TECHNOLOGY GUIDE

Intel Corporation

Intel® Software Guard Extensions (Intel® SGX) – Key Management on the 3rd Generation Intel® Xeon® Scalable Processor

Authors

Kapil Sood

Veronika Karpenko

Jon Strang

David Lu

Seosamh O'Riordain

Darragh Coen

1 Introduction

Intel® Software Guard Extensions (Intel® SGX) is a set of instructions that helps increase the security of application code and data. Developers can partition security sensitive code and data into an "SGX Enclave", which is executed in a CPU protected region.

NGINX is a prevalent open-source transport layer security (TLS) web service that is widely deployed in multiple applications. NGINX relies on the private key to perform authentication and key agreement operations for TLS handshake. As such, the security of these private keys is of utmost importance for customers.

Key Management Reference Application (KMRA) is proof-of-concept software created to demonstrate the integration of Intel® Software Guard Extensions (Intel® SGX) asymmetric key capability with a hardware security model (HSM) on a centralized key server. The goal of this document is to outline the steps to set up an NGINX workload to access the private key in an Intel® SGX enclave on the 3rd Generation Intel® Xeon® Scalable processor by using the Public-Key Cryptography Standard (PKCS) #11 interface and OpenSSL. This paper describes the use of Intel® SGX to help secure the NGINX private key on a general-purpose Intel® Xeon® platform. The unique security properties of Intel SGX, combined with the broad use of NGINX in web and cloud services, show an example of how Intel SGX can be used to more securely provision private keys into the enclave on a cloud platform, and then use those keys while they are protected inside the enclave.

Another significant contribution of this white paper is to illustrate the ease-of-use and ease-of-deployment for a complete end-to-end Intel SGX system. Customers can use this white paper and associated collateral as a reference to replicate Intel SGX deployments and customizations within their unique environments.

Intel SGX enclaves can be used for applications ranging from private key protection, security credentials management, and providing security services. In addition, industry security standards like ETSI NFV SEC have defined and published security requirements for a hardware-mediated execution enclaves like Intel SGX for purposes of network functions virtualization (NFV), 5G, and edge security. Intel SGX addresses multi-administration security requirements for these emerging cloud networking systems, which are increasingly software-defined and highly distributed. The intended audience for this document is Technical Architects, Solution Architects, Product Managers, Software Engineers, Engineering Managers, System and Security Architects, and Application Engineers.

This document can act as a reference to implement Intel SGX for Intel platforms to help secure workload keys, credential management, and Intel SGX deployment in a data center. It is highly recommended that readers refer to widely available documents for Intel SGX design and enabling.

This document is part of the Network Transformation Experience Kit, which is available at https://networkbuilders.intel.com/network-technologies/network-transformation-exp-kits.

Table of Contents

Figures

Tables

Document Revision History

REVISION	DATE	DESCRIPTION
001	February 2021	Initial release.
002	April 2021	Revised the document for public release to Intel® Network Builders.
003	August 2021	Added support for Docker containers.

1.1 Terminology

Table 1. Terminology

ABBREVIATION	DESCRIPTION
Ansible	Ansible is a radically simple IT automation engine that automates cloud provisioning, configuration management, application deployment, intra-service orchestration.
Ansible-playbook	Playbooks are the files where Ansible code is written.
BIOS	Basic Input/Output System is a set of computer instructions in firmware that controls input and output operations.
CA	Certificate authority
CDN	Content Delivery Network is a system of distributed servers (network). It delivers pages and other web content to a user, based on the geographic locations of the user, the origin of the webpage, and the content delivery server.
DCAP	Data Center Attestation Primitives. Intel® Software Guard Extensions Data Center Attestation Primitives (Intel® SGX DCAP) provides SGX attestation support targeted for data centers, cloud services providers, and enterprises.
ECDSA	Elliptic curve digital signature algorithm
Enclave	Ring 3 application software running inside the Intel® SGX protections
FLC	Flexible launch control
FW, UEFI FW	Firmware, Unified Extensible Firmware Interface FW
HSM	Hardware security module
Intel® SGX-TEM	Secure Guard Extensions – Trusted Environment Mode
kmaas	Key management as a service
KMRA	Key Management Reference Application
mTLS	Mutual transport layer security
OS	Operating system
PCCS	Provisioning Certificate Caching Service
PKCS	Public-Key Cryptography Standard
PKCS#11	Public-Key Cryptography Standard. The PKCS#11 standard defines a platform-independent API to cryptographic tokens, such as hardware security modules (HSM) and smart cards.
PSW	Platform software
SGX	Intel® Software Guard Extensions (Intel® SGX) is a set of instructions that increase the security of application code and data, giving them more protection from disclosure or modification.
SSL	Secure Sockets Layer is a networking protocol designed for securing connections between web clients and web servers over an insecure network, such as the internet.
TLS	Transport Layer Security

