Intel® TDX Connect Architecture Specification

354629 001-US

March 2023

Copyright © 2023 Intel Corporation. All rights reserved.

Notices and Disclaimers

Intel Corporation (“Intel”) provides these materials as-is, with no express or implied warranties.

All products, dates, and figures specified are preliminary, based on current expectations, and are subject to change without notice. Intel does not guarantee the availability of these interfaces in any future product. Contact your Intel representative to obtain the latest Intel product specifications and roadmaps.

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

Intel technologies might require enabled hardware, software, or service activation. Some results have been estimated or simulated. Your costs and results might vary.

No product or component can be absolutely secure.

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted that includes the subject matter disclosed herein.

No license (express, implied, by estoppel, or otherwise) to any intellectual-property rights is granted by this document.

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice.

Copies of documents that have an order number and are referenced in this document or other Intel literature may be obtained by calling 1-800-548-4725 or by visiting http://www.intel.com/design/literature.htm.

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

Table of Contents

Contents

Chapter 1: About This Document

1.1. Scope of this Document

This document describes Intel® TDX Connect architecture for CPU host side implementation of TDISP 1.0 (PCI-SIG) standard as extension of Intel VTd and Intel TDX Architectures.

Intel TDX Connect architecture supports TDI assignment of external PCIe devices using SR-IOV (Single Root I/O Virtualization) and Direct Device (DDA) models. This is the first generation of Intel TDX Connect architecture and its scope does not include support for advanced VTd, PCIe or CXL features.

This document is part of the **Intel TDX Connect Architecture Specification Set**, which includes the following documents:

Table 1.1: TDX Connect Architecture Specification Set

Document Name	Reference	Description
Intel® TDX Connect Architecture	[TDX Connect Spec]	System architecture overview and specification for Intel® TDX Connect
Intel® TDX Module 2.0 Module Base Architecture	[TDX Module Base Spec]	TDX module 2.0 base architecture overview and specification
Intel® TDX Module TDX Connect Architecture	[TDX Module TDX Connect Spec]	Architecture overview and specification for TDX module support for Intel® TDX Connect
Intel® TDX Module 2.0 Module ABI Reference	[TDX Module ABI Spec]	Detailed TDX module 2.0 Application Binary Interface (ABI) reference specification, covering the entire TDX module architecture
Intel® TDX Connect TEE-IO provisioning agent (TPA) Architecture	[TPA Spec]	Detailed specification for the Intel® TDX Connect TEE-IO provisioning agent (TPA)
Intel® TDX Connect SW Guide	[TDX Connect Software Guide]	Software guide for enabling VMM and OS with Intel® TDX Connect
Intel® TDX Connect Device Attestation Guide	[TDX Connect Attestation Guide]	An introductory overview of the device attestation for confidential computing and compatible with Intel® TDX Connect
Intel® TDX 2.0 Guest-Hypervisor Communication Interface	[TDX GHCI 2.0]	Specification of the software interface between the Guest OS (Tenant, Service TD VMs, and Architectural TD VMs) and the VMM required for enabling Intel TDX Module 2.0
Intel® TDX Connect TEE-IO Device Guide	[TDX Connect TEE-IO Device Guide]	An introductory overview on how to build TEE-IO device for confidential computing compliant with PCIe TDISP 1.0 and compatible with Intel® TDX Connect

This document is a work in progress and is subject to change based on customer feedback and internal analysis. This document does not imply any product commitment from Intel to anything in terms of features and/or behaviors.

Note: The contents of this document are accurate to the best of Intel's knowledge as of the date of publication, though Intel does not represent that such information will remain as described indefinitely in light of future research and design implementations. Intel does not commit to update this document in real time when such changes occur.

1.2. Document Organization

The document has multiple chapters:

- Chapter 2 contains the Intel® TDX Connect introduction and overview
- Chapter 3 contains the Intel® TDX Connect architecture

1.3. Glossary

Table 1.2: Intel TDX Glossary for TDX Connect

Acronym	Full Name	Description
ACS	**Access Control Service**	PCIe ACS defines a set of control points within a PCI Express topology to determine whether a TLP is to be routed normally, blocked, or redirected. ACS is applicable to RCs, Switches, and Multi-function Devices (See [PCIe Base spec])
ATS	**Address Translation Service**	ATS is a PCIe extended capability which uses a request-completion protocol between a Device and a root complex (RC) to provide translation services. In addition, a new AT field is defined within the memory read and memory write TLP to enable the RC to determine whether a given request has been translated or not via the ATS protocol (See [PCIe Base spec])
CMA/SPDM	**Component Measurement and Authentication**	Component Measurement and Authentication/SPDM (CMA/SPDM) defines optional security features based on the adaptation of the data objects and underlying protocol defined in [SPDM]. These provide mechanisms to perform security exchanges (where this term is used generically to refer to all defined capabilities of [SPDM]) with a component, or device/function (See [PCIe Base spec] and [SPDM Spec])
CPL	**Completion**	Completion PCIe TLP. All read, non-posted write, DMWR, and AtomicOp requests require Completion. Completions include a Completion header that, for some types of Completions, will be followed by some number of DWs of data (See [PCIe Base spec])
DOE	**Data Object Exchange**	Data Object Exchange (DOE) is a PCIe optional mechanism for system firmware/software to perform data object exchanges with a function or RCRB. Software discovers DOE support via the Data Object Exchange (DOE) extended capability structure (See [PCIe Base spec])
DSM	**Device Security Manager**	Device Security Manager (DSM) is a logical entity in the device that may be admitted into the TCB for a TVM by the TSM and enforces security policies on the device (See [TDISP spec])
FLR	**Function Level Reset**	FLR is a PCIe optional mechanism which enables software to quiesce and reset endpoint hardware with function-level granularity (See [PCIe Base spec])
IDE	**Integrity & Data Encryption**	Extended PCIe capability for Integrity & Data Encryption (IDE) for confidentiality, integrity, and replay protection of PCIe transport layer packets (See [PCIe IDE spec])
IDE_KM	**IDE Key Management**	IDE Key Management (IDE_KM) protocol builds upon [SPDM] and [Secured-SPDM], and can be used over multiple transports (See [PCIe IDE spec])
IOMMU	**Input–Output Memory Management Unit**	CPU-RC Input-Output Memory Management Unit (IOMMU) that translates device virtual addresses to physical addresses

Acronym	Full Name	Description
KCBAR	**Key Config Bar**	Intel CPU implementation of PCIe IDE key configuration interface (See [Intel RC IDE Programing Guide])
RC	**Root Complex**	PCIe root of an I/O hierarchy that connects the CPU/memory subsystem to the I/O. RC may support one or more PCI Express Ports. Each interface defines a separate hierarchy domain (See [PCIe IDE spec])
RP	**Root Port**	A PCIe port on a root complex that maps a portion of a hierarchy through an associated virtual PCI-PCI bridge (See [PCIe Base spec])
EP	**End Point**	Function that can be the Requester or Completer of a PCI Express. Endpoints are classified as either legacy, PCI Express, or root complex Integrated Endpoints RCiEPs (See [PCIe Base spec])
SPDM	**Secure Protocol and Data Model**	The Security Protocol and Data Model (SPDM) Specification which defines messages, data objects, and sequences for performing message exchanges between devices over a variety of transport and physical media. (See [SPDM spec])
TCB	**Trusted Computing Base**	A trusted computing base (TCB) is everything in a computing system that provides a secure environment for operations. This includes its hardware, firmware, software, operating system, physical locations, built-in security controls, and prescribed security and safety procedures.
TDI	**TEE Device Interface**	The unit of assignment for an IO-virtualization capable device. For example, a TDI may be an entire device, a non-IOV Function, or a VF (See [TDISP spec])
TEE	**Trusted Execution Environment**	A secure area of a main processor. It guarantees code and data loaded inside to be protected with respect to confidentiality and integrity.
TEE-IO	**Trusted Execution Environment for IO devices**	A conceptual framework for establishing and managing Trusted Execution Environments (TEEs) that include a composition of resources from one or more devices (See [TDISP spec])
TSM	**TEE Security Manager**	The TEE security manager (TSM) is a logical entity in a host that is in the TCB for a TVM and enforces security policies on the host (See [TDISP spec])
TVM	**TEE Virtual Machine**	A Trusted Execution Environment Virtual Machine as defined in the TEE Device Interface Security Protocol (TDISP) reference architecture

1.4. Notation

When specifying requirements or definitions, the level of commitment is specified following the convention of RFC-2119 keywords for use in RFCs to indicate Requirement Levels, as described in the following table:

Table 1.3: Requirement and Definition Commitment Levels

Keyword	Description
Must	"**Must**", "**Required**" or "**Shall**" means that the definition is an absolute requirement of the specification.
Must Not	"**Must Not**" or "**Shall Not**" means that the definition is an absolute prohibition of the specification.
Should	"**Should**", or the adjective "**Recommended**", means that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.
Should Not	"**Should Not**", or the phrase "**Not Recommended**" means that there may exist valid reasons in particular circumstances when a particular behavior is acceptable or even useful, but the full implications should be understood, and the case must be carefully weighed before implementing any behavior described with this label.
May	"**May**", or the adjective "**Optional**", means that an item is discretionary. An implementation may choose to include the item, while another may omit the same item, because of various reasons.

