Trusted Execution Environments (TEE) and the Open Trust Protocol (OTrP)

Hannes Tschofenig and Mingliang Pei

16th July 2017 -- IETF 99th, Prague

What do we mean by security?

Communication Security

Aims

- Prevent eavesdropping on communication
  - Solution: Encryption
- Prevent spoofing of end point/ server
  - Solution: Authentication

Components Required

- Standards based crypto algorithms and protocols
- Random number generator
- Key management
- Identity management

System/Software Security

Aims

- Protect system from malicious software
- Prevent unauthorized
- Allow recovery from attack

Components Required

- Isolation of and restricted access to certain data, resources and code
- Secure/protected storage
- Trusted boot
- Recovery functionality

Physical Security

Aims

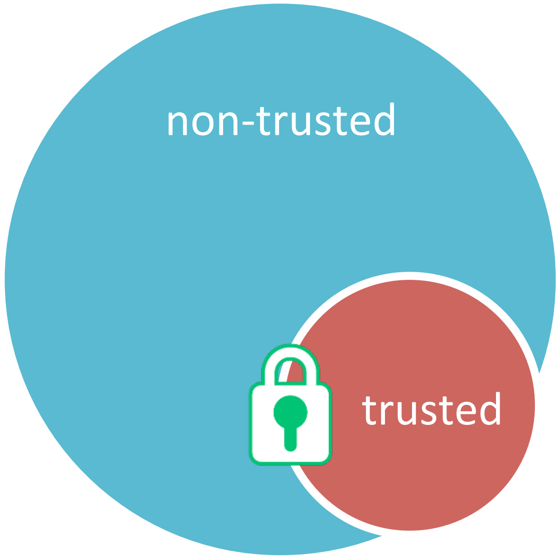
- Protecting against hardware attacks.
- Examples include:
  – Power analysis
  – Cutting internal chip tracks
  – Fault injection
  – etc

Components Required

- Specialised anti-tampering technology. E.g.
  – Deducing power and timing traces
  – Randomization of the pipeline
- Encrypted memory interfaces
- Implementations of algorithms that perform better against side channel attacks.

Security Principles

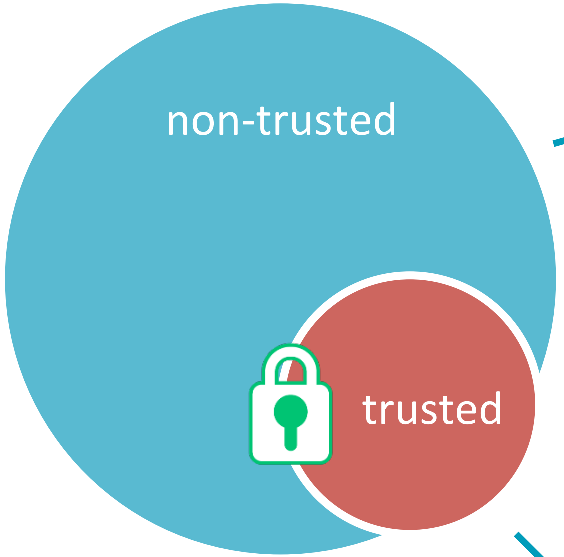
Security Principles: Isolation and Least Privilege



Isolation

- Isolate “trusted” resources from “non-trusted”
- What a developer calls trusted and untrusted is up to him/her.
- Access to trusted software only through dedicated APIs
- Non-trusted software run at lowest privilege possible
- Reduce attack surface of key components

Security Principles, cont.

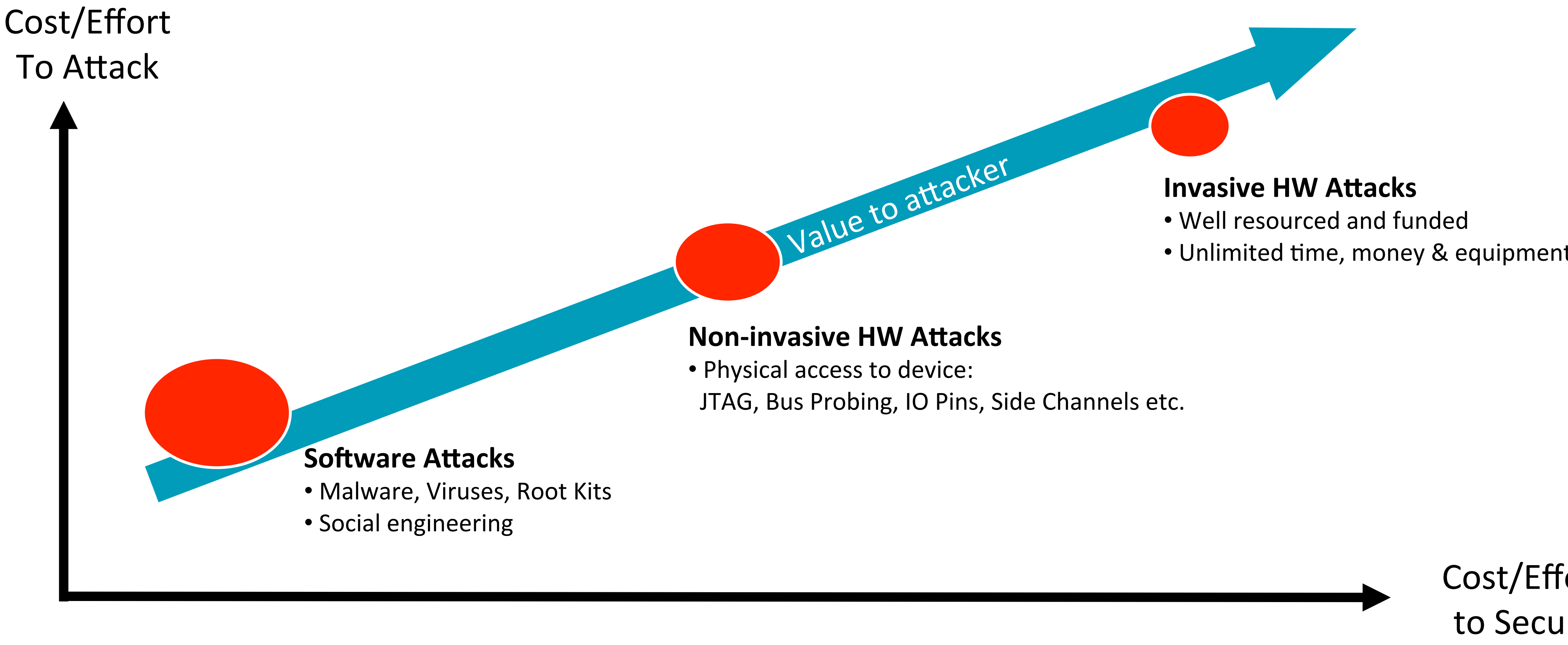


Non-Trusted
Majority of software
“Rich” operating system
Graphics
Applications
Data processing
etc

Trusted Software
- Crypto algorithms
- Keys
- Attack detection
- Sensitive data (e.g., fingerprint template)

Small, well reviewed code

Security Profiles



Solutions

Trusted Execution Environment: Why?

- Internet protocols today all rely on security protection
  - Use security protocols requiring cryptographic keys
  - Utilize cryptographic algorithms

- Operating systems (OSs), such as Android/Linux, are complex and sophisticated.

- Solution is to augment the OS with a more restrictive, and environment

- And extract the security components from applications / OS into this environment

