Intel® SGX Token for SoftHSM Configuration and Build Steps

Contents

SGX Token for SoftHSM Overview

The general architecture of SoftHSM is such that modular "tokens" (or TokenModule) may be compiled into the SoftHSM library (libsofthsm2.so) by modifying CMakeLists and MakeFiles. The token provides the crypto operations required by the SoftHSM library to comply with the PKCS#11 standard. The SoftHSM library provides a management layer that coordinates Slots, (PKCS#11) Tokens, Object Handles, and Sessions, which facilitate compliance of the PKCS#11 standard. The SoftHSM library specifies the "Token API" interface which provide the means to supply the cryptographic operations consistent with PKCS#11.

A SoftHSM token implements the Token API and associated crypto operations using its desired method, either in software, backed by a corresponding implementation (e.g. Botan, OpenSSL, etc.), like the default SW-based crypto Soft Token, or in hardware with similar backing. In the SGX Token for SoftHSM implementation, the token library doesn't perform any crypto operations itself but offloads them to SGX through the SGX Crypto API Toolkit, which utilizes a version of OpenSSL accessibly by the Trusted Execution Environment.

Figure 1: SoftHSM with pluggable Token diagram shows the high-level architecture of the SoftHSM library, and a TokenModule that is dynamically loaded and provides the crypto operations required of a SoftHSM token.

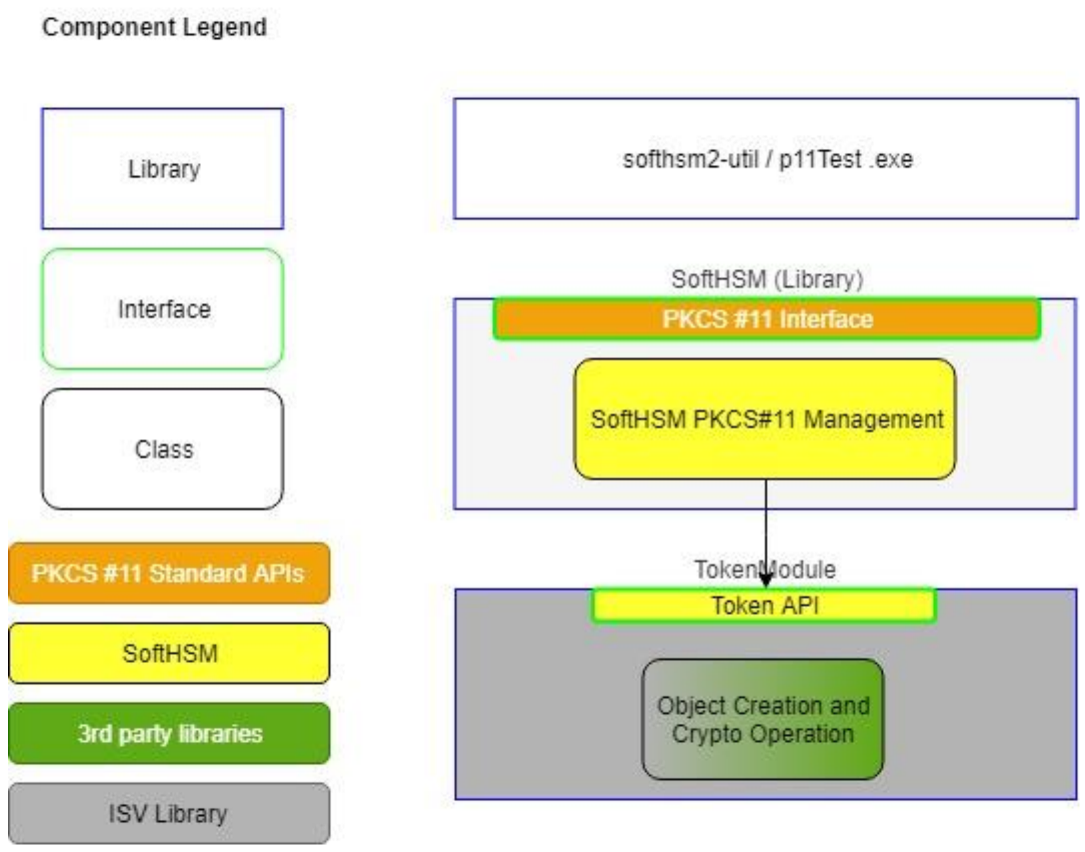

Figure 1: SoftHSM with pluggable Token diagram

Generally, per Figure 1: SoftHSM with pluggable Token diagram, the TokenModule may be substituted with a token library that conforms to the Token API interface and provides sufficient crypto operations to satisfy the PKCS#11 APIs for any particular flow.

