Technology Guide

# Intel Zero Trust Network Access Reference Architecture 23.03 Features Update

**Authors**

Liyan Wang

Ju Huang

Yun Lan

Heqing Zhu

Xiang Wang

## 1 Introduction

Intel® Zero Trust Network Access Reference Architecture (Intel® ZTNA Reference Architecture) delivers an optimized zero trust architecture with enhanced security and accelerated performance using Intel technologies. It's a software reference design to realize ZTNA. Intel® ZTNA Reference Architecture now includes security enforcement with Intel confidential computing technologies (Intel® SGX and Intel® TDX) and demonstrates improved network performance by accelerating encrypted network tunnels. This work provides a solid reference on how to build a more secure and performant ZTNA system with the 3rd and 4th Gen Intel® Xeon® Scalable processors that fulfills enterprise network security requirements.

The first software release of Intel® ZTNA Reference Architecture was published in June 2022 with core features including user authentication, service authorization, and WireGuard tunnels. These features were introduced in the whitepaper for the 22.06 release. This paper focuses on the new features in the 23.03 release which includes Intel TDX protection, full disk encryption, IPSec tunnels and automatic certificate management, built on platforms based on Intel Xeon processors.

This document is part of the Network Transformation Experience Kits.

## Table of Contents

## Figures

## Tables

## Document Revision History

| Revision | Date | Description |
|---|---|---|
| 001 | May. 2023 | Initial release. |

## 1.1 Terminology

Table 1. Terminology

| Abbreviation | Description |
|---|---|
| ACL | Access Control List |
| AES | Advanced Encryption Standard |
| AES-NI | Intel® Advanced Encryption Standard New Instructions |
| CISA | Cybersecurity & Infrastructure Security Agency |
| CSP | Cloud Service Providers |
| Enclave | Ring 3 application software running inside the Intel SGX protections |
| LUKS | Linux Unified Key Setup |
| NIST | National Institute of Standards and Technology |
| ZTNA | Zero Trust Network Access |
| IPSec | Internet Protocol Security |
| RBAC | Role Based Access Control |
| SGW | Secure Gateway |
| TD | Trust Domain |
| AH | Authentication Header |
| ESP | Encapsulating Security Payload |
| IKE | Internet Key Exchange protocols |

## 1.2 Reference Documentation

Table 2. Reference Documents

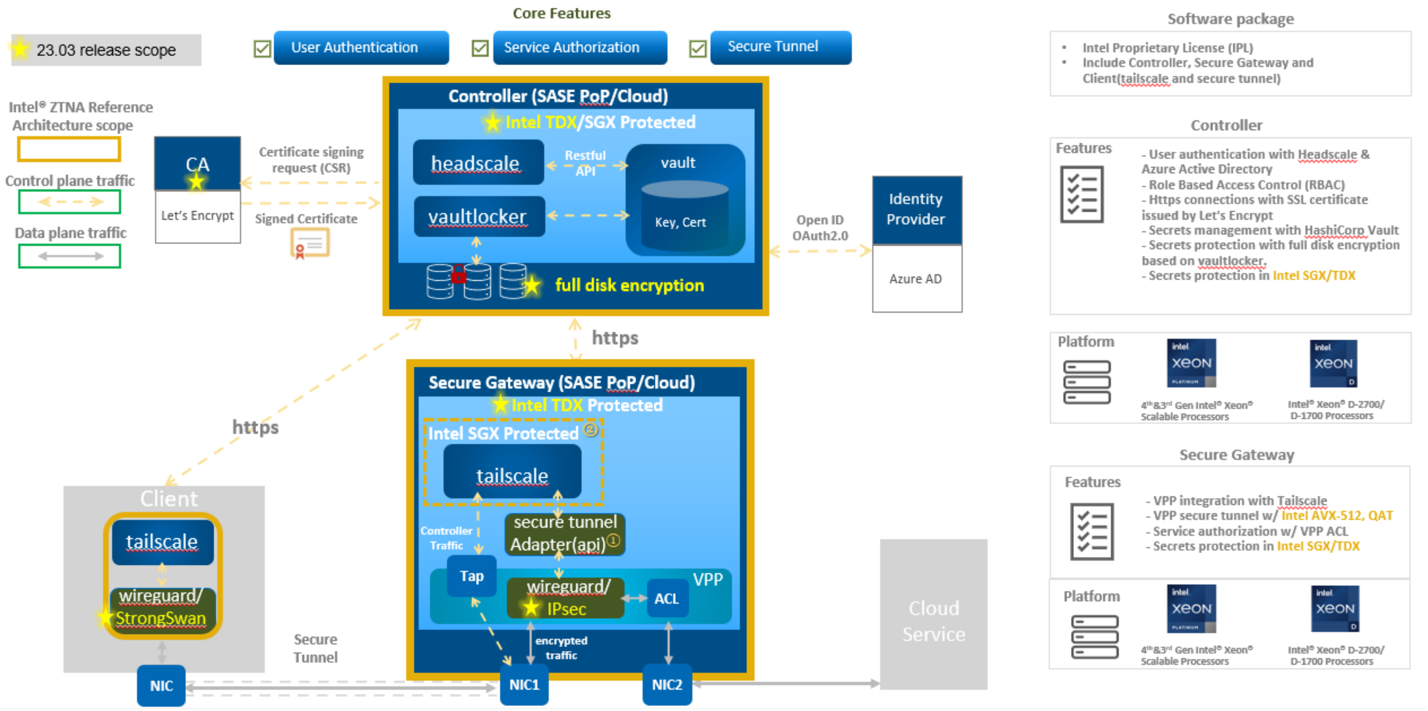
| Reference | Source |
|---|---|
| Intel® Xeon® Scalable Platform Built for Most Sensitive Workloads | https://www.intc.com/news-events/press-releases/detail/1423/intel-xeon-scalable-platform-built-for-most-sensitive |
| Crypto Acceleration: Enabling a Path to the Future of Computing | https://newsroom.intel.com/articles/crypto-acceleration-enabling-path-future-computing |
| Golang | https://go.dev/ |
| HashiCorp Vault | https://www.Vaultproject.io/<br>https://medium.com/hashicorp-engineering/hashicorp-Vault-performance-benchmark-13d0ea7b703f |
| Occlum | https://github.com/occlum/occlum |
| LUKS2 On-Disk Format Specification | https://gitlab.com/cryptsetup/LUKS2-docs/blob/main/luks2_doc_wip.pdf |
| Intel SGX Programming Reference and SDK for Linux | https://software.intel.com/content/www/us/en/develop/articles/intel-sdm.html#combined<br>https://download.01.org/intel-sgx/latest/linux-latest/docs/<br>https://github.com/intel/linux-sgx |
| National Institute of Standards and Technology FIPS Publication 197, Advanced Encryption Standard (AES) | https://csrc.nist.gov/publications/detail/fips/197/final |
| 3rd Generation Intel® Xeon® Scalable Processor - Achieving 1 Tbps IPSec with Intel® Advanced Vector Extensions 512 (Intel® AVX-512) Technology Guide | https://networkbuilders.intel.com/solutionslibrary/3rd-generation-intel-xeon-scalable-processor-achieving-1-tbps-ipsec-with-intel-advanced-vector-extensions-512-technology-guide |
| Create Intel SGX VM in the Azure portal | https://docs.microsoft.com/en-us/azure/confidential-computing/quick-create-portal |
| 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® Software Guard Extensions (Intel® SGX) - Key Management Reference Application (KMRA) on Intel® Xeon® Scalable Processors User Guide | https://networkbuilders.intel.com/solutionslibrary/intel-sgx-kmra-on-intel-xeon-processors-user-guide |
| Intel® Software Guard Extensions (Intel® SGX) – Securing Private Keys in an Encrypted Enclave for Your Service Mesh Demo | https://networkbuilders.intel.com/intel-software-guard-extensions-intel-sgx-securing-private-keys-in-an-encrypted-enclave-for-your-service-mesh-demo |
| Intel® Trust Domain Extensions (Intel® TDX) | https://www.intel.com/content/www/us/en/developer/articles/technical/intel-trust-domain-extensions.html |

