Design Document: Java API for Trusted Application

RUI YANG

rui.yang@aalto.fi

Abstract

Based on GlobalPlatform specification, Open-TEE paved a way for normal application developers to develop and deploy GP-compiant Trusted Applications (TA) in Trust Execution Environment (TEE). However, in order to develop a Client Application (CA) (especially Android application), there still lacks an efficient way of exposing the underlining C-based APIs to the application layer which is written in Java. In this design document, this problem will be adderssed in more details and the possibility to wrap the C APIs into Java APIs is discussed.

KEYWORDS: Open-TEE, Java API, JNI

1 Problem Description

Open-TEE is a virtual TEE which is based on GlobalPlatform (GP) TEE specifications. It can run on a mobile device either with real GP-Compliant TEE hardwares or without. If the mobile device is equipped with GP-Compliant TEE hardware, via one module of Open-TEE called "libtee" `https://github.com/Open-TEE/libtee` with corrsponding lower layer linux driver `https://github.com/Open-TEE/tee-engine-Driver`, Open-TEE can allow the communications between CAs in Rich Execution Environment (REE) and TAs in the real TEE.

If the mobile device does not have a real GP-Compliant TEE hardware, Open-TEE can run as an application in REE itself, which does not provide the security features of a TEE. TAs can be deployed in Open-TEE and the communications between CAs and TAs are provided.

The corresponding GP specifications are written in C as such Open-TEE is implemented in C. The problems are list as follows which will occur once Android application developers start using Open-TEE.

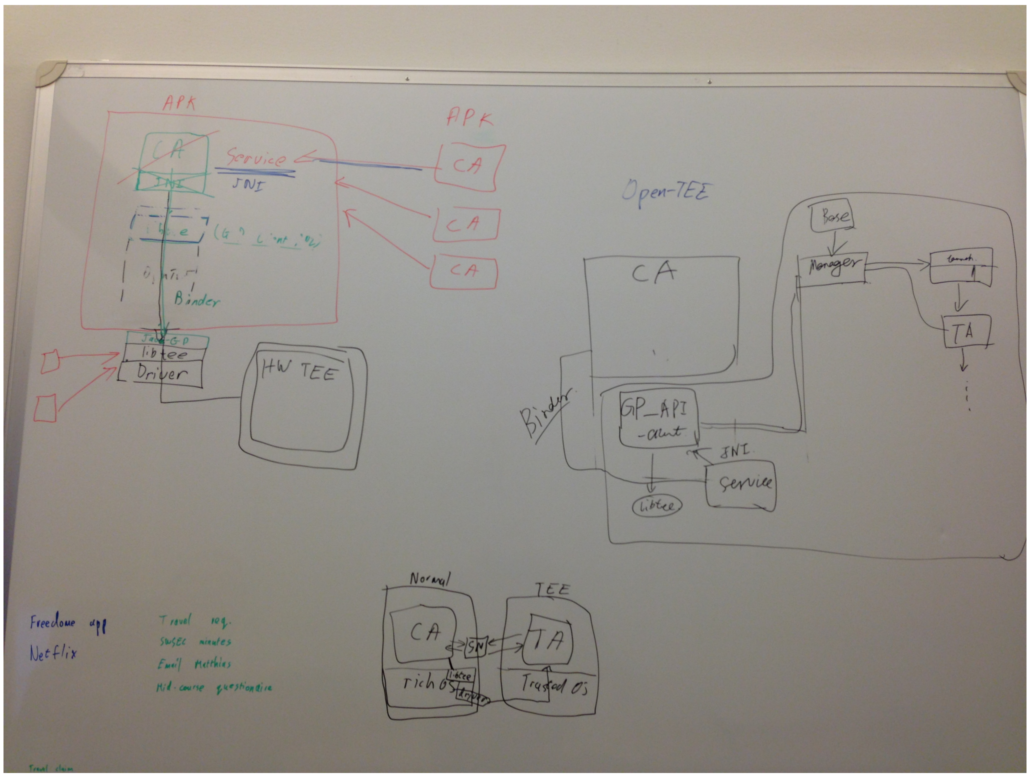

Figure 1: Draft for reminder

1.1 Problem One

The first problem is how to use the C APIs in the application layer. Java Native Interface (JNI) provides the ways to allow the communications between C and Java. So a middle layer between the application layer and C API layer should be provided to reduce the redundant works of application developers.

1.2 Problem Two

As stated above, the GP TEE specifications only focus on C. There is one notion called "Shared Memory" which is utilized in the specification to avoid big string of memory copies between the CA and TA if they wants to communicate with each other. In C programming language, sharing memory between two processes can be achieved by declaring that these two processes want to share part of their memory. However, since the CA in our scenario is written in Java which does not have the notions of referencing to the memory address in run time environment, which makes the notion of "Shared Memory" hard to implement.

2 Possible Solutions