TEEP BOF

Problem Statement

`draft-liu-opentrustprotocol-usecase`

IETF 100th, Singapore

Background

- Hardware based security is desirable
  - Today's processor technology supports various isolation concepts.
  - Well known are the concepts like the memory management unit, user and kernel space, and the hypervisor.
  - Additional isolation concepts where a Rich Execution Environment (REE) resides alongside a Trusted Execution Environment (TEE)

- TEE already widely deployed in the payment industry
- TEE already adopted in other standard bodies (GP, OneM2M, etc.)

Benefits of TEE

- A TEE provides hardware-enforcement that
  - The device has unique security identity
  - Any code inside the TEE is authorized code
    - Reduced risk for application compromise
  - Any data inside the TEE cannot be read by code outside the TEE
    - Safe area of the device to protect assets (great for key management)
  - Compromising REE and normal apps don’t affect TEE and code (called Trusted Application) running inside TEE

Background: Hardware Details

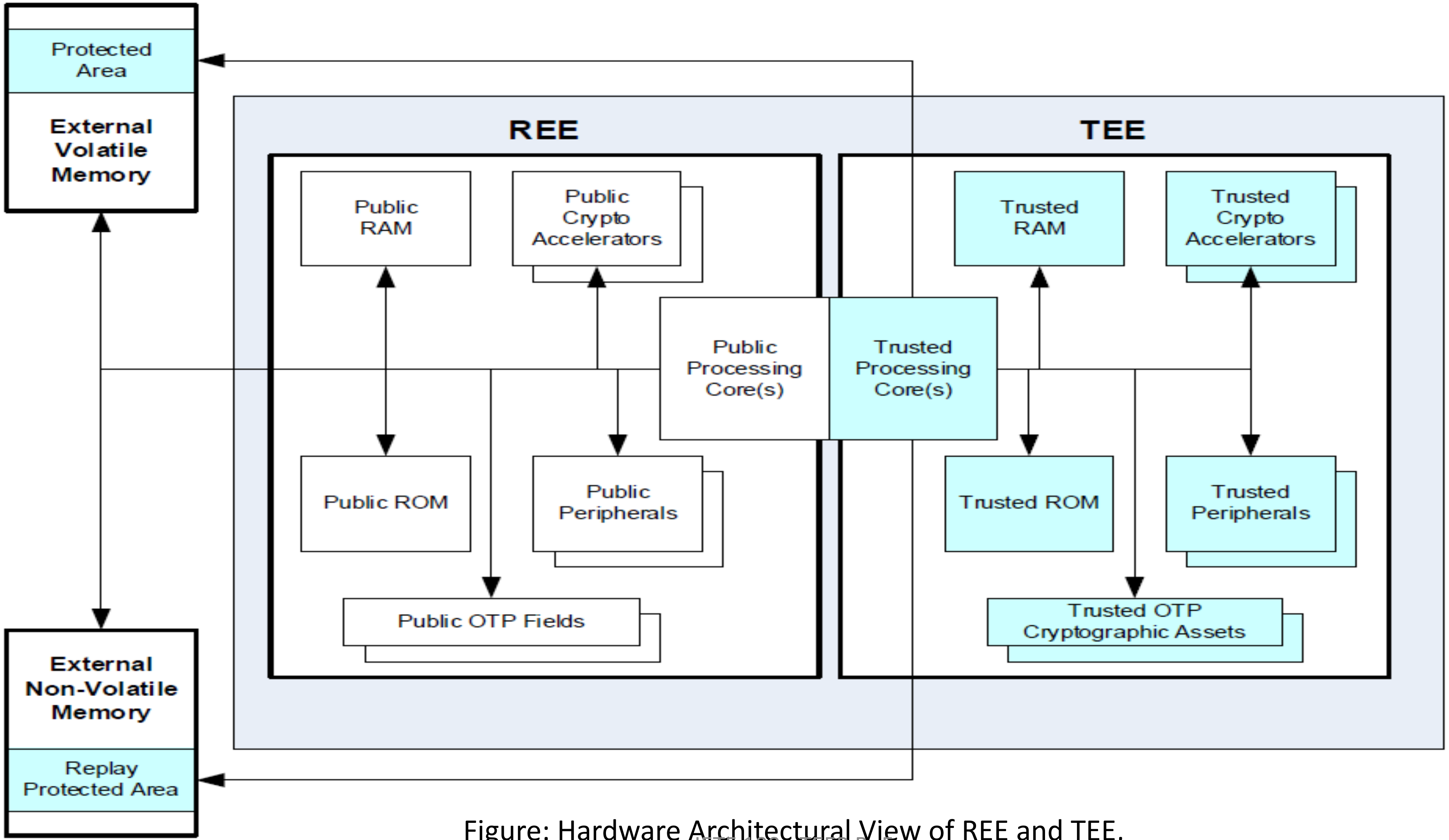

Protected Area
External Volatile Memory
REE
TEE
Public RAM
Public Crypto Accelerators
Trusted RAM
Trusted Crypto Accelerators
Public Processing Core(s)
Trusted Processing Core(s)
Public ROM
Public Peripherals
Trusted ROM
Trusted Peripherals
Public OTP Fields
Trusted OTP Cryptographic Assets
External Non-Volatile Memory
Replay Protected Area