- Trusted Execution Environments provide such an environment

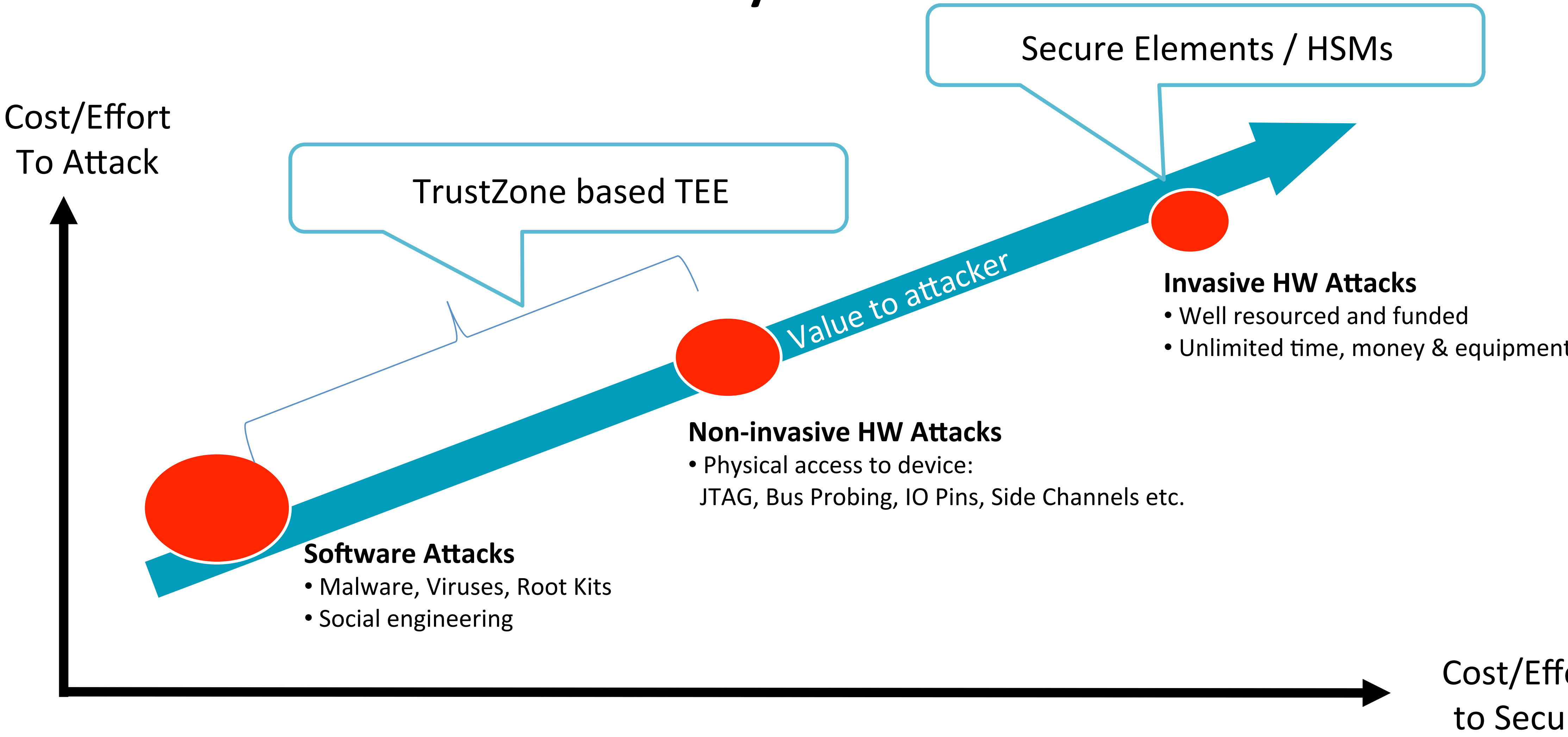
Trusted Execution Environment: What is needed?

- Lightweight OS that can support mutually distrusting Trusted Apps
  - Isolated environment for the execution of trusted code
  - Private memory spaces for code and data that cannot be snooped or modified by other system agents
- Well defined entry and exit interfaces
  - Designed to retain secrets when normal world clients are fully compromised

- Trusted Boot ROM*
- Trusted boot process*
- Cryptographic services
  - Symmetric Crypto
  - Asymmetric Crypto
  - Random Number Generator
- Cryptographic key store
  - Unique and shared keys
- Secure storage
  - For persistent data, such as keys

(*) Needed for ARM TrustZone but not necessarily for other TEEs (e.g., Int

Security Profiles



ARM Architecture Profiles

Application Profile
ARMv8-A

- 32-bit and 64-bit
- A32, T32 and A64 instruction sets
- Virtual memory system
- Supporting rich operating systems

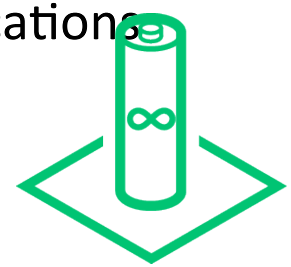
Real-time Profile
ARMv8-R

- 32-bit
- A32 and T32 instruction sets
- Protected memory system

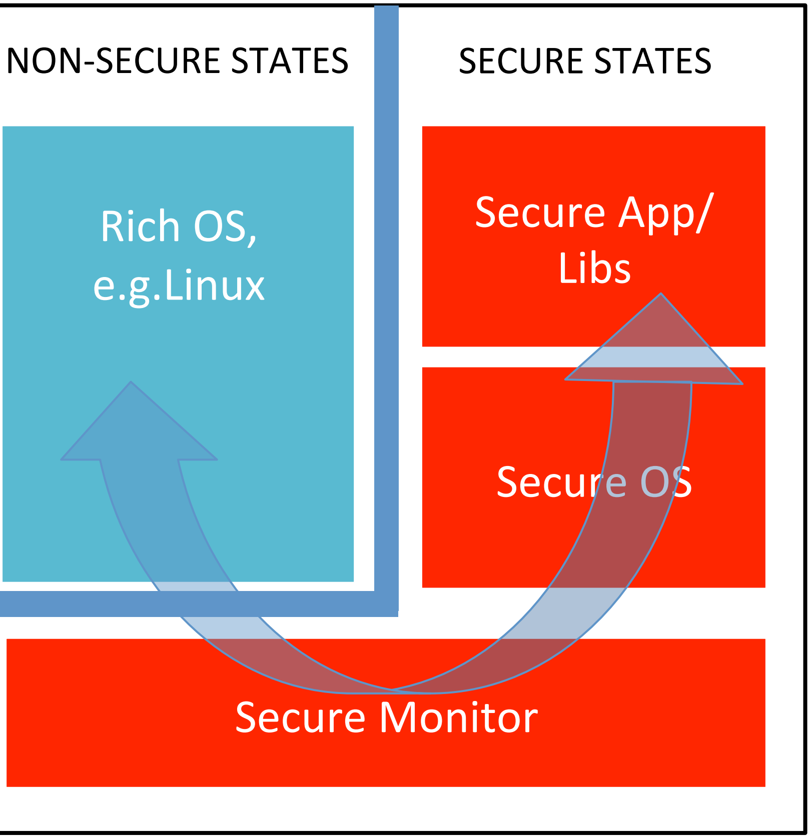
  (optional virtual memory)
- Optimized for real-time systems

Microcontroller Profile
ARMv8-M

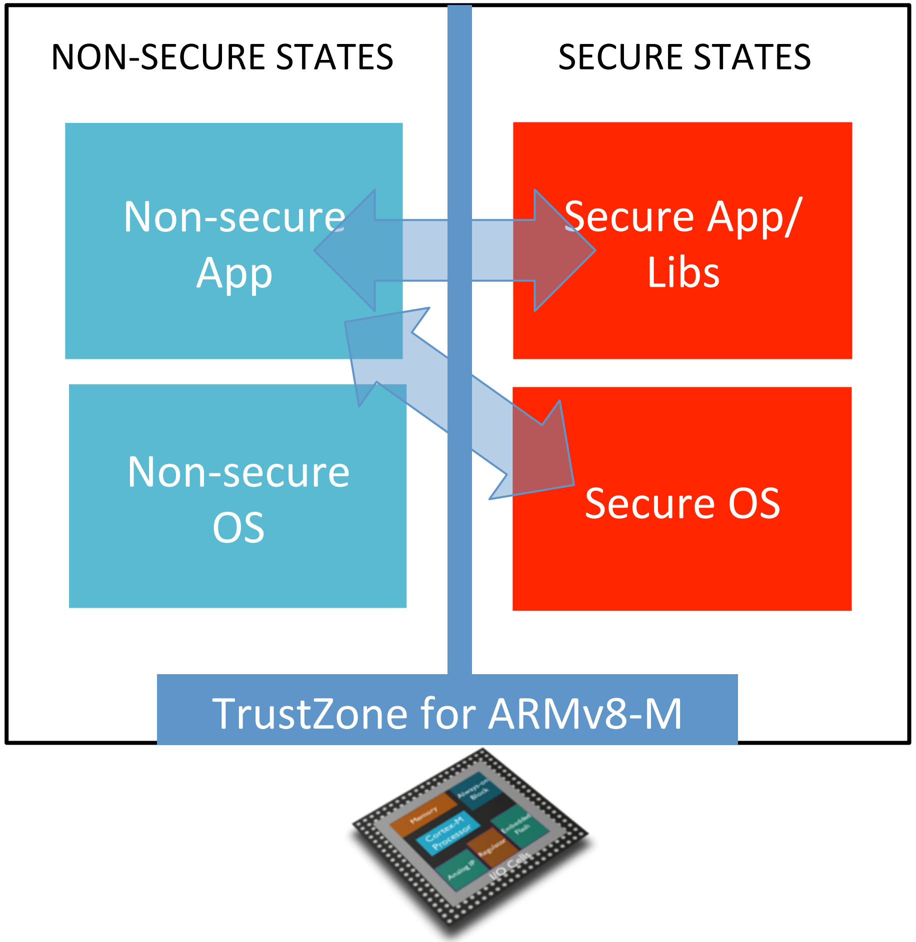
- 32-bit
- T32 / Thumb® instruction set only
- Protected memory system
- Optimized for microcontroller applications