1.5. Specification References

Table 1.4: Specification References

Document Name	Reference	Version & Date
Intel® 64 and IA-32 Architectures Software Developer's Manual Combined Volumes: 1, 2A, 2B, 2C, 2D, 3A, 3B, 3C, 3D, and 4	[Intel SDM]	325462-078US, December 2022
Intel® Virtualization Technology for Directed I/O Architecture Specification	[Intel VTd]	319433-040, June 2020
Root Complex IDE Key Configuration Unit - Software Programming Guide	[Intel RC-IDE Guide]	732838-001, June 2022
PCI Express Base Specification Revision 5.0, Version 1.0	[PCIe Base Spec]	Rev 5.0 Ver 1.0, May 2019
ECN - Data Object Exchange	[DOE Spec]	Rev 1.1, September 2022
TEE Device Interface Security Protocol (TDISP)	[TDISP Spec]	Rev 1.0 July 2022
Integrity and Data Encryption (IDE) – Revision A	[IDE Spec]	Rev A, October 2021
Security Protocol and Data Model (SPDM) Specification	[SPDM Spec]	Ver 1.2.1, June 2022
Secured Messages using SPDM Specification	[Secure SPDM Msg]	Ver 1.1, May 2022

Chapter 2: Introduction

2.1 IO Virtualization

IO virtualization refers to the virtualization and sharing of I/O devices across multiple VMs or container instances. There are multiple existing approaches for IO virtualization that may be broadly classified as either software-based or hardware-assisted.

With software based I/O virtualization, the hypervisor exposes a virtual device, such as a Network Inter- face Controller (NIC), to a VM. A software device model in the hypervisor or host OS emulates the behavior of the virtual device. The device model translates from virtual device commands to physical device commands before forwarding to the physical device.

Modern processors provide features to reduce virtualization overhead that may be utilized by VMMs to allow VMs direct access to hardware resources.

This includes capabilities for direct memory access (DMA) and interrupt remapping and isolation that can be utilized to minimize the overheads of IO virtualization.

Specifically, Intel supports the following hardware assisted IO virtualization schemes for direct data movement without needing software assistance:

- Direct Device Assignment: Assignment of entire device to a VM
- Single Root I/O virtualization (SR-IOV): Assignment of a device virtualized function.
- Scalable I/O virtualization (S-IOV): Assignment of low-level device interfaces composed by the VMM virtualize a device.

2.1. IO Virtualization with TDX vs. TDX Connect

Base Intel® TDX hardware prohibits devices from directly accessing TD private memory. Therefore, only software-based IO virtualization is supported. The untrusted hypervisor exposes a virtual device, such as a Network Interface Controller (NIC), to a TD using shared (untrusted) synthetic IO and para-virtualized device interfaces managed by the VMM (See [TDX Module Base Spec] "I/O support" section).

The software-based IO model is slow because the communication between the TD and the device is done through shared memory bounce-buffers which require the TD guest to copy and encrypt back and forth the data between the private memory buffers of applications running inside the TD and the shared IO buffer used by the device.

For some IO use cases, such as networking and storage, TD may employ software based cryptographic techniques for data protection, however this approach suffers from performance overhead vs. VTd direct IO virtualization low latency and high throughout. Besides the performance overhead, the cryptographic data protection does not allow the TD to offload computation to legacy GPU or FPGA accelerators and requires them to be in the TCB of the TD and properly protect its secret data.

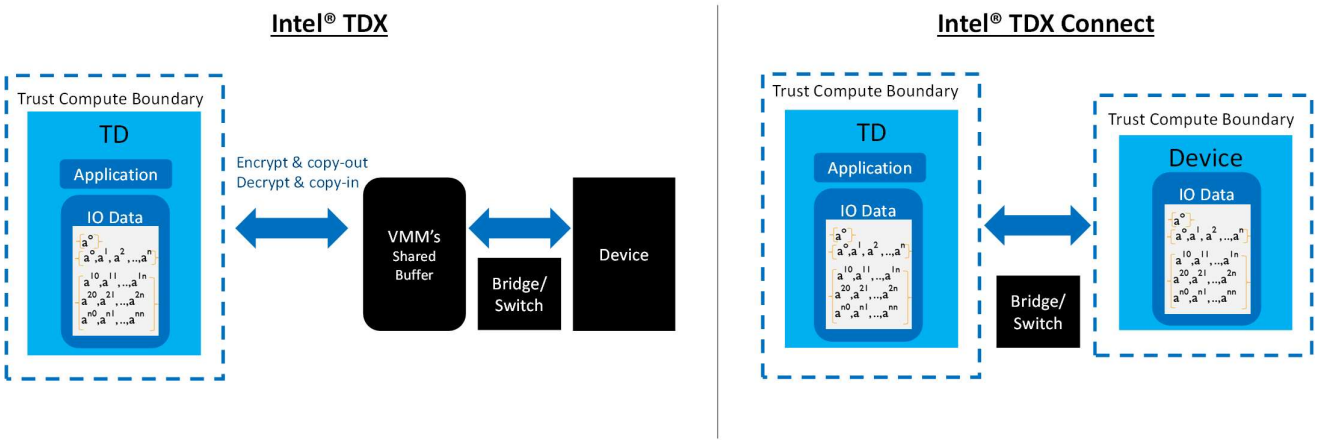

Figure 2.1: IO Virtualization with Intel® TDX vs. Intel® TDX Connect

Intel® TDX Connect is designed to improve IO virtualization for a TD TEE in two aspects:

- **Functionality**: Remove the need for TD and devices to use shared buffer for private data including the need to establish a secure transport-level session with the device (typically done using a proprietary protocol to adjust specific device data processing and transformation needs).
- **Performance**: Remove the additional resources and work needed for the data copy-encrypt or copy-decrypt back and forth between the shared TD-Device buffer and the private TD memory greatly improving the workload performance (with respect to bandwidth and latency).

Intel TDX Connect architecture introduces trusted device assignment to a TD, extending its TD Trusted Compute Base (TCB) and to a TEE-IO Device Interface (TDI) while protecting its data in:

- **CPU**: Extending Intel TDX hardware with private MMIO and DMA access control and data isolation
- **Transport**: End to end data protection using PCIe selective IDE streams
- **Device:** Extending the TDX module with SPDM, IDE and TDISP support allowing TDs to extend their TEE and TCB only to TEE-IO devices they choose to trust

2.2. System Overview

The host platform hardware may include one or more sockets each with multiple-cores, memory controllers and IO hubs. The cores and the IO hubs share a coherent system cache and fabric which is connected to the system memory with memory controllers implementing Intel® Multi-Key TME (TME-MK) encryption engine.

Each IO hub includes an IOMMU and related logic for connecting devices to the coherent fabric within the SOC. A subset of the IO hubs may include PCIe root-complexes and that are used to connect discrete PCIe devices, while others may only support within-package integrated devices.

The integrated devices are not included in the Intel TDX Connect scope while (a subset of) the discrete PCIe root-complex IO hubs have Intel TDX Connect support with bifurcated RPs that may support PCIe IDE extended capability.

