Writing a SGX application

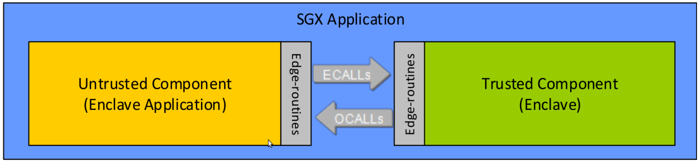
Tiago M. Dias
(tiago.dias@isel.pt)

December 2021

Writing a SGX application

Partitioning

- The first step in designing an Intel SGX enabled application is to identify the assets it needs to protect, the data structures where the assets are contained, and the set of code that operates on those data structures and then place them into a separate trusted library, i.e `the enclave`.


SGX Application
Untrusted Component (Enclave Application)
Edge-routines
ECALLs
OCALLs
Edge-routines
Trusted Component (Enclave)

Curiosity

Partitioning also plays a key role in preparing an Intel SGX application to manage power events!

Writing a SGX application

Partitioning

Recommendation

Conduct a security analysis of the application and properly partition it making the decision about what code and data is placed in the enclave.

Writing a SGX application

Enclave

- An enclave is a monolithic software entity that reduces the Trusted Computing Base (TCB) for an application to a trusted runtime system, ISV code and 3rd party trusted libraries.
  - A bug in one component may compromise the security properties of the enclave.
  - ISVs should attempt to minimize the enclave size.
- The code in an enclave is no different than code that exists as part of a regular application.
- Enclave functions can rely on special versions of the C/C++ runtime libraries, STL, synchronization and several other trusted libraries that are part of the Intel SGX SDK.
- Enclave source code is built as a shared object!

Writing a SGX application

Interface

- After defining the trusted (enclave) and untrusted (application) components of an Intel SGX enabled application, the developer should carefully define the interface between untrusted application and enclave.
- An enclave must expose an API for applications to call in (`ECalls`) and advertise what services provided by the untrusted domain are needed (`OCalls`).
- Since `ECalls` expose the interface that an untrusted application may use, the enclave attack surface should be reduce by limiting the number of ECalls.
- An enclave has no control on which `ECall` is executed, or the order in which `ECalls` are invoked. Thus, an enclave cannot depend on `ECalls` occurring in certain order.

Writing a SGX application

Enclave Definition Language (EDL) files

- Enclave Definition Language (EDL) files are meant to describe enclave trusted and untrusted functions and types used in the function prototypes.

```
enclave {
     //Include files
     //Import other edl files
     //Data structure declarations to be used as parameters of the function prototypes in edl
     trusted {
          //Include header files if any
          //Will be includedd in enclave_t.h
          //Trusted function prototypes
     };
     untrusted {
          //Include header files if any
          //Will be included in enclave_u.hhead
          //Untrusted function prototypes
     };
};
```

Attention

The untrusted block is always optional.

Writing a SGX application

Enclave Definition Language (EDL) files

- Enclave Definition Language (EDL) files are meant to describe enclave trusted and untrusted functions and types used in the function prototypes.

```
enclave {
    //Include files
    //Import other edl files
    //Data structure declarations to be used as parameters of the function prototypes in edl
    trusted {
        //Include header files if any
        //Will be includedd in enclave_t.h
        //Trusted function prototypes
    };
    untrusted {
        //Include header files if any
        //Will be included in enclave_u.hhead
        //Untrusted function prototypes
    };
};
```

Attention

The trusted block is optional only if it is used as a library EDL, and this EDL would be imported by other EDL files.

Writing a SGX application

Enclave Definition Language (EDL) files :: Pointers

- Pointers should be decorated with either a pointer direction attribute in, out or a user_check attribute explicitly:
  - in and out serve as direction attributes.
  - in an out may be combined.

```
enclave {
     trusted {
          public void test_ecall_user_check([user_check] int * ptr);
          public void test_ecall_in([in] int * ptr);
          public void test_ecall_out([out] int * ptr);
          public void test_ecall_in_out([in, out] int * ptr);
     };
     untrusted {
          void test_OCall_user_check([user_check] int * ptr);
          void test_OCall_in([in] int * ptr);
          void test_OCall_out([out] int * ptr);
          void test_OCall_in_out([in, out] int * ptr);
     };
};
```

Writing a SGX application

Enclave Definition Language (EDL) files :: Buffer Size Calculation

- The direction attribute is usually followed by a `size` or `count` modifier.
- The generalized formula for calculating the buffer size using these attributes is:

$$\text{Total number of bytes} = \texttt{count} \times \texttt{size}$$

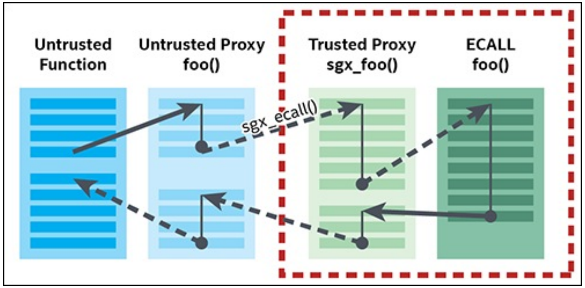
- If `count` is not specified, then it is assumed to be equal to 1.
- If `size` is not specified, then it is assumed to sizeof (element pointed by the pointer).

```
enclave{
     trusted {
          // Copies cnt * sizeof(int) bytes
          public void test_count([in, count=cnt] int* ptr, unsigned cnt);
          // Copies cnt * len bytes
          public void test_count_size([in, count=cnt, size=len] int* ptr, unsigned cnt, size_t
     len);
     };
};
```

Writing a SGX application

Proxy functions

- Edger8r Tool uses the EDL file to create proxy (C wrapper) functions for both enclave exports (used by ECALLs) and imports (used by OCalls).
- Each ECall and OCall gets a pair of proxy functions: a trusted half and an untrusted half.
- A program does not call the ECall and OCall functions directly; it calls these proxy functions.



Writing a SGX application

Enclave Development Basics

The typical enclave development process includes the following steps:

1. Define the interface between the untrusted application and the enclave in the EDL file.
2. Implement the application and enclave functions.
3. Build the application and enclave.
4. Run and debug the application in simulation and hardware modes.
5. Prepare the application and enclave for release.

Attention

In the build process,

- `Edger8r Tool` generates trusted and untrusted proxy/bridge functions;
- `Enclave Signing Tool` generates the metadata and signature for the enclave.

References

- Intel Software Guard Extensions Programming Reference
- Intel Software Guard Extensions Enclave Writer's Guide
- Intel Software Guard Extensions (Intel SGX) Developer Guide