| Reference | Source |
|---|---|
| Intel® AVX-512 and Intel® QAT - Accelerate WireGuard Processing with Intel® Xeon® D-2700 Processor Technology Guide | https://networkbuilders.intel.com/solutionslibrary/intel-avx-512-and-intel-qat-accelerate-wireguard-processing-with-intel-xeon-d-2700-processor-technology-guide |
| Zero Trust - Zero Trust Reference Architecture Technology Guide | https://networkbuilders.intel.com/solutionslibrary/zero-trust-zero-trust-reference-architecture-technology-guide |
| Zero Trust - Harden Data Security with Intel Xeon Processors | https://networkbuilders.intel.com/solutionslibrary/zero-trust-harden-data-security-with-intel-xeon-processors |

# 2 Intel® ZTNA Reference Architecture

## 2.1 Overview

Intel® ZTNA Reference Architecture consists of 4 components, including the client, the controller, the secure gateway, and service as shown in Figure 1Figure 1. The yellow boxes are the current scope of Intel® ZTNA Reference Architecture, and the features marked with a yellow star are newly added in the 23.03 release, including Intel® TDX protection, full disk encryption, IPSec tunnel support and automatic certificate management.


Core Features
23.03 release scope
User Authentication
Service Authorization
Secure Tunnel
Intel® ZTNA Reference Architecture scope
Control plane traffic
Data plane traffic
CA
Let's Encrypt
Certificate signing request (CSR)
Signed Certificate
Controller (SASE PoP/Cloud)
Intel TDX/SGX Protected
headscale
Restful API
vault
Key, Cert
vaultlocker
full disk encryption
Open ID OAuth2.0
Identity Provider
Azure AD
https
https
Secure Gateway (SASE PoP/Cloud)
Intel TDX Protected
Intel SGX Protected ②
tailscale
Controller Traffic
secure tunnel Adapter(api)①
Tap
wireguard/ IPsec
ACL
VPP
encrypted traffic
Client
tailscale
wireguard/ StrongSwan
Secure Tunnel
NIC
NIC1
NIC2
Cloud Service
Software package
Intel Proprietary License (IPL)
Include Controller, Secure Gateway and Client(tailscale and secure tunnel)
Controller
Features
- User authentication with Headscale & Azure Active Directory
- Role Based Access Control (RBAC)
- Https connections with SSL certificate issued by Let's Encrypt
- Secrets management with HashiCorp Vault
- Secrets protection with full disk encryption based on vaultlocker.
- Secrets protection in Intel SGX/TDX
Platform
4th&3rd Gen Intel® Xeon® Scalable Processors
Intel® Xeon® D-2700/ D-1700 Processors
Secure Gateway
Features
- VPP integration with Tailscale
- VPP secure tunnel w/ Intel AVX-512, QAT
- Service authorization w/ VPP ACL
- Secrets protection in Intel SGX/TDX
Platform
4th&3rd Gen Intel® Xeon® Scalable Processors
Intel® Xeon® D-2700/ D-1700 Processors

① Secure tunnel adapter (API): include WireGuard binapi, StrongSwan and govici

② Intel SGX Protected: Intel SGX only protect tailscale in Secure Gateway, Intel TDX protect all Secure Gateway

Figure 1. Intel® ZTNA Reference Architecture Overview

A brief introduction to the entire architecture is as follows:

- In the controller, Headscale and HashiCorp Vault are integrated to authenticate users. Secrets management and third-party authentication systems are supported by Vault. Full disk encryption is implemented with Vaultlocker, which is based on LUKS, to protect data at rest. All processes and sensitive data in the controller can be protected by Intel confidential computing technologies such as Intel SGX and Intel TDX.
- In the secure gateway (SGW), we use VPP to accelerate the data forwarding of the secure tunnel and to implement role-based access control (RBAC). Tailscale is integrated as an agent of the SGW to interact with the controller. All processes and sensitive data in SGW can be protected by Intel SGX and Intel TDX.
- In the client, Tailscale works as the agent to connect with the controller and to establish a secure tunnel with the SGW.