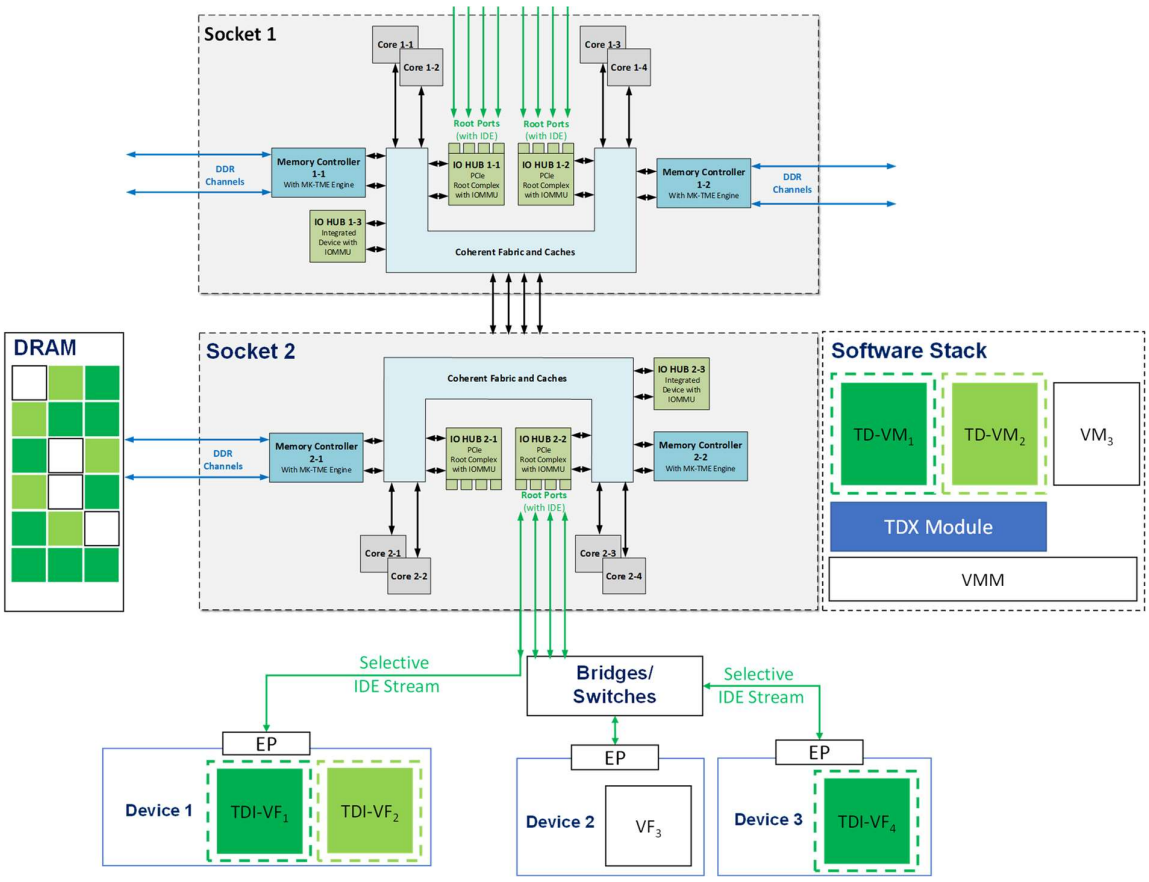

Figure 2.2: TDX Connect System Model

Each PCIe root complex may be connected using bifurcated PCIe link downstream root port to the endpoint device directly or via any topology of bridges and switches.

The devices under a PCIe root-complex could include TDISP-compliant and/or regular PCIe devices. A TDISP-compliant device could host one or more TEE-IO Device Interfaces (TDI) which could be assigned to a single TD at any point-in time a time or alternatively, as regular device interface (e.g., VF) which can be assigned to TDs (via shared memory) or to regular VMs.

The platform system software resembles that of a platform with TDX support. It includes a hypervisor that may host one or more VMs and may include an installed TDX Module with Intel TDX Connect support.

2.3. Security Model

Intel TDX Connect Security Model consists of the following key concepts:

1. Only the TD owner can decide which TEE-IO device interface (TDI) is trustworthy
2. TDI may use DMA to access TD private memory only allowed if the TD explicitly allowed it and only while its exclusively assigned to that TD
3. TD may use trusted access (TEE-TLP) to TDI MMIO space only if the TD is the current owner of that TDI.

The Intel TDX Connect trust model requires each TD to explicitly accept a device into its trust boundary. Device being trusted by one TD does not imply that it is in trust boundary of other TDs that have not accepted the device into their trust boundary.

According to TDISP, such a device is in the trust boundary of all TDs that have accepted it into their trust boundary, and it must maintain per-TD isolation per TDI (e.g., VF).

A single TDI cannot be shared between TDs, however, a multi-function device (supports multiple TDIs) can be trusted by multiple TD-VMs and use a single IDE selective stream to secure the data-path between the CPU host and the TEE-IO endpoint device.

Intel TDX Connect TCB does not include switches and bridges, therefore it is required by the host VMM to setup selective IDE stream to guarantee end-to-end IDE protection between TD-VM running on the host CPU, and the TDI running on the TEE-IO device.

The below diagram illustrates the Intel TDX Connect trust relationship in which:

- The TDX Connect host platform and TSM are trusted by all TD-VMs
- Device-1 is trusted by both TD-VMs since TDI-VF$_1$ and TDI-VF$_2$ are assigned to TD-VM$_1$ and TD-VM$_2$ respectively
- Device-2 is not trusted by any TD-VM because VF$_3$ not assigned to any TDI-VM
- Device-3 is only trusted by TD-VM$_1$ because TDI-VF$_4$ is assigned to TD-VM$_1$
- Selective IDE stream used to protect the interconnect between the host CPU and the device, the bridges and switches use IDE pass-through mode and are not assumed to be trusted by any TD-VM

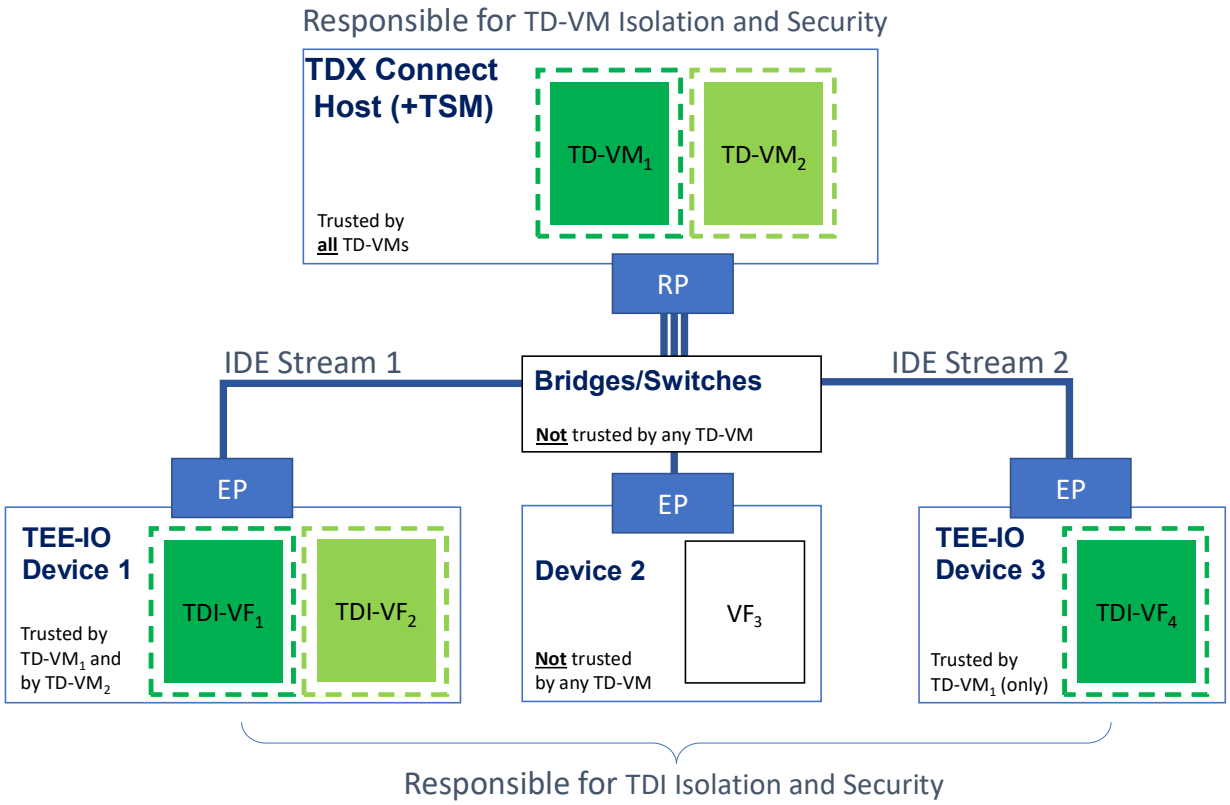

Figure 2.3: TDX Connect Security Model

2.4. Capabilities and Features

The following table describes the capabilities and features in scope of Intel TDX Connect architecture.

