TEEP WG

Problem Statement Recap

IETF 101, London
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

Figure: Hardware Architectural View of REE and TEE, Global Platform, TEE System Architecture v1.1
Despite such widely available TEE environment

• Trusted App development and distribution are hard
  – Much less than that for normal apps via App Store
  – Trust and management issues due to multiple parties involved in the scenario
Example use cases for TEE apps

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
**Entity Roles and Example Experience**

1. **App Developer**
   - App developer builds two components:
     1) Normal App
     2) Trusted App
   - Developer includes a TAM library into normal app to handle the OTrP interaction.

2a. **App developer uploads their Normal App to a suitable app store. Trusted App could be optionally bundled inside the Normal App.**

2b. **App developer sends their trusted app to a TAM provider. Optional if Trusted App was distributed via Normal App.**

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

4. **Normal App on first start communicates to TAM, and installs Trusted App into the TEE.**

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

**CAs**
- 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.

**Entity Roles and Example Experience**

IETF 101 - TEEP WG
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 Dev:
- What TEEs / FW devices to trust?
- how to identify a remote device?
- How to update my apps?

App Developers

Trusted Applications

Normal Applications

TEE Providers

Device Manufactures

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

How to get FW and TEE packaged and verifiable?

Firmware Providers

Is FW trustworthy?

IETF 101 - TEEP WG
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 Trusted Apps
  – 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 TAM to work with multiple TEE & device vendors and flavors
  – Such a protocol better provides security
Software Updates for IoT (SUIT) WG relationship

• TEEP focuses on more on trusted “apps” after boot, whereas SUIT focuses more on “firmware” for boot
• TEEP focuses on installation of code into a Trusted Execution Environment (whether for IoT or not), whereas SUIT focuses on installation of code on an IoT device (whether in a TEE or not)
• TEEP focuses more on initial provisioning of code the first time, whereas SUIT focuses more on subsequent updates to already-provisioned code
Q&A