requires RNG seed
- Default seed for dev is zero
- For i.MX6:
  - Retrieve RNG from CAAM TRNG on bootup
  - Not implemented yet

Peripheral Access Configuration

- SoCs have DMA masters beside CPU
- Those masters may be default secure and can access secure world memory
- For i.MX6:
  - Access policies configurable via Central Security Unit (CSU)

i.MX6 CSU



- Upstream configures correctly for i.MX6UL
- Other i.MX6/7 SoCs trivial to add (given Security Reference Manual)

Trusted Bootup

- Use platform verified/secure boot
- Verifies OP-TEE version to prevent replacements
- For i.MX6:
  - Implement High Assurance Boot (HAB), also required for HUK
  - Not implementable upstream, needs to be handled by integrator

Storage Rollback protection

- To protect from rollback attacks, employ eMMC RPMB FS
- Simple FAT filesystem
- For all platforms:
  - Enable with CFG_RPMB_FS=1
  - Deploy during manufacturing with CFG_RPMB_WRITE_KEY=1
  - Ensure to disable emulation in TEE Supplicant with RPMB_EMU=0
  - Support upstream

Conclusion

- No platform is currently ready to deploy OP-TEE in production
  - i.MX6 is slowly getting there
- Vendor implementations may include the necessary bits
  - Still requires code review and cross reference to platform manual

Outlook (Wishlist)

- Clock and access coordination between OP-TEE and Linux
- Deeper device tree integration for OP-TEE
- CI infrastructure to test each commit to OP-TEE master for i.MX6/7

Thank you

Questions?