Platform Security
Secure Systems Group

Rui Yang, Thomas Nyman, Andrew Paverd, N. Asokan

Java API for Trusted Execution Environment

- To interact with trusted applications (TAs), Android developers have to write **complex native code** using the **GlobalPlatform TEE Client API**.
- Designed a Java API for using TEE from Android.

API Design

(a) Full functionality – Java API provides full coverage of the GP TEE Client API.

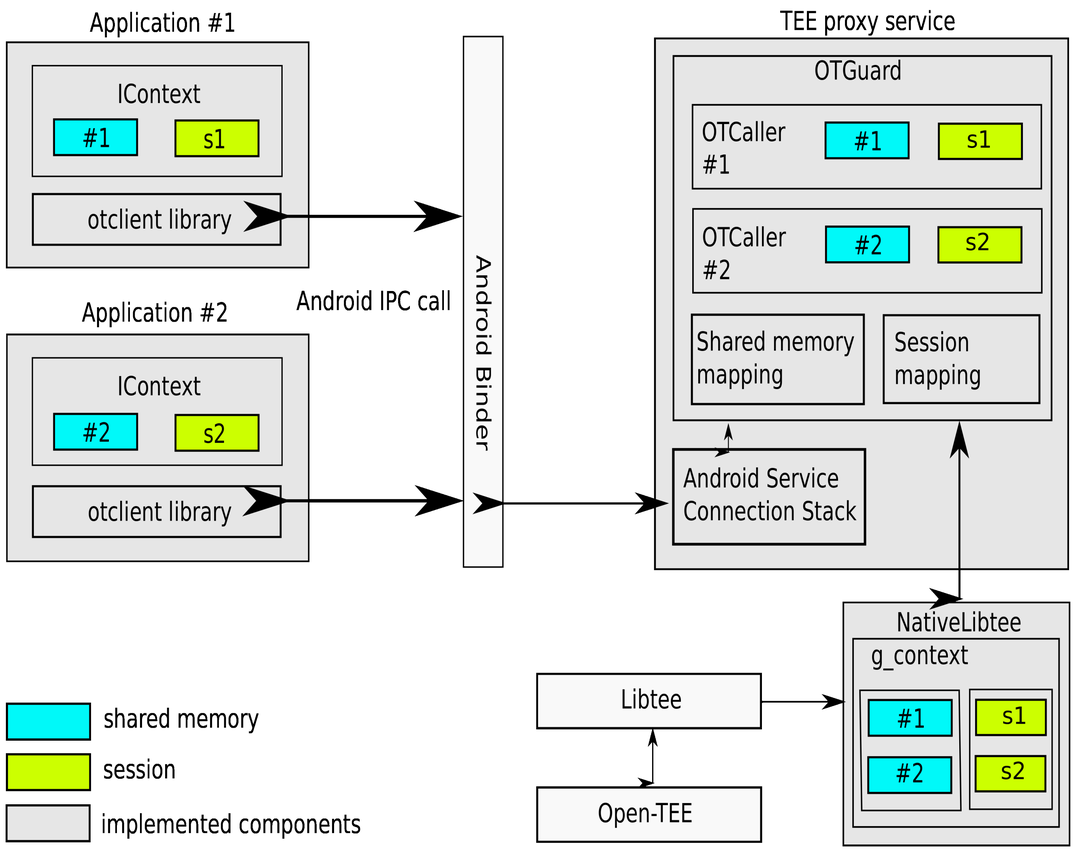
(b) Java conventions

1. **object oriented** – all data types transformed to Java classes. API functions mapped to corresponding classes;
2. **exceptions** – errors indicated using new Java exceptions;
3. **factory method based design pattern**.

(c) Usable – relieve burdens for use-case specified Java Native Interface to native code.

(d) Challenges – GP TEE Client API is specified in terms of C binding:

1. Comply with SE Android restrictions;
2. Mapping C constructs to Java:
   * error codes to exceptions;
   * union represented as class inheritance;
   * single values to enums.




```
ret = TEEC_InitializeContext(&context, ...);
if( ret != TEEC_SUCCESS ) return ret;
ret = TEEC_OpenSession(&context, &session, ...);
operation.params[0].memref.parent = &shared_memory;
operation.params[1].value.a = a;
operation.params[1].value.b = b;
ret = TEEC_InvokeCommand(&session, CMD_DO_ENC,
&operation, &retOrigin);
```

GP TEE Client API example

```
try {
    ITEEClient.IContext context = client.initializeContext(...);
    ITEEClient.ISession session = context.openSession(...);
    ITEEClient.IValue value = client.newValue(a, b, ...);
    ITEEClient.IRegisteredMemoryReference rmr =
client.newRegisteredMemoryReference(shared_memory, ...);
    ITEEClient.IOperation operation = client.newOperation(rmr,
value);
    session.invokeCommand(CMD_DO_ENC, operation);
} catch (TEEClientException e) {
    retOrigin = e.getReturnOrigin();
}
```

Java API example

Architecture Design

(a) Exposing TEE functionality as a TEE proxy service – applications use the TEE as an ordinary Android service.

(b) Support multiplexing – resource isolation between different applications.

(c) Compatible with SE Android – using Android Binder to communicate with proxy service.

Prototype implementation:

1. Fully implemented the Java API;
2. Utilizing Open-TEE and OmniShare.

Aalto University

HUAWEI

https://open-tee.github.io/