White Paper

Intel® Trust Domain Extensions

Table of Contents

01. Introduction 1

02. Intel TDX – Technical Explanation 1

A. MEMORY CONFIDENTIALITY AND INTEGRITY 3

B. ADDRESS-TRANSLATION INTEGRITY. 4

C. CPU-STATE CONFIDENTIALITY AND INTEGRITY 6

D. SECURE INTERRUPT AND EXCEPTION DELIVERY 6

E. REMOTE ATTESTATION 6

03. Summary 8

01. Introduction

Intel® Trust Domain Extensions (Intel® TDX) is introducing new, architectural elements to deploy hardware-isolated, virtual machines (VMs) called trust domains (TDs). Intel TDX is designed to isolate VMs from the virtual-machine manager (VMM)/hypervisor and any other non-TD software on the platform to protect TDs from a broad range of software. Intel TDX enhances control of data security and IP protection for the cloud tenant, while helping maintain the cloud-service provider's (CSP) role of managing resources and cloud-platform integrity.

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, multi-key, total-memory-encryption (MKTME) 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 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.

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.

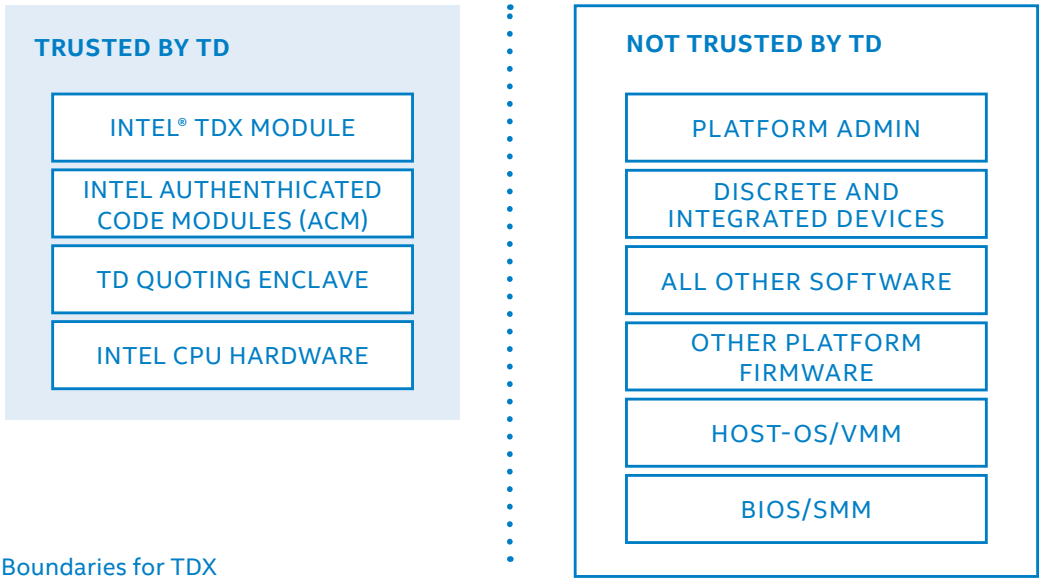


Figure 5.1. Trust Boundaries for TDX

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).

The SEAM-memory range offers cryptographic and confidentiality protection using AES in XTS mode with ephemeral 128-bit, memory-encryption keys and cryptographic-integrity protection using a SHA-3-based, message-authentication code (MAC) (28-bit).

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.