Figure: Hardware Architectural View of REE and TEE,
Global Platform, TEE System Architecture v1.1

Background: Software Details

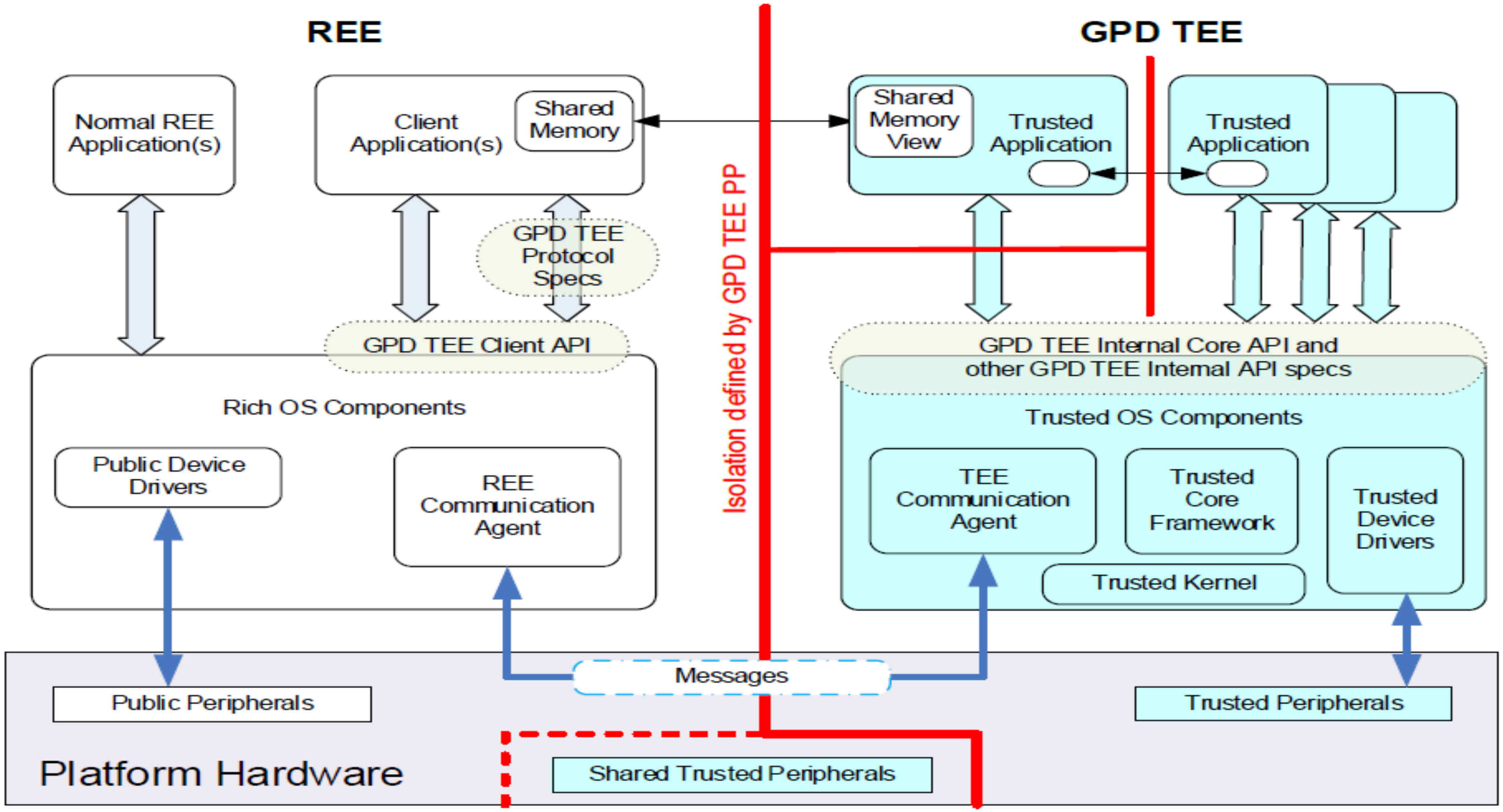


Figure: TEE Software Architecture,
Global Platform, TEE System Architecture v1.1