1.2 Reference Documentation

Table 2. Reference Documents

REFERENCE	SOURCE
Intel® SGX Programming Reference and SDK for Linux	https://software.intel.com/content/www/us/en/develop/articles/intel-sdm.html#combined https://download.01.org/intel-sgx/latest/linux-latest/docs/ https://github.com/intel/linux-sgx
PKCS#11 Specification	http://docs.oasis-open.org/pkcs11/pkcs11-base/v2.40/pkcs11-base-v2.40.html
ETSI NFV Security Standards (SEC001, SEC012, SEC013, others)	http://www.etsi.org/technologies-clusters/technologies/nfv

REFERENCE	SOURCE
Intel® SGX Resources	https://software.intel.com/content/www/us/en/develop/topics/software-guard-extensions.html https://software.intel.com/content/www/us/en/develop/download/intel-software-guard-extensions-intel-sgx-developer-guide.html https://software.intel.com/content/www/us/en/develop/articles/intel-sdm.html
Intel® SGX Crypto-Toolkit Open Source	https://github.com/intel/crypto-api-toolkit
Intel® SGX ECDSA Attestation DCAP and APIs	https://download.01.org/intel-sgx/latest/dcap-latest/linux/docs/ https://github.com/cloud-security-research/sgx-ra-tls https://github.com/intel/SGXDataCenterAttestationPrimitives
Intel® SGX Flexible Launch Control (FLC)	https://github.com/intel/linux-sgx/blob/master/psw/ae/ref_le/ref_le.md https://software.intel.com/content/www/us/en/develop/blogs/an-update-on-3rd-party-attestation.html
Intel® SGX Open Source Projects	https://github.com/intel/intel-sgx-ssl https://github.com/intel/sgx-ra-sample
Intel® SGX Security Analysis	https://www.intel.com/content/www/us/en/security-center/default.html https://software.intel.com/security-software-guidance/
Intel® Software Guard Extensions (Intel® SGX) - NGINX Private Key on 3rd Generation Intel® Xeon® Scalable Processor User Guide	https://networkbuilders.intel.com/solutionslibrary/intel-sgx-securing-an-nginx-private-key-3rd-generation-intel-xeon-scalable-processor-user-guide

2 Overview

Intel® SGX is a set of instructions on Intel CPUs that helps increase the security of application code and data. Developers can partition security sensitive code and data into an Intel SGX enclave, which is executed in a CPU protected region. The developer creates these enclaves on untrusted platforms and uses Intel CPU-based attestation to ensure the integrity of their Intel SGX enclave setup. After the enclave is verified, the remote attester can provision secrets securely into the enclave. In KMRA, we provision the NGINX private key. The use of Intel SGX enclave is reserved for applications and cannot be used by an OS or BIOS driver/module.

3 Deployment

3.1 Intel® SGX Deployment

3.1.1 Intel® SGX-Trusted Environment Mode (TEM) Security Model and Usages

Intel SGX removes the privileged software (OS, VMM, SMM, devices) and unprivileged software (Ring 3 applications, VMs, containers) from the trust boundary of the code running inside the enclave and enhances the security of sensitive application code and data. Intel SGX enclave trusts the Intel CPU for execution and memory protections. Intel SGX encrypts memory to help protect against memory bus snooping and cold boot attacks for enclave code and data in host DRAM.

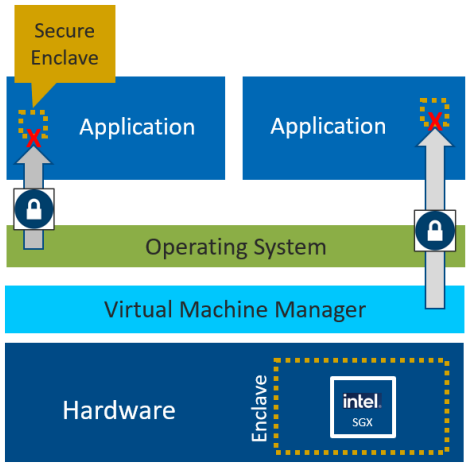

Figure 1: Intel® SGX Security Model

Intel SGX relies on the system unified extensible firmware interface (UEFI) BIOS and OS for initial provisioning, resource allocation, and management. However, after an Intel SGX enclave starts execution, it runs on a cryptographically isolated environment separate from the OS and UEFI BIOS, which can thereafter only launch a DoS attack on the Intel SGX enclave. Readers are highly encouraged to refer to Intel SGX collateral, including specifications, security analysis, SDK, drivers, and open source applications, from Intel, academia, and customers.

Intel SGX allows confidential computing services to be delivered on Intel servers. Confidential computing is an emerging industry paradigm where applications can be run in a cryptographically protected environment. It can allow any application (whole or partial) to run inside an enclave, and by limiting the trusted compute base, Intel SGX puts application developers in control of their own application security. However, it is recommended that developers keep the Intel SGX code base small, test for software side channel resistance, and follow other secure software development guidelines included in the Intel® SGX Developer Guide.