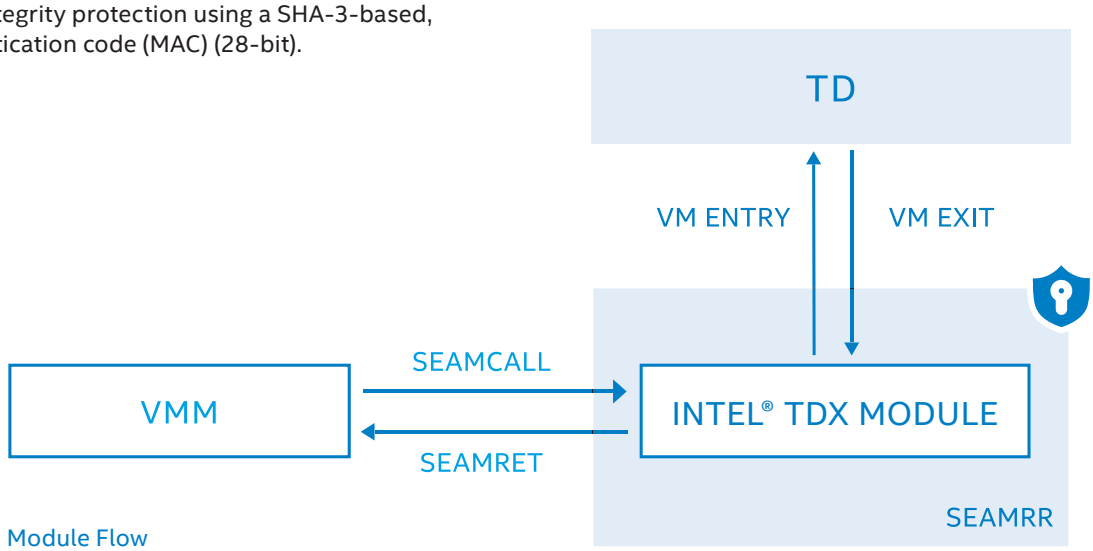


Figure 5.2. SEAM Module Flow

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 various reasons like virtualizing the hardware capabilities. Intel TDX can restrict capabilities of the VMM to either hide or allow the TD to use the feature. If the VMM chooses to allow the TD to use the feature, Intel TDX seeks to prevent the VMM from modifying the enumerations and information provided by instructions like CPUID, capability MSRs, and control registers when accessed by the TD. The TD aims to reliably use these hardware instructions without needing special, operating-system modifications but with similar performance that software in a legacy VM would expect.

Intel TDX is designed to enforce execution controls on the TD to cause execution of certain instructions like IN/OUT, HLT, RD/WRMSR, etc. in a TD to then cause a Virtualization Exception (#VE) to the TD OS, so that these instructions can be emulated by the TD OS in a secure manner for the TD software.

The intention is that a VM exit to enter SEAM-VMX-root operation occurs in response to certain instructions and events in SEAM-VMX-non-root operation. A TDCALL instruction is provided to help the TD call for service by enabling a VM exit to the module.

When a TD exits to the module, the module is designed to save registers (e.g., the general-purpose registers, control register, MSRs, debug registers, performance-monitoring counters, extended-state registers, and other CPU state associated with the TD) into the state-save area allocated for that TD. The module then scrubs these registers before returning execution control to the VMM, to help prevent leakage of TD state. When a TD is subsequently resumed, the CPU state of the TD is meant to be restored from this state-save area by the module.

A SEAMREPORT instruction is introduced for the Intel TDX module to help create an evidence structure—a report—that is cryptographically bound to the platform hardware with a MAC. The report structure created by SEAMREPORT subsequently helps generate a remote-attestation quote.

A SEAMRET instruction is introduced for the module to return execution control to the VMM.

Intel TDX is designed to provide a TD the following capabilities:

A. Memory Confidentiality And Integrity

B. Address-Translation Integrity

C. CPU-State Confidentiality And Integrity

D. Secure Interrupt And Exception Delivery

E. Remote Attestation

A. MEMORY CONFIDENTIALITY AND INTEGRITY

CRYPTOGRAPHY.

