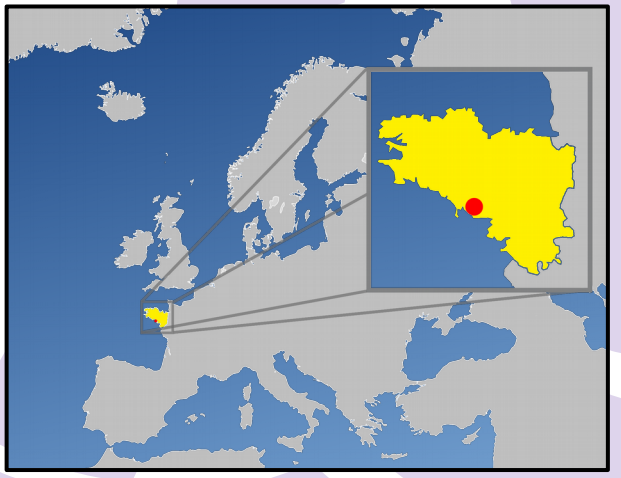
IOT BZH

Secure boot
Secure software update

Yannick Gicquel
SW Engineer
yannick.gicquel@iot.bzh

- Specialized on Embedded & IoT
- Contributing to AGL Project for Renesas
- Expertise domains:
  - System architecture
  - Security
  - Application Framework
  - Graphics & Multimedia
  - Middleware
  - Linux Kernel
- Located in Brittany, France



Agenda

1. Overall context of updates for cars

- Updates characteristics,
- Security requirements,

2. Secure boot

- Concept,
- U-Boot signature,

3. Enforcement solution

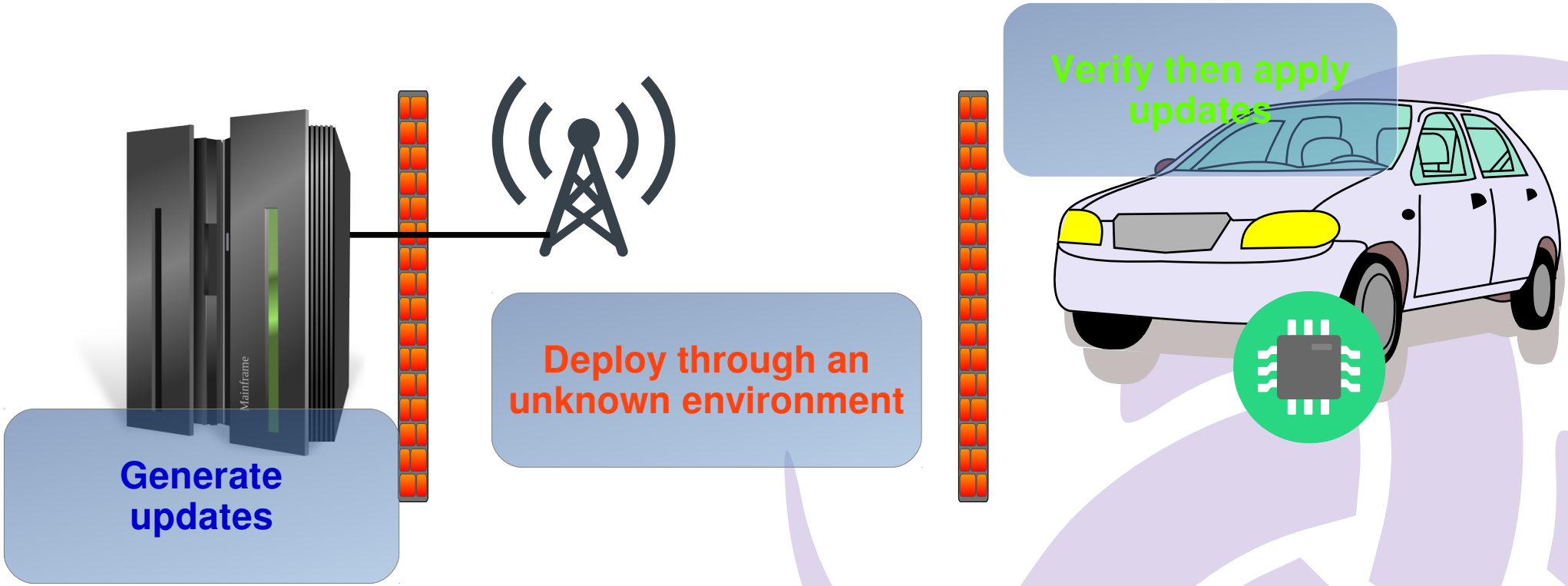
- Trusted Execution Environment
- OP-TEE

What are updates?