Intel SGX enclaves can be used for applications ranging from private key protection, security credentials management, and security service provider. The KMRA described in this specification focuses on customer private key protection. However, it is expected that the broader Intel SGX enabling, provisioning, attestation, and application service delivery components of this KMRA system are applicable to multiple customer use cases.

Intel SGX includes Intel architecture instruction set extension instructions, which can be used by supervisor mode (like operating system software) for enclave page cache (EPC) page management, and to create, initialize, and manage enclaves. The user mode instructions allow an application to enter/exit an enclave, for attestation and key management. A detailed and updated set of Intel® SGX ISA can be found in the Intel SGX Programing Reference document.

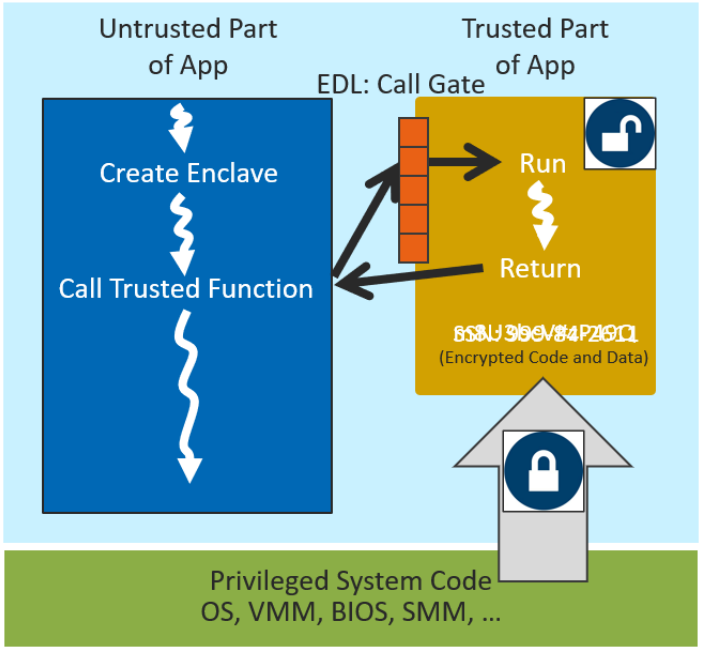

Figure 2: Application Refactoring for an Enclave

Intel® SGX TEM implementation on 3rd Generation Intel® Xeon® Scalable processor provides memory confidentiality.

3.1.2 Flexible Launch Control (FLC)

Flexible launch control (FLC) allows the platform owner (not Intel) to control which enclaves are launched on that platform. The launch enclave (LE) is an Intel SGX architectural enclave that is responsible for generating a launch token for an application enclave, which is used with the SGX instructions. The default signer for the LE is Intel, but FLC allows the platform owner to specify the LE authorizing party as part of the BIOS Intel SGX setup.

3.1.3 Data Center Attestation Primitives (DCAP)

Attestation is the process of cryptographically demonstrating that the software is instantiated as expected on the platform. Intel SGX allows infrastructure, data center, or platform owners (for example, CoSPs, CSPs, OEMs, Enterprises) to deploy their own ECDSA attestation infrastructure for Intel SGX enclave attestation.

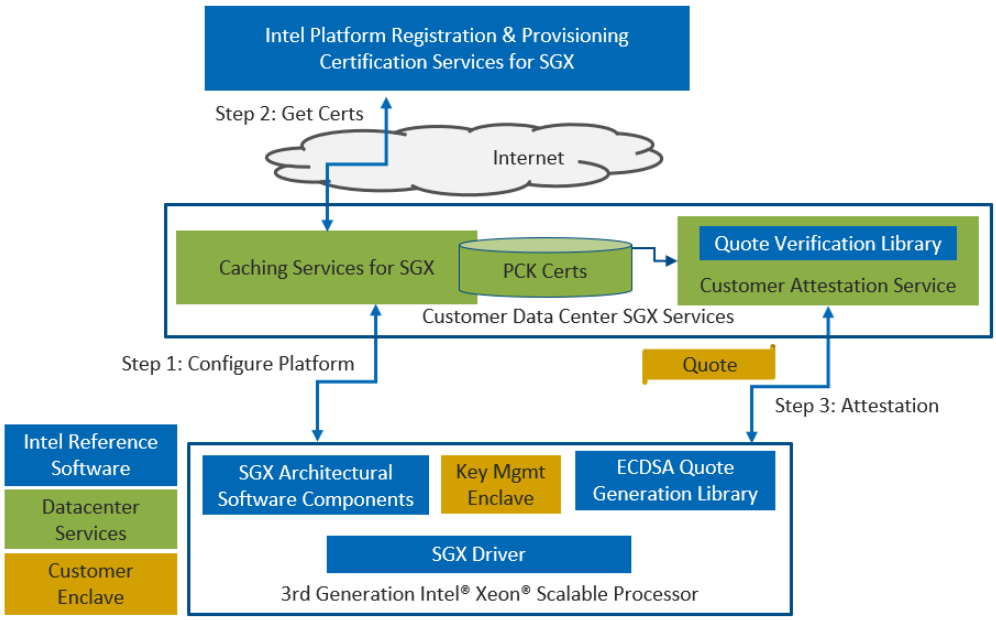

Figure 3: Intel® SGX Data Center Attestation Deployment Architecture

Figure 3 shows the flow for enabling a data center customer to deploy their remote attestation service to provide attestation services to their tenants.