Intel TDX uses the MKTME engine to enable memory encryption and enhance it by adding support for cryptographic-integrity protection using a SHA-3 based MAC (28-bit) on each cache line in addition to AES-XTS 128-bit-memory encryption. Additionally, a 1-bit-ownership tag (TD-bit; included in the MAC) is associated with each cache line to identify if the line is associated with a memory page assigned to a TD. SHA-3-256 (KECCAK [512]) is meant to be used as the underlying function for MAC generation with the 256b output truncated to 28b for storage and verification.

Each key supported by MKTME is identified by a KeyID or key identifier. The CPU is designed to provide a PCONFIG instruction used by the Intel-TDX module to program a CPU-generated, unique, and ephemeral AES-XTS 128-bit key to each KeyID. The keys programmed into the MKTME are tailored not to be accessible by software or by using external interfaces to an SOC.

FUNCTIONAL SHARING OF KEYID(S).

The set of KeyID(s) supported by MKTME can be partitioned into two groups of KeyID(s)—a set of private KeyID(s) and a set of Shared KeyID(s). PCONFIG is designed to only allow private KeyIDs to be programmed with keys when called from SEAM mode (i.e., by the TDX-SEAM module).

TD ACQUIRING KEYID(S).

On TD creation, the module is intended to assign each TD a unique, private KeyID (Fig. 5.3.)

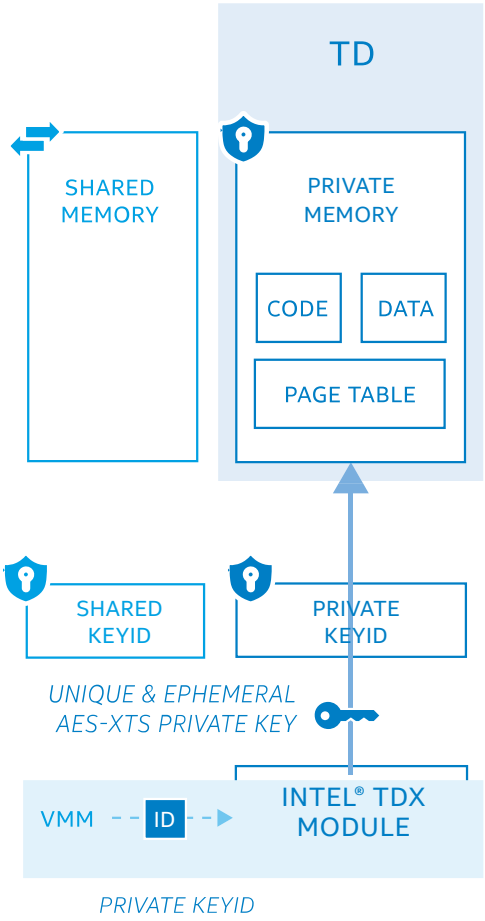


Figure 5.3.

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-bit associated with the line in memory seeks to detect software or devices attempting to read memory encrypted with private KeyID, using a shared KeyID, to reveal the ciphertext. On such accesses, the MKTME returns a fixed pattern to prevent ciphertext analysis.

The integrity protection provided by 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, then a subsequent access by a TD to this corrupted memory would lead to a MAC-verification failure. A MAC-verification