Table 2-1: Intel TDX Connect Supported Capabilities and Features

<table>
<tr><th>Focus Area</th><th>Feature or Capability</th><th>Intel TDX Connect Support</th><th>Details</th></tr>
<tr><td rowspan="2">PCIe IDE</td><td>Link IDE stream</td><td>No</td><td>Using Link IDE streams is enabled for legacy devices but not for TEE-IO devices</td></tr>
<tr><td>Selective IDE stream</td><td>Yes</td><td>Up to 4 IDE stream per root complex with number IDE stream register blocks per RP depends on per RP bifurcation.
<table>
<tr><td>Bifurcation</td><td>Selective</td><td>Link</td></tr>
<tr><td>1x16</td><td>4</td><td>1</td></tr>
<tr><td>2x8</td><td>3</td><td>1</td></tr>
<tr><td>4x4</td><td>1</td><td>1</td></tr>
<tr><td>8x2</td><td>N/A</td><td>N/A</td></tr>
<tr><td>16x1</td><td>N/A</td><td>N/A</td></tr>
</table></td></tr>
<tr><td rowspan="2">Trusted MMIO</td><td>TD to TEE-IO device</td><td>Limited</td><td>Limited to MMIO high (above 4GB)</td></tr>
<tr><td>Peer-to-Peer</td><td>Limited</td><td>Only via-host IOMMU and root complex. Direct access via bridges and switches depends on Intel TDX Connect ATS support (see below)</td></tr>
<tr><td>Trusted DMA</td><td>TEE-IO Device to TD Memory</td><td>Yes</td><td>Device may access TD private or shared memory</td></tr>
<tr><td>ATS</td><td>Trusted Address Translation and Translated Requests</td><td>No</td><td>Intel TDX Connect threat model requires Secure-ATS to ensure translated requests can only access system memory of the TDs who accepted them into their TCB. This is a future extension for Intel TDX Connect</td></tr>
<tr><td>Nested IOMMU Translations</td><td>Support for TDs to setup first level page tables to enable DMA access control or SVM</td><td>No</td><td></td></tr>
<tr><td>PF, FV Assignment</td><td>Assignment of PF and VF (SR-IOV)</td><td>Yes</td><td></td></tr>
<tr><td>SVM</td><td>Trusted PRS Support</td><td>No</td><td></td></tr>
<tr><td>Scalable IOV Support</td><td>Assignment of Scalable IOV R2 Device Support</td><td>No</td><td></td></tr>
<tr><td>CXL Type 1,2</td><td>Support for CXL devices participating host memory coherency</td><td>No</td><td>TEE-IO standard and requirements for CXL Type 1 and 2 devices are immature. This may be a future Intel TDX Connect extension.</td></tr>
<tr><td>CXL Type 3</td><td>Support for CXL memory buffer devices</td><td>N/A</td><td>This feature is out of scope for this Intel TDX Connect specification and will be described in other TDX future specifications.</td></tr>
</table>

2.4.1. TEE-IO Transactions and Requirements

The table below refers to transaction and configuration requirements with respect to TEE-IO transactions from TDs and to TDI private MMIO, from TDI to its TD owner private memory or P2P between TDI to another TDI or non-TD device assigned to a TD.

Table 2-2: Trusted Transactions and Requirements

	Requirement	Requirement Details
[TXN-1]	TEE-IO Enabling	Intel TDX Connect architecture must ensure TEE-IO transactions are only allowed if IOMMU has been enabled for Intel TDX Connect.
[TXN-2]	TEE-IO MMIO Decode Consistency	Intel TDX Connect architecture must check prior to Intel TDX Connect enabling per PCIe root complex (and IOMMU) that the untrusted platform software configuration of MMIO decoding is locked (cannot be modified) and consistent (source to target routing is configured in all SOC registers to prevent misrouting of TEE-IO transactions).
[TXN-3]	TEE-IO DMA Decode Consistency	Intel TDX Connect architecture must check prior to Intel TDX Connect enabling per PCIe root complex (and IOMMU) that the untrusted platform software configuration of system memory decoding is locked (cannot be modified) and consistent (source to target routing is configured in all SOC registers to prevent misrouting of TEE-IO transactions).
[TXN-4]	TEE-IO IDE and SPDM	Intel TDX Connect architecture must ensure TEE-IO transactions are only allowed via root complex, and root ports are enabled and configured with IDE selective streams using IDE_KM over SPDM session established with TEE-IO devices by the Intel TDX Connect TSM (TDX Module and TPA).
[TXN-5]	TEE-IO MMIO Source	Intel TDX Connect may only allow TEE-IO transactions to MMIO from a TD or TDI (P2P) using private GPA mapped in the TD Secure-EPT.
[TXN-6]	TEE-IO MMIO Range	Host platform software may only configure TEE-IO MMIO routing via selective IDE stream using Intel TDX Connect ABI. These IDE streams may only be configured with address association registers that enclose the associated RP prefetchable memory range and may only select address ranges above 4GB.
[TXN-7]	TEE-IO MMIO Target	TDSIP defines DSM rules with respect to device TEE/Non-TEE memory access. Intel TDX Connect may not enforce any of device TEE vs non-TEE memory access control.
[TXN-8]	TEE-IO DMA Source	Intel TDX Connect architecture must only allow TEE-IO transactions to TD private memory if and only if the TD explicitly accepted the TDI into its TCB (using TDISP protocol). The TDI may not access any other TD private memory space until properly stopped and reclaimed from its previous TD owner.
[TXN-9]	TEE-IO DMA Range and Target	IOMMU GPA mapping of TDI to TD private memory space must be consistent with the TD GPA to HPA mapping and permissions. Intel TDX Connect may not support finer control over TDI access permissions to TD on the current architecture (e.g., by allowing TD to configure IOVA to GVA translation tables).
[TXN-10]	TEE-IO MSI/X	Intel TDX Connect may not support MSI/X TEE-IO transactions
[TXN-11]	TEE-IO P2P via root complex	Intel TDX Connect architecture must allow P2P TEE-IO transactions only via the root complex and only with non-translated TLP.
[TXN-12]	TEE-IO P2P Direct	Intel TDX Connect architecture may not support direct P2P between TEE-IO devices and may not allow TDs to generate TDISP P2P configuration messages.

	Requirement	Requirement Details
[TXN-13]	TEE-IO ATS	Intel TDX Connect may not support TEE-IO transactions related to PCIe ATS: - Translation and Device TLB Request and Completions - Translated TEE-IO Request
[TXN-14]	Non-TEE-IO Transactions	Intel TDX Connect support must support non-TEE transactions and must guarantee such transactions would never become a TEE-IO transaction or system memory with TDX private memory semantics.
[TXN-15]	Implicit Non-TEE-IO Transactions	Intel TDX Connect architecture must guarantee that any access with shared semantics (HPA access not using TDX private KeyID) will result with non-TEE transaction.
[TXN-16]	Device P2P TEE-IO Access	Intel TDX Connect architecture may allow TEE-IO device to perform trusted P2P accesses only if request is non-translated, requestor and the responder are both TDI assigned and accepted into the TCB of the same TD owner and the address on the target TDI is a private HPA (using TDX private KeyID).

2.4.1. Standard and Device Interoperability

The DOE support below refers to the message transport format of messages generated by the TDX module/TPA (i.e., for SPDM, IDE_KM and TDISP protocols). Intel TDX Connect host platform does not expose a DOE mailbox (which is required by a TEE-IO device but not by the TEE host platform).

5

Table 2-3: DOE Support and Requirements

<table>
<tr><th></th><th>Requirement</th><th>Details</th></tr>
<tr><td>[DOE-1]</td><td>Supported DOE versions</td><td>Intel TDX Connect Architecture supports DOE 1.0 with additional enhancements for Data Object types added to DOE 1.1.</td></tr>
<tr><td>[DOE-2]</td><td>Supported DOE object types</td><td><table><tr><td>Vendor-ID</td><td>Data Object Type</td></tr><tr><td>0001</td><td>01 – CMA/SPDM</td></tr><tr><td>0001</td><td>02 – Secure CMA/SPDM</td></tr></table></td></tr>
<tr><td>[DOE-3]</td><td>Non-Supported DOE object types</td><td><table><tr><td>Vendor-ID</td><td>Data Object Type</td></tr><tr><td>0001</td><td>00 – DOE Discovery (VMM/OS DOE discovery is a device scope feature)</td></tr><tr><td>0001</td><td>03 – CMA/SPDM with Connection ID</td></tr><tr><td>0001</td><td>04 – Secure CMA/SPDM with Connection ID</td></tr><tr><td>0001</td><td>05 – Async Message</td></tr></table></td></tr>
</table>