TrustZone for ARMv8-A

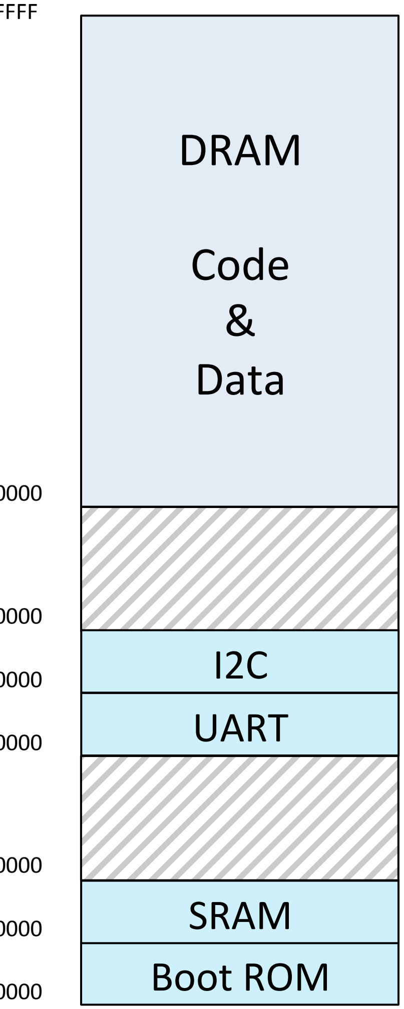


TrustZone for ARMv8-M

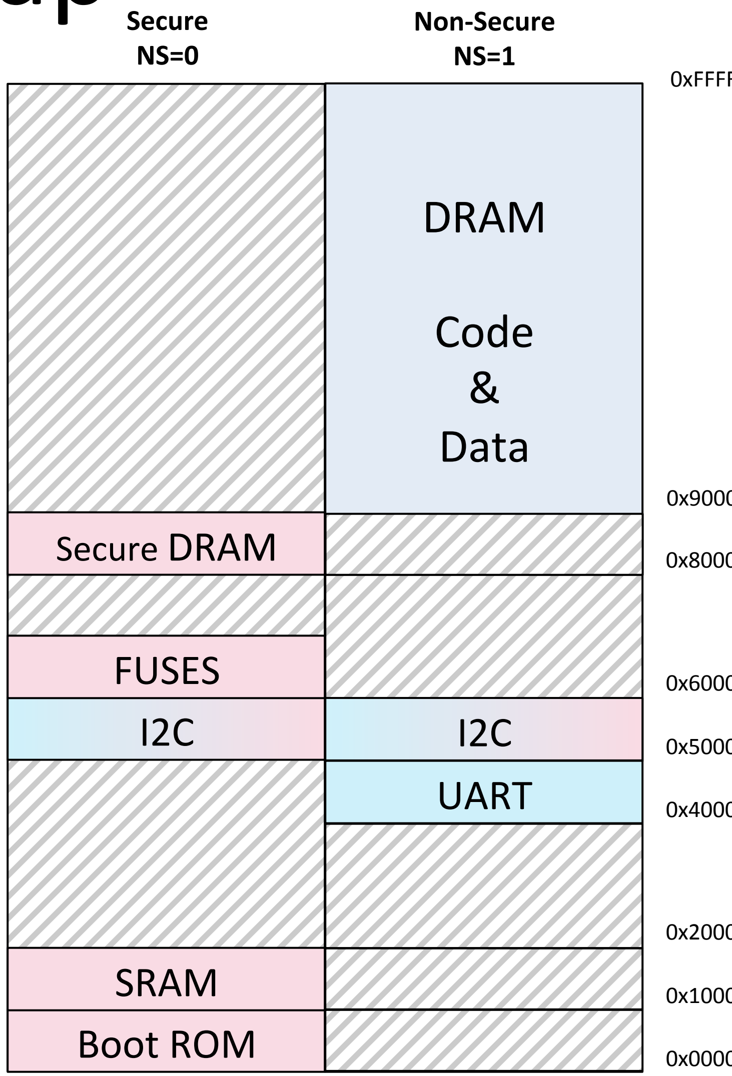


Secure transitions handled by the processor
to maintain embedded class latency

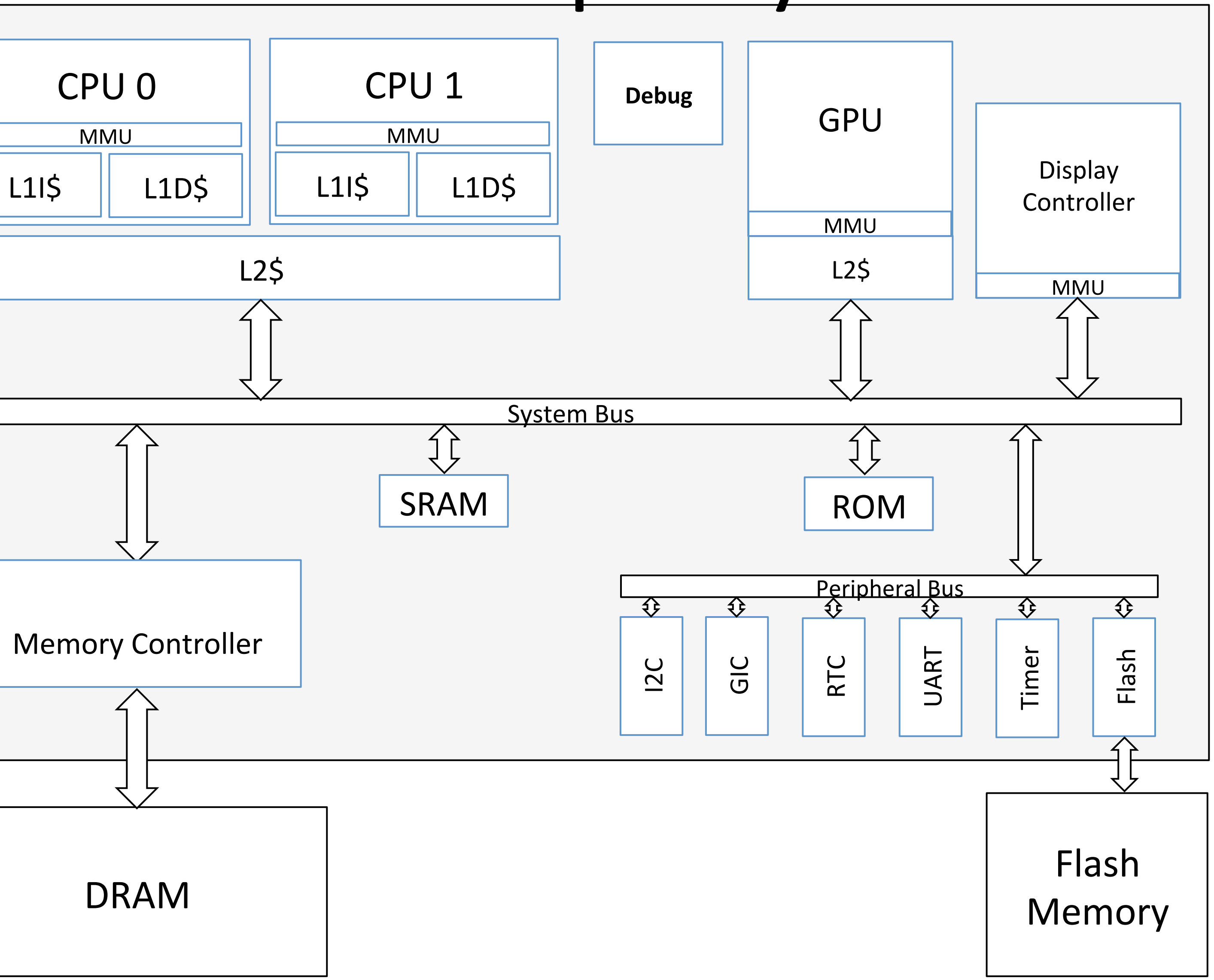
Secure Memory Map



- Normal physical memory map contains
  - DRAM for code and data
  - I/O peripherals
  - On chip ROM and SRAM
- The Secure state acts like "33rd address bit"
  - Doubling size of physical address map
- Key resources become secure only
  - Boot ROM and internal SRAM
- I/O devices are segregated
  - Secure only, Non-Secure or shared access
- DRAM can be partitioned
  - Using address space controller



