White Paper

Intel® Trust Domain Extensions

Table of Contents

01. Introduction

Providing better protection for data in the enterprise and cloud environments has long been a high concern of IT departments, governments, and users of personal services. The rise in number and size of data breaches has highlighted how much personal data is essential to many businesses and services that we have become dependent on and how this datafication is being monetized by many companies. The security concerns around this datafication trend were further heightened in 2013 when Edward Snowden made revelations about unwarranted access to data.

The reaction to this datafication trend by the major technology companies is changing their stances on the protection and privacy of data they are processing as being an important asset to protect. So much that big technology providers are now turning privacy into a marketable advantage, and the increase in the distrust of third-party processing vendors has increased that some are looking to become blind to the computations they host.

The release of Intel(r) Software Guard Extensions (SGX) in 2015, put the ability to do application computations that were much harder to introspect by the owner of the platform within reach of every-day PC platforms and their server derivatives, and it sparked a paradigm and a race to build-out ecosystems where this could be done at scale. We call this paradigm Confidential Computing.

The main premise behind this paradigm is that those that are controlling the platform and those that have data on the platform being processed are two separate entities. In typical installations the platform owner has full access to what is being processed on the platform. This is the hierarchical nature of memory management and access control on modern computing platforms.

In this paper we introduce Intel® Trust Domain Extensions (Intel® TDX). An architectural technology to deploy hardware-isolated, Virtual Machines (VMs) called Trust Domains (TDs). Intel TDX is designed to isolate TD VMs from the Virtual-Machine Manager (VMM), hypervisor and other non-TD software on the host platform. Intel TDX may be used to enhance confidential computing by helping protect TDs from a broad range of software attacks and which also helps reduce the TD Trusted Computing Base (TCB). Intel TDX is designed to enhance a platform user's control of data security and IP protection. Intel TDX can also enhance the Cloud-Service Providers (CSP) ability to provide managed cloud services without exposing tenant data to adversaries.

In this paper we describe Intel TDX technology and how it forms one of the pillars of our Confidential Compute offering.

02. Intel TDX – Technical Explanation

The Intel-TDX solution is built using a combination of Intel Virtual Machine Extensions (VMX) instruction-set-architecture (ISA) extensions, Intel total memory-encryption multi-key, (Intel® TME-MK) technology, and a CPU-attested software module.

Intel TDX solution can provide the following capabilities to TDs:

- Memory and CPU state confidentiality and integrity to help keep the sensitive IP and workload data secure from most software-based attacks and many hardware-based attacks. The workload now has a tool that supports excluding the firmware, software, devices, and operators of the cloud platform from the trusted-computing base (TCB). The workloads can use this tool to foster more secure access to CPU instructions, security, debug, and other technologies. The workload can now have this ability irrespective of the cloud infrastructure used to deploy the workload.
- Remote attestation enables a relying party (either the owner of the workload or a user of the services provided by the workload) to establish that the workload is running on an Intel-TDX-enabled platform located within a TD prior to providing that workload data. Remote attestation aims to allow the owners and consumers of the service to digitally determine the version of the TCB on which they are relying on to help secure their data.

Intel TDX also augments defense of the TD against limited forms of attacks that use physical access to the platform memory, such as offline, dynamic-random- access memory (DRAM) analysis (*e.g.*, cold-boot attacks) and active attacks of DRAM interfaces, including capturing, modifying, relocating, splicing, and aliasing memory contents. Intel TDX does not defend against replay of memory through physical attacks.

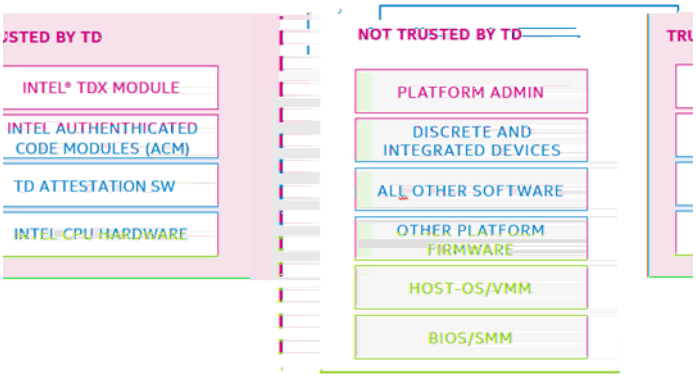

Figure 1 Trust Boundaries for TDX

The VMM continues to be the resource manager, and TDs do not have privileges to deny service to the VMM. Protecting a TD against denial of service by the VMM is not a security objective of Intel TDX.