Table 2-4: SPDM Support and Requirements

<table>
<tr><th></th><th>Requirement</th><th>Details</th></tr>
<tr><td>[SPDM-1]</td><td>Supported SPDM version</td><td>Intel TDX Connect supports SPDM version 1.2</td></tr>
<tr><td>[SPDM-2]</td><td>Support for SPDM session establishment, management, and teardown with TDISP-compliant devices</td><td>It must be feasible to establish, use and teardown SPDM sessions with TDISP-capable devices for use by TDs.</td></tr>
<tr><td>[SPDM-3]</td><td>Supported granularities for SPDM session establishment for TDIs</td><td>SPDM does not mandate a specific device granularity for session establishment. For example, all the following configurations are allowed depending on device support for the DOE mailbox:
1 device -> N function -> 1 DOE (in function 0 only) -> 1 SPDM.
1 device -> N function -> N DOE -> N SPDM
1 device -> N function -> N DOE -> 1 SPDM (other N-1 DOE is used for other purpose)
1 device -> N function -> N*M DOE (each function has M DOE) -> N SPDM</td></tr>
<tr><td>[SPDM-4]</td><td>Number of SPDM sessions per TDISP compliant device/ device interface</td><td>This is not mandated by the specification and is dependent on the device. For example, the following is allowed:
1 DOE -> 1 SPDM
1 DOE -> N SPDM (with DOE connection ID support in DOE 1.1)</td></tr>
<tr><td>[SPDM-5]</td><td>Number of SPDM sessions per SoC</td><td>256 per IO stack</td></tr>
<tr><td>[SPDM-6]</td><td>Number of outstanding SPDM requests</td><td>Intel TDX Connect architecture shall support unlimited number of outstanding secure SPDM message requests (per SPDM session).</td></tr>
<tr><td>[SPDM-7]</td><td>List of SPDM messages supported</td><td>Intel TDX Connect supports the following SPDM protocol request and response messages:
<table>
<tr><th>SPDM Request</th><th>SPDM Response</th></tr>
<tr><td>GET_VERSION</td><td>VERSION</td></tr>
<tr><td>GET_CAPABILITIES</td><td>CAPABILITIES</td></tr>
<tr><td>NEGOTIATE_ALGORITHMS</td><td>ALGORITHMES</td></tr>
<tr><td>GET_DIGESTS</td><td>DIGESTS</td></tr>
<tr><td>GET_CERTIFICATE</td><td>CERTIFICATE</td></tr>
<tr><td>GET_MEASUREMENTS</td><td>MEASUREMENTS</td></tr>
<tr><td>KEY_EXCHANGE</td><td>KEY_EXCHANGE_RSP</td></tr>
<tr><td>FINISH</td><td>FINISH_RSP</td></tr>
<tr><td>END_SESSION</td><td>END_SESSION_ACK</td></tr>
</table></td></tr>
<tr><td>[SPDM-8]</td><td>SPDM algorithms</td><td>Intel TDX Connect supports the SPDM algorithms defined in CMA 1.0 and CMA 1.1.</td></tr>
</table>

	Requirement	Details
[SPDM-9]	List of SPDM messages NOT supported	Intel TDX Connect does not require any other messages defined in SPDM specification.

Table 2-5: IDE Support and Requirements

	Requirement	Details
[IDE-1]	IDE Version Support	Intel TDX Connect supports PCI-SIG, IDE Rev A
[IDE-2]	IDE Host Enumeration	Platform software must enumerate IDE support on both host and endpoint device to determine the common IDE supported features for both Intel TDX Connect platform and TDISP device. Intel RC IDE is compatible with TEE-IO but does not support the TEE-IO enumeration via IDE-ECAP. Instead, TDX Connect (and host TEE-IO support) shall be enumerated using root complex and IDE discovery mechanism (see [Intel RC-IDE Guide]) and TDX module extended feature enumeration (see [TDX Module ABI Spec]).
[IDE-3]	Support for Selective and Link IDE streams	Intel TDX Connect CPU implements 4 IDE streams on each x16 PCIe hierarchy. Number of Link and selective IDE stream registers per RP depends on per RP bifurcation: 1X16 – 1 link IDE, 4 selective IDE configuration registers per port 2x8 – 1 link IDE, 3 selective IDE configuration registers per port 4x4 – 1 link IDE, 1 selective IDE configuration registers per port 8x2 – No link IDE stream, and no selective IDE configuration registers (not usable for Intel TDX Connect)
[IDE-4]	IDE selective stream	For each IDE selective stream, there is one Address and one RID Association Register set per IDE stream. For correct binding of selective IDE streams between host and endpoint device, the platform software must program same stream IDs with equal RID and Address association ranges and symmetric keys using IDE KM protocol for all sub stream before setting the IDE enable control.
[IDE-5]	Intel TDX Connect Enable/Disable and Host IDE Register Protection	To enable Intel TDX Connect, host platform must first enable TDX mode per PCIe hierarchy and each enabled RP using Intel TDX Connect ABI. Once Intel TDX Connect is enabled, IDE-ECAP and key programming registers must prevent write access to platform software and may only allow write access to TDX module (SEAM root mode). This protection may only be disabled by a platform reset or by platform software using Intel TDX Connect ABI (see [Intel TDX Connect Module Spec])
[IDE-6]	IDE Management and Binding with SPDM Session	Once Intel TDX Connect is enabled, the host platform software must use Intel TDX Connect ABI for setting up Link or selective IDE stream and configuring their keys using IDE KM protocol. Before IDE stream is created and managed using Intel TDX Connect ABI, the platform software must establish the SPDM session with an IDE device using Intel TDX Connect ABI (see [Intel TDX Connect Module Spec])
[IDE-7]	IDE stream Setup for TDISP vs. non-TDISP Device	Host platform must use selective IDE for TDISP device and may use either Link or selective IDE stream for non TDISP device

<table>
<tr><td>[IDE-8]</td><td>IDE_KM Protocol Support</td><td>Intel TDX Connect ABI interface for IDE_KM supports the following response messages:
<table>
<tr><th>IDE_KM Request</th><th>IDE_KM Response</th></tr>
<tr><td>KEY_PROG</td><td>KP_ACK</td></tr>
<tr><td>K_SET_GO</td><td rowspan="2">K_GOSTOP_ACK</td></tr>
<tr><td>K_SET_STOP</td></tr>
</table></td></tr>
<tr><td>[IDE-9]</td><td>Selective IDE stream Control</td><td>Selective IDE stream control address association registers must not be programmed to include addresses below 4GB (MMIOL)</td></tr>
<tr><td>[IDE-10]</td><td>IDE TLP reserved bit checking</td><td>RP-IDE hardware may not implement reserved bit checking for IDE TLP prefix bit 7</td></tr>
</table>

Table 2-6: TDISP Support and Requirements

<table>
<tr><th></th><th>Requirement</th><th>Details</th></tr>
<tr><td>[DISP-1]</td><td>Supported TDISP version</td><td>Intel TDX Connect supports TDISP v1.0</td></tr>
<tr><td>[DISP-2]</td><td>Supported TDISP Device Interface Modes</td><td>Intel TDX Connect Architecture support full device (PF) and or virtual function (VF) assignment to a TD</td></tr>
<tr><td>[DISP-3]</td><td>TEE-IO Device Interface Assignment life cycle management</td><td>Platform software must enable TEE-IO device interface (TDI) assignment to TD trust boundary using Intel TDX Connect ABI. TDISP devices are first assigned to TDs at PENDING state. Meaning, TD, and devices cannot generate trusted TLP transactions till this assignment is made PRESENT by the TD explicitly calling TDX- guest ABIs.
For more details refer to [Intel TDX Connect Module Spec].</td></tr>
<tr><td>[DISP-4]</td><td>TEE-IO Device Interface Removal by Host Platform (Graceful)</td><td>To remove a TDI from a TD and before it can be assigned to another TD, the platform software must use Intel TDX Connect ABI to disable the TDI device associated selective IDE stream or to transition its TDISP state to CONFIG_UNLOCKED. For more details refer to [Intel TDX Connect Module Spec].</td></tr>
<tr><td>[DISP-5]</td><td>List of TDISP protocol messages supported</td><td>Intel TDX Connect supports the following TDISP requests and response messages:
<table>
<tr><th>TDISP Request</th><th>TDISP Response</th></tr>
<tr><td>GET_TDISP_VERSION</td><td>TDISP_VERSION</td></tr>
<tr><td>GET_TDISP_CAPABILITIES</td><td>TDISP_CAPABILITIES</td></tr>
<tr><td>LOCK_INTERFACE_REQUEST</td><td>LOCK_INTERFACE_RESPONSE</td></tr>
<tr><td>GET_DEVICE_INTERFACE_REPORT</td><td>INTERFACE_REPORT</td></tr>
<tr><td>GET_DEVICE_INTERFACE_STATE</td><td>GET_DEVICE_INTERFACE_STATE</td></tr>
<tr><td>START_INTERFACE_REQUEST</td><td>START_INTERFACE_RESPONSE</td></tr>
<tr><td>STOP_INTERFACE_REQUEST</td><td>STOP_INTERFACE_RESPONSE</td></tr>
</table>
Other message types are currently not supported.</td></tr>
<tr><td>[DISP-6]</td><td>TEE-IO Enumeration</td><td>Before using Intel TDX Connect ABI, platform software must use TDX ABI to enumerate Intel TDX Connect support.
For more details refer to [Intel TDX Connect Module Spec].</td></tr>
</table>