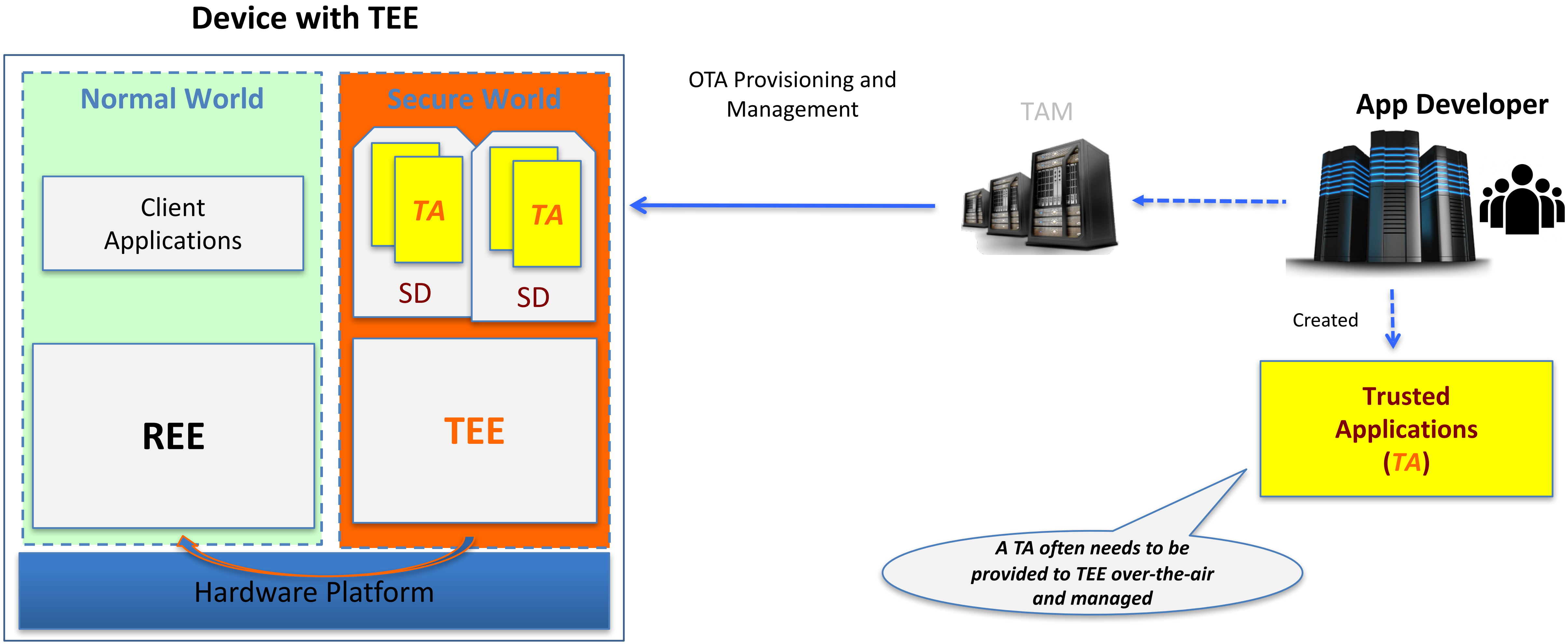
Despite such widely available TEE environment

- Trusted application development and distribution are hard
  - Much less than that for normal apps via App Store
  - Trust and management issues because TEE itself deems authorized trustworthy code