The SGX Token implementation consists of 3 major components:

1. SGX Token library (TokenModule) for SoftHSM logic
2. SGX Crypto API Toolkit
    a. P11 Provider (Untrusted component)
    b. P11 Enclave (Trusted component)
3. Intel® SGX OpenSSL

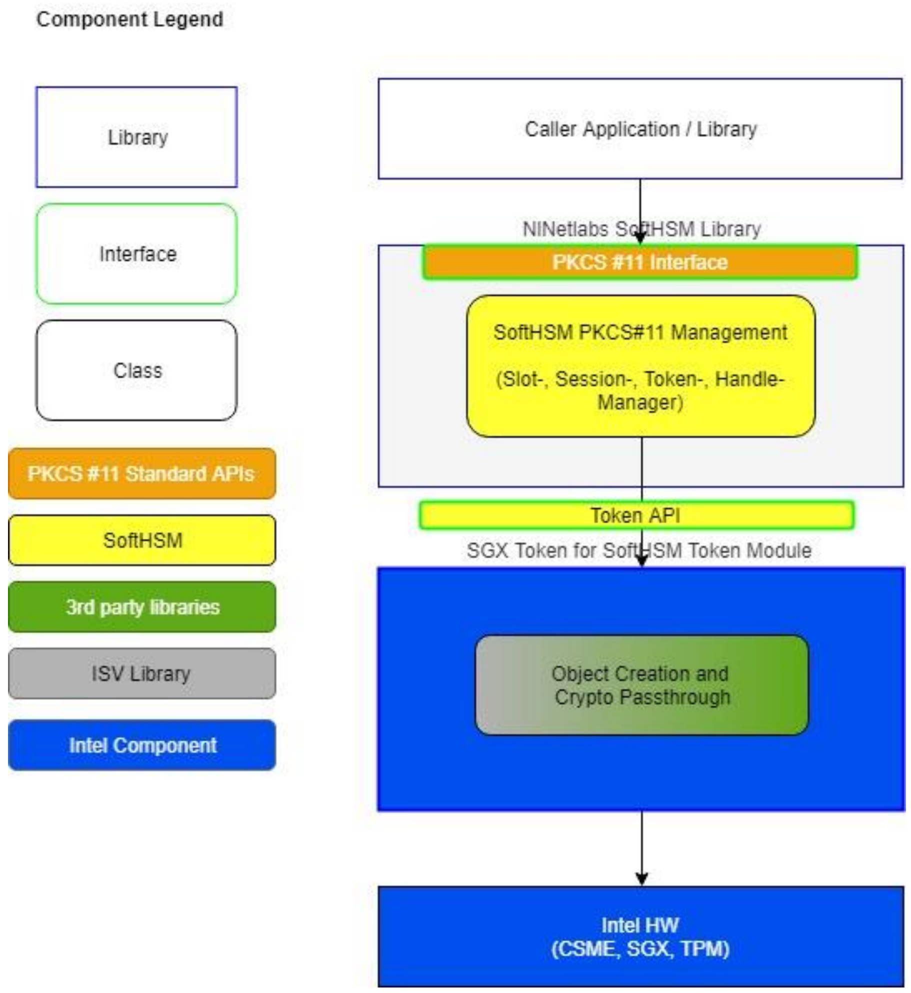
Component Legend
Library
Interface
Class
PKCS #11 Standard APIs
SoftHSM
3rd party libraries
ISV Library
Intel Component
Caller Application / Library
NINetlabs SoftHSM Library
PKCS #11 Interface
SoftHSM PKCS#11 Management
(Slot-, Session-, Token-, Handle-Manager)
Token API
SGX Token for SoftHSM Token Module
Object Creation and Crypto Passthrough
Intel HW
(CSME, SGX, TPM)

Figure 2: SGX Token for SoftHSM Implementation

The SGX Token for SoftHSM implementation, indicated in blue in Figure 2: SGX Token for SoftHSM Implementation, is a set of libraries that provide the APIs required by the SoftHSM token interface, and the crypto operations provided by the Intel® SGX OpenSSL library, via the p11Provider and p11Enclave libraries together known as the SGX Crypto API Toolkit.