3rd and 4th Gen Intel Xeon Scalable processors and Intel Xeon D-2700/1700 processors are target platforms for Intel® ZTNA Reference Architecture as they deliver advanced security features and high performance for zero trust systems. We have applied Intel architecture specific optimizations to the Intel® ZTNA Reference Architecture on these platforms. For example, WireGuard and IPSec tunnels are accelerated by Intel® AVX-512. Also, on 4th Gen Intel Xeon Scalable processors, the controller and SGW can be run inside a TD VM, so that data in memory can be encrypted and on 3rd Gen Intel Xeon Scalable processors, all credentials (private key, etc.) of the controller and the SGW core processes are protected by Intel SGX. Intel® ZTNA Reference Architecture is delivered as a software package under the Intel Proprietary License (IPL).

## 2.2 Workflow

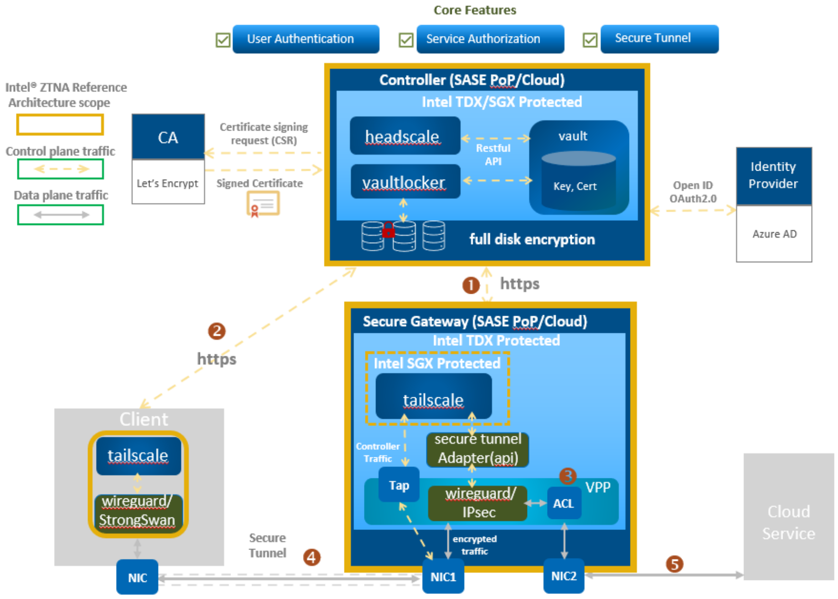


Figure 2. Brief Workflow of Intel® ZTNA Reference Architecture

This section summarizes the user authentication and service authorization workflow of the Intel® ZTNA Reference Architecture system as shown in Figure 2Figure 2. For a detailed description, please refer to the whitepaper for the 22.06 release.

1. The SGW sends a registration message to the controller with its own public key, machine information, and public IP address. Then it will be asked to input a username and password for Azure Authentication. If authenticated, the controller will save the SGW as a node and store the related information to Vault.
2. The client connects to the zero-trust network in the same way. The client sends a registration message to the controller with its own public key, endpoint information, and so on. Once the client is authenticated by Azure, the controller saves the client as a node and then returns the SGW information and routing rules to the client.
3. At the same time, the SGW gets updated information from including the new client endpoint and new ACL policies. This information will be configured into VPP to achieve role-based access control using the VPP ACL module.
4. A secure tunnel is established between client and SGW. The supported tunnel protocols are WireGuard and IPSec.
5. Finally, the client can access the service via the established secure tunnel.