Example use cases

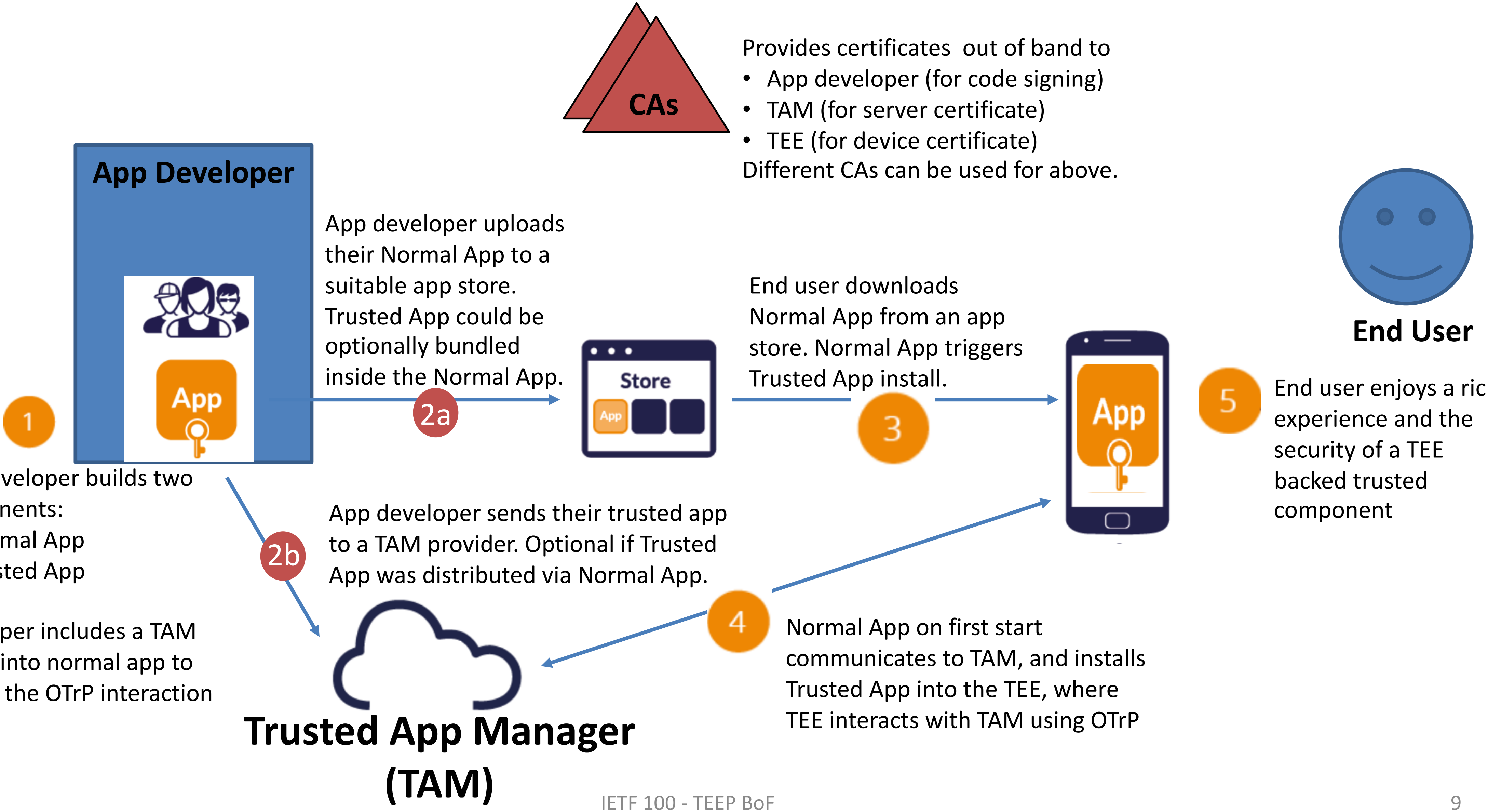
1. Payment
   – Only authorized code can make payments or see payment data, to protect against financial loss
2. IoT
   – Only authorized code can access physical actuator/sensor, to protect against safety issues
3. Confidential cloud computing
   – Only tenant (not cloud hoster) can access data

Desirable hardware based security for critical applications



A TA often needs to be provided to TEE over-the-air and managed

Entity Roles and Experience



Provides certificates out of band to
- App developer (for code signing)
- TAM (for server certificate)
- TEE (for device certificate)

Different CAs can be used for above.

App developer builds two components:
1) Normal App
2) Trusted App

Developer includes a TAM library into normal app to handle the OTrP interaction

App developer uploads their Normal App to a suitable app store. Trusted App could be optionally bundled inside the Normal App.