The SGX Crypto API Toolkit is the backend supplier of cryptographic operations for the SGX Token, and enables the SGX Token to utilize SGX as a HW root-of-trust for PKCS#11 Cryptoki cryptographic operations.

Component Description

Green (3rd-party) components are:

- OpenSSL: Intel®-ported version of the OpenSSL cryptographic library

Gray/Yellow components are:

- SoftHSM: PKCS#11-compliant library which consumes different token implementations that provide the cryptographic operations for the PKCS#11 Cryptoki. Provides the front-end interface for performing PKCS#11 operations and requires a token implementation to provide the cryptographic backend, such as through SW (Soft Token) or HW (SGX Token).

Additionally, Yellow is either provided or specified by SoftHSM, such as the token interface.

The Blue components are Intel-provided:

- Intel® SGX Token for SoftHSM: Token implementation for SoftHSM that supplies the operations specified by the Token API, backed by SGX through the SGX Crypto API Toolkit.
- Intel® SGX Crypto API Toolkit: Cryptographic library interface utilizing SGX as the Trusted Execution Environment
- Intel® SGX: Intel Software Guard Extensions Trusted Execution Environment, which provides a programming model and generic secure environment for sensitive operations.

Build Instructions

The SGX Token for SoftHSM may be reproduced by:

1. Download all required source code
2. Install prerequisite software dependencies
3. Update build files
4. Build and install all required components

Requirements

Software

- Ubuntu 16.04
- SGX Token for SoftHSM
- SoftHSM
- SGX Crypto API Toolkit
- SGX SDK and PSW
- SGX OpenSSL

Hardware Requirements

- Platform with Intel® Core processor 6th generation (SkyLake) or later

Source Code Locations and Version

- SGX Token for SoftHSM
    - https://github.com/intel/softhsm-tokens/sgx
- SoftHSM
    - https://gitlab.nlnetlabs.nl/NLnetLabs/SoftHSM/tags/3.0.0-pre1
- SGX Crypto API Toolkit 1.2.2.0
    - https://github.com/intel/crypto-api-toolkit
- SGX SDK and PSW 2.3.1
    - https://github.com/intel/linux-sgx
    - https://github.com/intel/linux-sgx-driver
- SGX OpenSSL 2.4
    - https://github.com/intel/intel-sgx-ssl
- OpenSSL 1.1.0i
    - https://www.openssl.org/source/old/1.1.0/

Setup

A number of pre-requisites need to be downloaded and installed prior to building source. It may be necessary to install the following items.

The following list of dependencies may be required for installation and can be installed by running, from a terminal, "sudo apt-get install [library name]":

1. sudo apt-get update

1. sudo apt-get install git cmake curl alien pkg-config uuid-dev libxml2-dev libsystemd-dev build-essential ocaml automake autoconf libtool wget python libssl-dev libcurl4-openssl-dev protobuf-compiler libprotobuf-dev debhelper

Build

The following build order is required to ensure all dependencies are installed appropriately for building the main SGX Token library:

1. SGX SDK & PSW
2. SGX Crypto API Toolkit with SGX OpenSSL
3. SoftHSM & SGX Token

SGX SDK, PSW, Driver and dependencies

Verify Intel® Management Engine Driver is installed:

1. sudo modprobe mei

Install iclsClient (Intel Capability Licensing Service):

2. Register and download from: https://registrationcenter.intel.com/en/products
   a. Search for Intel(R) Capability Licensing Service for Linux

(from location where .rpm was downloaded)

3. sudo alien --scripts iclsClient-1.45.449.12-1.x86_64.rpm
4. sudo dpkg -i iclsclient_1.45.449.12-2_amd64.deb

Install Dynamic Application Loader (DAL) Java Host Interface (JHI):

5. git clone https://github.com/intel/dynamic-application-loader-host-interface.git

(cd to dynamic-application-loader-host-interface directory)

6. cmake .
7. make
8. sudo make install
9. sudo systemctl enable jhi

Clone SGX SDK:

10. git clone https://github.com/intel/linux-sgx.git
11. cd to SGX SDK directory
12. ./download_prebuilt.sh
13. make [DEBUG=1]

Build SDK Installer:

14. cd to SGX SDK directory
15. make sdk_install_pkg [DEBUG=1]

You can find the generated Intel(R) SGX SDK installer sgx_linux_x64_sdk_${version}.bin located under linux/installer/bin/, where ${version} refers to the version number.