To help enforce the security policies for the TDs, a new mode of the CPU called Secure-Arbitration Mode (SEAM) is introduced to host an Intel-provided, digitally signed, but not encrypted, security-services module. The Intel-TDX module is hosted in a reserved, memory space identified by the SEAM-range register (SEAMRR). Under the design, the CPU only allows access to SEAM-memory range to software executing inside the SEAM-memory range, and all other software accesses and direct-memory access (DMA) from devices to this memory range are aborted. SEAM is also designed to not have any memory-access privileges to other protected, memory regions in the platform, including the System-Management Mode (SMM) memory or Intel® Software Guard Extensions (Intel® SGX) protected memory (Figure 5.2).

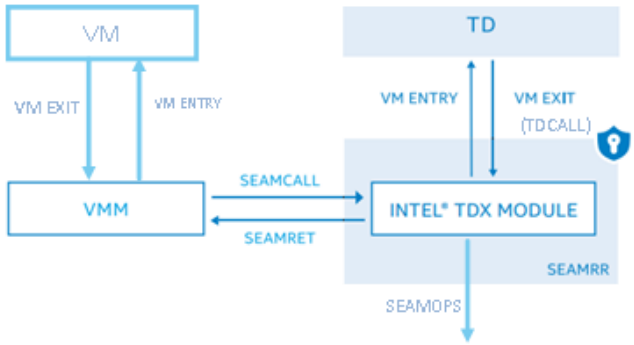

Figure 2 SEAM Module Flow

The SEAM-memory range offers cryptographic confidentiality protection using AES in XTS mode with an ephemeral 128-bit, memory-encryption key and can operate in one of two available modes for memory integrity protection (to enable various memory configurations). Memory integrity may be enforced by either (the default) cryptographic-integrity protection scheme or a logical-integrity protection scheme. The cryptographic-integrity scheme uses a SHA-3-based, message-authentication code (MAC) (28-bit) that helps prevent host/system software accesses as well as detects state-tamper from software (e.g., rowhammer) and some hardware attacks. The logical-integrity protection scheme is designed to prevent host/system software accesses only.

In order to install the module for Intel TDX, a new, Intel® Trusted Execution Technology (Intel® TXT) authenticated-code module (ACM), called the SEAM Loader (SEAMLDR), is provided to help verify the digital signature on the Intel TDX module and load it into the SEAM-memory range. By design, the measurement and security-version number (SVN) of the module are recorded into hardware-measurement registers by the SEAMLDR and then loaded into SEAM-memory range in response to the VMM invoking the SEAMLDR ACM in order to load the module, which has no persistence. The cloud- service provider can additionally apply its own security policies on the acceptable version of Intel TDX that should be loaded on its servers.

A SEAMCALL instruction introduced for the VMM is designed to place the CPU in SEAM-VMX-root operation and invoke the module. The Intel TDX module is designed to provide an interface to the VMM to create, delete, and schedule execution of TDs. The Intel TDX module acts as the trusted intermediary to help implement security policies, actions, and necessary mitigations for the TDs.

As part of TD creation, the VMM provides the memory pages for the TD code, data, and TD-associated-metadata structures such as the virtual-machine-control structure (VMCS) and the state-save area used to save the TD state when it is not executing.

Intel TDX uses Intel® 64 architecture, including the VMX architecture, to help manage the TDs. The Intel-TDX module is designed to perform VM entry to SEAM-VMX, non-root operation using VMRESUME and VMLAUNCH-VMX instructions to execute the TD.

The Intel-TDX module helps ensure that the execution controls active for a TD do not allow the VMM or other untrusted entities to intercept TD accesses to TD-assigned resources like control registers, model-specific registers (MSRs), debug registers, performance-monitoring counters, Time-Stamp Counter, etc. The TDX module aims to implement security policies for the TDs. An example of such policy would be the module using the Indirect-Branch- Prediction Barrier (IBPB) when switching TDs to help keep a TD-indirect-branch prediction from being influenced by code in a previously executed TD.