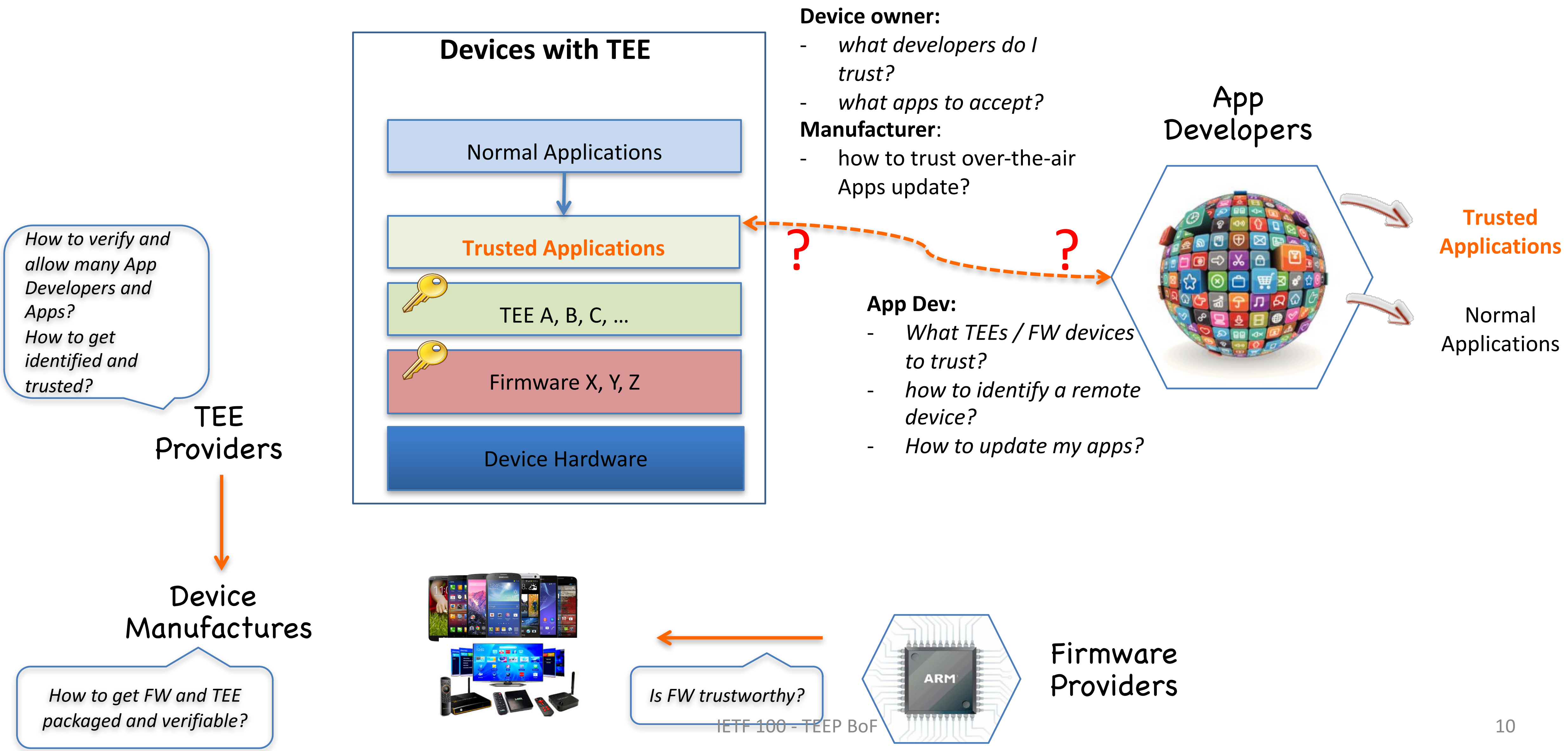
App developer sends their trusted app to a TAM provider. Optional if Trusted App was distributed via Normal App.

End user downloads Normal App from an app store. Normal App triggers Trusted App install.

Normal App on first start communicates to TAM, and installs Trusted App into the TEE, where TEE interacts with TAM using OTrP

End user enjoys a rich experience and the security of a TEE backed trusted component

Gaps to utilize hardware based security



Devices with TEE

Normal Applications

Trusted Applications

TEE A, B, C, …

Firmware X, Y, Z

Device Hardware

?

?

Device owner:

- *what developers do I trust?*
- *what apps to accept?*

Manufacturer:

- how to trust over-the-air Apps update?

App Developers

Trusted Applications

Normal Applications

App Dev:

- *What TEEs / FW devices to trust?*
- *how to identify a remote device?*
- *How to update my apps?*

How to verify and allow many App Developers and Apps?
How to get identified and trusted?

TEE Providers

Device Manufactures

How to get FW and TEE packaged and verifiable?

Is FW trustworthy?

Firmware Providers

The Problems

- Adoption gap for App Developers
  - – Applications have to be provisioned somehow into the TEE
  - – Many device manufacturers + many device types (e.g., phones, tablets, networking equipment, servers) + multiple TEE providers
    - An application provider needs to support

- Lack of standards to manage TAs
  - – Via proprietary techniques today
  - – Need to answer
    - How is mutual trust based and verified
      - – App Developers / TAM trusts Device's TEE / FW
      - – Device trusts App Developers and Apps to be installed and updated
    - What messages for mutual communication
    - What permissions that different entities should have