# 3 Intel® TDX and Crypto Technologies

## 3.1 Intel® Trust Domain Extensions (Intel® TDX)

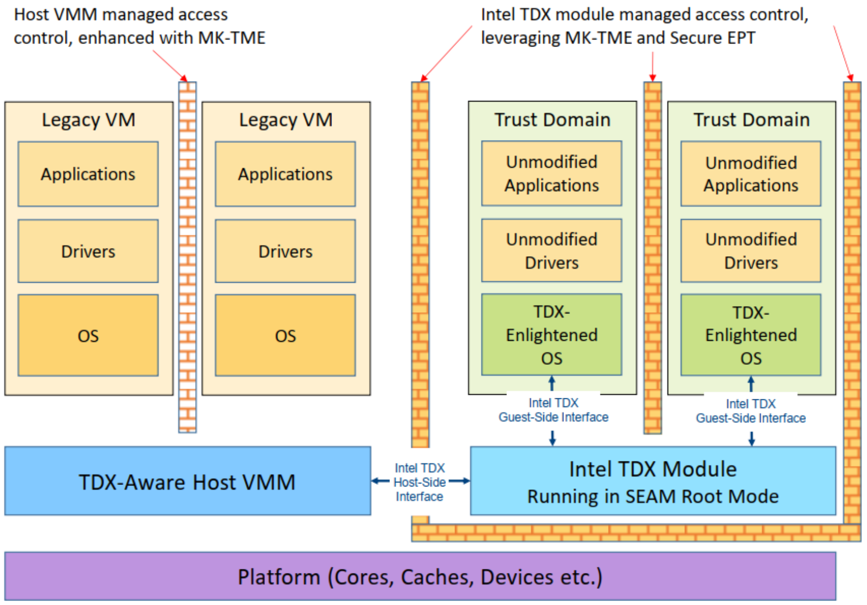

Host VMM managed access control, enhanced with MK-TME
Intel TDX module managed access control, leveraging MK-TME and Secure EPT
Legacy VM
Applications
Drivers
OS
Legacy VM
Applications
Drivers
OS
Trust Domain
Unmodified Applications
Unmodified Drivers
TDX-Enlightened OS
Intel TDX Guest-Side Interface
Trust Domain
Unmodified Applications
Unmodified Drivers
TDX-Enlightened OS
Intel TDX Guest-Side Interface
TDX-Aware Host VMM
Intel TDX Host-Side Interface
Intel TDX Module Running in SEAM Root Mode
Platform (Cores, Caches, Devices etc.)

Figure 3. Intel Trust Domain Extensions (Intel TDX)

More and more enterprises are migrating services to the cloud but cloud tenants require more protection to the data security (data in use). In typical installations the cloud service providers (CSP) have full access to what is being processed on the platform. Intel® Trust Domain Extensions (Intel® TDX) is an architectural technology to deploy hardware-isolated Virtual Machines (VMs) called Trust Domains (TDs). It is designed to isolate TD VMs, hypervisor, and other non-TD software on the host platform. This protects TDs from a broad range of software and hardware exploits and attacks which enhances the cloud-service provider's (CSP) ability to provide managed cloud services without exposing tenant data to adversaries or third parties. Each TD VM runs in an isolated and encrypted memory and has a unique cryptographic key for memory data encryption. With Intel® TDX, customers can be more confident that their data will be private and protected while being processed in the public cloud.

### 3.2 Intel Crypto Technologies

Crypto operations are at the core of secure data in transit and at rest. The communication between the client and the SGW rely on cryptographic tunneling protocols such as SSL, IPSec, and WireGuard. Secure tunnels protect data in transit. Sophisticated solutions even establish an addtional tunnel between the SGW and the remote services and then, by stitching together the two separate tunnels, enforce stronger security. Secure tunnels are compute intensive. Intel technologies such as Intel® QuickAssist Technology (Intel® QAT), Advanced Encryption Standard-New Instructions (AES-NI) and Advanced Vector Extensions-512 (AVX-512) instruction-set found in Intel processors can significantly accelerate the performance of encryption.

For data at rest, full disk encryption is a useful solution. This usually involves a data encryption key and a key encryption key based on Linux Unified Key Setup (LUKS) on-disk format specification. AES algorithms are widely used for encryption at rest and included as part of compliances by NIST who recommends both AES-256 and AES-128 for long-term storage use. Full disk encryption is mostly based on symmetric encryption algorithms as defined in LUKS. Please refer to Zero Trust - Harden Data Security with Intel Xeon Processors for more details.

## 4 Open-Source Software Updated in 23.03

There is a strong open-source software ecosystem for zero trust systems, including key and secrets management solutions, Linux user space utility tools, Linux kernel device-mapper crypto modules, etc. Intel contributes to these open-source communities to a large extent. HashiCorp Vault, Tailscale, Headscale and Vector Packet Processing (VPP) have been introduced in the whitepaper for 22.06 release. In this section, we only focus on the newly adopted open-source components in Intel® ZTNA Reference Architecture 23.03 release.

### 4.1 Vaultlocker

Vaultlocker is an open-source software framework to automate full disk encryption. As a broker between Cryptsetup and Vault, Vaultlocker maintains encryption keys in Vault and provides wrapper interfaces to supply the key to Cryptsetup for full disk encryption. It sets roles and permissions for key storage using the key-value secrets engine of Vault. Furthermore, Vaultlocker uses Cryptsetup to encrypt specified disk partition. Once the partition setup is completed, all data writes to the disk partition are encrypted automatically and all data reads from the disk partition are decrypted automatically.

### 4.2 strongSwan

strongSwan is an open-source IPSec-based VPN solution. It can provide encryption and authentication between the server and client to establish a secure connection. strongSwan is essentially a key based application that uses Internet Key Exchange protocols (IKEv1 and IKEv2) to establish a security association (SA) between two endpoints. IKE provides strong authentication between peers and obtains a unique encrypted session key. In addition to authentication and keys, IKE also provides methods for exchanging configuration information (such as virtual network addresses) and negotiating IPSec SAs (also known as CHILD_SAs). IPsec SAs define which network transfers are secure and how they are encrypted and authenticated. With strongSwan integration with VPP, the key negotiation and tunnel establishment can be completed automatically. We use the high-performance data plane of VPP to accelerate IPSec tunnels.

### 4.3 ACME Tool

acme.sh is an open-source ACME client software that provides an automatic method to manage digital certificates. acme.sh can automatically request TLS certificates from the third-party certificate authority (CA) and renew them before expiration. The supported CA list includes ZeroSSL, Let's Encrypt, and BuyPass, among others. We chose Let's Encrypt as our public CA because of its stability and free 90-day certificates. When Let's Encrypt issues a certificate, they first validate the domain name's ownership using "challenges," as defined by the ACME standard. acme.sh supports multiple challenge modes including HTTP-01 and DNS-01. We've integrated acme.sh to manage certificates for the controller so that it can establish secure HTTPs connections with the client and the SGW.