The TD would have full use of debug and performance-monitoring features if such use is authorized for the TD at creation. If they are not authorized, these features would be disabled when the TD executes. Debug and performance monitoring attributes of the TD are included in the TD- attestation report.

Intel TDX is designed to allow a VMM to limit the features made available to the TD for

TD ACQUIRING KEYID(S).

On TD creation, the module is intended to assign each TD a unique, private KeyID (Fig. 3.)

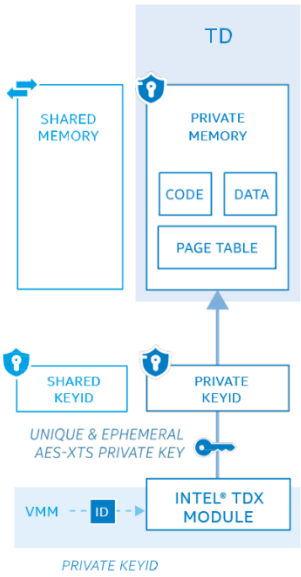

Figure 3

ACCESS CONTROL TO PRIVATE KEYID(S).

The CPU is designed to disallow software other than the Intel TDX module and TDs from making memory accesses using a private KeyID. Attempting to access a private KeyID by software outside the SEAM mode would cause a page-fault exception (#PF). Similarly, DMA from devices using a private KeyID would be aborted.

MEMORY ENCRYPTED BY PRIVATE KEYID(S).

The TD-ownership bit associated with the line in memory seeks to detect software or device attempts to read memory encrypted with private KeyID, using a shared KeyID, to reveal the ciphertext. On such invalid accesses, the TME-MK returns a fixed pattern to help prevent ciphertext analysis.

When cryptographic-integrity protection is enabled, the MAC helps ensure that the TD reads back the same data it had last written to its private memory. If the memory integrity was violated per the attack model, then a subsequent access by a TD to that corrupted memory would lead to a MAC-verification failure. With logical-integrity a similar property is obtained for software attacks based on access control via the TD-ownership bit.

A MAC-verification or a TD-ownership check failure would be fatal to the TD and lead to its termination, but other TDs and platform software would not be impacted. Encryption of TD-private memory helps defend against some classes of physical attacks on memory (e.g., cold-boot attacks). Cryptographic-integrity mode adds cryptographic defense to defend against software tamper attacks (e.g., rowhammer) and some hardware-based corruption attacks. In contrast, the logical-integrity mode is designed to prevent software access to TD ciphertext. Sophisticated attacks such as replay of memory via physical access are not detected or protected-against by the current generation of memory protection for Intel TDX.

B. ADDRESS-TRANSLATION INTEGRITY.

TDs have access to two classes of memory—private memory that holds the confidential data of the TD and shared memory used to communicate with untrusted entities external to a TD. The private memory of the TD improves cryptographic confidentiality and integrity protection using a unique, ephemeral-memory-encryption key assigned to that TD.

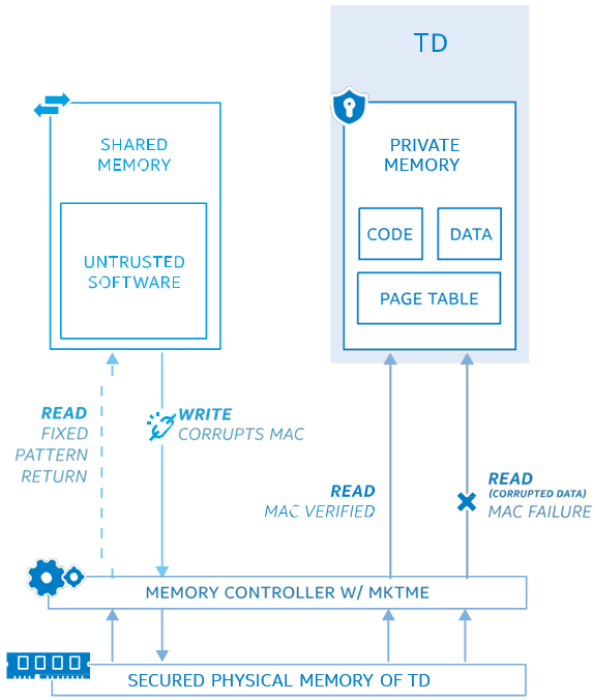

Figure 4

Shared memory is used to communicate with the agents outside the TD to perform I/O operations such as network access, storage services, invoking hypervisor services, etc.