- In software engineering:
  - Deploy another revision of an application or service,
  - New features activation or enhancements,
  - Zero-day security fixes,
- In automotive:
  - Multiples programmable sub-systems,
  - Local updates (usb-stick, dvd) or remotes updates,
  - IVI systems as a update gateway for other components,

Updates infrastructure

Connected cars needs a secured infrastructure,



Security should tight each stages to a whole process,

Requirements for secure update

- **Reliable update agent**
  - Resilient to some technicals failures,
  - Ensure the update process won't break the car systems,
  - Otherwise, *safety* issues can occurs,
- **Trusted infrastructure**
  - Deployed updates should be *authenticated,*
  - Updates *integrity* should be checked before being applied,
  - *Confidentiality* should be ensured,

1. Overall context of updates for cars
   - Requirements,
   - Nature of updates

2. Secure boot
   - Concept,
   - U-Boot signature,

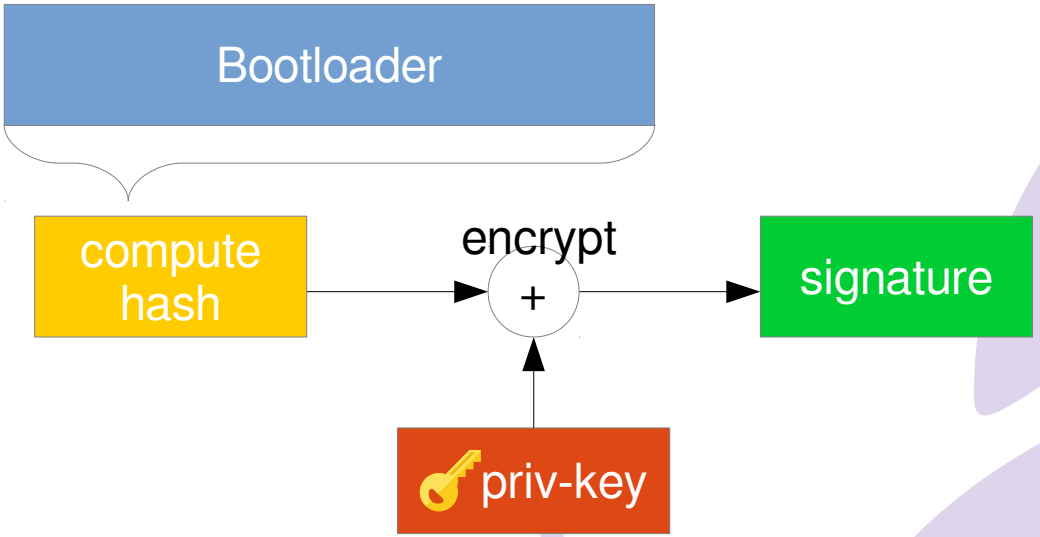
3. Enforcement solution
   - Trusted Execution Environment
   - OP-TEE

Secure boot

- **Feature**
  - Establish a root of trust to ensure the integrity of the whole software stack,
- **How?**
  - Using cryptography and signatures of digital contents,
  - At generation: Signing software,
  - At runtime: Verify all signatures,
- **Scope**
  - From hardware power-on to kernel startup,
  - Following secure boot: RootFS integrity, (dm-verity, dm-integrity, linux ima/evm)

Secure boot: signing

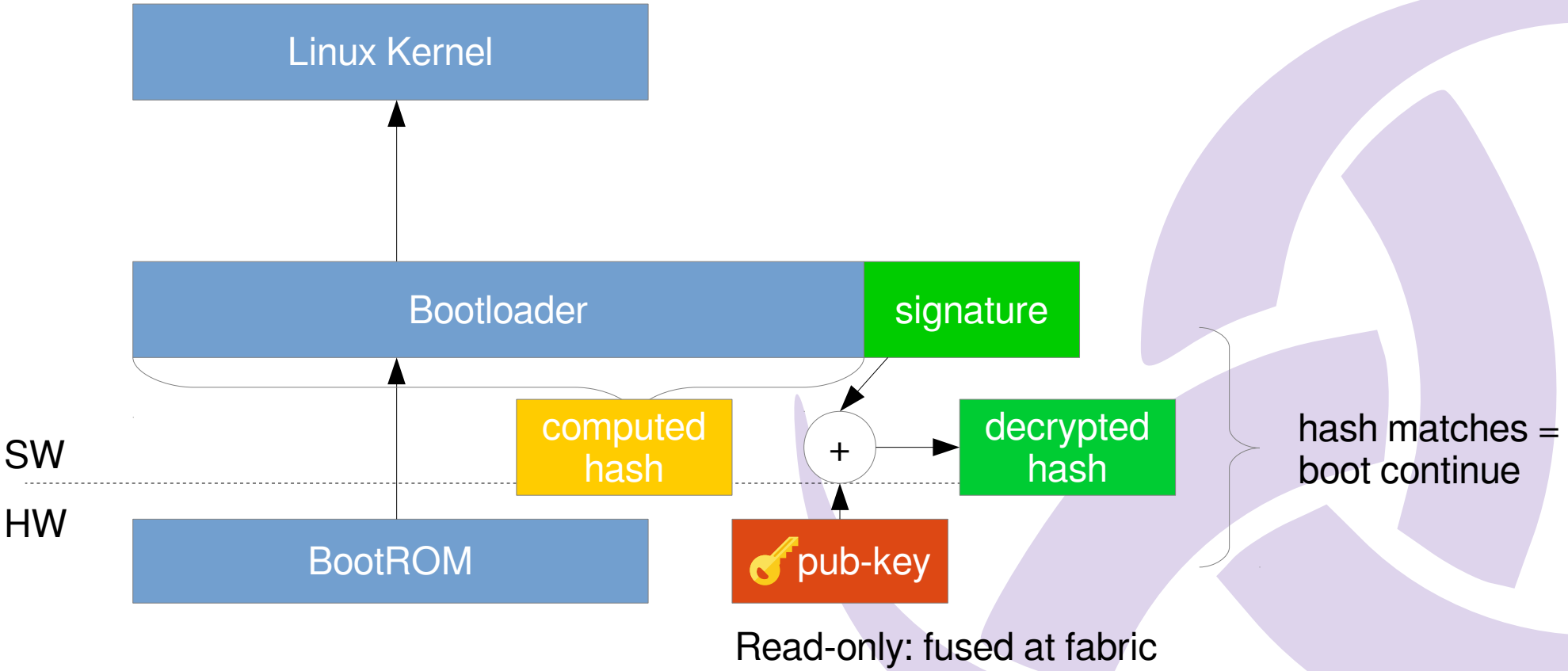
Software are signed after build using private key,