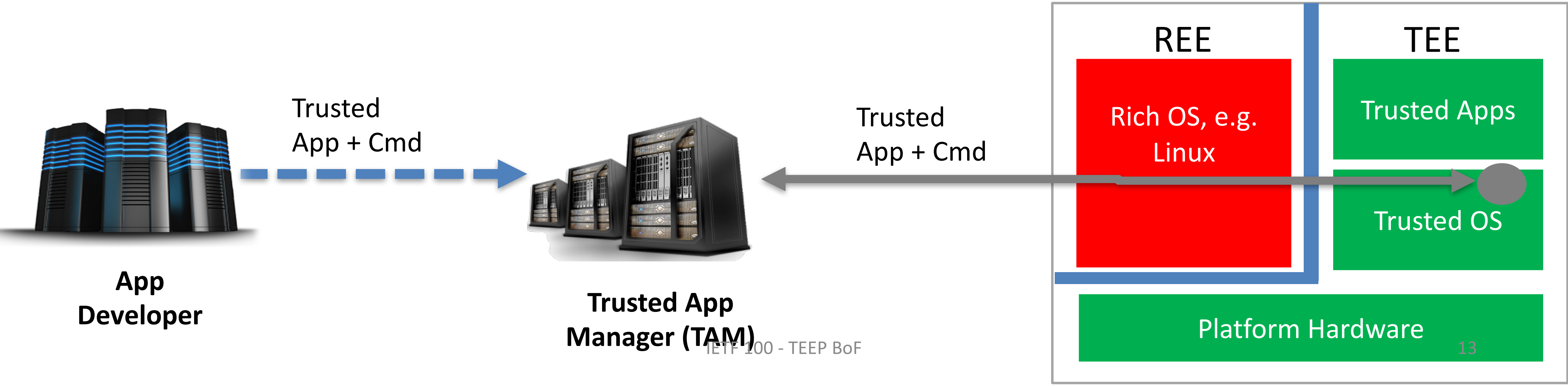
- Fragmentation is growing - IoT accelerated that fragmentation

Goal

- Define a standardized protocol for providing and managing trusted applications in various devices with TEE
  - Grow the adoption of trusted applications to reduce the inherent security weakness with rich OS
  - Non-lock in for broad device types and providers
    - E.g., allow a common TAM to work with multiple TEE & device vendors and flavors
  - Such a protocol better provides security

IETF Work TBD: A Protocol

- To illustrate the idea a proposal has been put together -- the Open Trust Protocol (OTrP)
- OTrP is currently a JSON/JOSE-based application layer security protocol that runs between a TAM and a component in the TEE OS
  - Open for draft update in WG (e.g. JSON vs. CBOR, mandatory transport protocol support etc.)



App Developer

Trusted App + Cmd

Trusted App Manager (TAM)