The highest order bit of the guest-physical address (GPA) is designated as a “Shared” bit in order to indicate if that GPA maps private memory (when “Shared” bit is 0) or maps shared memory (when “Shared” bit is 1). The TD is designed to locate all its private code and data in private memory mapped using a private GPA. The TD- assigned, private key helps encrypt and integrity-protect all memory accesses that use GPA with “Shared” bit set to 0. All shared-memory accesses that use GPA with “Shared” bit set to 1 may be encrypted and integrity-protected with the aid of a shared key that the hypervisor manages (Figure 5).

The VMM helps allocate and map memory used by the TDs into the GPA of the TD using an extended-page table (EPT) that provides the GPA to physical-address (PA) translation.

When a TD is executing, the design designates that two EPTs will be active for the TD: a secure EPT used to provide private GPA to PA translations and a shared EPT used to provide shared GPA to PA translations (Figure 6).

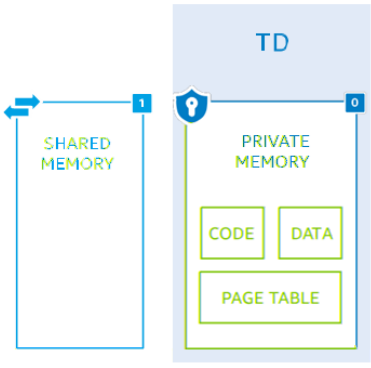

Figure 5

The Intel-TDX module helps provide secure-EPT management functions to the VMM to add or remove mappings from the secure EPT and enforce security policies around those operation with the aim of preserving the integrity of the memory layout. The memory used to build secure EPT is designed to be encrypted and integrity- protected using the unique, per-TD, memory-encryption key.

The CPU helps prevent a TD from locating page-table structures and executable code in shared memory. CPU would cause a page fault (#PF) on code fetches or page-table accesses if they are in shared memory.

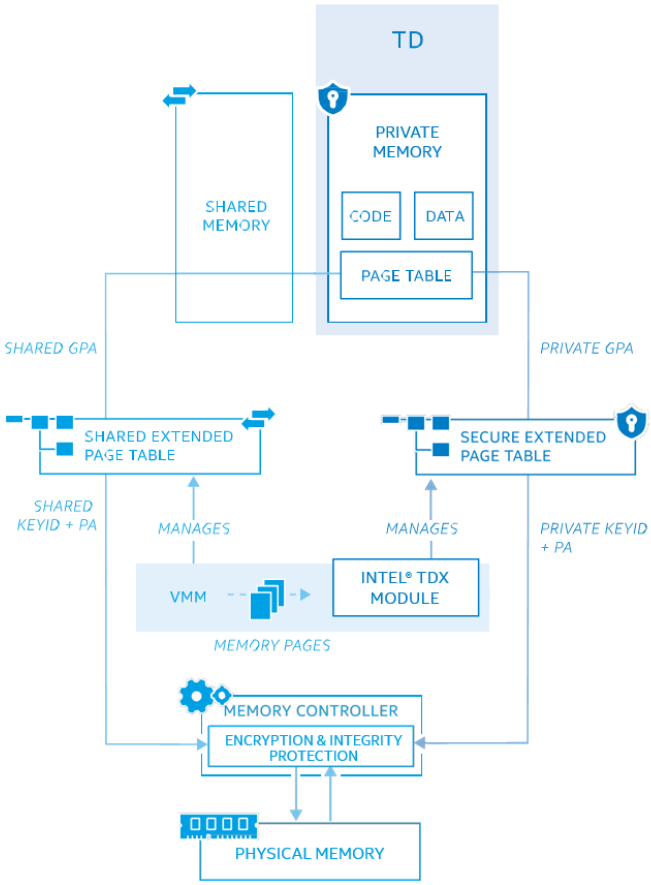

Figure 6

The Intel-TDX module is designed to maintain a tracking-data structure called Physical-Address-Metadata Table (PAMT) so that a page mapped into the secure-EPT of a TD cannot be mapped into secure EPT of any other TD. The module for Intel TDX also uses PAMT to help map a page to only one GPA in the secure EPT and provide functions to the VMM to add 4K, 2M, or 1G translations to the secure EPT. This module is then designed to track the page sizes and page types in the PAMT to track the correctness of the secure-EPT operations and the required invalidations of TLB entries when a page is unmapped from the secure EPT. The PAMT helps ensure that all memory allocated to a TD is initialized to a known state before first access by the TD.