Secure boot: verification

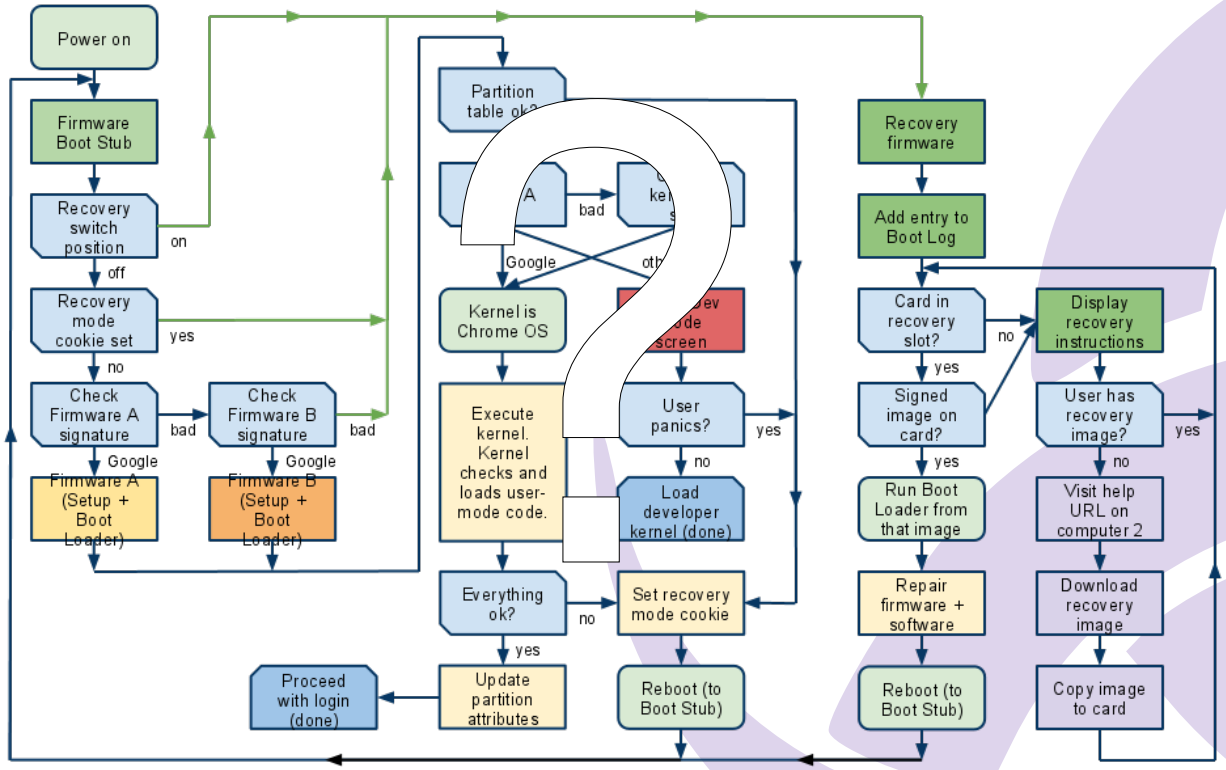
Principles

- Each software stage ensures integrity of next one,
- Rely on HW security features to store the key in read-only mode,