## 5 Features Updated in 23.03

### 5.1 Protection using Intel® TDX

Intel TDX is designed to provide the ability to isolate tenant VMs from (VMM)/hypervisor and any other non-TD software running on the platform. It enforces cryptographic isolation among the security domains, thereby mitigating cross-domain attacks. It also enhances the CSP's ability to manage cloud services without exposing tenant data to adversaries. There is a common requirement that the credentials (private keys, tokens, etc) in ZTNA solutions are kept confidential. In a VM-based deployment, Intel TDX improves data security for ZTNA solutions by encrypting these credentials in use.

The Intel® ZTNA Reference Architecture controller provides user authentication and maintains sensitive data at rest (in storage) with full disk encryption. The Intel® ZTNA Reference Architecture SGW supports WireGuard/IPsec tunnels which use cryptographic keys for handshakes. All of this processing should be run in TD VMs to reduce leaks and attacks.

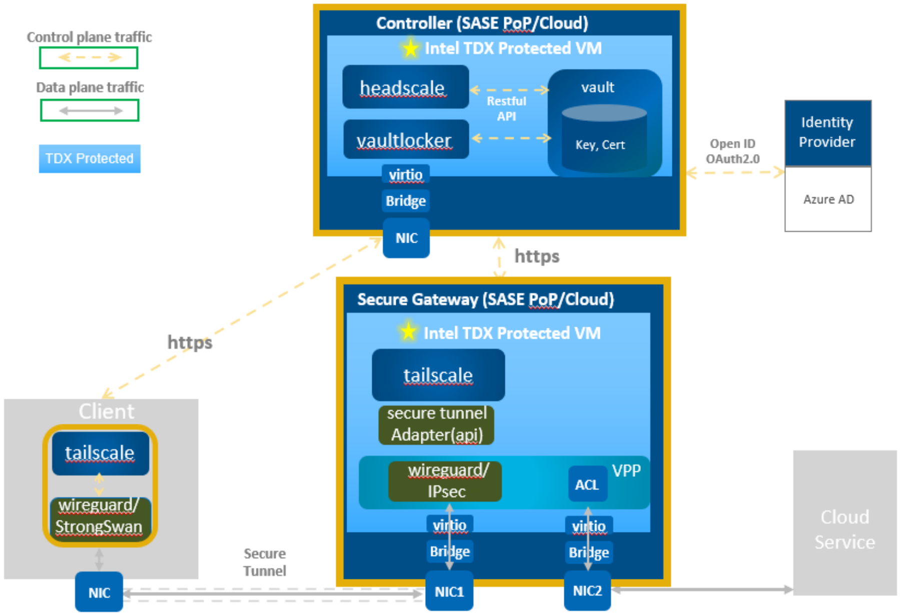


Figure 4. Intel® ZTNA Reference Architecture protected by TD VM

Running a TD VM first requires enabling Intel TDX through the BIOS. Then, Intel TDX-related software must be installed on the host to build and start the guest VM. Finally, Intel® ZTNA Reference Architecture workloads can be deployed in TD VMs just like normal VMs. To enable Intel TDX in BIOS and build the Intel TDX software stack, please refer to the whitepaper Linux*Stacks for Intel® Trust Domain Extension 1.0.

The Intel tdx tool also provides an easy way to build the entire Intel TDX stack, including the Intel TDX host kernel, Intel TDX Qemu, Intel TDX Libvirt, TDVF, the Intel TDX guest kernel, grub2, and shim. TD VM runs with v5.15 kernel. As Intel® ZTNA Reference Architecture uses the Ubuntu 22.04 operating system, please navigate to the build/ubuntu-22.04 folder to build, install and start a TD VM.

The Intel® ZTNA Reference Architecture SGW uses VPP as its data plane. Currently user space virtio-pmd cannot work in TD VMs since network hardware devices are not a part of the trust domain by default. We instead create AF-XDP interfaces in VPP as a workaround.

## 5.2 Full Disk Encryption

Protecting data at rest is often a mandatory modern security consideration. The Cybersecurity & Infrastructure Security Agency (CISA) defines data security as a pillar of their Zero Trust Maturity Model to secure data at rest. Encryption is the most common way of protecting data at rest. Disk encryption, as a dominant solution to encrypt data at rest, delivers comprehensive protection and is fully transparent to user applications. Intel® ZTNA Reference Architecture delivers a full disk encryption solution based on Intel confidential computing technology as well as a new reference design to realize zero trust security standards. The system consists of Vault, Vaultlocker and LUKS software. Vaultlocker works as the broker between Cryptsetup and Vault, maintains encryption keys in Vault and provides wrapper interfaces to supply the key to Cryptsetup for full disk encryption. Figure 5 shows the workflow of full disk encryption.

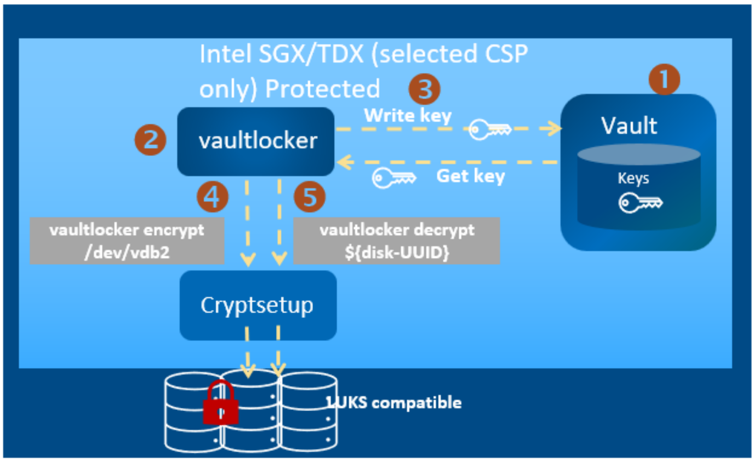

Intel SGX/TDX (selected CSP only) Protected
1
Vault
Keys
2
vaultlocker
3
Write key
Get key
4
5
vaultlocker encrypt /dev/vdb2
vaultlocker decrypt ${disk-UUID}
Cryptsetup
LUKS compatible