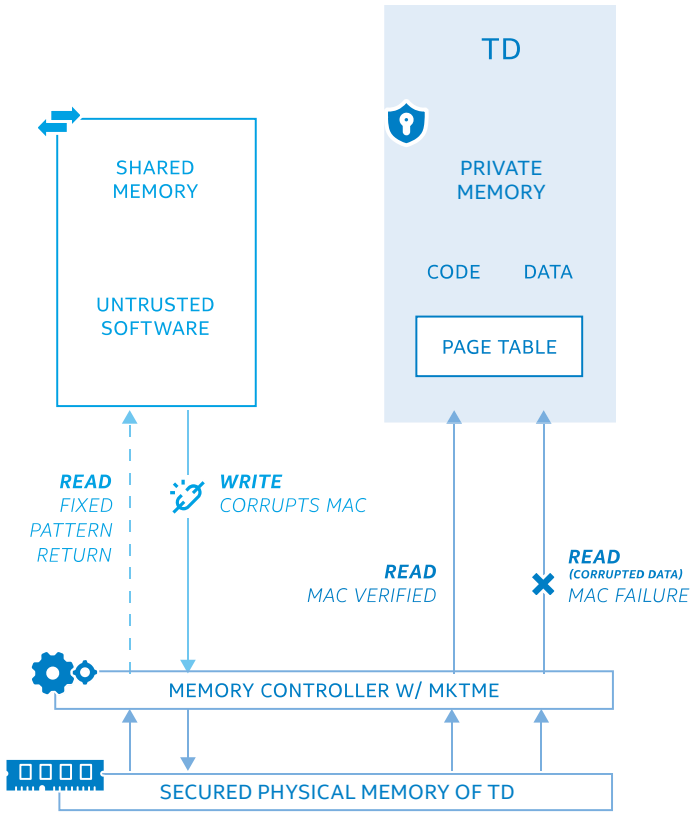


Figure 5.4.

failure would be fatal to the TD and lead to its termination, but other TDs and platform software would not be impacted. Cryptographic confidentiality and integrity on TD-private memory helps defend against some classes of physical attacks such as cold-boot attacks and simple-corruption attacks. More sophisticated attacks involving replay of memory contents using physical access are not detected and/or protected against by the current generation of memory protections 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.

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

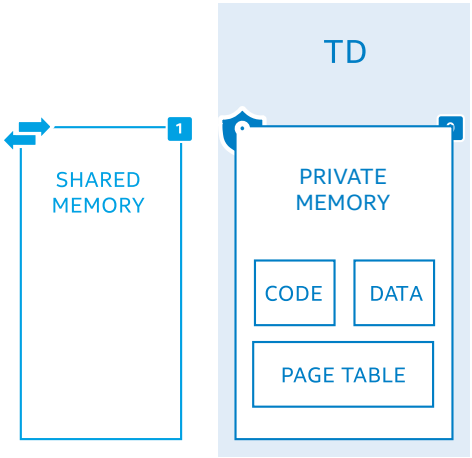


Figure 5.5.

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 5.5).

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 5.6).