1. Intel SGX PCK Cert ID Retrieval Tools run on the Intel SGX capable platform.
2. The Caching Service retrieves the PCK certificates from the Intel SGX provisioning certification service (PCS) for Intel SGX-enabled platforms. These retrieved data elements are signed by Intel, and, upon download and verification, these data elements are stored in the caching service. Refer to Intel® SGX ECDSA Attestation DCAP and APIs document for detailed steps on registration and API key procurement.
3. Intel SGX quote generation library (QGL) generates ECDSA-based remote attestation quotes using Intel SGX architectural software. Intel SGX QGL exposes a set of APIs that the application can use to generate the quote.

Intel provides quote verification library (QVL) reference code, which offers APIs to enable ECDSA quote verification. The ECDSA Quote format is used both by the QGL on the platform and by the QVL in the data center customer attestation service.

3.1.4 Enclave Remote Attestation

Remote attestation is the foundational security step upon which the Intel SGX enclave runs the intended code and data. The process of enclave build on the target server is performed by the untrusted software, and therefore, only after a successful verification of the instantiated enclave should any secrets be provisioned into the enclave. The enclave library should NOT contain any security sensitive material (for example, keys) during initialization. These materials should be securely provisioned directly into the enclave or generated by the enclave after it is successfully created and initialized.

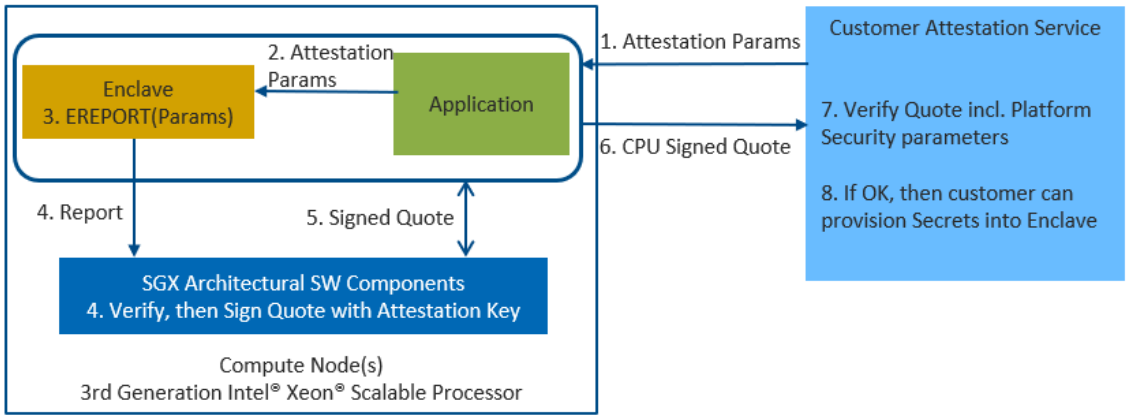

Figure 4: Intel® SGX Remote Attestation

The following steps describe a process for application enclave attestation. Refer to Intel SGX reference documentation for details.

1. The remote customer attestation server may initiate an attestation protocol, including sending a random challenge to the untrusted application running Key Management enclave.
2. The untrusted application sends these parameters to the Key Management enclave. The enclave may add any additional parameters (for example, an enclave-generated RSA public key) and create a list of parameters to be signed by CPU.
3. Enclave calls the Intel SGX API instructions via Intel SGX SDK to generate an Intel SGX signed report.
4. Enclave sends this report to the Intel SGX architectural software, which verifies this report. If verified, the CPU signs it with the customer's ECDSA attestation key.
5. The CPU sends the signed report (quote) to the application.
6. Application sends this signed quote to the remote customer attestation server along with any additional parameters (for example, root keys).
7. The remote attestation server verifies this quote and verifies any additional security parameters (for example, root keys) necessary for authenticating the endpoint enclave.
8. After verification, the key server can provision secrets into the application enclave on the compute platform.