Secure boot policy

- **When integrity checks failed**
  - A boot policy should be defined,
  - This can differs from vendors, products requirements,
  - Tight to the whole system design,



U-Boot signature

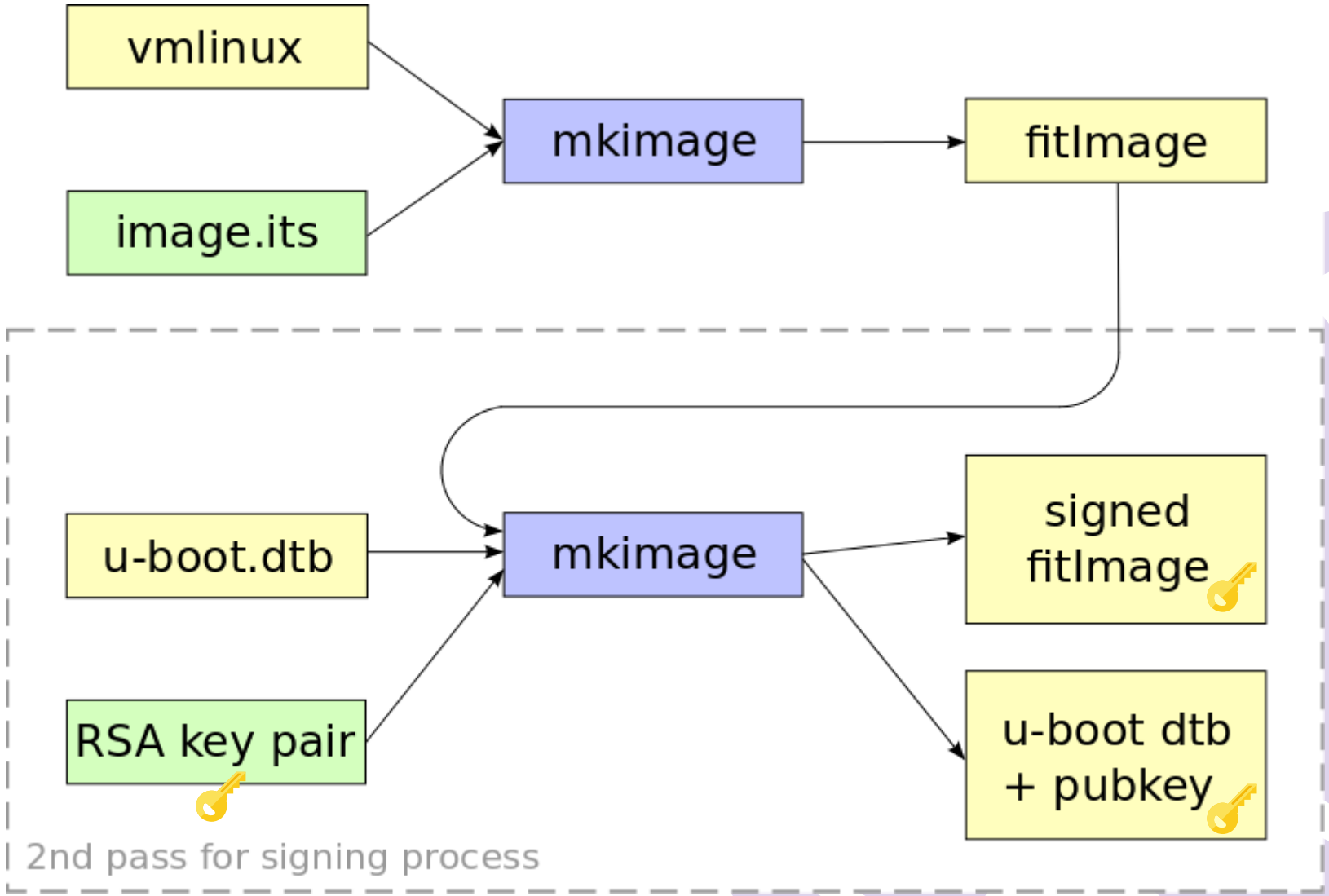
- Seals Linux Kernel & U-Boot after their builds,

- **Requirements**
  - U-Boot release v2013.07,
  - Linux kernel should be embedded in a *fitImage,*
  - An RSA key-pair (RSA-2048) is required for the signing process,