Trusted App + Cmd

REE

Rich OS, e.g. Linux

TEE

Trusted Apps

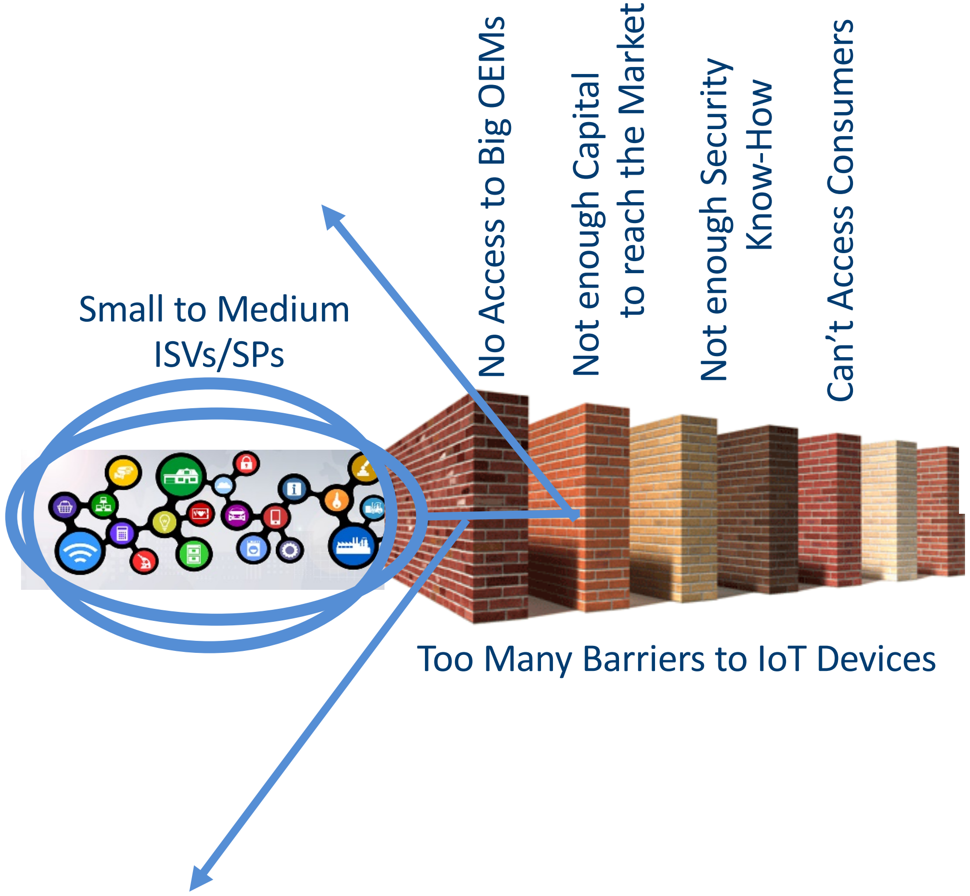
Trusted OS

Platform Hardware

Q&A

Backup

Small to Medium ISV’s & SPs have a Problem



Small to Medium ISVs/SPs

No Access to Big OEMs

Not enough Capital to reach the Market

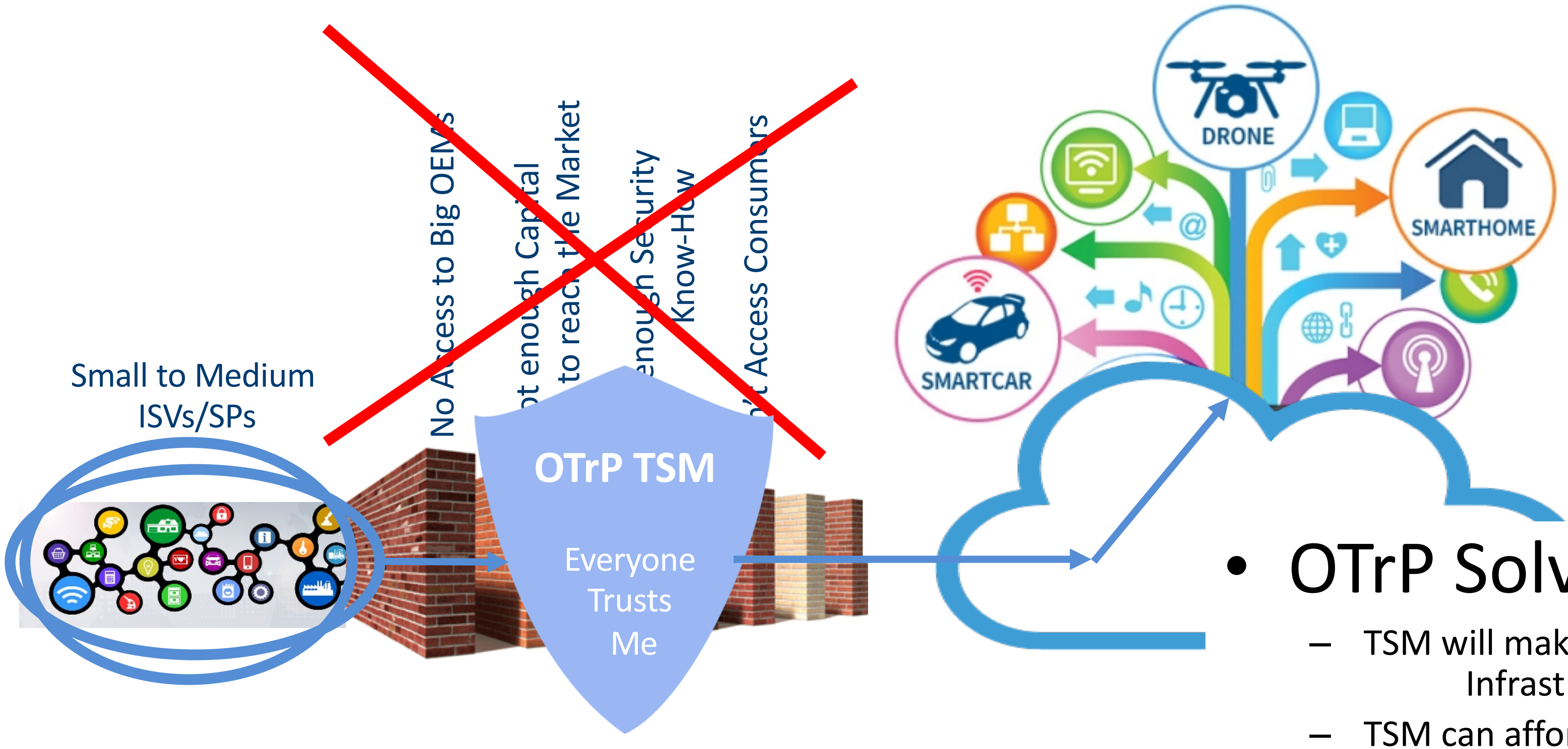
Not enough Security Know-How

Can’t Access Consumers

Too Many Barriers to IoT Devices

- Small ISV’s and Service Providers
  - Don’t have the clout to talk to big OEMs
  - Don’t have the capital to build large infrastructure
  - Don’t have the Brains & Brawn to tackle security on the devices

