Writing an optee application

Peng Xu

May 17, 2019
1. Why we need TEE-based application?
2. What are the difference between them and normal C/C++ program?
3. Basic elements we need to develop TEE application (Intel SGX)?
4. Three fundamental modules in main() (Intel SGX)?
SGX Application - SampleEnclave

1. App
   - App.cpp/App.h
   - Edger8rSyntax/Functions.cpp

2. Enclave
   - Enclave.h/Enclave.cpp
   - Enclave.edl
   - Enclave_private.pem
   - Edger8rSyntax/Functions.cpp
   - Edger8rSyntax/Functions.edl

3. Include

4. Makefile
OPTEE Application - mathematics operation

1. Host(Client Application)
   ▶ host.c/.h
   ▶ Makefile

2. TA(Trusted Application)
   ▶ math.c/.h
   ▶ Makefile

3. Both Host and TA sides are written in C
Client Application

1. Basic data structure
   ▶ TEEC_Result res;
   ▶ TEEC_Context ctx; /*Represents a connection between a client application and a TEE. */
   ▶ TEEC_Session sess; /*Represents a connection between a client application and a trusted application.*/
   ▶ TEEC_Operation op; /*Holds information and memory references.*/
   ▶ TEEC_UUID uuid; /*UUID values are used to identify Trusted Applications.*/
Client Application

1. Basic data structure
   - TEEC_Result res;
   - TEEC_Context ctx; /* Represents a connection between a client application and a TEE. */
   - TEEC_Session sess; /* Represents a connection between a client application and a trusted application. */
   - TEEC_Operation op; /* Holds information and memory references. */
   - TEEC_UUID uuid; /* UUID values are used to identify Trusted Applications. */

2. Basic functions
   - TEEC_InitializeContext(&ctx) /* Initialize a context */
   - TEEC_OpenSession(&ctx,&sess,&uuid) /* Open a session */
   - TEEC_InvokeCommand(&sess, cmd, &op,&err_origin)
   - TEEC_CloseSession(&sess); /* Close the session */
   - TEEC_FinalizeContext(&ctx); /* Destory the context */
**Client Application**

1. **TEEC_Context ctx; */*Represents a connection between a client application and a TEE. */*
   
   ```
   typedef struct {
      /* Implementation defined */
      int fd;
      bool reg_mem;
   } TEEC_Context;
   ```

2. **TEEC_Session sess; */*Represents a connection between a client application and a trusted application. */*
   
   ```
   typedef struct {
      /* Implementation defined */
      TEEC_Context *ctx;
      uint32_t session_id;
   } TEEC_Session;
   ```
Trusted Application

1. Basic data structure
   ▶ TEE_Result
   ▶ TEE_Param

2. Basic functions
   ▶ TA_CreateEntryPoint(void) /* Called when the instance is created */
   ▶ TA_OpenSessionEntryPoint(param_types, params[4], **sess_ctx) /* Called when a new session is opened to the TA. */
   ▶ TA_InvokeCommandEntryPoint(*sess_ctx, cmd_id, param_types, params[4]) /* Called when a TA is invoked */
   ▶ TA_CloseSessionEntryPoint(&sess_ctx) /* Called when a session is closed */
   ▶ TA_DestroyEntryPoint(void) /* Called when the instance is destroyed */
Logic functions interfacing between CA and TA

1. Arguments preparation
   - op.paramTypes = TEEC_PARAM_TYPES();
   - op.params[0].value.a = 42; Prepare the argument
   - TEEC_InvokeCommand(&sess, opcmd, &op, &err_origin)

2. Command receiving and processing
   - TA_InvokeCommandEntryPoint(*sess_ctx, cmd_id, param_types, params[4])
     { ... 
     switch (cmd_id) { 
     case a: return inc_value(param_types, params);
     case b: return dec_value(param_types, params); ... 
     }
     }
   - inc_value() { params[0].value.a++; }
Questions?