Figure 5. Workflow of Disk Encryption

1. Vault is configured by enabling the key-value store secrets engine and adding an authentication method which uses the client application role, secret id, and access policy (create, read, update, delete, and list).
2. Vaultlocker is configured with the host address of Vault and the client role, secret id and secrets engine name configured in step 1.
3. Vaultlocker generates a disk encryption key and stores it in Vault.
4. Vaultlocker retrieves the key from Vault with a preconfigured role and permissions and triggers Cryptsetup to encrypt a specified disk partition.
5. A file system is created for the encrypted partition and the encrypted disk is mounted.

Once the setup is done, all full disk encryption operations are fully transparent to applications. Every disk write and read from applications to the encrypted partition will trigger the dm-crypt kernel module which in turn calls the kernel crypto module to conduct encryption for each write request and decryption for each read request. For more detailed information about disk encryption please refer to Zero Trust - Harden Data Security with Intel Xeon Processors.

## 5.3 Secure Tunnel - IPSec

IPSec consists of a group of protocols including AH, ESP, and IKE. It provides secure encrypted communication between a pair of hosts, between a pair of secure gateways (SGW), or between a host and a secure gateway. It supports network-level peer authentication, data origin authentication, data integrity, data confidentiality, and replay protection.

IPSec tunnels offer two modes of operation: route-based IPSec and policy-based IPsec. Policy-based IPSec, as the name implies, determines whether packets are tunneled up or down by policy matching. Route-based IPSec, which creates an IPSec logical interface, introduces packets by way of routing to the IPSec tunnel. Policy-based mode is a more traditional approach with greater versatility, while route-based mode has higher performance and is more suitable for a variety of complex networking environments.

Intel® ZTNA Reference Architecture supports IPSec tunnels as an alternative to WireGuard with the expectation of meeting more customer requirements. It uses VPP as the data plane and the VPP sswan plugin to communicate with strongSwan. The architecture supports only policy-based IPSec due to limitations of current versions of the VPP sswan plugin. Yet, to achieve unified configuration management for multiple types of network tunnels and ACL combinations with single-arm routing, route-based IPSec would be a better choice for the Intel® ZTNA Reference Architecture and it will be supported in the future. On the SGW side, we introduce strongSwan to implement IPSec IKE negotiation, and the result is configured to VPP IPSec and adapt it to the ACL configuration in VPP. On the client side, we introduce strongSwan to implement the IPSec IKE association with SGW. The workflow of IPSec tunnel establishment is shown inFigure 6.

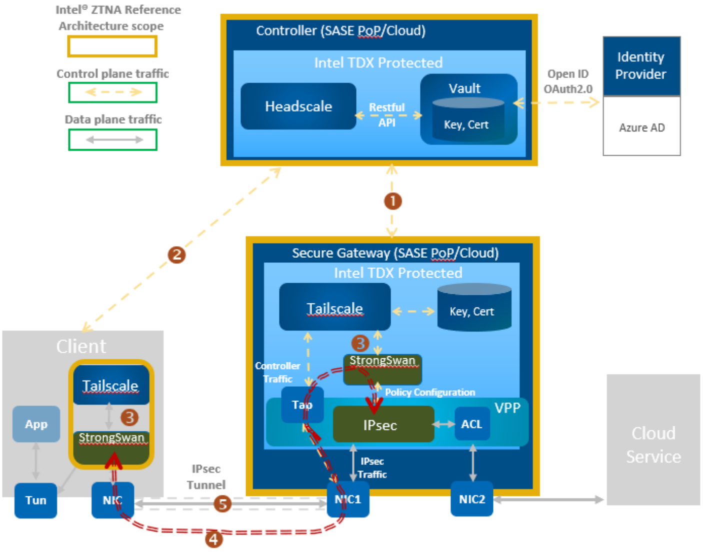


Figure 6. Workflow of IPSec Tunnel

1. The SGW registers with the controller and sends it related information such as NIC1's IP address and its public key.
2. The client registers with the controller and sends it related information such as the IP address of its NIC and its public key.
3. After the client has authenticated with the controller it will exchange IP addresses, public keys, and other information with the SGW. This information is used to configure strongSwan on both the client and the SGW. The client initiates IKE negotiation to the SGW through strongSwan, and the SGW configures the VPP IPSec module with the result of the negotiation.
4. The client and the SGW establish an IPSec tunnel for communication.

After the tunnel is established, the client can access the remote service through the secure tunnel and SGW.