2.4.2. Device (DSM) and Host (TSM) Requirements

Table 2-7: DSM and TEE-IO Device Requirements

Intel TDX Connect Device Requirements		
	Requirement	**Requirement Details**
[DSM-1]	TEE MMIO Ranges	Intel TDX Connect architecture does not support TEE-IO transactions to MMIO low (< 4GM) or Configuration Space. Device must expose all TEE MMIO resources using 64 BARs. IDE selective stream bound to TDI and TDI TEE MMIO ranges must be configured to MMIO high ranges.
[DSM-2]	Device Address Width	Devices address width (also reported by TDISP capabilities) must be at least 52 bits
[DSM-3]	TEE-IO P2P	Device may send and handle trusted P2P TLPs only via the root complex IOMMU. Intel TDX Connect CPU host does not support setting P2P IDE selective streams required for direct P2P via switches (and via the root complex).
[DSM-4]	TEE-IO PASID	Device must not use PASID for TEE-IO transactions. During TDI configuration it is recommended to disable PASID.
[DSM-5]	TEE-IO ATS	Device must not use ATS requests for TEE-IO transactions. During TDI configuration it is recommended to disable ATS.
[DSM-6]	TEE-IO PRS	Device must not use PRS requests for TEE-IO transactions. During TDI configuration it is recommended to disable PRS.
[DSM-7]	TEE-IO CXL.mem/cache	Device must not use CXL.mem/cache for TEE-IO transactions. During TEE-IO enabling or TDI configuration it is recommended by the device to downgrade the device to use CXL.io TEE-IO transactions or support PCIe only mode.
[DSM-8]	TEE-IO MSI/X	Device must not use MSI/X requests for TEE-IO transactions. Device configuration logic must not set the MSI/X locking flag as part as device interface TDISP lock request.

Table 2-8: TSM (TDX Module and TPA Requirements)

	Requirement	**Details**
[TSM-1]	Intel TDX Connect Enumeration	TDX module must expose platform Intel TDX Connect feature support per the entire platform and per specific IOMMU (PCIe hierarchy)
[TSM-2]	Intel TDX Connect Mode	TDX module must enable VMM dynamic enable/disable Intel TDX Connect mode on each root complex instance which has Intel TDX Connect support and was configured by BIOS to enable Intel TDX Connect
[TSM-3]	IOMMU, PCIe Configuration Lock	When Intel TDX Connect mode is enabled, TDX module must also enable hardware access controls to ensure IOMMU and PCIe registers relevant to Intel TDX Connect security cannot be tampered with by untrusted software, firmware, and hardware (e.g., devices).
[TSM-4]	SPDM Setup	TPA TD and TDX Module must support SPDM session establishment, maintenance, and teardown.
[TSM-5]	Attestation	TPA TD and TDX Module must support SPDM gathering for device certificates, measurements, and policy information. The device information can be transported to TVM via untrusted channel, however, the TDX module and the TPA must provide means for the TVM to validate such device information integrity and freshness.

[TSM-6]	IDE Setup	TDX module must support VMM ability to setup selective (or Link for non-TIDSP, IDE only cases) streams with a device using IDE_KM and Secure SPDM transport. The Secure SPDM transport must use TPA TD established SPDM Session.
[TSM-7]	TDI (FUNCTION_ID) Assignment	TDX module must support TDI Assignment to TD ensuring each FUNCTION_ID may only be assigned to a single TD at a time. TDX module must ensure such TDI must is associated with a selective IDE.
[TSM-8]	TDISP Life Cycle Support	TDX module must provide VMM and TD interface to manage TDI assignment according to TDISP protocol. TDX module must restrict VMM to only generate TDI lock, stop and get interface state messages. TDX module must provide TD interface to manage TDI assignment according to TDISP protocol. TDX module must restrict TD to only generate TDI get interface report, get interface state, and start interface messages. TDX module may not support any non-mandatory TDISP protocol message type.
[TSM-9]	Private MMIO Management	TDX module must track each private MMIO, can be assigned at most to a single TDI (FUNCTION_ID), and must provide VMM API for assigning private MMIO pages to TDI and mapping them to TDs as PENDING. TDX module must provide TD interface to verify and enable such MMIO mappings (TVM must call TDX module API to accept each MMIO page which appears as TEE MMIO range in the device TDISP report).
[TSM-10]	Private DMA Management	TDX module must provide trusted DMA mapping management interfaces to VMM to manage TEE-IO DMA translation to TD private memory. Such VMM interfaces must be mapped as PENDING. TDX module must provide TD interface to verify and enable such DMA mappings (TVM must call TDX module API to accept each DMA mapping per FUNCTION_ID as it appears in TDISP report message.
[TSM-11]	TDI Teardown	TDX Module must enable VMM initiated teardown of TDI bindings from a TD. TDX module must support graceful teardown in which TDI is first stopped and then removed from a TD. For survivability, in case TD or device are not operational, TDX module must provide mechanism to teardown TDI resources (e.g., MMIO, DMA mappings) from a TD by first disabling the TDISP bound IDE selective stream and then reclaiming all TDI resources using the same interfaces used for graceful tear down.
[TSM-12]	TD GPA.S bit position	Intel TDX Connect may not allow VMM to configure the GPA.S bit position to be other than 52 (MAX_PA).
[TSM-13]	Support for TD Live Migration	Intel TDX Connect supports live TD migration co-existence as long as TDI are removed before TD live migration begins and cannot be assigned during TD live migration.
[TSM-14]	Support for TDX Module Seamless Update	Intel TDX Connect supports seamless (impact less) TDX module update. During TDX module update TD may not be operational however device DMA to TD private memory is possible.
[TSM-15]	Support for TD Partitioning	Intel TDX Connect supports TD partitioning such that TDI assignment from VMM perspective is done to L1 VMM. Intel TDX Connect must support L1 TD ability to accept the TDI assignment to its own context to the context of one of the L2 VMs.

Chapter 3: Architecture Overview

Intel TDX Connect framework is the Intel implementation of TDISP allowing direct assignment through establishment of trust between a TD and a TDI hosted by a TDISP compatible device.

The Intel TDX Connect framework goals are to enable the following:

- Establishing trust relationship between a TD and a TDISP device.
- Securing the data-path PCIe interconnect between the host and device.
- Supporting TDISP assignment and removal life cycle in a trusted manner.

Specifically, this framework defines the following ingredients:

1. TEE-IO Capable device and DSM (see [TDISP Spec])
2. PCIe Integrity and Data Encryption (see [IDE Spec])
3. CPU hardware access controls to protect and isolate TEE-IO transactions within the host SOC
4. TDX module extension to support secure TEE-IO Device Setup and direct TDI assignment to TD

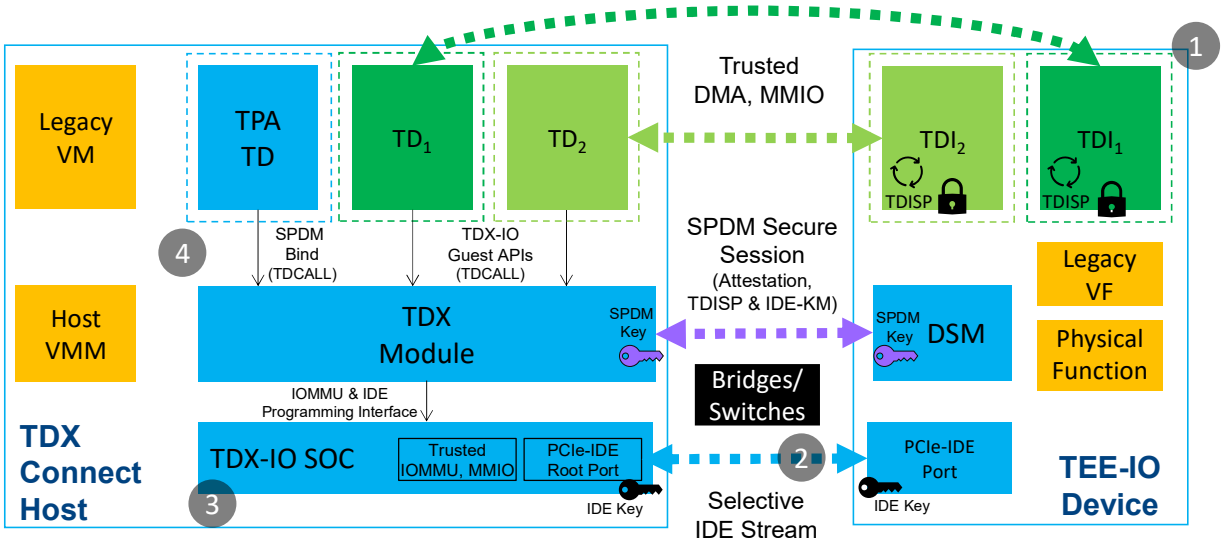

Figure 3.1: Intel TDX Connect Framework Ingredients

3.1. *TEE-IO Capable Device and DSM*

TEE-IO capable devices, as defined in the TEE Device Interface Security Protocol (TDISP) specification, must provide the following capabilities:

1. **Authentication, Attestation, and Key Negotiation:** Implementation of the DMTF SPDM 1.2 specification for runtime authentication, firmware and configuration measurement reporting, and session key negotiation.
2. **Integrity and Data Encryption (IDE):** The Intel TDX Connect-capable devices support the PCIe/CXL IDE extension to PCIe to provide confidentiality, integrity, and replay protection of data transferred to or from the device.
3. **Secure Life-Cycle Management:** Implement TDISP state machine for tracking and enforcing device interface secure life cycle of configuration, lock, reporting and run enabling secure TDI to TD assignment and removal according to TEE Device Interface Security Protocol (TDISP) specification.
4. **Data Security:** Support intra-device resource access control with the same quality of isolation and security properties as the host for data provided to the device by the TD.

The authenticity of a device is determined by digital signatures using well-established techniques based on public key cryptography. A device proves its identity by generating digital signatures using a private key. The TD can cryptographically verify the signatures using the public key associated with that private key. The private key used by the device to prove its identity and authenticity is provisioned into the device by its vendors during or after hardware manufacturing.

A trusted root certificate authority (CA) generates a root certificate (Root-Cert) provisioned to the TD. This allows the TD to verify the validity of the digital signatures generated by the device during runtime authentication. The root CA also endorses a per-part public/private key pair indirectly through the certificate chain, where the private key is provisioned

to or generated by the device. A device carries a certificate chain with the root being the Root-Cert and the leaf being the device certificate (Device-Cert), which contains the public key corresponding to its private key.

At run time, a TD can retrieve the certificate chain(s) from the device and send a unique challenge to the device. The device then signs the challenge with the private key. The TD verifies the signature using the public key of the device as well as any intermediate public keys within the certificate chain using the root certificate as the trusted anchor.

TEE-IO capable devices also implement measurement registers which hold the cryptographic hash value of the firmware/software or configuration data of the device. In response to a request from the TD, the device provides its measurements signed with the private key. This allows the TD to establish the identity and measurements of the firmware/software/configurations of the device.

For more details refer to [TDISP Spec]

3.2. *Integrity and Data Encryption*

Intel TDX Connect requires the establishment of a secure connection with a device such that:

1. Data that flows between the TD and a trusted device must remain confidential, have integrity, and be replay-protected, using the keys derived by the TSM.
2. Metadata associated with transfers (e.g., logical addresses of DMA transfers in the TD's address space) must be integrity and replay protected.

PCIe Integrity & Data Encryption (IDE) specification provides confidentiality, integrity, and replay protection for TLPs transmitted and received between two Ports. It flexibly supports a variety of use models, while providing broad interoperability. The cryptographic mechanisms used by IDE are aligned with industry best practices and can be extended as security requirements evolve. The security model considers threats from physical attacks on links, including cases where an adversary can examine data intended to be confidential, modify TLP contents, reorder and/or delete TLPs, using lab equipment, purpose-built interposers, or malicious extension devices. TLPs can be protected as they transit switches, extending the security model to address attacks mounted by reprogramming switch routing mechanisms or using malicious switches.

IDE establishes an IDE stream between two ports. When there are no switches between the Pports, then it is possible to secure all, or only selected, TLP traffic on the Link. For cases with and without switches between the ports, it is possible to secure selected TLP traffic.

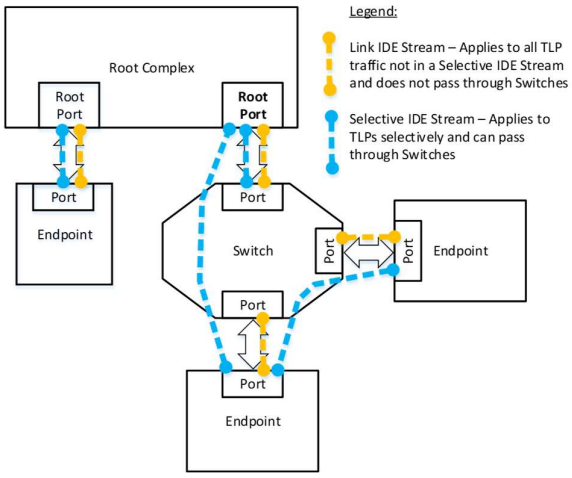

Figure 3.2: Selective vs. Link IDE streams

The Intel TDX Connect security architecture excludes switches from the TCB of TD and devices participating in the Intel TDX Connect framework. To that effect, the Intel TDX Connect architecture exclusively uses the selective IDE streams to protect the TLPs flowing between the SOC root complex and the Intel TDX Connect capable endpoint devices.

TLPs protected by IDE are called IDE TLPs. All IDE TLPs must use the IDE TLP prefix, which must precede all other end-end TLP prefixes.

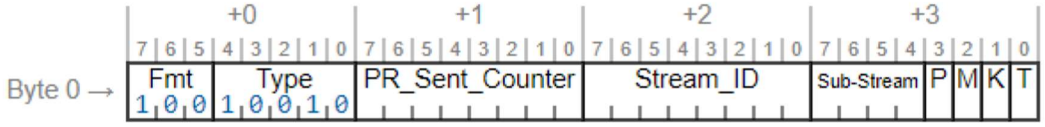

Figure 3.3: IDE TLP prefix

IDE uses AES in Galois/Counter Mode (GCM). For IDE TLPs, TLP data payload content forms the "Plaintext", also known as P, as defined in [AES-GCM], and the TLP Header and certain other elements form the "Additional Authenticated Data", also known as A, as defined in [AES-GCM]. The MAC size, also known as t, as defined in [AES-GCM], must be 96b.

When a selective IDE stream is enabled, matching the TC value in its stream Control Register determines that a transmitted TLP is associated with it. Additionally:

- For ID-Routed Messages and Completions, the destination is in the selective IDE RID Association Register block within the range between the RID Base and RID Limit
- For Memory Requests, Requests, the address is in the selective IDE Address Association Register block within the range between the Memory Base and Memory Limit

Received TLPs must be processed according to the stream ID indicated in the IDE TLP Prefix.

The unit of device assignment to a TD is a TEE Device Interface (TDI). The TDI may be a physical function (direct device assignment) or SR-IOV virtual function (VF). To assign a device interface to a TD, a selective IDE stream is first established with the device and set as a default IDE stream. On the host root port, the selective IDE address association register is setup with the address range decoded by the device and the RID association register is set up with the requester ID range decoded by the device.

Once the stream has been established between the host root port and the device all address-routed TLPs sent by the host CPU where the address matches the address association registers of that stream, including those originated by non-TD software, are sent as IDE TLPs using the key associated with that stream. Likewise, all ID routed messages and completions where the destination matches the RID range configured for that stream, including those originating from non-TD software, are sent as IDE TLPs using the key associated with that stream.

When the device generates TLPs, the stream used for those TLPs is determined in a device specific manner. For Intel TDX Connect usages, the TDISP lock interface message is used to configure the Default stream ID into the device such that all TLPs generated by the device for the TDI are sent as IDE TLPs using the default stream-id. Likewise, once the stream-id is configured into the TDI, the device only accepts IDE TLPs where the stream-id in the IDE TLP prefix matches the default stream-id.

Note that according to [TDISP Spec], all TDIs hosted by a TEE-IO device must be locked and bound to the same stream-ID which is the default stream.

When the TDI is assigned to a TD and is in TDISP RUN state, it generates TLPs using the default stream and with the T bit set to 1. Similarly, access to the TDI private MMIO space (TEE-MMIO) must be made using the default IDE stream ID and with the T bit set and only when the TDI is in TDISP RUN state.

Other device interfaces or when TDI is not in RUN state, must set the T bit to 0 and must not allow access to TD private resources (such as TEE-MMIO or TD private memory).