3.2 Customer Private Key Security - Use Case

Securing customer keys has become an industry and customer priority. This task is especially important for third-party software running on a cloud infrastructure, or on hosted environments as in edge and 5G deployments, and in commercial infrastructure systems. All types of workloads that deal with keys have a genuine business reason to ensure that these keys are never exposed in clear or stolen by adversaries. The monetary and reputation costs of loss of private key can be high, and private key replacements in large deployments is not a trivial activity. Based on customer-prioritized usage for key security, we focus on using NGINX as the primary example application and illustrate an Intel SGX easy-to-deploy reference solution on 3rd Generation Intel® Xeon® Scalable processor, which enhances private key security for NGINX workloads.

3.2.1 NGINX KMRA Flow with Intel® SGX

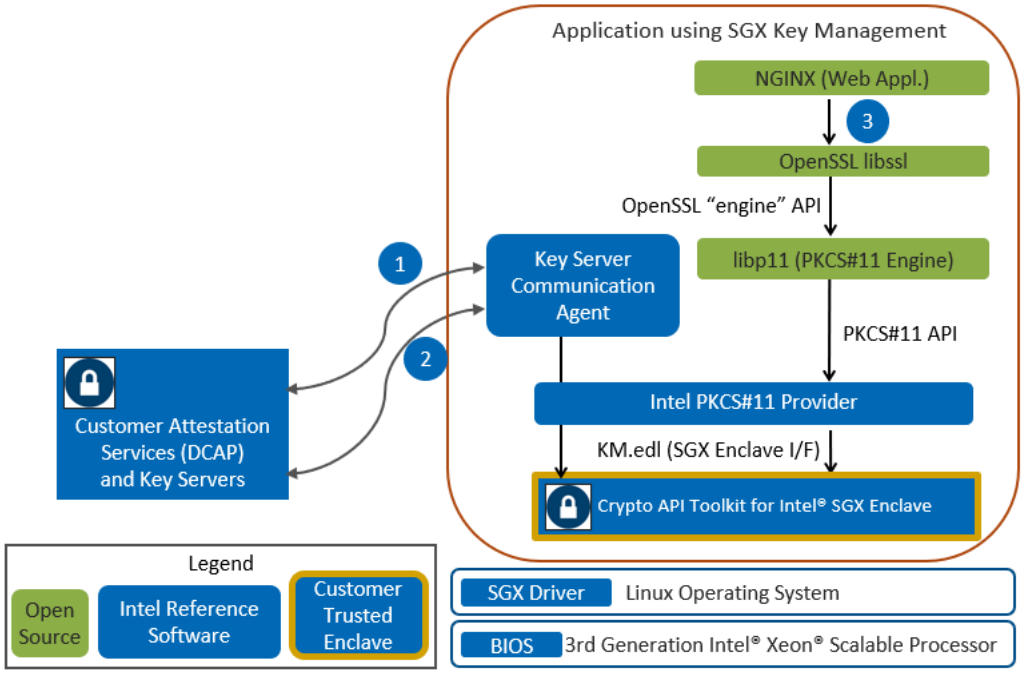

Figure 5. NGINX KMRA Flow with Intel SGX

As shown in Figure 5, the key management flow of NGINX with Intel SGX has three main steps.

3.2.1.1 Step 1 - SGX Enclave Launch with DCAP Attestation

A compute node has Intel SGX enabled and Crypto API Toolkit for Intel® Software Guard Extensions (Crypto API Toolkit for Intel® SGX) installed. An Intel SGX quote is generated inside the Crypto API Toolkit for Intel SGX enclave for DCAP attestation. The Intel SGX quote is attested on the key server side.

3.2.1.2 Step 2 - Customer Key Delivery into Enclave

The wrapped private key is provisioned by the key server into the Crypto API Toolkit for Intel® SGX enclave. The private key is never exposed in the clear outside of the key server or the SGX enclave.

3.2.1.3 Step 3 - NGINX Application Uses the Key Protected Inside the Enclave

The NGINX workload can more securely access the private key through the PKCS#11 interface using the libp11 engine configured with OpenSSL. NGINX can establish a TLS connection using the private key from the Crypto API Toolkit for Intel® SGX enclave. Since the private key is never exposed in the clear outside of the enclave, NGINX uses PKCS#11 APIs to perform private key operations inside the enclave.

3.2.2 KMRA Software Design and Architecture

KMRA is proof-of-concept software created to demonstrate the integration of Intel SGX asymmetric key capability running in the NGINX application, with a backend attestation and key server.

KMRA service node is a centralized key server that provisions wrapped keys to the compute node. A Flask REST API runs on the service node with Connexion verifying all incoming requests. The REST API uses a Cython wrapper for C to interact with SoftHSMv2, to wrap and extract keys through the PKCS#11 interface.