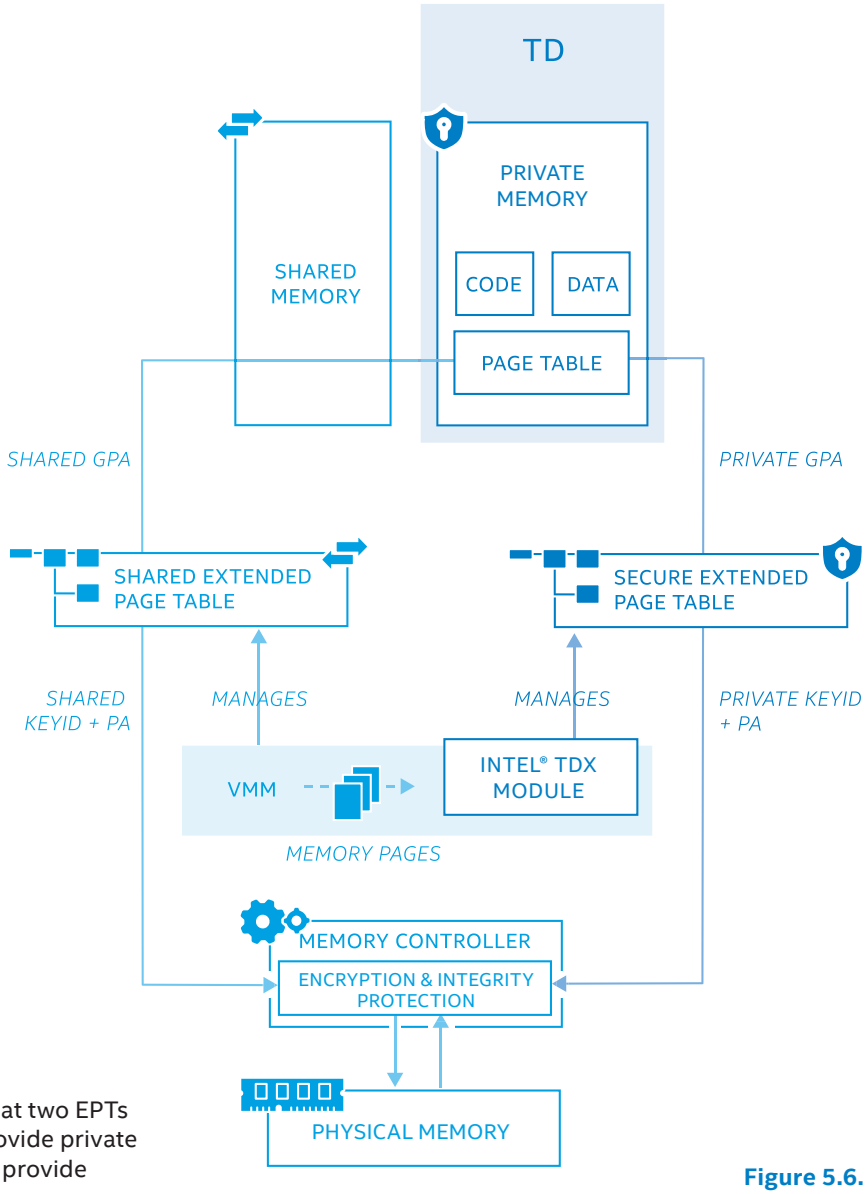


Figure 5.6.

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 operations 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.

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 in order 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 AND 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 5.7).

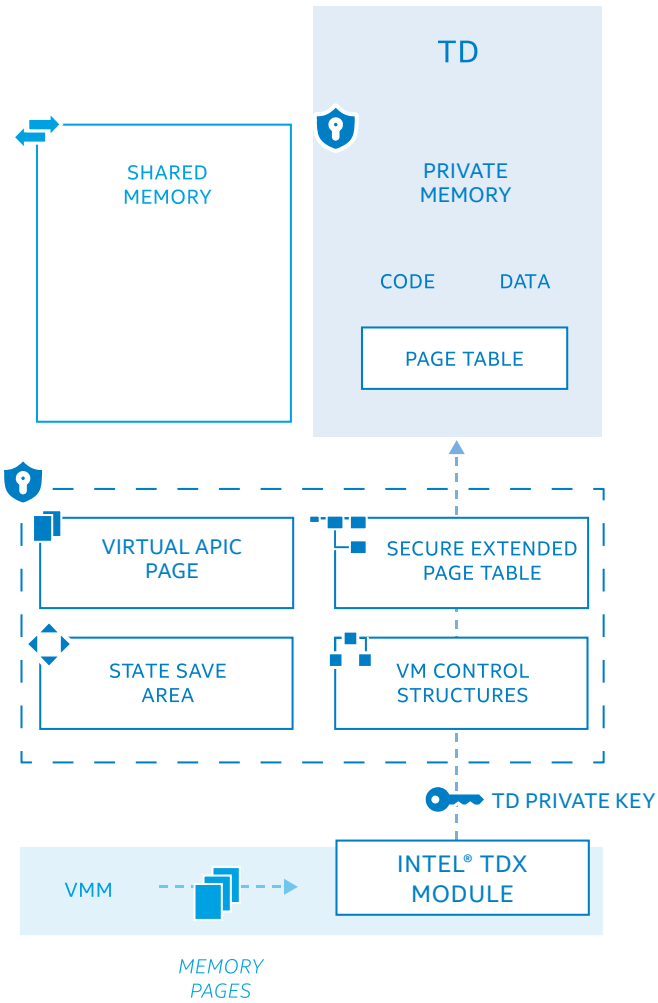


Figure 5.7.