## 5.4 Automatic Certificate Management

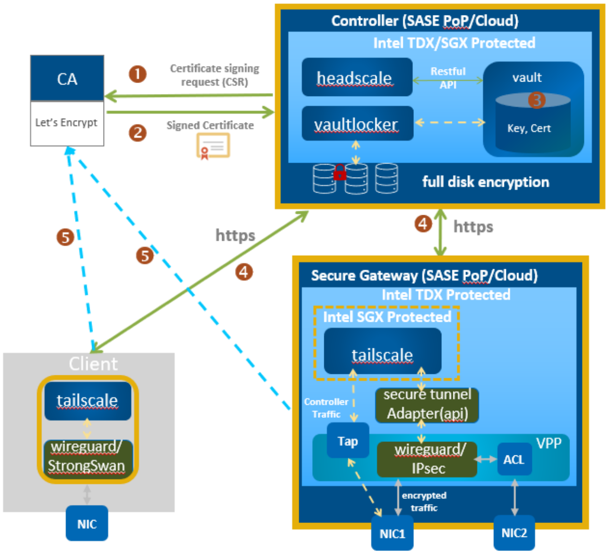


Figure 7. Workflow of Certificate Configuration

TLS is recognized as a secure network protocol. For secure network access in the Intel® ZTNA Reference Architecture, we can configure the controller to provide HTTPS service. To do that, we acquire a TLS certificate from public CA and deploy it to the controller. Let's Encrypt is a free, automated, and open certificate authority by the nonprofit Internet Security Research Group (ISRG). It provides TLS certificates to 300 million websites with high stability. It supports the ACME protocol to realize automatic certificate management. Every server with a domain name can request certificates from Let's Encrypt and upgrade from HTTP to HTTPS. In the Intel® ZTNA Reference Architecture we use acme.sh as an ACME protocol client to manage certificates. The workflow of certificate configuration is shown in Figure 7.

1. Controller generates a Certificate Singing Request (CSR) which includes public key, organization, country, and common name etc. Common name points to the fully qualified domain name of controller. And then, the controller sends it to the server of Let's Encrypt.
2. Once Let's Encrypt receives the CSR, it will validate if you control the domain name in that certificate using "challenge". The challenge will be handled automatically by ACME.sh. After passing the challenge, Let's Encrypt will issue the signed certificate to controller.
3. Controller receives and saves the certificate to specified path and configures Headscale. Headscale starts by using that certificate in HTTPS mode.
4. When client and SGW create connections with the controller, the controller will send its certificate to prove its identity.
5. The Client and SGW will validate the certificate leveraging Let's Encrypt as a trusted entity. Once verified, the controller can establish secure connections with both client and SGW.

# 6 Deployment and Demo

This section shows the basic deployment of Intel® ZTNA Reference Architecture solution. It covers detailed install instructions for core components and setups.

## 6.1 Deployment in Docker

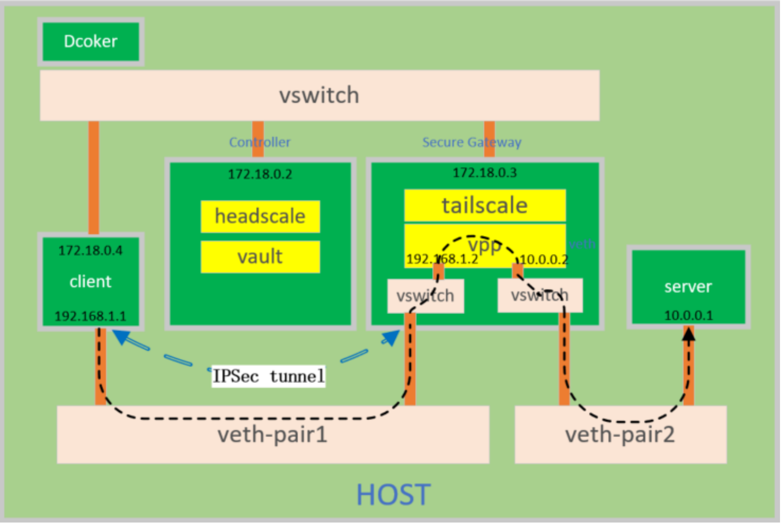


Figure 8. Topology of Deployment in Docker

Docker is an open-source engine that makes it easy to create a lightweight, portable, and self-sufficient container for any application. You can get a trial of Intel® ZTNA Reference Architecture in docker environment efficiently and easily.

1. Create Dockerfile and add the installation files of Intel® ZTNA Reference Architecture to the building image.
2. Build an image and create 4 dockers, then configure network and namespace of dockers. Use the veth pair to make dockers interconnected.
3. Install controller, SGW, client and server in each of the 4 dockers.
4. SGW registers to controller first, then controller enables the advertise routes of SGW.
5. Client registers to controller and establishes an IPSec tunnel with SGW. Client then would be able to ping the server.

The network topology is shown in Figure 8Figure 8, which demonstrate a comprehensive overview of Intel® ZTNA Reference Architecture.