To validate the client, mutual TLS is implemented on the service node where each client certificate is verified. The client certificate must be generated by a mutual CA. Subject OUN extracted from the certificate maps to permissions and keys in the configuration file. KMRA compute node is a client running on an Intel SGX-enabled platform. The client sends a request to the service node containing an Intel® SGX quote, a public key from Crypto API Toolkit for Intel SGX, and a unique ID to identify the key pair to extract. The client constructs the requests by using json-c and sends requests by using libcurl. The Intel SGX quote of a client is validated by the quote verification library (QVL) on the service node before the encrypted keys are released by the SoftHSM key server and sent back to the compute node for provisioning into the enclave. When the client receives a response containing wrapped keys, the server certificate is validated, and the keys are imported into Crypto API Toolkit for Intel SGX. For NGINX to access the secured private key provisioned by the service node, a libp11 engine is configured with OpenSSL. The libp11 engine is an interface for NGINX to access keys secured by Crypto API Toolkit for Intel SGX.

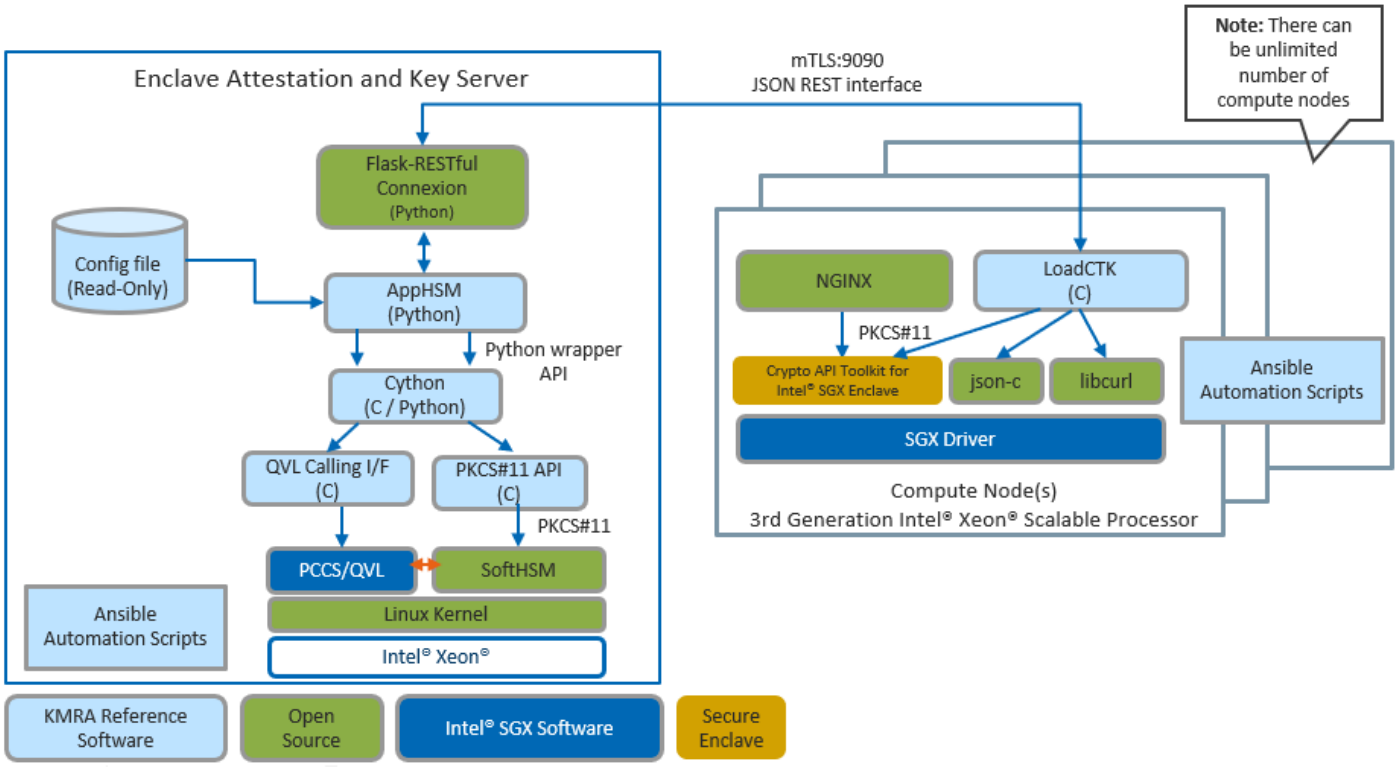

Figure 6: KMRA NGINX/Intel® SGX Key Management Software Design

3.2.3 Intel® SGX PKCS#11 Provider Crypto - Toolkit

The Crypto API Toolkit for Intel® SGX is based on SoftHSMv2, an open-source cryptographic library that provides a PKCS#11 interface to the application. Most PKCS#11 enabled applications (for example. HSMs, OpenSSL) can use this toolkit. This toolkit is a functional subset implementation of the PKCS#11 standard[1].