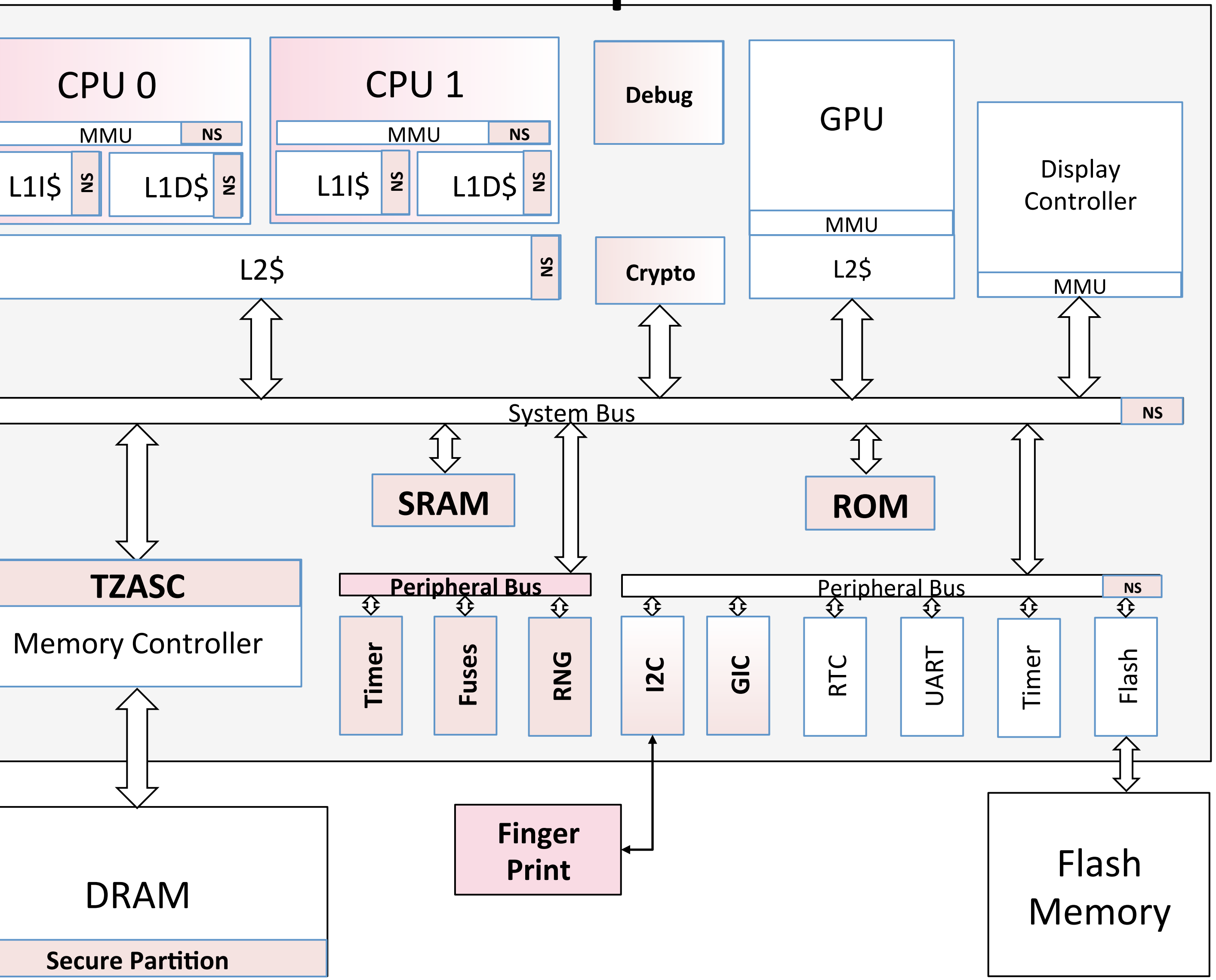
Example System on Chip (SoC)



- CPU cluster
  - MMUs and caches
- Bus mastering devices
  - GPU and Display controller
- Boot ROM and SRAM
- Memory Controller to DRAM
- Peripheral bus
  - Standard peripherals

Example SoC with TrustZone

KEY:
Trusted
Normal

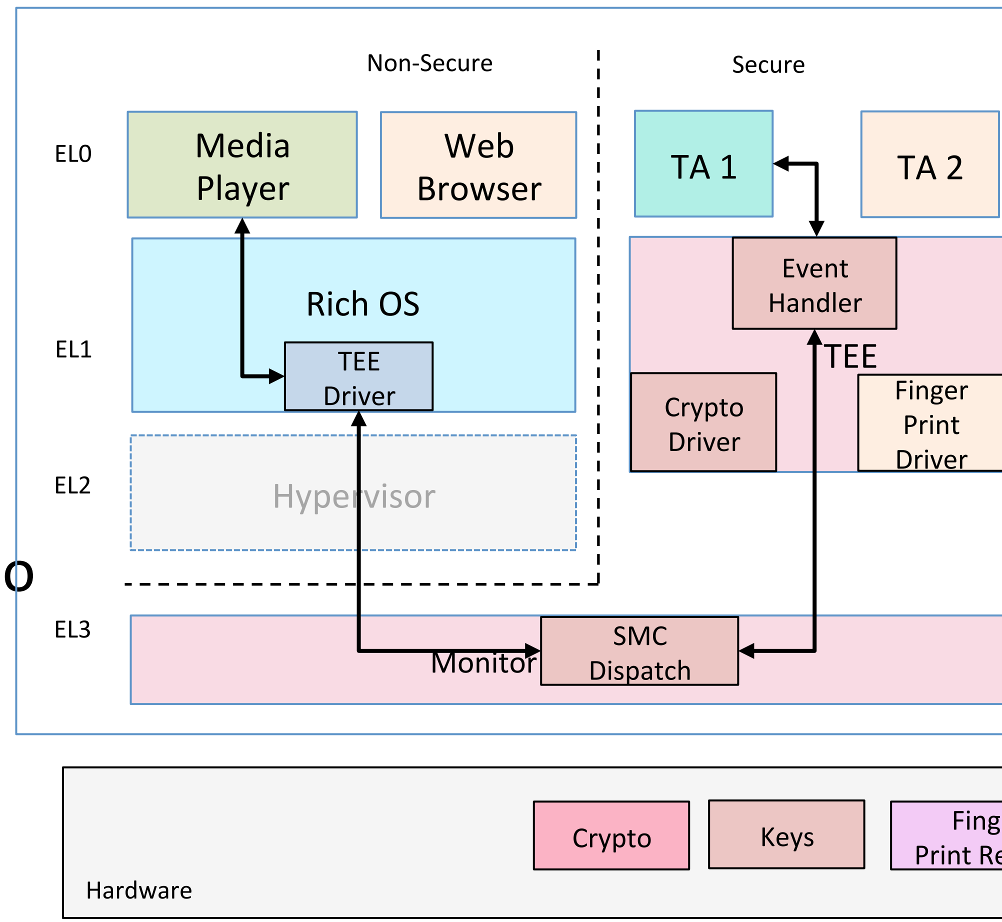


- Secure state added to CPU
  - MMU and Caches
- NS tags in buses
- Boot ROM and SRAM secured
- Debug and profiling secured
- Secure only peripherals added
- Shared peripherals modified
- DRAM partitioned for Secure
- Crypto HW accelerator
- External Secure Peripherals

- Existing Non-Secure HW remains unchanged
  - Never able to generate NS=0 transactions

TrustZone Software Stack

- Trusted Execution Environment with
  - Lightweight operating system offering security services
  - Trusted Apps, which can be installed, updated and deleted
- EL3 Monitor provides Secure / Non-Secure switching
- OS integration requires TEE driver issues SMCs to TEE



FIDO Example

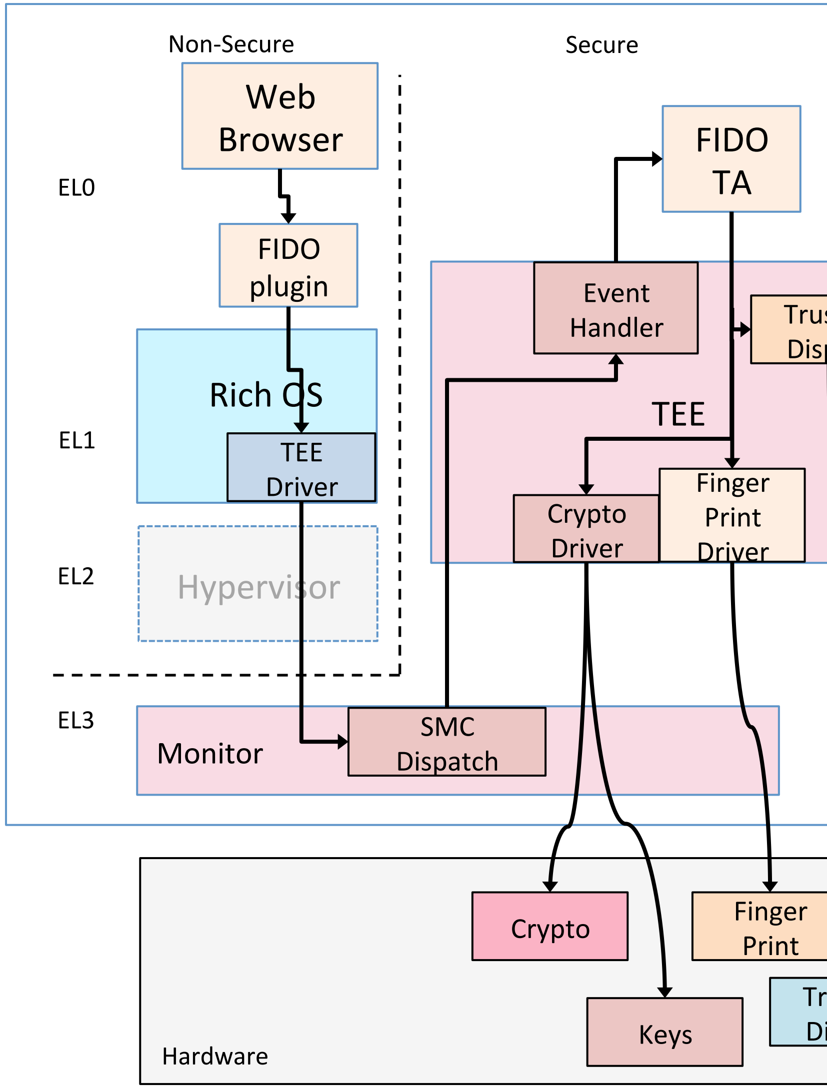
FIDO is an attempt to get rid of username/ password-based authentication

Process:

- Web service challenges device
- Challenge passed onto FIDO authenticator
- Performs user verification (e.g., fingerprint)
- Cryptographically sign the challenge
- Send response to web service
- User now securely logged in
- For transaction confirmation, trusted display is used.