Build PSW Installer:

16. cd to SGX SDK directory
17. On Ubuntu 16.04, to keep debug symbols in the Intel(R) SGX PSW installer, before building the Intel(R) SGX PSW, export the environment variable to ensure the debug symbols not stripped:
    a. [optional] export DEB_BUILD_OPTIONS="nostrip"
    b. make deb_pkg [DEBUG=1]

The installers for the PSW will be as:

- *libsgx-urts_${version}-${revision}amd64.deb* and
- *libsgx-enclave-common_${version}-${revision}_amd64.deb*

These should be located under linux/installer/deb:

- *${version}* refers to the version number
- *${revision}* refers to the revision number of the package.

Install SGX SDK:

18. cd linux/installer/bin
19. ./sgx_linux_x64_sdk${version}.bin
    a. When asked "Do you want to install in current directory", answer NO
    b. When asked "Please input the directory which you want to install in:": enter /opt/intel/

Verify SGX SDK is installed properly:

20. cd to 'linux-sgx'/SampleCode/LocalAttestation
21. source /opt/intel/sgxsdk/environment
22. make SGX_MODE=SIM
23. ./app

This will build the simulation version of SGX. Later, the Debug or Release version can be built and tested.

Build and install SGX Driver:

24. git clone https://github.com/intel/linux-sgx-driver.git
25. cd to linux-sgx-driver directory

Check if matching kernel headers are installed:

26. dpkg-query -s linux-headers-$(uname -r)

Install matching headers:

27. sudo apt-get install linux-headers-$(uname -r)

Build driver:

28. make

Install driver:

29. sudo mkdir -p "/lib/modules/"`uname -r`"/kernel/drivers/intel/sgx"
30. sudo cp isgx.ko "/lib/modules/"`uname -r`"/kernel/drivers/intel/sgx"
31. sudo sh -c "cat /etc/modules | grep -Fxq isgx || echo isgx >> /etc/modules"

32. sudo /sbin/depmod
33. sudo /sbin/modprobe isgx

Install PSW:

34. sudo dpkg -i ./libsgx-urts_2.4.100.48613-xenial_amd64.deb ./libsgx-enclave-common_2.4.100.48613-xenial1_amd64.deb

At this point, the SGX AESM service should be installed and running, which can be verified by running:

- sudo service aesmd status

If there are issues, or it hasn't been started, stop and start the service either with:

1. sudo service aesmd stop
2. sudo service aesmd start

or

1. sudo service aesmd restart

Note: You may need to configure a proxy if to allow the service to provision.

SGX Crypto API Toolkit

35. Download and unzip the package from https://github.com/intel/intel-sgx-ssl/releases/tag/v2.4
36. Download OpenSSL version 1.1.0i
37. Copy openssl-1.1.0i.tar.gz to intel-sgx-ssl-2.4/openssl_source directory
38. cd to intel-sgx-ssl-2.4/Linux directory
39. make all [DEBUG=1] test
40. sudo make install

41. Clone Crypto API Toolkit version 1.2.2.0 from https://github.com/intel/crypto-api-toolkit
42. cd to crypto-api-toolkit
43. ./build.sh [Debug/Release]
44. Copy all contents in /bin/ to /usr/local/lib/softhsm/
    a. Folder contents should include files 1) libp11sgx.so 2) p11Enclave.signed.so and 3) p11SampleApp

At this stage, all SGX SDK, PSW, and Driver components should be installed for building the Crypto API Toolkit. Additionally, the Crypto API Toolkit should be built with its supported version of OpenSSL. This set satisfies the foundational requirements for building and using the SGX Token for SoftHSM.

Build SGX Token

SoftHSM is required to utilize the SGX Token, thus it must be obtained and its build files modified to compile the SGX Token into it.