[1] http://docs.oasis-open.org/pkcs11/pkcs11-base/v2.40/pkcs11-base-v2.40.html

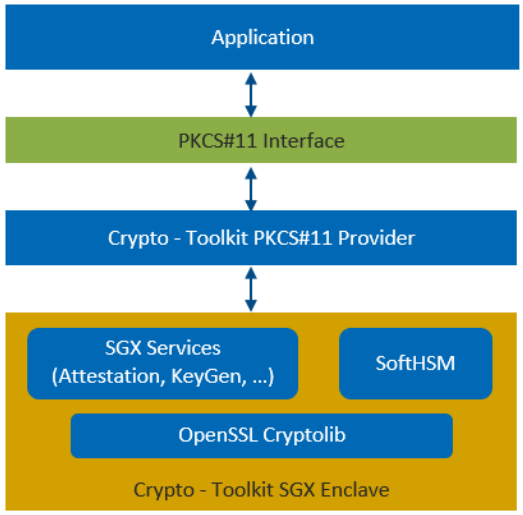

Figure 7: Crypto API Toolkit for Intel® SGX Software Architecture

The Crypto API Toolkit for Intel® SGX may not support the exhaustive list of all PKCS#11 APIs and mechanisms defined in the standard. Rather, it supports algorithms and parameters that are sufficient for common cloud applications. This toolkit can be used as a baseline reference enclave. It is expected to be extended by customers with additional capabilities and security validation, as required.

The toolkit embeds Intel's SGX SSL crypto library, and in addition supports Intel SGX enclave capabilities for purposes of local attestation to the quoting enclave and for remote attestation.

3.2.4 Enclave Attestation using KMRA Key Server

Intel® SGX provides an enclave remote attestation mechanism. This mechanism allows a remote provider to verify the following:

1. The enclave is running on a real Intel processor inside an Intel SGX enclave.
2. The platform is running at the latest security level (also referred to as the *trusted computing base* [TCB] version).
3. Identity of the enclave.
4. The enclave has not been compromised.

After verifying these parameters and any additional security checks, the remote attestation and key server can then provision secrets more securely into the enclave.

The KMRA REST API server allows remote access. It is an interface that allows the client to request keys from the key server to be more securely sent to the Intel SGX-enabled node running ctk_loakdey. The REST API requires the client to send a quote generated in the Intel SGX enclave, a public key from the enclave and a client certificate. The REST API uses mutual TLS to authenticate a client. The certificates for AppHSM and ctk_loadkey are generated by the same certificate authority (CA). The CA certificate is used to authenticate the client. The client certificate also has a Subject OUN field with a specific client ID. Key permissions for this client ID are outlined in the apphsm.conf configuration file. The mutual TLS certificates are not related to the wrapped keys provisioned to the client.

The following list provides a detailed key flow sequence:

- Client generates RSA key pair (rsa_pub, rsa_priv) out-of-band on service node (key server).
- Client generates customer public/private key pair (cust_pub/cust_priv) as a session object on the compute node. An attestation quote sgx_quote_t is generated using Crypto API Toolkit for Intel SGX and attests the hash of cust_pub and the enclave. Client sends REST API request containing a (cust_pub) to AppHSM to trigger the Intel SGX quote verification library.
- The quote provides proof to the service node that the client's enclave is running with Intel SGX protections on a trusted Intel SGX-enabled platform, with a valid TCB. If quote is correct, the hash of cust_pub is verified with sgx_quote_t, cust_pub is imported into the HSM as a session object, and a symmetric wrapping key (aes_swk) is generated as a session object.
- AppHSM creates wrapped_priv_key by wrapping RSA private key (rsa_priv) with SWK (aes_swk) using CKM_AES_KEY_WRAP_PAD. AppHSM also creates wrapped_swk by wrapping SWK (aes_swk) with the imported customer public key (cust_pub) using RSA OAEP. Wrapped keys wrapped_priv_key and wrapped_swk are returned to the client on the compute node.

- Client unwraps keys into Crypto API Toolkit for Intel SGX using cust_priv secured in the enclave. Cust_pub and cust_priv are destroyed after unwrapping when the session ends. OpenSSL on the client node is configured to retrieve protected key using Libp11 engine and provide it to NGINX workload.

The REST API has two URIs implemented.

1. To get the KMRA server version

```
GET /sys/version
```

   - HTTPS 201 – OK
   - HTTPS 401 – Unauthorized

   Example command:

```
curl https://localhost:5000/sys/version -X GET -k --cacert ./ca.crt -key
./ctk_loadkey.key --cert ./ctk_loadkey.crt
```

   Example response:

```
{ "version": "0.01", "api_version": "0.01_API"}
```

2. To verify quote and hash, import key, and return wrapped keys

```
POST /sgx/keys/export
```

   - HTTPS 201 – OK
   - HTTPS 401 – Unauthorized
   - HTTPS 400 – Bad Request
   - HTTPS 500 - Internal Server Error

   Example command:

```
curl  https://localhost:5000/sgx/keys/export -X POST -k
--cacert ./ca.crt --key ./ctk_loadkey.key --cert ./ctk_loadkey.crt
-H "Content-Type: application/json" -d
'{'HsmObject': {'h_unique_id': 'xxxx'}, 'SgxObject': {'RsaPublicKey':
{'ExponentLen': 0, 'Exponent': 'xxxx', 'ModulusLen': 0,
'Modulus': 'xxxx'}, 'SgxQuote': 'xxxx'}}'
```