OTrP is Striking a Market Need



OTrP TSM Punctures the Barriers
For Small to Medium Sized ISV’s & SP’s

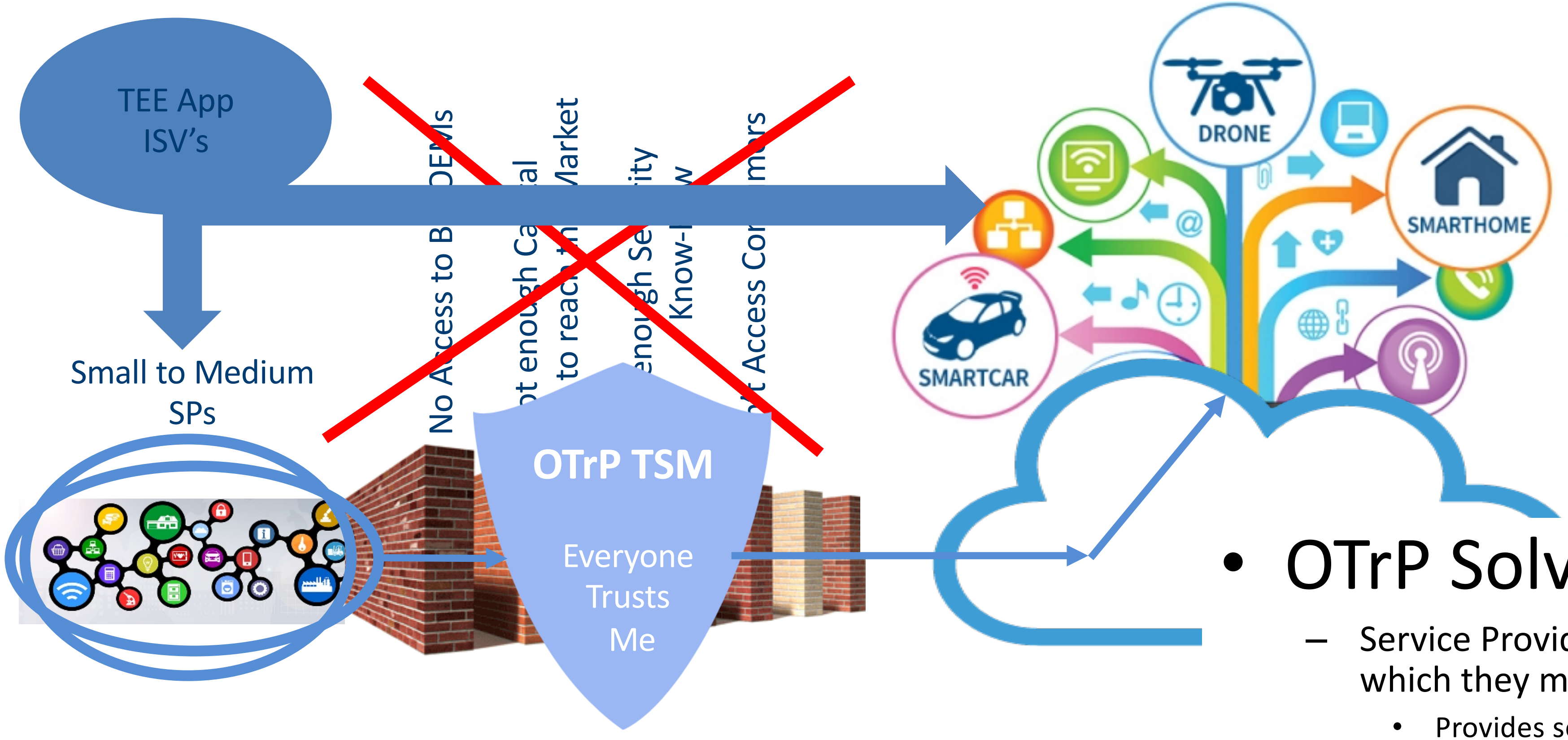
Large SP’s can benefit from OTrP because they can scale their infrastructure investment to their available market easily at lower cost

- OTrP Solves Their Problems
  - TSM will make deals with big OEMs & Infrastructure players
  - TSM can afford to build out infrastructure, because costs are leveraged across many ISVs and SPs
  - TSM will hire the Brains & Brawn and manage the security (ISVs/SPs only need a single certificate)
  - OTrP TSM is a ready-to-go Cloud solution

OTrP Addresses Security Know-H

Secure App Problem

Even with access to the TEE,
a Service Provider
may not really have the
Security Expertise
to create their own
Trusted Applications
to run inside the TEE, or re-sign
someone else's apps



The Service Provider does not have the *knowledge* to build trusted apps for different platforms and TEEs.
The Security Domain in OTrP allows the service provider to *just buy trusted apps from ISVs*, not have to even re-sign those apps or manage their attestation, and install them into their own TEE

- OTrP Solves Their Problems
  - Service Provider is given a Security Domain into which they may place their applications
    - Provides separation between different SP's applications
  - Allows Security Domain to host off-the-shelf/common trusted applications which are bound specifically to the Service Provider
    - Common Secure Key Manager
    - Common Cloud Agent