Software and hardware stack needed for operation

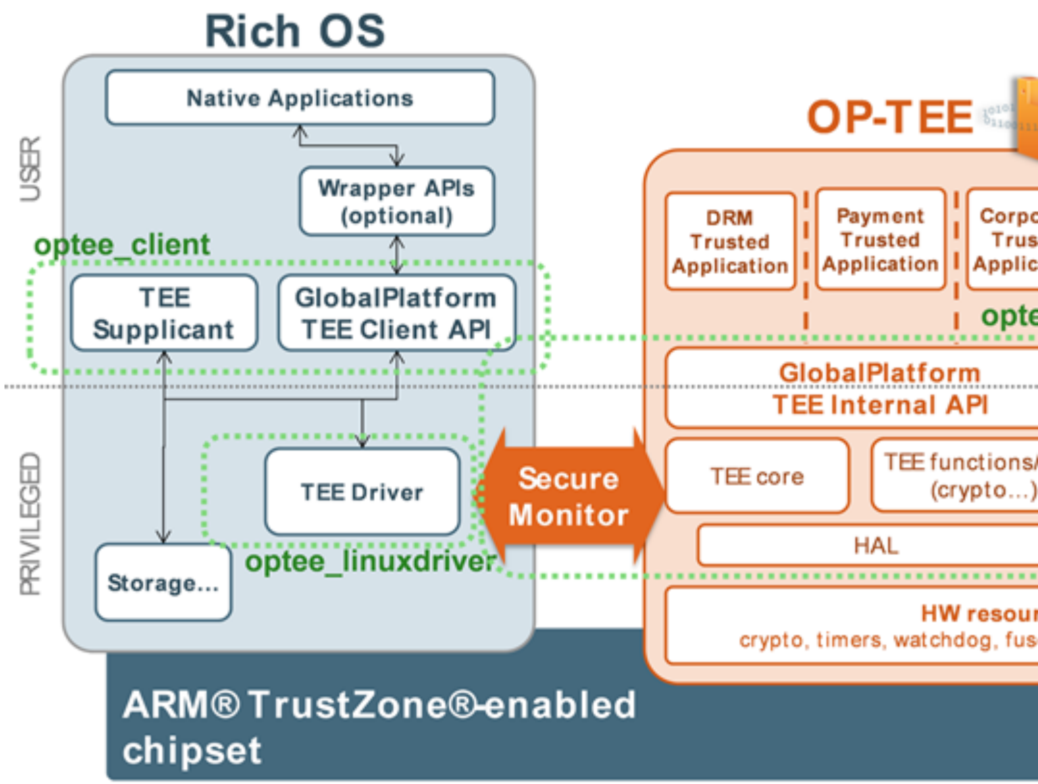
- Networking, Rich OS, Secure OS, HW
- FIDO Authenticator functionality in TEE; FIDO private key and fingerprint never leaves the TEE



Running Code

Open Source Software Available

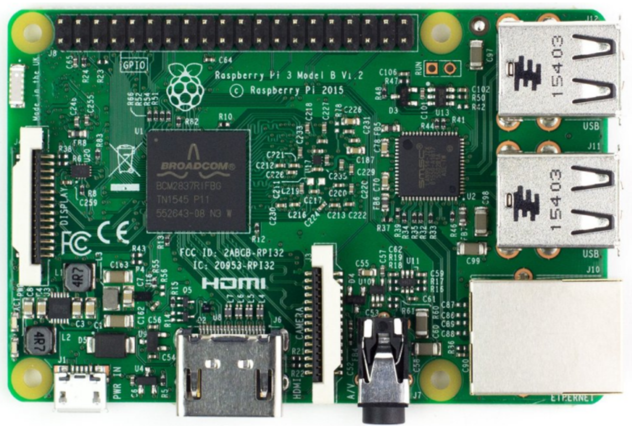
- Many developers of TEE technology
  - Chip companies, OEMs, OS platform owners, Independent Software Vendors, OSS
- ARM Trusted Firmware:
  - Link: https://github.com/ARM-software/arm-trusted-firmware
  - AArch64 reference implementation containing trusted boot, monitor and runtime firmware
- OP-TEE
  - Link: https://github.com/OP-TEE/
  - Reference implementation of secure world OS.
- GlobalPlatform provides common set of API's and services



Trying TrustZone @ Home

TrustZone on Raspberry Pi3

- Sequitur Labs port of Linaro's OP-TEE environment to the Raspberry Pi 3
- Press release:
- http://linuxgizmos.com/trustzone-tee-tech-ported-to-raspberry-pi-3/
- Code:
- https://github.com/OP-TEE/build/blob/master/docs/rpi3.md
- Video: https://www.youtube.com/watch?v=3MnLrHoQcyI



USB Armory

- Hardware: http://inversepath.com/usbarmory.html
- ~100 EUR
- The **USB armory** from **Inverse Path** is an open source hardware design, implementing a flash drive sized computer with TrustZone.
- Example apps available:

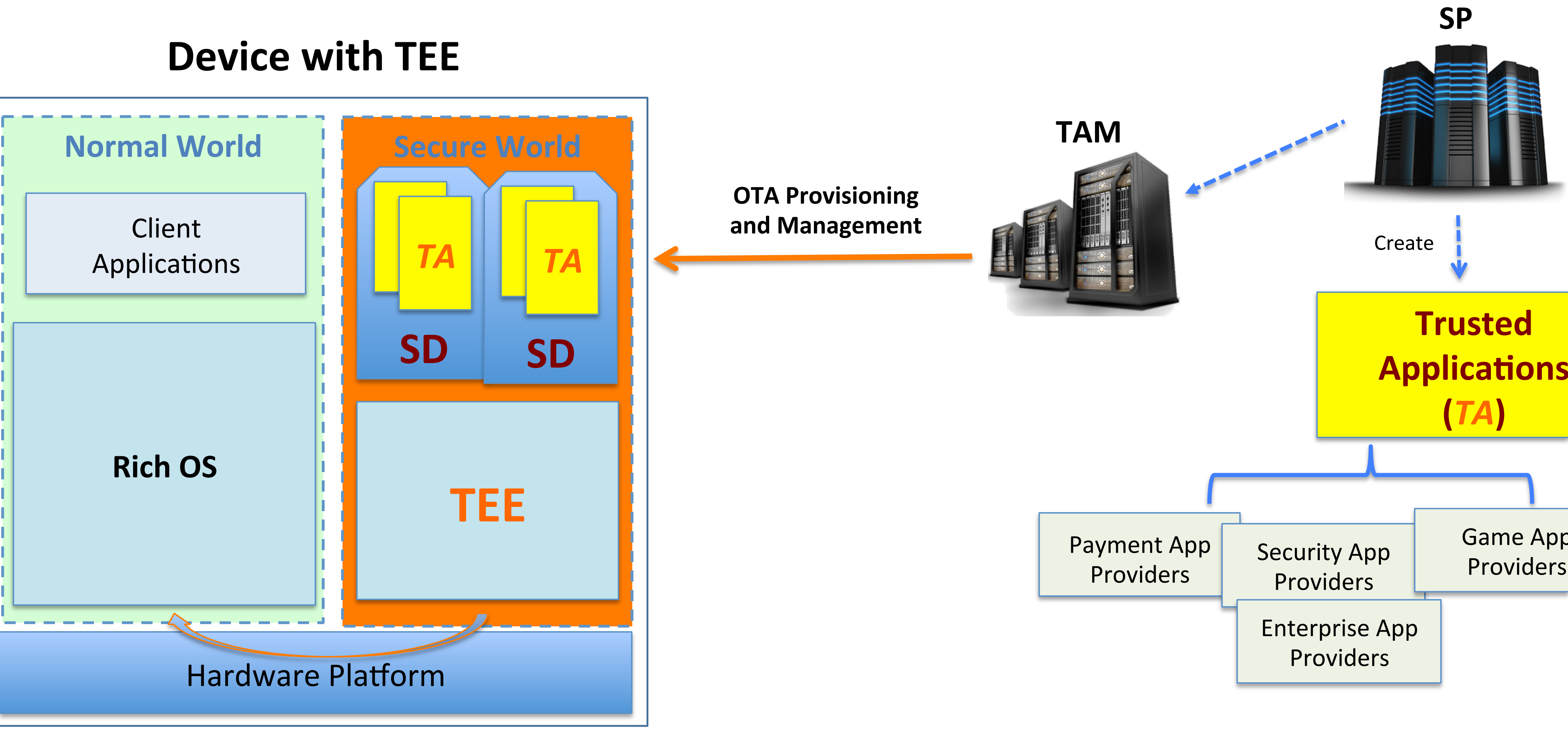
  https://github.com/inversepath/usbarmory/wiki/Applications

Summary

1. Isolation helps to improve security of software.
2. TrustZone provides the CPU and system isolation.
3. Open source code available for you to play with.

