User Guide

Service Mesh – mTLS Key Management in Istio and Envoy for Intel® Xeon® Scalable Processors

Authors

Xintong Chen

Ramesh Masavarapu

1 Introduction

A service mesh is a configurable, low-latency infrastructure layer designed to handle a high volume of network-based inter-process communication (IPC) among application infrastructure services using application programming interfaces (APIs). The service mesh ensures that the communication layer between microservices is fast, reliable, and secure. Some of the key features include service discovery, security, traceability, and observability.

The service mesh can be implemented by multiple open-source software solutions. Istio and Envoy are the two popular open-source projects that implement service mesh architecture. Istio is the control plane and Envoy is the data plane.

Security within a service mesh deployment is safer using Intel® Software Guard Extensions (Intel® SGX) as the keys are secured in a hardware enclave and not in the clear. This solution improves the key management during mTLS connections.

The purpose of this document is to showcase the usage of the private keys of mTLS communication between service mesh workloads by Intel SGX. The private keys are stored and used inside the Intel® SGX enclave(s) and will never be stored in clear anywhere in the system. Users can use the private key in the enclave by Key handle provided by Intel® SGX to communicate with other applications. The importance of using Intel® SGX solution cannot be understated for any Cloud Service Provider as security is one of the key factors of concern for tenants who deploy their workloads. Customers who deploy their solutions on the cloud should work with the provider in deploying their workload on an Intel® Xeon® SGX enabled SKU.

The solution is available only on 3rd Gen Intel® Xeon® Scalable processors and later.

This document is part of the Network Transformation Experience Kits.

Table of Contents

Tables

Document Revision History

Revision	Date	Description
001	December 2022	Initial release.
002	January 2023	Revised for public distribution on Intel Network Builders.

1.1 Terminology

Table 1. Terminology

Abbreviation	Description
API	Application Programming Interface
Intel® SGX	Intel® Software Guard Extensions
Intel® SGX-TEM	Intel® Software Guard Extensions – Trusted Environment Mode
mTLS	Mutual Transport Layer Security

1.2 Reference Documentation

Table 2. Reference Documents

Reference	Source
Intel® Software Guard Extensions (Intel® SGX) – Key Management Reference Application (KMRA) on Intel® Xeon® Processors Technology Guide	https://networkbuilders.intel.com/solutionslibrary/intel-sgx-kmra-on-intel-xeon-processors-technology-guide
Intel® SGX Resources	https://www.intel.com/content/www/us/en/developer/tools/software-guard-extensions/overview.html
Service Mesh – Istio and Envoy Optimizations for Intel® Xeon® Scalable Processors Solution Brief	https://networkbuilders.intel.com/solutionslibrary/service-mesh-istio-envoy-optimizations-intel-xeon-sp-solution-brief
Service Mesh - Crypto Accelerations in Istio and Envoy with Intel® Xeon® Scalable Processors User Guide	https://networkbuilders.intel.com/solutionslibrary/service-mesh-crypto-accelerations-istio-envoy-intel-xeon-sp-user-guide
Service Mesh - Envoy Regular Expression Matching Acceleration with Hyperscan User Guide	https://networkbuilders.intel.com/solutionslibrary/service-envoy-regular-expression-matching-acceleration-hyperscan-user-guide
Service Mesh - TCP/IP eBPF Bypass in Istio and Envoy with Intel® Xeon® Scalable Processors User Guide	https://networkbuilders.intel.com/solutionslibrary/service-mesh-tcp-ip-bypass-istio-envoy-intel-xeon-sp-user-guide

2 Overview

Private keys used by Envoy in the data plane to perform the TLS handshaking are not encrypted nor protected. They exist in plaintext in both the memory and filesystems. By using Intel® SGX for enhanced security for both Istio and Envoy, the private keys can now exist in the Intel SGX enclave throughout their life cycle. This means that they are generated, used, and destroyed in the Intel SGX enclave so that attackers would not have a chance to get access to it.

3 Prerequisites

The following are the prerequisites for building and running the Intel SGX based Istio and Envoy applications:

- A Kubernetes cluster with one or more nodes with Intel® SGX supported hardware
- The Intel® SGX device plugin for Kubernetes
- The Intel® SGX AESM daemon
- Linux kernel version 5.11 or later on the host with the in-tree SGX driver
- Git, or a similar tool, to obtain the source code
- Docker, or a similar tool, to build container images

4 Installation

This section provides steps on how to install Intel® SGX based Envoy and Istio applications.