- **Default boot policy:**
  - Boot stopped if check failed,

- **Software signing**
  - *mkimage* tool is used in 2 passes

U-Boot signature



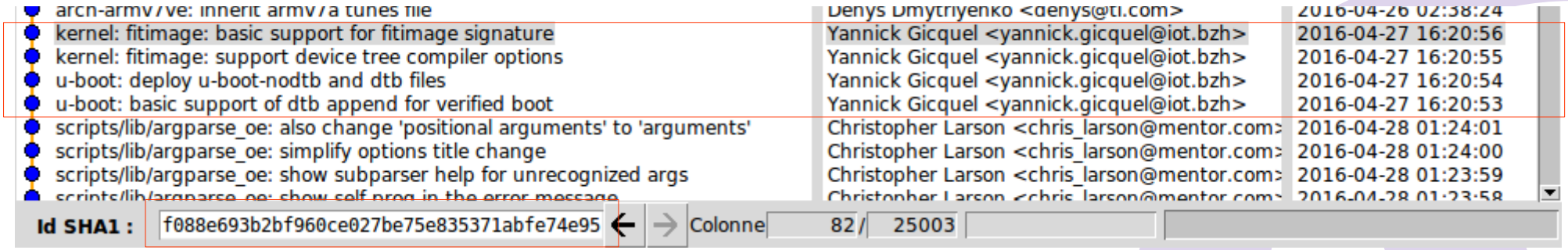
Signing with Open-Embedded

2013.07
U-Boot fitImage
+ signature support

2015.11
Yocto 2.0 introduce
fitImage support

2016.04
Yocto 2.1
released

2016.11
Yocto 2.2 will support
signed fitImage



How to sign the fitImage in OpenEmbedded build system?

UBOOT_SIGN_KEYDIR = "/keys/directory"

UBOOT_SIGN_KEYNAME = "dev" # keys name in keydir (eg. "dev.crt", "dev.key")

UBOOT_MKIMAGE_DTCOPTS = "-I dts -O dtb -p 2000"

UBOOT_SIGN_ENABLE = "1"

1. Overall context of updates for cars
   - Requirements,
   - Nature of updates

2. Secure boot
   - Concept,
   - U-Boot signature,

3. Enforcement solution
   - Trusted Execution Environment
   - OP-TEE

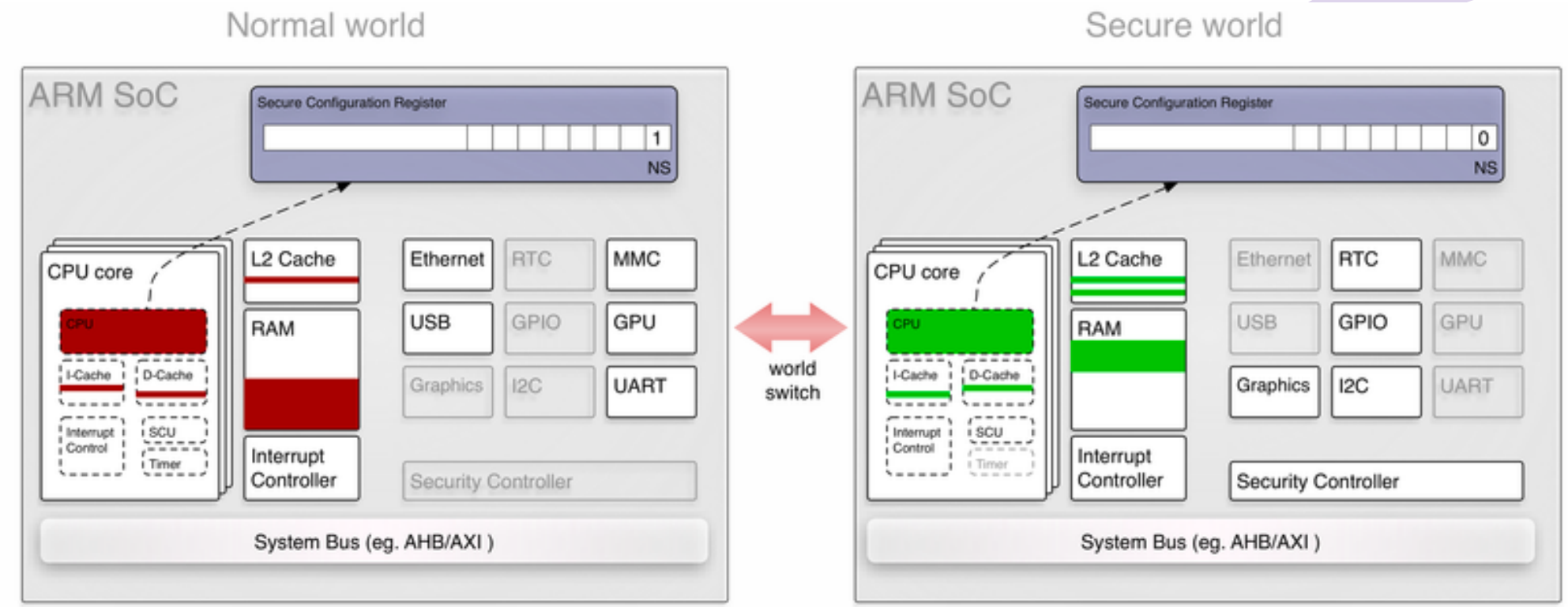
Trusted Execution Environment

- **Objectives**
  - It adds another bastion in case of Linux kernel security breach,
  - OS Virtualisation approach for security purpose,
  - Leverage HW capabilities to introduce privileges separations,