Virtual address translated to physical addresses using either the secure EPT or shared EPT would be cached in the CPU TLB. The TLB is designed to associate a tag with the translation to identify the TD that created the translations.

The secure-EPT-based, address-translation architecture enables mapping large/huge pages into the secure/shared EPT as well as caching the translations as large/huge pages when appropriate. Address translations for a TD are similar to a legacy VM and involve two levels of page walk. Therefore, software in a TD would incur similar overhead as compared to address-translation overhead for software in a legacy VM.

C. CPU-STATE CONFIDENTIALITY & INTEGRITY

When a TD is created, the module for Intel TDX would require the VMM to provide a set of memory pages to be used to host the virtual-machine-control structures (VMCS), the state- save area for the TD, etc. The goal of the module is to use its page-allocation trackers to enforce that these pages have not been simultaneously assigned to other TDs by the VMM. The Intel-TDX module would then initialize and configure these structures using the TD-assigned, private key to help provide cryptographic confidentiality and integrity protection to the TD-CPU state (Figure 7).

D. SECURE INTERRUPT & EXCEPTION DELIVERY

Interrupts and exception delivery to the TD are designed to use the VMX-APIC virtualization and virtual-interrupts architecture. The APIC-virtualization architecture aims to provide CPU emulation of many registers of the APIC, track the state of the virtual APIC, and deliver virtual interrupts. The CPU would do all the above in VMX-non-root operation without requiring a VM exit from the TD. The use of VMX- APIC virtualization and virtual-interrupts architecture would then deliver interrupts into TD(s) efficiently and avoid the need for modifications to the operating system in the TD to emulate the APIC.

The goal of the CPU is to track the state of the virtual APIC using a virtual-APIC page and related state in the VMCS of that TD. The virtual-APIC page and the VMCS would be initialized by the Intel-TDX module using the TD-private key at TD creation.

The VMX-posted-interrupts-processing architecture is designed to allow a VMM or devices to deliver virtual interrupts directly to a TD through a posted-interrupt descriptor processed by the CPU hardware in response to a notification interrupt. The virtual interrupts in the posted- interrupt descriptor would then be delivered to the TD through the virtual APIC by the CPU. The VMX architecture is enhanced to help block any attempts to deliver exception vectors to a TD as virtual interrupts.