D. SECURE INTERRUPT AND 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.

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 TD associate with themselves

- 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 of these 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. This "Report" structure is designed to be integrity-protected using a MAC. The CPU would then provide an EVERIFYREPORT 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 from an off-platform challenger (1), the TD would then request the module provide 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 attestation (a Quote) (7, 8, and 9) by the TD-quoting Enclave. The TD-quoting enclave would then verify the MAC on the report using EVERIFYREPORT and convert the report, if verified, into a Quote by signing the report using the TD's asymmetric-attestation key. The Quote would next return to the challenger (10). Finally, the challenger is designed to use an attestation-verification service (11 and 12) to perform quote verification.

To support attestation infrastructure for Intel TDX, Intel has built a general-certification infrastructure, built upon the Intel® Software Guard Extensions Data Center Attestation Primitives (Intel® SGX DCAP), to help certify TD-quoting Enclaves with a certificate chain rooted to an Intel-issued certificate. The foundation of this infrastructure, shown in Fig. 5.9, is an Intel-provided enclave called the Provisioning-Certification Enclave (PCE), which is designed to act as a local, Certificate Authority for local TD-quoting Enclaves (i.e., running on the same platform as each other). The TD-quoting Enclave(s) would then generate their own Attestation Keys using their preferred method and algorithm (1). The TD-quoting Enclave aims to provide the PCE with the attestation-public key (2). The PCE is designed to authenticate the request and issue a certificate-like structure identifying the TD-quoting enclave and the Attestation Key (3). This structure would then be signed by a device and TCB-specific, signing key called the Provisioning-Certification Key (PCK). Intel aims to publish certificates and certificate-revocation lists (CRLs) for the PCKs in all genuine, Intel platforms. This would result in a complete, signature chain from the Quotes to an Intel-certification authority (CA) (4). The resulting Quote would be verified by anyone with the complete, certificate chain and CRLs.

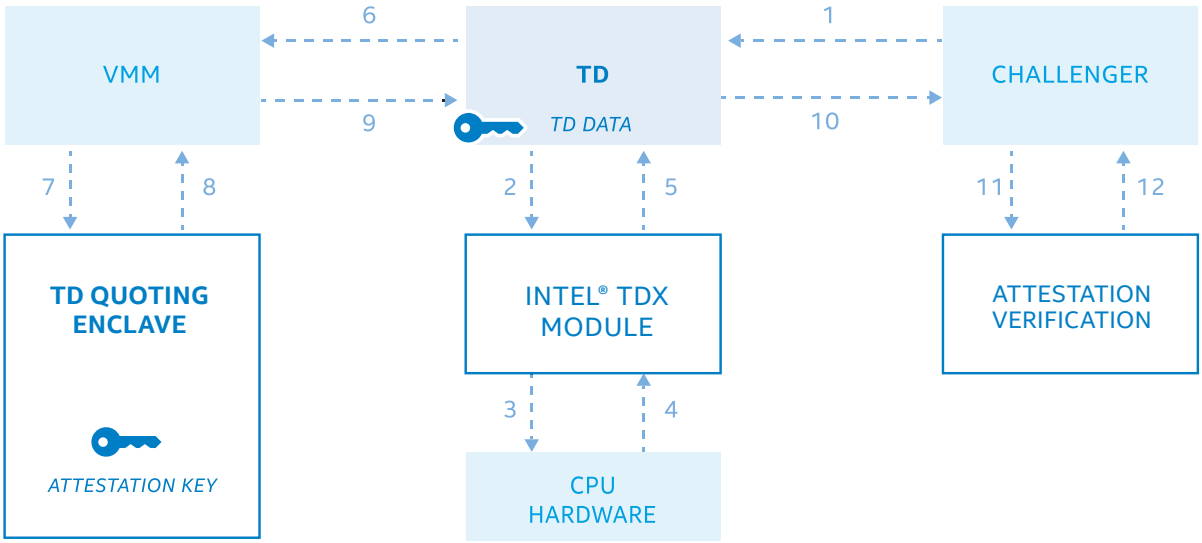
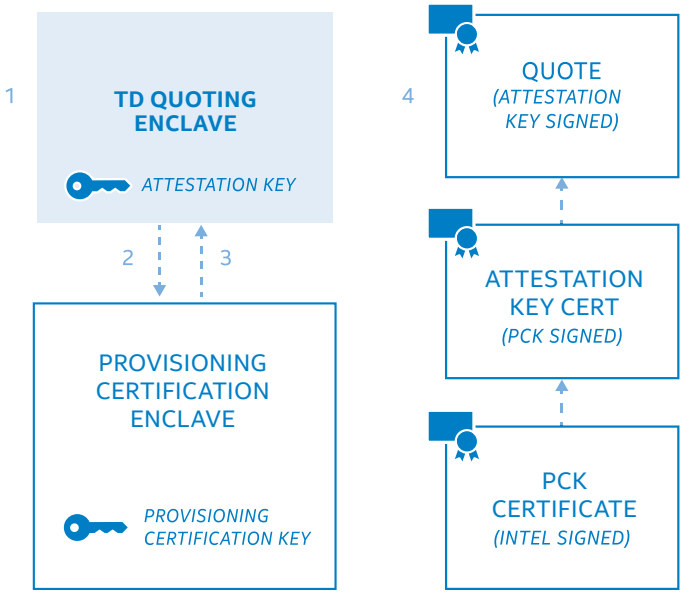