Open Trust Protocol (OTrP): Problem Statement

emand of hardware based security with TEE and TA



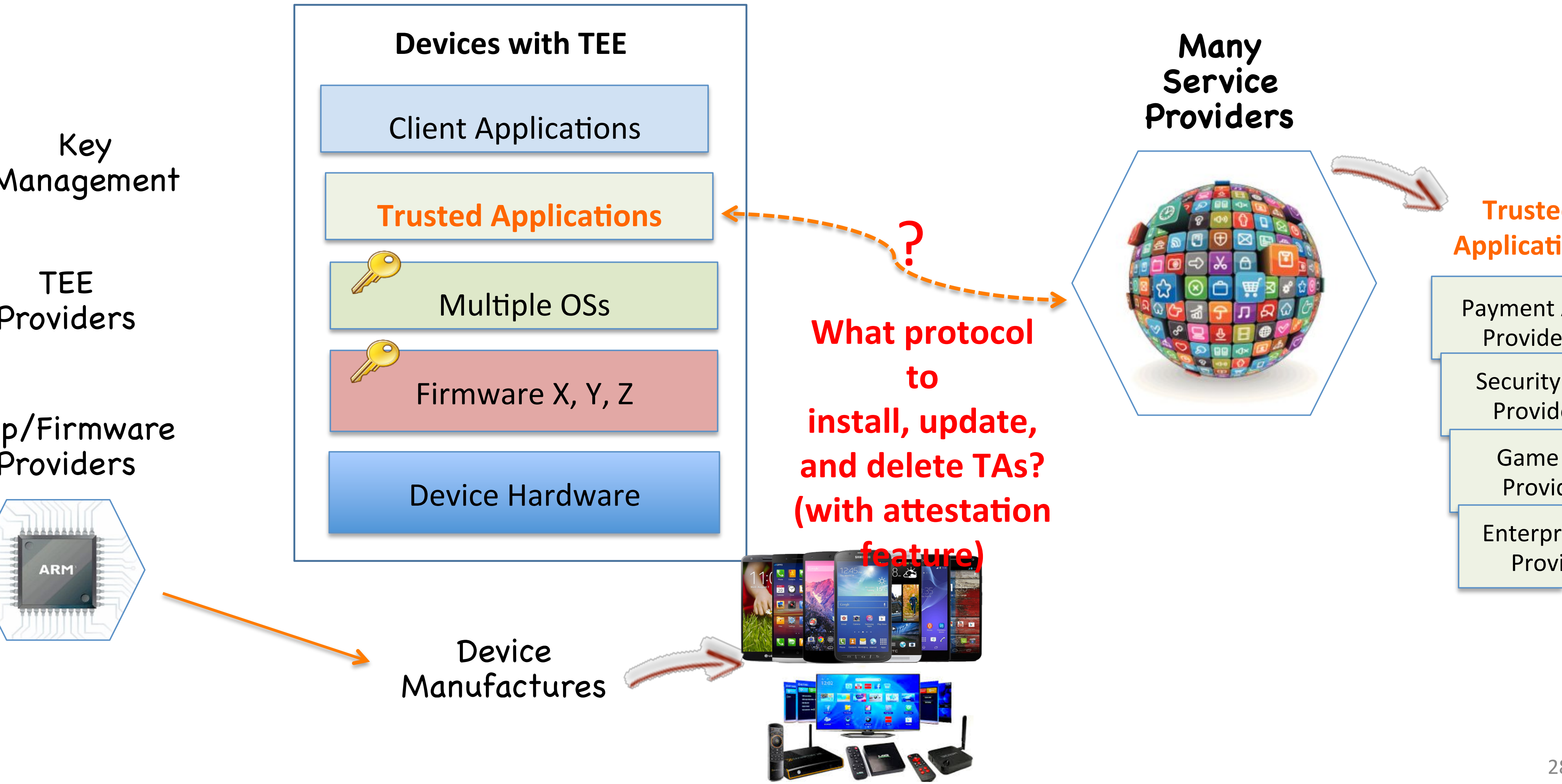
The Challenge

- Adoption gap for service providers
  - Gap between devices with hardware security and a wish to push Trusted Apps to devices with different TEEs and vendors

- Fragmentation is growing - IoT accelerated that fragmentation

- Lack of standards to manage TAs
  - Devices have hardware based Trusted Execution Environments (TEE) but they do not have a standard way of managing those security domains and TAs

Gaps to utilize hardware based security



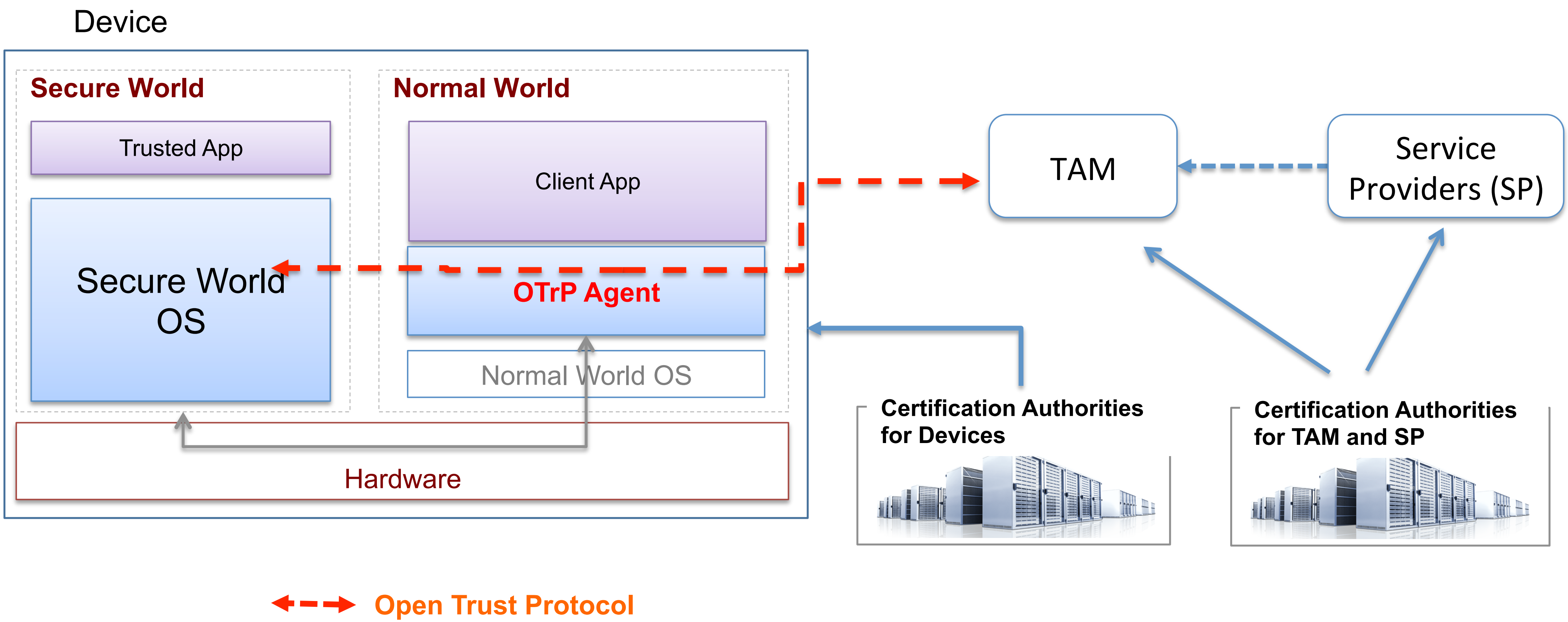
Open Trust Protocol (OTrP)

Open Trust Protocol (OTrP)

- An interoperable Trust Application Management protocol across broad application providers and diverse TEE OS providers
- Designed to work with any hardware security based TEE that aims to support a multi-vendor environment
- Focus on re-use of existing schemes (CA and PKI) and ease of implementation (keeping message protocol high level)