Download and configure SoftHSM:

45. clone tag 3.0.0-pre1 from https://gitlab.nlnetlabs.nl/NLnetLabs/SoftHSM/
46. cd to SoftHSM

In a separate directory:

47. git clone https://github.com/intel/softhsm-tokens/sgx
48. Copy the SGX directory that was cloned and rename to "hard_token_sgx"
49. Copy hard_token_sgx to /SoftHSM/src/lib/
    a. Note that another similar directory is contained here called "soft_token"
    b. This is the default SW token that will be replaced with the SGX Token
50. Update the following file types in their respective directories to point to hard_token_sgx instead of soft_token. All files should have contained a location pointing to soft_token, and after being updated, point to hard_token_sgx instead. It is suggested to also rename the soft_token directory to another name to ensure that no accidental inclusion of its source files will be made.
    a. CMakeList
        i. src/lib
        ii. src/lib/token_mgr
        iii. src/lib/object_store
        iv. src/lib/slot_mgr
    b. Makefile.am
        i. src/lib
        ii. src/lib/hard_token
        iii. src/lib/object_store
        iv. src/lib/data_mgr
        v. src/lib/handle_mgr
        vi. src/lib/slot_mgr
        vii. src/lib/token_mgr
        viii. src/lib/session_mgr

At this point, SoftHSM has been configured to build with the SGX Token instead of the default SW token (soft_token). You may follow the build steps provided with SoftHSM in its readme, or simply run the following commands:

51. cd to SoftHSM
52. sh ./autogen.sh
53. ./configure
54. make
55. sudo make install
    a. This will install the SoftHSM libraries to /usr/local/lib/softhsm and include the files:
        i. libp11sgx.so
        ii. libsofthsm2.a
        iii. libsofthsm2.la
        iv. libsofthsm2.so
        v. p11Enclave.signed.so
    b. Additionally, a utility application will be built and output to SoftHSM/src/bin/util as softhsm2-util

To verify installation is successful:

56. cd to SoftHSM/src/bin/util

57. run ./softhsm2-util --init-token --slot 0 --label sgxtoken --so-pin 12345 --pin 1234
    a. This should initialize the token with the provided SO and User PINs

Congratulations, you have successfully build SoftHSM with the SGX Token!

You may utilize existing PKCS#11 applications to call into SoftHSM's PKCS#11 interface with crypto operations backed by the SGX Token, who uses SGX to generate and manage crypto keys and operations. By analyzing the SGX Token or SGX Crypto API Toolkit source code, you may identify the necessary interfaces, structures, and operations needed to build a similar token, or ensure proper handling of sensitive materials.

Additionally, the Key Handler class in the SGX Token, where PKCS#11 validation is performed per each specific crypto operation, as well as in the Crypto API Toolkit, may be updated to allow or disallow specific policies on key management, or add additional functionality within the secure environment.

Use pre-built SGX SDK, PSW, Driver and dependencies

Alternatively, pre-built versions of the SDK, PSW, and Driver may be used for installation instead of building them manually.

Install SGX Driver, SDK, and PSW:

58. Download SGX 2.3.1 from the Installers page:
    a. https://download.01.org/intel-sgx/linux-2.3.1/ubuntu16.04/
59. sudo chmod 777 sgx_linux_x64_driver_4d69b9c.bin
60. sudo chmod 777 sgx_linux_x64_sdk_2.3.101.46683.bin
61. ./sgx_linux_x64_driver_4d69b9c.bin
62. ./sgx_linux_x64_sdk_2.3.101.46683.bin
    b. When asked "Do you want to install in current directory", answer No
    c. When asked "Please input the directory which you want to install in:": enter /opt/intel/
63. sudo dpkg –i ./libsgx-enclave-common_2.3.101.46683-1_amd64.deb

If you want to build the SoftHSM test applications for exercising the SGX HW Token, follow the steps listed in **Error! Reference source not found.**.

If you want to build the source manually, follow the steps listed in **Error! Reference source not found.**.

Build p11Test Application

Optionally, a PKCS#11 test application may be built, which tests the SGX HW Token library, and requires the building of the SoftHSM PKCS#11 library, which serves as the PKCS#11 interface to the SGX HW Token library for testing the crypto backend.

Note that the tests provided in p11Test are specific to each implementation and thus assumptions about a token are pre-defined. These assumptions would need to be updated for each token, including the SGX Token, else test failures are expected.

1. cd to /src/lib/test
2. copy /src/bin/util/libsofthsm2.so, libp11sgx.so, and p11Enclave.signed.so to this directory
3. make p11test_DEPENDENCIES=p11test_LDADD=CPPFLAGS=-DP11M=\\\"./libsofthsm2.so\\\" p11test
   a. Review README for exact build arguments
4. make p11Test

Run tests:

5. Run tests with summary and then output for each test
   a. ./p11test
6. Run tests with output immediately after each test
   a. ./p11test direct

This will run a series of tests using the SoftHSM management library, using the SGX Token for SoftHSM with SoftHSM.