## 6.2 Demo on Multi-cloud Deployment

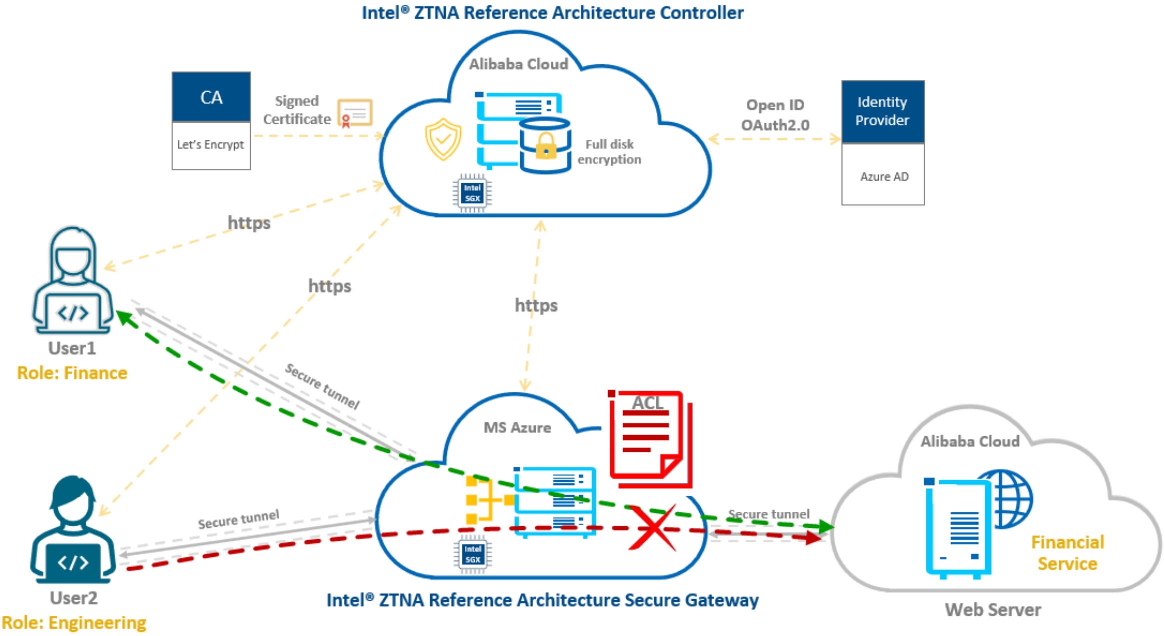


Figure 9. Topology of Multi-cloud Deployment

The Intel® ZTNA Reference Architecture system supports multi-cloud deployments. The topology of the deployment is shown in Figure 9Figure 9. The controller and web servers are deployed in different regions of Alibaba Cloud. The SGW is deployed in Azure. We simulate a real enterprise network access scenario where user1 is a financial staff member and user2 is an engineer. The financial service runs on an Alibaba Cloud instance and thus, based on the roles of the users, we'd expect that only user1 can access to the server. The Intel® ZTNA Reference Architecture implements role-based access control using the VPP ACL module. The workflow of the demo is as follows:

1. One-click deployment of all cloud components with Intel Multi Cloud Networking Automation (MCNAT) tool.
2. The controller configures the ACL rules, etc.
3. The SGW and webserver register to the controller. The controller sends the ACL out to the SGW.
4. User1 wants to access the webserver before registration but fails.
5. User1 registers to the controller and authenticates with Azure Oauth2 and then User1 can access the webserver normally.
6. User2 registers to the controller and authenticates with Azure Oauth2.
7. User2 wants to access the webserver but is restricted by the ACL.

To access the demo video for a detailed introduction, please contact your Intel sales person or the document authors.

# 7 Summary

Intel® Zero Trust Network Access Reference Architecture builds a good reference implementation of zero trust system. It differentiates with other solutions by applying the latest Intel security technologies including confidential computing and crypto acceleration. Until now, Intel ZTNA Reference Architecture has covered features including user authentication with Azure AD, service authorization with RBAC, secure WireGuard/IPSec tunnel, full disk encryption and secrets protection with Intel SGX and Intel TDX, providing a completed end-to-end multi-cloud network security solution. This solution also demonstrates an automatic multi-cloud deployment. There are continuously development efforts to enhance it with latest Intel processor features. This solution provides a good reference for more secure, higher-performance, and more flexible deployment. To access this software, please contact your Intel sales person, or the document authors.

## Appendix A Platform Configuration

w/o TDX

| Name | Intel M50CYP2SBSTD |
|---|---|
| Vendor | Intel Corporation |
| Product Name | M50CYP2SBSTD |
| BIOS Version | SE5C620.86B.01.01.0004.2110190142 |
| SGX | Yes |
| SGX EPC size | 2GB (max 64GB) |
| TDX | No |
| OS | Ubuntu 20.04.6 LTS |
| Kernel | 5.4.0-148-generic |
| IRQ Balance | enabled |
| CPU Model | Intel(R) Xeon(R) Platinum 8380 CPU @ 2.30GHz |
| Base Frequency | 2.30GHz |
| CPU Family | 6 |
| CPU Model | 106 |
| CPU(s) | 160 |
| Thread(s) per Core | 2 |
| Core(s) per Socket | 40 |
| Socket(s) | 2 |
| NUMA Node(s) | 2 |
| Turbo | enable |
| Memory Installed | 128GB |

w/ TDX

| Name | Intel ArcherCity |
|---|---|
| Vendor | Intel Corporation |
| Product Name | ArcherCity |
| BIOS Version | EGSDCRB1.SYS.9207.P03.2211041115 |
| SGX | Yes |
| SGX EPC size | 2GB (max 64GB) |
| TDX | Yes |
| OS | Ubuntu 22.04.1 LTS |
| Kernel | 5.15.0-mvp |
| IRQ Balance | enabled |
| CPU Model | Intel(R) Xeon(R) Platinum 8475B @ 3.80GHz |
| Base Frequency | 3.80GHz |
| CPU Family | 6 |

| | |
|---|---|
| CPU Model | 143 |
| CPU(s) | 192 |
| Thread(s) per Core | 2 |
| Core(s) per Socket | 48 |
| Socket(s) | 2 |
| NUMA Node(s) | 2 |
| Turbo | enable |
| Memory Installed | 256GB |

| Software Configuration | Software version | Location |
|---|---|---|
| Host OS | Ubuntu 20.04.6 LTS | https://ubuntu.com/ |
| Kernel | 5.4.0-148-generic | https://www.kernel.org/ |
| Headscale | v0.15.0 | https://github.com/juanfont/headscale |
| Tailscale | v1.22.2 | https://github.com/tailscale/tailscale |
| FDio VPP | stable/2302 | https://github.com/FDio/vpp |
| Vault | 1.10.3 | https://www.vaultproject.io/ |
| SGX SDK&PSW | 2.16.100.4 | https://github.com/intel/linux-sgx |
| Occlum | 0.26.3 | https://occlum.io/ |
| Tdx-tools | 2022ww44 | https://github.com/intel/tdx-tools |
| Acme.sh | V3.0.5 | https://github.com/acmesh-official/acme.sh |
| Vaultlocker | 1.0.6 | https://github.com/openstack-charmers/vaultlocker |
| StrongSwan | 5.9.6 | https://github.com/strongswan/strongswan |