Overview

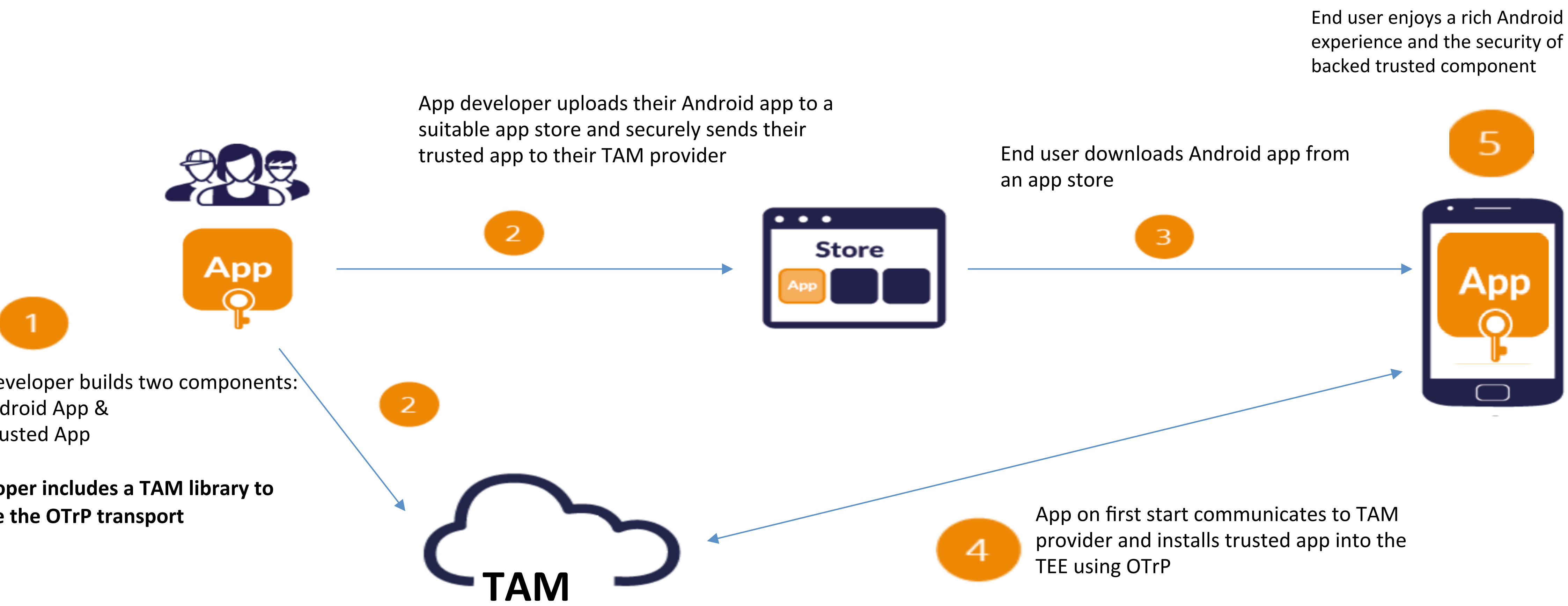
- CAs issue certificates to OTrP actors (TEE, TAM, SP)
- TAM and TEE exchange messages
- An OTrP Agent relays the OTrP message between TAM and TEE



Design Choices

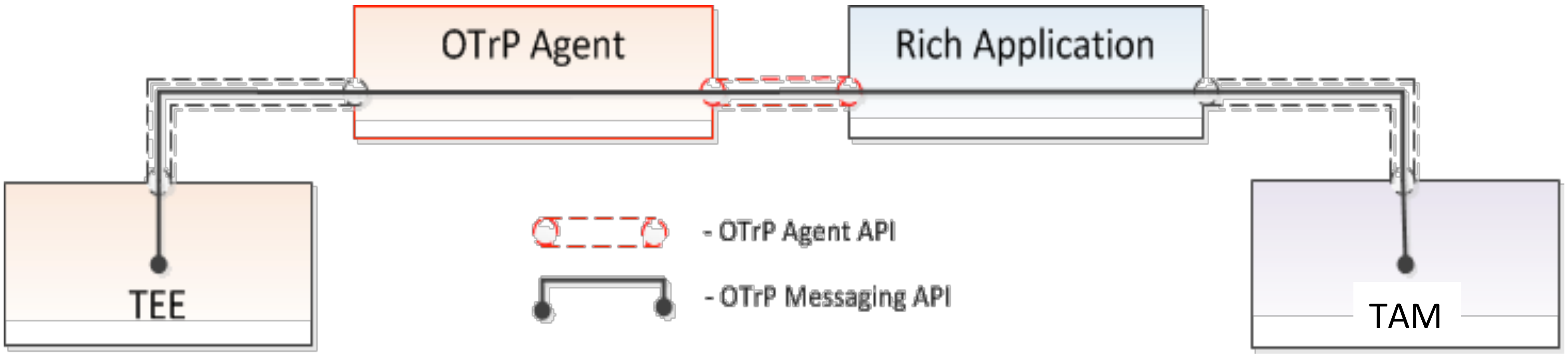
- **Uses asymmetric keys and PKI**
  - Manufacturer-provided keys and trust anchors
  - Enables attestation between TAM and TEE-device
- **JSON-based messaging between TAM and TEE**
  - Messages for attestation
  - Messages for security domain management and TA management
  - Use JOSE (JSON signing and encryption specifications) – CBOR alternative spec available.
- **OTrP Agent in REE relays message exchanges between a TAM and TEE**
- **Device has a single TEE only**

Envisioned User Experience

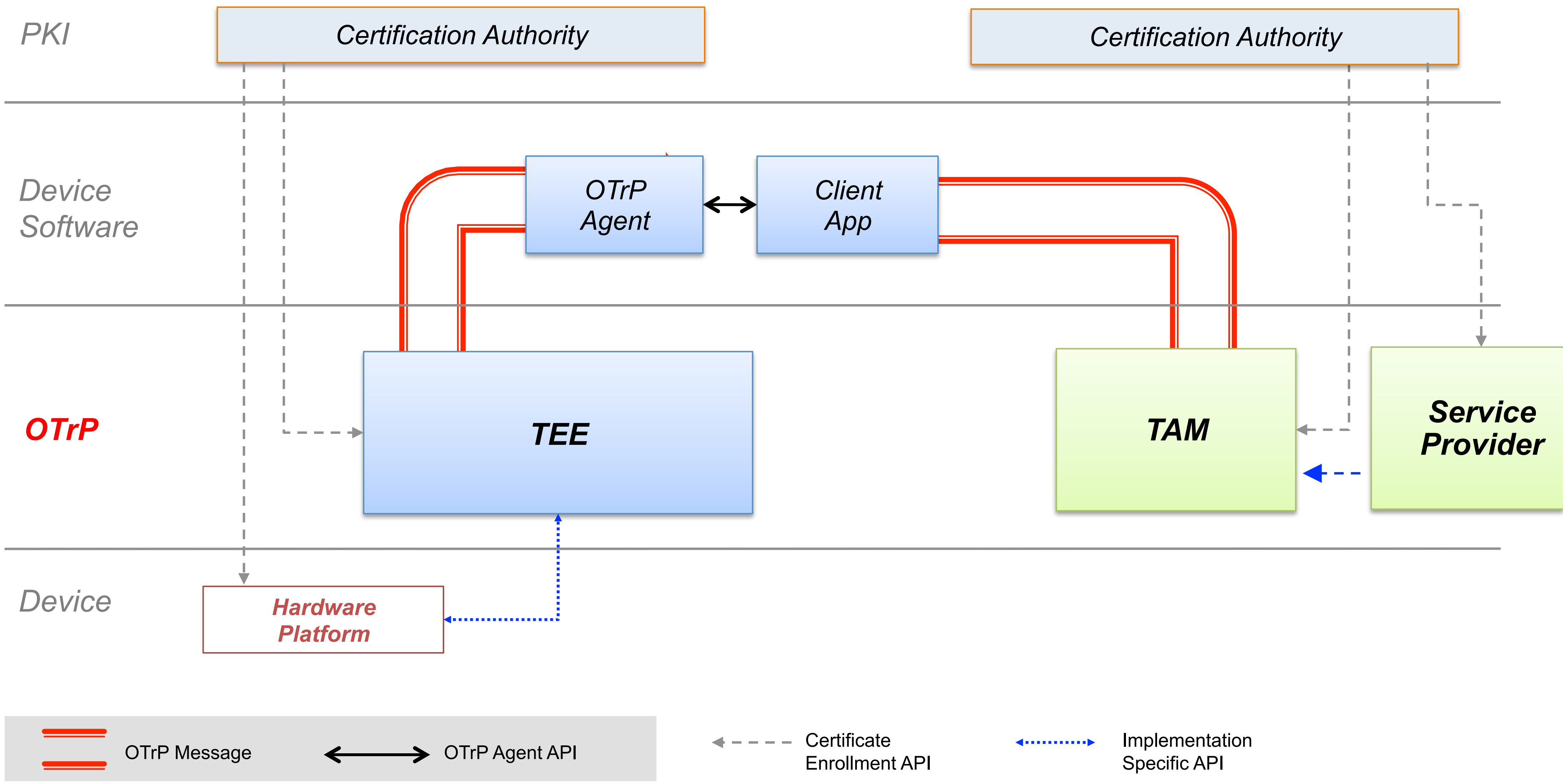


OTrP Agent

- Responsible for routing OTrP messages to the appropriate TEE
- Most commonly developed and distributed by TEE vendor
- Implements an interface as a service, SDK, etc.



