Writing a SGX application

Peng Xu

April 30, 2019

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

ECALL & OCALL

1. Enclave Calls (ECALLs)
   - The application can invoke a pre-defined function inside the enclave, passing input parameters and pointers to shared memory within the application.
   - Those invocations from the application to the enclave are called ECALL.
2. Outside Calls (OCALLs)
   - When an enclave executes, it can perform an OCALL to a pre-defined function in the application(non-secure world).
   - Contrary to an ECALL, an OCALL cannot share enclave memory with the application, so it must copy the parameters into the application memory before the OCALL.

***.edl

1. EDL: Enclave description language
2. Import ECALL/OCALL from sub-directory EDLs

***.edl

1. EDL: Enclave description language
2. Import ECALL/OCALL from sub-directory EDLs
3. trusted {public void ecall_array_in([in] int arr[4]);};
   - *[]* can be used to declare an array
   - *[user_check]* the buffer pointed by 'arr' is not copied into the enclave either. But enclave can modify the memory outside.
   - *[in]* buffer for the array will be allocated inside the enclave
   - *[out]* buffer for the array will be allocated inside the enclave, but the content of the array won't be copied
   - *[in,out]* buffer for the array will be allocated inside the enclave, the content of the array will be copied either

***.edl

1. EDL: Enclave description language
2. Import ECALL/OCALL from sub-directory EDLs
3. trusted {public void ecall_array_in([in] int arr[4]);};
   - *[]* can be used to declare an array
   - *[user_check]* the buffer pointed by 'arr' is not copied into the enclave either. But enclave can modify the memory outside.
   - *[in]* buffer for the array will be allocated inside the enclave
   - *[out]* buffer for the array will be allocated inside the enclave, but the content of the array won't be copied
   - *[in,out]* buffer for the array will be allocated inside the enclave, the content of the array will be copied either
4. untrusted {void ocall_function_allow(void) allow(ecall_function_private);}
   - OCALL 'ocall_function_allow' can invoke ECALL 'ecall_function_private' in App side.

App.h/App.cpp

1. main()
   - Initialize the enclave
   - Logical functionalities
   - Destroy the enclave
2. initialize_enclave
   - read the token from saved file
   - call sgx_create_enclave to initialize an enclave instance
   - save the launch token if it is updated
3. Logical functionalities
   - edger8r_function_defination()
   - ECALL: ecall_function_public(*global_eid*);

Enclave.h/Enclave.cpp

1. Logical function declaration and implementation
   - Enclave.h int printf(const char* fmt, ...);
   - Enclave.cpp int printf(const char* fmt, ...){
     ...;
     ocall_print_string(buf);
     ...;
     }
   - ocall_print_string(buf) is declared and defined in App.h/App.cpp

Enclave.edl

1. Top enclave file
   - include "user_types.h" /* buffer_t */
   - from "Edger8rSyntax/Functions.edl" import *;
     Import ECALL/OCALL from sub-directory EDLs.
   - trusted {
     };
   - untrusted {
     void ocall_print_string([in, string] const char *str);
     };
     invokes OCALL

Enclave_private.pem

1. sign private key
2. how to use?
   - $(Signed_Enclave_Name): $(Enclave_Name) @$(SGX_ENCLAVE_SIGNER) sign -key Enclave/Enclave_private.pem -enclave $(Enclave_Name) -out $@ -config $(Enclave_Config_File) @echo "SIGN = $@"
3. how to generate this private key?
   - openssl rsa -in private.pem -outform PEM -pubout -out public.pem

Edger8rSyntax/Functions.cpp

1. Logical function implementation
   - void ecall_function_public(void)
   - int ecall_function_private(void)

Edger8rSyntax/Functions.edl

1. public ECALL can be called directly in App.
   public void ecall_function_public(void);
2. private ECALL cannot be called directly in App.
   int ecall_function_private(void);

Makefile

1. SGX SDK Settings
   - SGX_SDK ?= /opt/intel/sgxsdk
   - SGX_DEBUG ?= 1
   - SGX_MODE ?= HW
   - SGX_ARCH ?= x64
   - SGX_COMMON_CFLAGS, SGX_COMMON_CXXFLAGS, etc

Makefile

1. SGX SDK Settings
   - SGX_SDK ?= /opt/intel/sgxsdk
   - SGX_DEBUG ?= 1
   - SGX_MODE ?= HW
   - SGX_ARCH ?= x64
   - SGX_COMMON_CFLAGS, SGX_COMMON_CXXFLAGS, etc
2. App side
   - App_Cpp_Files := App/App.cpp $(wildcard App/Edger8rSyntax/*.cpp) $(wildcard App/TrustedLibrary/*.cpp)
   - App_Include_Paths := -IInclude -IApp -I$(SGX_SDK)/include
   - App_C_Flag := -fPIC -Wno-attributes $(App_Include_Paths)
   - App_Cpp_Flags, App_Link_Flags, App_Cpp_Objects and so on

Makefile

1. SGX SDK Settings
2. App side
3. Enclave side
   - Enclave_Cpp_Files := Enclave/Enclave.cpp $(wildcard Enclave/Edger8rSyntax/*.cpp) $(wildcard Enclave/TrustedLibrary/*.cpp)
   - Enclave_Include_Paths := -IInclude -IEnclave -I$(SGX_SDK)/include -I$(SGX_SDK)/include/tlibc -I$(SGX_SDK)/include/libcxx
   - Enclave_C_Flags := $(Enclave_Include_Paths) -nostdinc -fvisibility=hidden -fpie -ffunction-sections -fdata-sections
   - Enclave_Link_Flags, Enclave_Cpp_Objects, Enclave_Name, Signed_Enclave_Name, Enclave_Config_File and so on

Other files

1. Enclave.config.xml
2. Enclave.lds
3. Config.xml
4. etc.

Question?

Questions?