- **Implementations**
  - ARM: TrustZone,
  - Intel: Trusted Execution Technology

TrustZone

- Two executions contexts: normal world & secure world,
- Peripherals visibility can be configured for each world,
- Integrated into the system on chip,



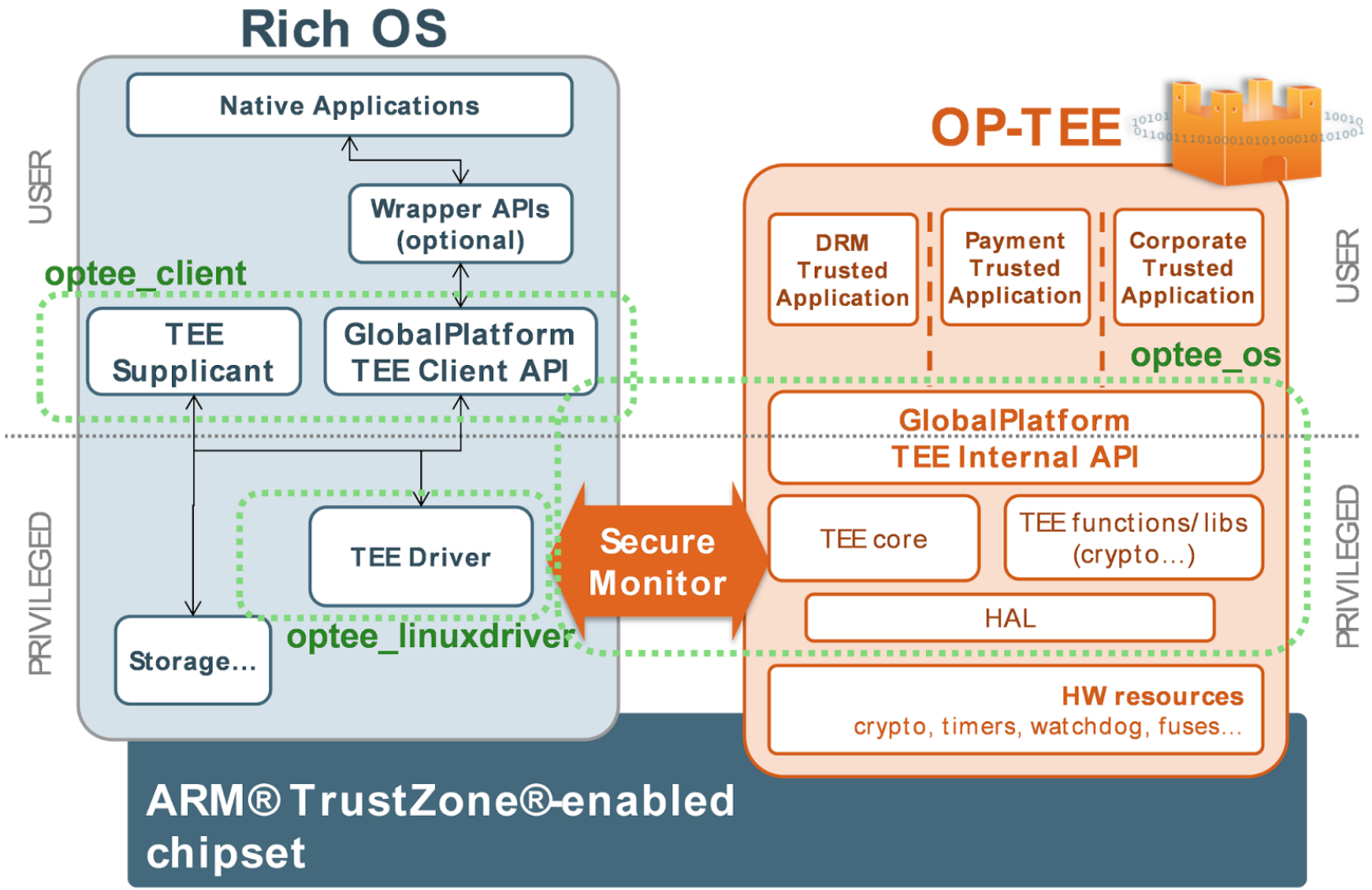
Credit: http://genode.org/documentation/articles/trustzone

OP-TEE

- **Open-source Portable TEE,**
  - Initiated by ST in 2007, then handled by Linaro,
  - Implements Global Platform API on top of ARM TrustZone, https://github.com/OP-TEE/
- **Features**
  - Protected storage,
  - SW isolation,
  - Device integrity.
- **TEE Core API specify**
  - Trusted Storage API for Data and Keys,
  - Cryptographic Operation API,
  - Time API,