1. Install Istio.

```
# set the KUBECONFIG based on your configuration
$ export KUBECONFIG="$HOME/.kube/config"

# Install Istio
# These flags can be customized according to your needs
istioctl install -f ./sgx-mTLS/istio-config.yaml -y \
    --set values.global.proxy.logLevel=debug \
    --set values.global.logging.level=all:debug \
    --set values.global.proxy.sgx.enabled=true \
    --set values.global.proxy.sgx.certExtensionValidationEnabled=true \
    --set values.gateways.sgx.enabled=true \
    --set values.gateways.sgx.certExtensionValidationEnabled=true
```

2. Verify that the pods are running.

By default, Istio will be installed in the istio-system namespace.

```
# Ensure that the pod is running state
$ kubectl get pods -n istio-system
NAME                                    READY   STATUS    RESTARTS   AGE
istiod-5fcf786bf4-6wz24                 1/1     Running   2          3d23h
```

5 Deployment of Sample App

1. Create test namespace.

```
# create test namespace
$ kubectl create ns foo
```

2. Deploy the sample applications.
   - Create `httpbin` deployment.

```
kubectl apply -f <(istioctl kube-inject -f
https://raw.githubusercontent.com/istio/istio/master/samples/httpbin/httpbin.yaml) -n foo
```

   - Create sleep deployment

```
kubectl apply -f <(istioctl kube-inject -f
https://raw.githubusercontent.com/istio/istio/master/samples/sleep/sleep.yamlP) -n foo
```

   - A successful deployment appears as follows:

```
$ kubectl get pods -n foo
NAME                              READY  STATUS        RESTARTS      AGE
httpbin-7c78f947d7-tgh92      2/2    Running       4                 3d23h
sleep-b7cd7d89-rrrwf          2/2    Running       6                 3d23h
```

   - Test Pod resources

```
$ kubectl exec "$(kubectl get pod -l app=sleep -n foo -o jsonpath={.items..metadata.name})" -c
sleep -n foo -- curl -s http://httpbin.foo:8000/headers | grep X-Forwarded-Client-Cert
    "X-Forwarded-Client-Cert":
"By=spiffe://cluster.local/ns/foo/sa/httpbin;Hash=cd5d0504234e80c701c4fe01ef49f3fe048a63d1cdd5b
9ffe3dd67ae3d93396b;Subject=\"CN=spiffe://cluster.local/ns/foo/sa/sleep\";URI=spiffe://cluster.
local/ns/foo/sa/sleep"
```

The above `httpbin` and `sleep` applications enable Intel® SGX and store the private keys inside an SGX enclave, complete the TLS handshake and establish a connection with each other and communicate as expected.

6 Sample Use Cases

You can use this Intel SGX security enhanced Istio Service Mesh for any application scenarios. This Intel® SGX security enhanced Istio can also be used for example in the following use cases:

- Any east-west traffic between the sidecars and gateway for any applications
- Envoy as an ingress proxy server

7 Summary

As the communication between different microservices happens through an Envoy side car proxy using mTLS, securing the keys becomes very important. By using Intel® SGX solution, customers can safely enhance the security of the keys within a service mesh environment. The importance of using Intel® SGX solution cannot be understated for any Cloud Service Provider as security is one of the key factors of concern for tenants who deploy their workloads. Customers who deploy their solutions on the cloud should work with the provider in deploying their workload on an Intel® Xeon® SGX enabled SKU.

Performance varies by use, configuration and other factors. Learn more at www.Intel.com/PerformanceIndex.

Performance results are based on testing as of dates shown in configurations and may not reflect all publicly available updates. See backup for configuration details. No product or component can be absolutely secure.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

Intel technologies may require enabled hardware, software or service activation.

Intel does not control or audit third-party data. You should consult other sources to evaluate accuracy.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

© Intel Corporation. Intel, the Intel logo, and other Intel marks are trademarks of Intel Corporation or its subsidiaries. Other names and brands may be claimed as the property of others.

0123/DN/WIT/PDF 761906-002US