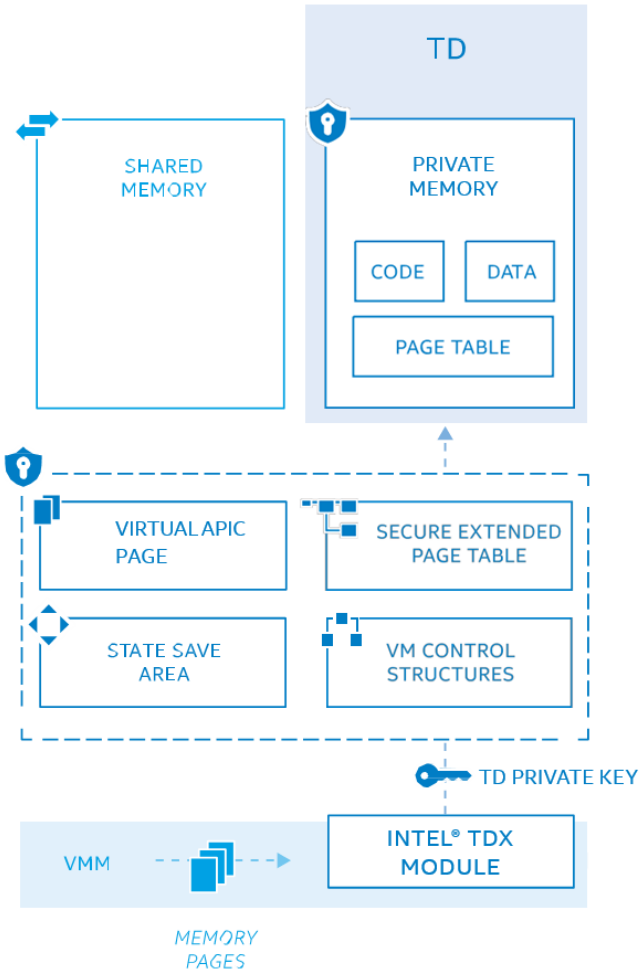

Figure 7

The TD-interrupt-virtualization architecture helps deliver interrupts into a TD without violating TD assumptions such as interrupt priorities and masking. The VMX architecture is designed to disallow injection of exceptions into a TD. Intel- TDX module provides functions the VMM can use to deliver virtual, non-maskable interrupt (NMI) into a TD without violating the x86 NMI architecture.

E. REMOTE ATTESTATION

Remote attestation helps a remote provider (also known as a relying party) have increased confidence that the software is running inside a TD, on a genuine, Intel-TDX system, and at a given security level, which is also referenced as the TCB version. Attestation results can provide:

Data that TDs associate with itself

- The goal is for this information to be provided by the software in the TD when requesting an attestation. For example, the TD software might include the public key it would like to use to communicate with the relying party as part of the attestation.

TD Measurements provided by Intel-TDX module

- At TD creation, the Intel TDX module is designed to initialize the measurement registers for the TD. As part of the TD creation, the VMM would request the module to add a set of pages to the TD. The module would then extend a static- measurement register called TD-measurement register (TDMR) with the measurements of the initial pages added to the TD along with metadata associated with these pages. It also seeks to provide the TD a set of runtime-extendable- measurement registers (RTMR) that would be extended by the code in the TD with measurements of additional code and data at runtime. The goal of the attestation is to include all measurement registers.

Details of additional unmeasured state provided by Intel TDX module

- Some states of the TD, like the attributes of the TD, identities of the TD owner (MROWNER), etc. would not be measured but included in the attestation results for the TD.

SVNs of elements in TDX TCB provided by CPU HW

- Each element of the Intel-TDX TCB would be assigned an SVN. A TCB is considered up to date if all components of the TCB have SVNs greater than or equal to a threshold published by the author of the component(s). For the hardware, these SVNs are known collectively as the CPUSVN and include the SEAMLDR SVN. The module is designed to be in the TCB of the TD, and the module's SVN should also be reflected in the attestation.

A goal of Intel TDX is to use an Elliptic-Curve-Digital-Signature-Algorithm (ECDSA)-based, asymmetric-attestation key representing the Intel-TDX-TCB version in order to sign an assertion (a Quote) with the information listed above. Intel TDX was designed so that, if a vulnerability is mitigated or otherwise addressed by an update to the platform, relying parties can verify that the update has been installed. The process of updating the platform attestation to reflect the update is called TCB Recovery. A new attestation key would be created to reflect the update in the platform's attestations. The new TCB would be reflected in the attestations that occur following the replacement of the attestation key.

The Intel-TDX architecture is designed to utilize an Intel-SGX enclave, called the TD-quoting enclave, to generate the remote attestation for a TD. The CPU would provide a new instruction, SEAMREPORT, to be invoked only by Intel-TDX module and create an evidence structure that is cryptographically bound to the platform hardware for consumption by the TD-quoting enclave.

The SEAMREPORT instruction is designed to take the attestation information provided by the TD software, the TD measurements, and additional information provided by the Intel-TDX module as input and generate a "Report" structure that includes the SVNs of the TDX-TCB elements.

Figure 8

This “Report” structure is designed to be integrity-protected using a MAC. The CPU would then provide an EVERIFYREPORT2 instruction to be used by enclaves to verify the MAC on the report structure to help ensure the report structure was created on the same platform as the one on which it is executing.

The TD-quoting enclave is designed to then generate the Quote for the report using an asymmetric attestation key. When a TD receives an attestation request, shown in Fig 8, from an off-platform challenger (1), the TD would then request the module provides the TD a report (2) that includes the attestation information along with TD-provided data. The Intel-TDX module would also invoke the SEAMREPORT instruction (3) to request the CPU generate a “Report” structure (4) that includes the TD-provided data, the measurements of the TD as maintained by the module, and SVNs of all elements in TDX TCB. The report structure is designed to be handed to the TD (5), and then the TD would request the VMM (6) convert the report into a remote atte

hypervisor to support this flow while maintaining the primary security objective of TD memory and CPU state confidentiality and integrity during migration across compatible platforms.