GLOBALPLATFORM™
THE STANDARD FOR MANAGING APPLICATIONS ON SECURE CHIP TECHNOLOGY

OP-TEE Software architecture

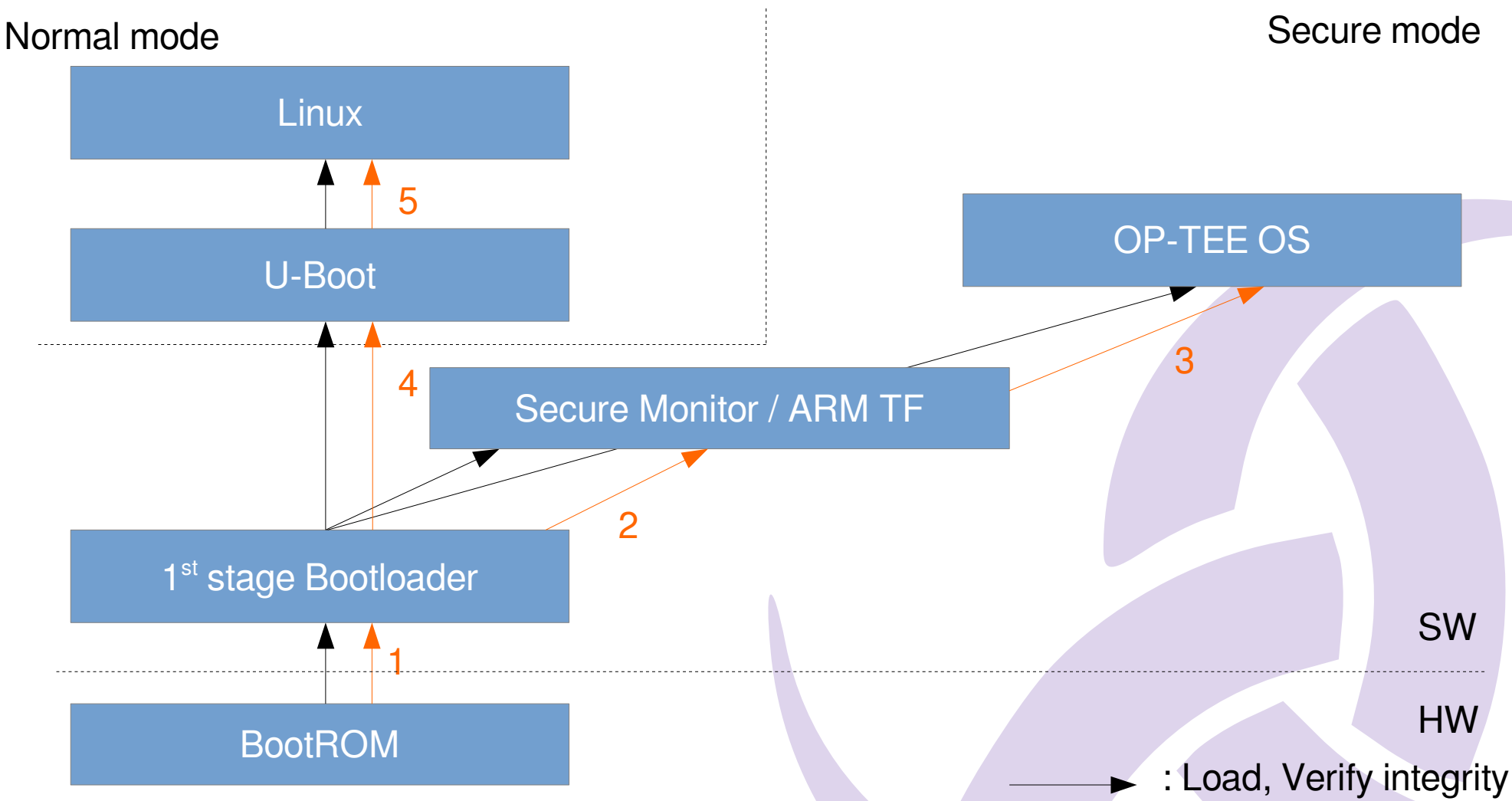


OP-TEE

- **OP-TEE OS Characteristics**
  - Trusted OS – Requires ~256KiB of RAM, ~320KiB of ROM
  - 22000 tests on the API,
  - Strong isolation of TA with stack canary protections,
  - Use Secure-RAM HW capability,