Figure 5.8.

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 from an off-platform challenger (1), the TD would then request the module provide 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 attestation (a Quote) (7, 8, and 9) by the TD-quoting Enclave. The TD-quoting enclave would then verify the MAC on the report using EVERIFYREPORT and convert the report, if verified, into a Quote by signing the report using the TD's asymmetric-attestation key. The Quote would next return to the challenger (10). Finally, the challenger is designed to use an attestation-verification service (11 and 12) to perform quote verification.

To support attestation infrastructure for Intel TDX, Intel has built a general-certification infrastructure, built upon the Intel® Software Guard Extensions Data Center Attestation Primitives (Intel® SGX DCAP), to help certify TD-quoting Enclaves with a certificate chain rooted to an Intel-issued certificate. The foundation of this infrastructure, shown in Fig. 5.9, is an Intel-provided enclave called the Provisioning-Certification Enclave (PCE), which is designed to act as a local, Certificate Authority for local TD-quoting Enclaves (i.e., running on the same platform as each other). The TD-quoting Enclave(s) would then generate their own Attestation Keys using their preferred method and algorithm (1). The TD-quoting Enclave aims to provide the PCE with the attestation-public key (2). The PCE is designed to authenticate the request and issue a certificate-like structure identifying the TD-quoting enclave and the Attestation Key (3). This structure would then be signed by a device and TCB-specific, signing key called the Provisioning-Certification Key (PCK). Intel aims to publish certificates and certificate-revocation lists (CRLs) for the PCKs in all genuine, Intel platforms. This would result in a complete, signature chain from the Quotes to an Intel-certification authority (CA) (4). The resulting Quote would be verified by anyone with the complete, certificate chain and CRLs.

Figure 5.9.



03. Summary

Intel is introducing new, architectural elements to help deploy hardware-isolated VMs called trust domains (TDs):

- 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 to help 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.
- Physical-address-metadata table (PAMT) to help track page allocation, page initialization, and TLB consistency.
- Multi-key, total-memory-encryption (MKTME) engine designed to provide memory encryption using AES-128-XTS and integrity using 28-bit MAC and a TD-ownership bit.
- Remote attestation designed to provide evidence of TD executing on a genuine, Intel-TDX system and its TCB version.

0720/RR/MESH/PDF 343961-001US