Scope


PKI
Certification Authority
Certification Authority
Device
Software
OTrP
Agent
Client
App
OTrP
TEE
TAM
Service
Provider
Device
Hardware
Platform
OTrP Message
OTrP Agent API
Certificate
Enrollment API
Implementation
Specific API

Operations and Messages

✓ Remote Device Attestation

Command	Descriptions
GetDeviceState	• Retrieve information of TEE device state including SD and TA associated to a TAM

✓ Security Domain Management

Command	Descriptions
CreateSD	• Create SD in the TEE associated to a TAM
UpdateSD	• Update sub-SD within SD or SP related information
DeleteSD	• Delete SD or SD related information in the TEE associated to a TAM

✓ Trusted Application Management

Command	Descriptions
InstallTA	• Install TA in the SD associated to a TAM
UpdateTA	• Update TA in the SD associated to a TAM
DeleteTA	• Delete TA in the SD associated to a TAM

Keys

Certificate Authority

CA Certificate

Service Provider

SP Key pair
and Certificate

TAM

TAM Key pair
and Certificate

Trust Anchors: trusted Root CA list of TEE certificates

Device TEE

TEE Key pair
and Certificate

TFW Key pair
and Certificate (optic

Trust Anchors: trusted Root CA list of TAM & TFW

*** Key pair and Certificate:***
used to issue certificate

*** Key pair and Certificate:***
used to sign a TA

*** Key pair and Certificate:***
sign OTrP requests to be verified by TEE

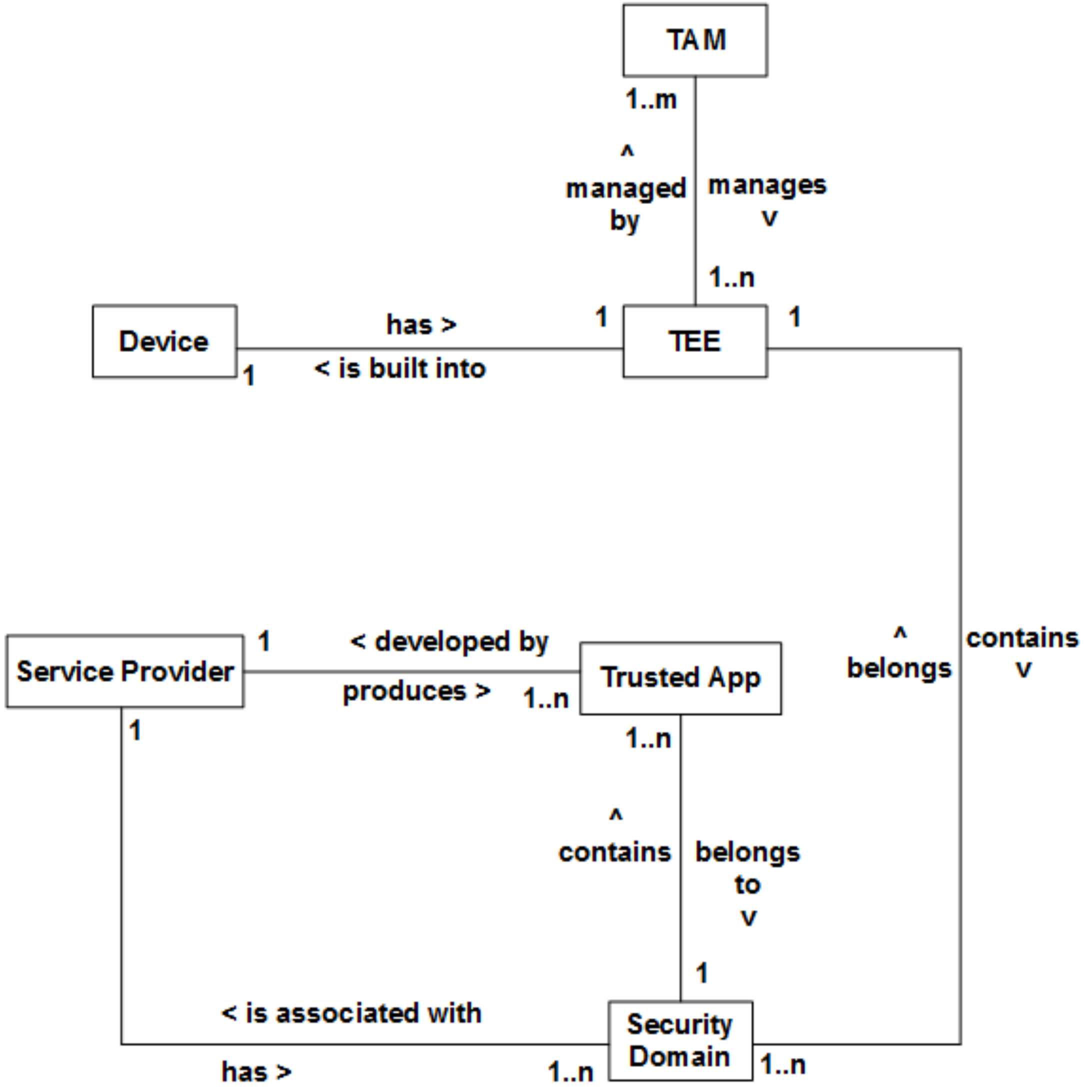
*** Key pair and Certificate:***
device attestation to remote TAM and SP.

*** SP AIK in runtime***
for use by SP (encrypt TA data / verify)

*** Key pair and Certi***
evidence of secure and trustworthy firr

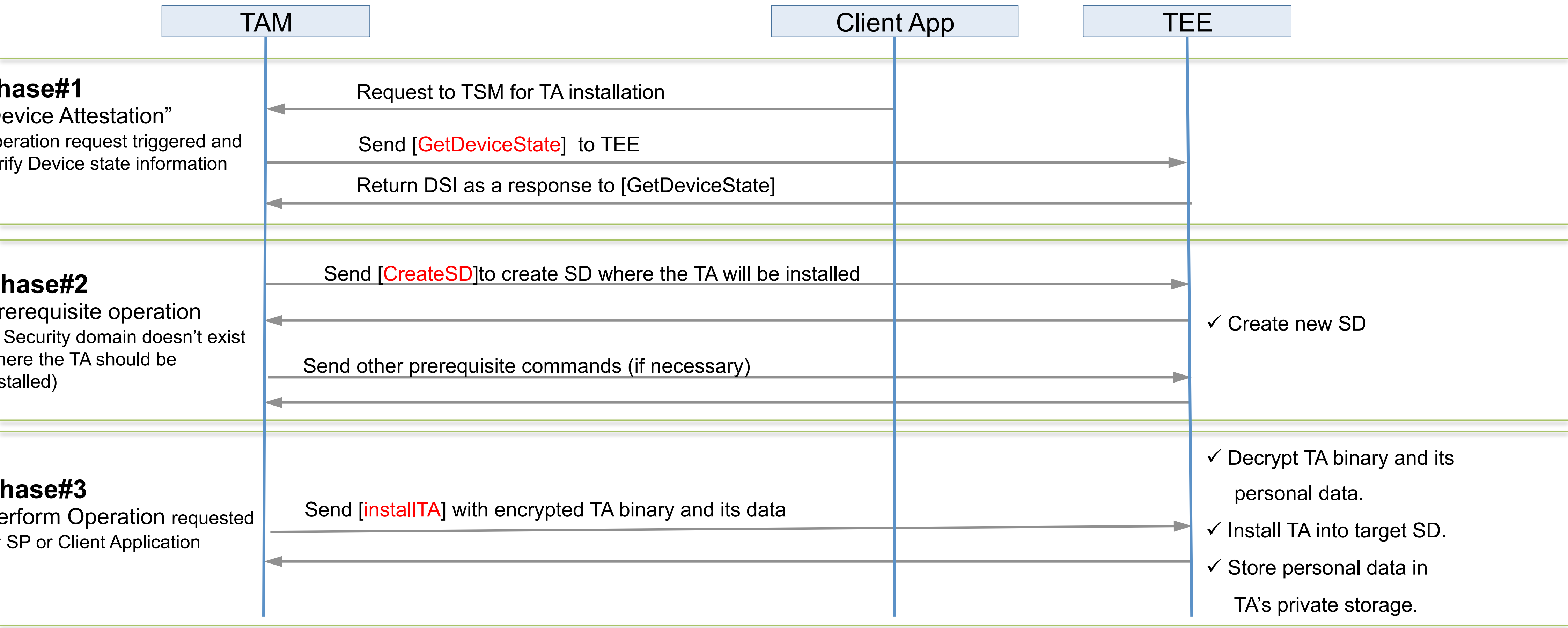
ation Identity Key, TFW: Trusted Firmware

Entity Relationships



Sample Protocol Usage Flow

- Security of the Operation Protocol is enhanced by applying the following three Measures:
  - ✓ Verifies validity of Message Sender's Certificate
  - ✓ Verifies signature of Message Sender to check immutability
  - ✓ Encrypted to guard against exposure of Sensitive data



Summary

- Some TEEs, such as TrustZone, are open to companies to install their favourite secure world OS.
- Vendors want to have a choice regarding Trusted Application Managers.
- This creates an interoperability challenge for managing (installing, updating, deleting) Trusted Applications on a TEE.
- OTrP provides a protocol for such a TA management (offering attestation capabilities).