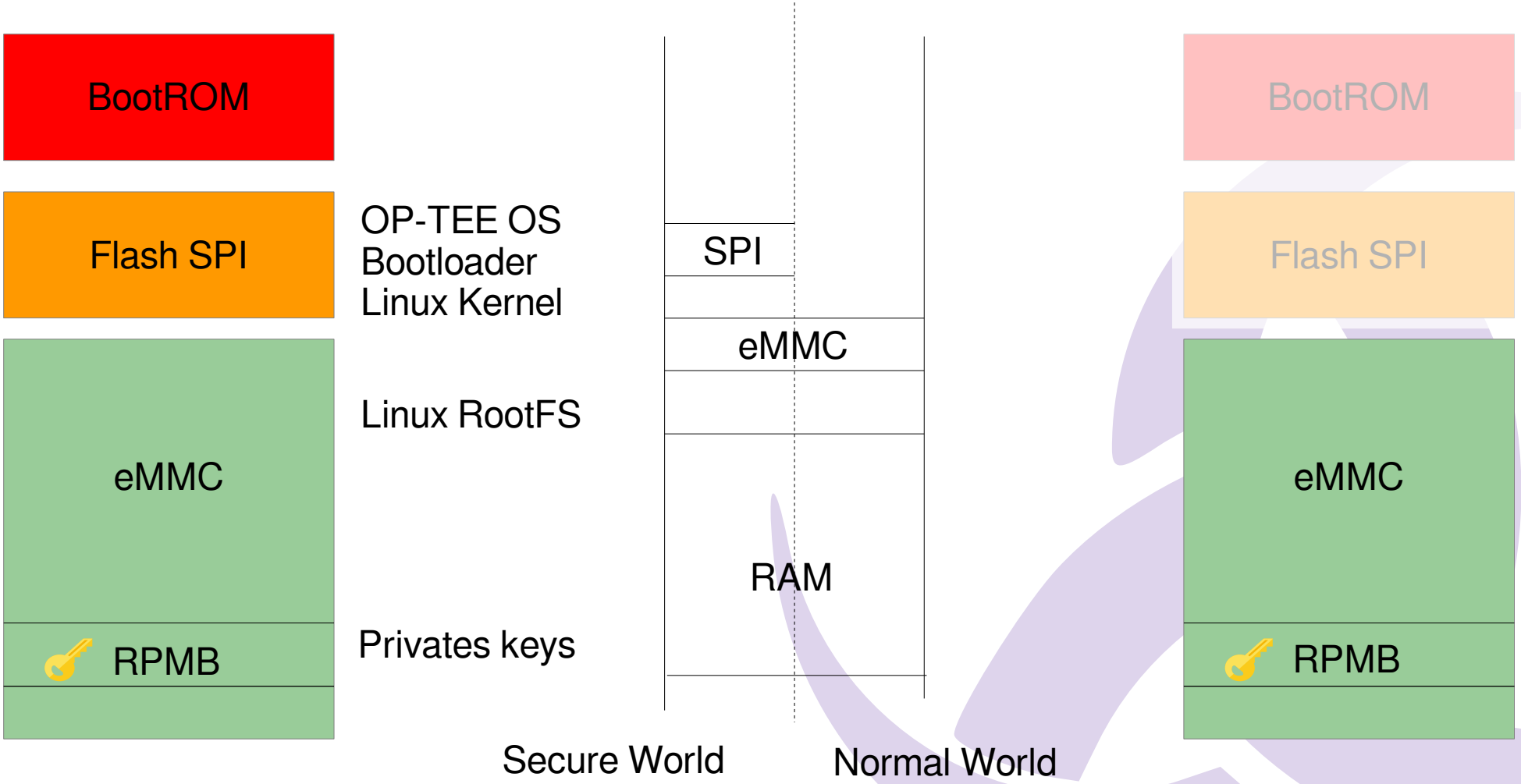
- **Secured Applications**
  - Two binaries blobs:
    - User space program (Normal world),
    - TA: Trusted Application (Secure world).
  - TA are signed, and identified by a UUID,
  - TA integrity are checked by the trusted OS before execution.

Boot sequence



Protected storage

- HW isolation to protect sensitive binaries & data:



OP-TEE in Open-embedded

- **Layer for AGL**
  - Enable a QEmu machine with OP-TEE OS + samples applications:

    https://github.com/iotbzh/meta-optee

- **Following steps**
  - Propose for staging for AGL to get an easier access to an “op-tee ready” environment.
  - Linaro on the way to publish upstream recipes they aim to maintain,
  - Protected storage for OTA client,

To summarize

- **Securing updates**
  - Not just a set of tools but a whole process,
  - Secure boot & boot policy are important to fulfill security requirements,
  - Virtualisation enhance the whole system security,
- **AGL distribution**
  - Balance between generic implementation & specific design,
  - Consolidation of tools in the build system,

Thanks

Upcoming discussions about SOTA:

Thursday, July 14 • 14:50 - 15:30

BoFs: How Do You Update Your Embedded Linux Devices - Daniel Sangorrin, Toshiba

Thursday, July 14 • 16:00 - 16:40

BOF-Discussion: CI, Testing and SOTA Updates One-Stop?! - Jan-Simon Moeller, The Linux Foundation