G. TD PARTITIONING

TD Partitioning provides the ability to support unmodified or unenlightened legacy VM as TD, thus providing CSP customers with the ability to lift and shift workloads into hardware isolated VMs or Trust Domains (TDs) without having to make any changes to the Guest OS. TD Partitioning extends the base TDX architecture by allowing TDs to contain multiple virtual machines (VMs). A TD may contain up to 4 VMs. The primary VM (known as the L1 VM) may act as a virtual machine monitor (known as the L1 VMM). Up to 3 nested VMs (known as the L2 VMs) serve as the guest of the L1 VMM.

H. VM PRESERVING UPDATES

TD Preserving Architecture provides the ability to update TDX Module at runtime, while persisting TD State and minimum TD interruption. The Trusted Compute Base (TCB) associated with each TD will be recovered and a new TCB tied to the updated TDX Module is consumed as part of new TD(s) launch with subsequent attestation. Existing TDs and their relevant metadata state will be securely saved in encrypted memory owned by the ACM thus protecting the secure surface. For Cloud Service Providers (CSP), using traditional TDX Module update methodologies will require reboot on the node, resulting in downtime in orders of minutes on the platform and offered Cloud Services. TD Preserving Update Flows will furnish updates in orders of milliseconds, this making the 99.999% Cloud availability goal of CSPs attainable.

03. Threat Model Overview

In order to meet the goals of Confidential Computing, platform owners need the ability to isolate tenant VMs (Workloads) from the VMM and other system software on the platform.

In this context TDX architecture considers 2 primary adversaries:

1. System Software Adversary: Administrative insiders (e.g., DC Admin, developer or technician), VMM, SMM, BIOS, which can launch a malicious SW-only attempt to either extract workload secrets and assets or spoof workload data/memory in an operational environment.
2. Hardware Adversary: CSP insiders (e.g., technician) that can attempt a HW attack to extract Cloud Tenant's secrets or spoof Tenants data/memory by hooking into system interfaces (DDR bus).

A System Software adversary has the ability to read and write system memory, including private memory of TDs, access to TD CPU state and registers, ability to execute from TD memory, ability to program critical system hardware devices (e.g., DMA engines), ability to program page tables/EPT to cause faults or aliasing, and the ability to inject virtual interrupts/exceptions into TDs. Thus, some examples of the software attacks include direct content/memory injection, capture and replay of content/memory, EPT remapping attacks.

A Hardware adversary has physical access to the platform and the ability to read and write the physical memory of the platform. It can thus launch attacks like DRAM freeze attacks, where physical memory can be read and analyzed offline, as well as an ability to inject content/overwrite physical memory and capture & replay physical memory at same or different location. TDX does not prevent all types of hardware attacks rather it uses cryptographic protections on memory to improve defenses against a limited set of techniques.

TDX uses Intel TME-MK to provide per-VM encryption or confidentiality protection. However, for technologies like TME and TME-MK, the Hypervisor or VMM is within the trust boundary or the TCB. TDX adds integrity protection, etc. with the goal of having all system software, including the Hypervisor, out of the trust boundary of the platform user's data or the workload. Thus, TDX would provide better protection or mitigation against compromised Hypervisor/software attacks. But the hypervisor still manages and controls the platform resource allocation like memory for a Guest VM, so providing protection against denial-of-service attack from compromised VMM is out of scope for TDX. Below is a table summarizing some common software and hardware attacks and which of these technologies (TME, TME-MK or TDX) provide mitigation against them.

Software Attacks	TME	TME-MK	TDX
Hardware Attacks			

04. Summary

Intel is introducing new architectural elements to help deploy hardware isolated VMs called trust domains (TDs). TDX1.0 Introduces:

- Secure-Arbitration Mode (SEAM) – a new mode of the CPU designed to host an Intel-provided, digitally-signed, security-services module called the Intel-TDX module.
- Shared bit in GPA to help allow TD to access shared memory.
- Secure EPT designed to translate private GPA to provide address-translation integrity and to prevent TD-code fetches from shared memory. Encryption and integrity protection of private-memory access using a TD-private key is the goal.

- PhysicalEx