   Example response:

```
{ "wrappedSWK": "L3NneC9rZXlzL2V4cG9ydCB3cmFwcGVkU1dL","wrappedKey":
"L3NneC9rZXlzL2V4cG9ydCB3cmFwcGVkS2V5"}
```

3.3 Automated Intel® SGX Deployment

KMRA provides ansible environment setup scripts for installing Intel® SGX components and KMRA dependencies. The Intel® Software Guard Extensions (Intel® SGX) - NGINX Private Key on 3rd Generation Intel® Xeon® Scalable Processor User Guide contains step-by-step instructions for running the Ansible scripts.

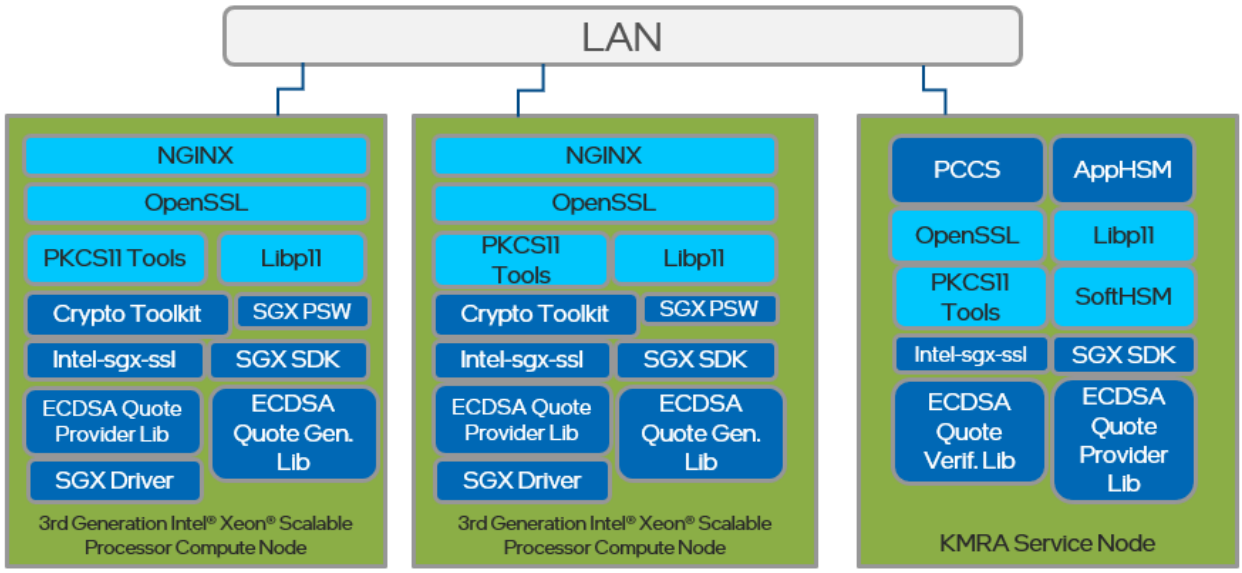

Figure 8: Ansible Deployment of KMRA Components Across Multiple 3rd Generation Intel® Xeon® Scalable Processor Compute Nodes and a Service Node

3.4 KMRA Supports Containers

KMRA now supports containers. A container is a standard unit of software that packages code and all its dependencies, so the application runs quickly and reliably from one computing environment to another. A Docker container image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries, and settings.

Follow these steps to set up your KMRA container.

1. Install the AESMD container from https://github.com/intel/intel-device-plugins-for-kubernetes.
2. Install the SGX drivers from the KMRA Ansible scripts.
3. Follow the steps in the container READ ME file.

The Dockerfiles are set up in three steps. The PCCS container is set up in nine steps. The Ctk_loadkey containers are set up in as few as three steps. All of the steps are well documented in the Intel® Software Guard Extensions (Intel® SGX) - NGINX Private Key on 3rd Generation Intel® Xeon® Scalable Processor User Guide and in the READ ME file on GitLab.

4 Summary

Intel® SGX provides a more secure environment for application owners to run their applications' sensitive code and data inside an Intel SGX enclave, enhancing protection of their enclave code and data from privileged software and applications. This guide demonstrates an end-to-end reference architecture for using Intel SGX to protect private key for NGINX application. This document focuses on making Intel SGX easy to use and deploy, illustrating the overall system architecture and software design components that enabled the usage. This example can be extended to additional Intel SGX usages, applications, and deployments. It is recommended that readers refer to this Intel SGX reference architecture and associated collateral to assist in development and follow their security best practices in the deployment of their Intel SGX systems.

0821/DN/WIT/PDF 635272-003US