In summary, the Intel TDX Connect architecture uses the PCIe IDE extension as follows:

- Selective IDE streams are used to secure the TLPs flowing between SOC and devices with TD-assigned interfaces. Such devices may have some interfaces assigned to TDs and other interfaces assigned to non-TD software.
- Transactions originated from TDs set the T-bit in the IDE TLP prefix and the devices accept TLPs accessing TD-assigned interfaces only if the T-bit is 1. The T-bit helps the device differentiate between TD and non-TD accesses. The TD Secure EPT is used to allow access to MMIO address ranges of the device interfaces assigned to that TD. This way, it prevents one TD from accessing the MMIO address range of other device interfaces assigned to other TDs.
- Devices generate IDE TLPs with T-bit set to 1 for device interfaces assigned to TDs. The DMA TLPs with T-bit set to 1 are translated by the IOMMU using trusted DMA translation tables (discussed later) that allow the device interface to access its assigned TD's private memory. The trusted DMA translation tables prevent a TD-assigned device from performing DMA to memory of other TDs.

3.3. Trusted IO Access Controls

IDE provides end-to-end protection over the PCIe link and on the edges of the data transport between TD and its assigned TDI. The TEE-IO device DSM ensures that DMA access originated from a TDI will use the default IDE stream and will set the T bit on the IDE prefix. TDX Connect CPU and TSM access controls provide isolation for private DMA and MMIO accesses between TD and its assigned TDI(s) inside the CPU.

3.3.1. Trusted MMIO Security Objectives

Intel TDX Connect architecture enabling of trusted MMIO access from TD to TDI must cope with the following problems:

3.3.1.1. *Malicious Interference*

Non-TD software, other devices not trusted by TD, or another TD maliciously access MMIO range of a TD assigned interface. The malicious interference problem is addressed as follows:

- The TDX module must maintain a MMIO page assignment tracker to ensure each MMIO page can only be mapped to a single TD via its Secure-EPT. VMM calls TDX module API to map the MMIO space of the assigned device interface its TD owner in the TD Secure-EPT. To ensure MMIO space mapping is done correctly and exclusively (i.e., no other TD can access the assigned device MMIO space), TD uses the TDX module API to accept the MMIO pages exclusively assigned to it in order to ensure no other TD can generate trusted MMIO accesses (with TDX KeyID) to TD assigned MMIO space.
- A root complex generates TLPs with T bit set only if the MMIO access was translated through a TD Secure-EPT i.e., the access was made with a TDX KeyID. Untrusted software (e.g., Legacy VM, VMM, SMM, BIOS, etc.) cannot generate MMIO accesses with a TDX KeyID. For such MMIO accesses, made with shared KeyID, the root complex will clear the T bit in the IDE TLP. A device interface who follows TDISP must reject any IDE TLP with T bit set to 0 as unsupported TLPs if they access a TD assigned interface register.
- Device interface assigned to a TD is locked and bound to an IDE stream. Such device interfaces only accept IDE TLPs with T bit set to 1 and received on the stream to which they are bound. All other TLPs are rejected as unsupported TLPs. This binding of stream ID and locking of the interface to a TD is done using the TEE Device Interface Security Protocol messages (see: [PCI-SIG,TDISP) specification). Unauthorized devices cannot generate TLPs to access the registers of the TD assigned interface, as these unauthorized devices do not have the IDE encryption keys.

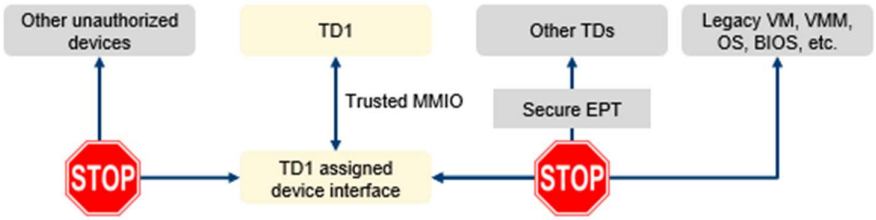

Figure 3.4: Malicious Interference attack

3.3.1.2. *Address remapping attack*

VMM may attempt to cross connect MMIO ranges of two different devices when mapping them into the secure-EPT. A variant of the attack involves mapping the MMIO GPA range to DRAM, other device interface MMIO pages or switching the order of pages of a contiguous MMIO HPA range.

For example, as illustrated in the figure below, a malicious host VMM can map GPA3 to point PA4 belongs to another device or switch the mapping of GPA1 and GPA2 to PA2 and PA1 respectively.

The remap attacks are addressed by TDX Connect as follows:

- MCEHCK and TDX module lock and verify BIOS configuration MMIO allocated to of external PCIe devices is consistent and deterministic on the entire platform.
- TDX module maintains MMIO page assignment to TDI tracker. TDX module host VMM API for MMIO mapping enforces that MMIO pages can only be mapped as private once and to a TD which is the current owner of the TDI.

- As part as TDISP life cycle, the TD receives the device interface configuration report. This report contains the private MMIO ranges and offsets (order) in which they must be mapped into the private GPA space of the TD.
- TDX module exposes API for TD guest to accept MMIO ranges device interface configuration report. The TDX module accept API verifies the MMIO pages are mapped as expected by converting the HPA offset to a specific HPA and checking it matches the VMM mapping.

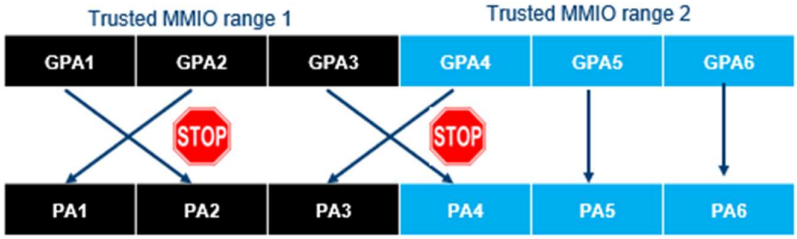

Figure 3.5: Address remap attack

3.3.1.3. *Overlap Attacks*

VMM tries to assign a MMIO range to two devices to exploit any priority decode bugs. The address decoders in the root complex could be misconfigured by the BIOS to set up multiple root ports to decode a given MMIO range. The VMM may set up the decoders in the switches downstream of the root complex to cause such overlaps. The overlap attack is addressed as follows:

- MCHECK and TDX module lock and verify MMIO address decoding between Core, SOC fabric, root complex and root ports are consistent and deterministic such that there are no MMIO routing aliases or dropping due to overlapped decoding rules to MMCFG, DRAM or other ranges.
- TEE-IO devices must ensure that their BARs do not overlap and treat any BAR re-programming as error when their TDI or IDE streams are in locked state (for details see [PCI-SIG, TDISP and IDE]).
- Intel TDX Connect architecture does not include switches in the trust boundary and setup selective IDE streams that are pass through for the switches. This avoids having to trust any configurations in the switches as mis-programming including creating such overlaps that lead to redirection of encrypted and integrity protected traffic and thus avoid compromising the confidentiality of the data. Note that denial of service is not a security objective.
- Another flavor of this attack involves reducing MMIO decoding windows to silently drop some of the trusted transactions. This attack is addressed by the TDX module locking the root complex and root port MMIO routing configuration ranges ensuring selective IDE streams and MMIO pages can only be programmed whiting these ranges.

Figure 3.6: Overlap Attack

3.3.1.4. *Redirection Attack*

VMM configures switches to redirect or drop MMIO accesses. The redirection attack is addressed as follows:

- Intel TDX Connect architecture does not include switches in the trust boundary and setup selective IDE streams that are pass through for the switches. This avoids having to trust any configurations in the switches as mis-programming including creating such overlaps that lead to redirection of encrypted and integrity protected traffic and thus avoids compromising the confidentiality of the data. Note that denial of service is not a security objective.
- A specific way such redirections can be done is by the VMM tampering with the selective IDE stream address and ID association registers selective IDE stream. This attack is addressed by the host side IDE access controls mechanisms preventing ensuring only the TDX module is able to program the IDE stream registers.

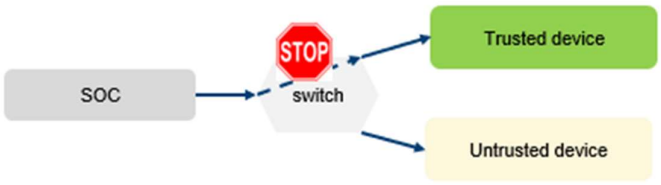

Figure 3.7: Redirection Attacks.

3.3.1.5. *Man-in-the-middle (MITM)*

Attacker, eavesdrop, modifies, replays or injects MMIO request or completion. The MITM attack is addressed as follows:

- Intel TDX Connect architecture does not include switches in the trust boundary and setup selective IDE streams that are pass through for the switches. This avoids having to trust any configurations in the switches or security mechanism around observation interfaces (e.g., debug ports) in the switches. The PCIe transactions between the SOC RC and the TEE-IO device are protected by the selective IDE stream encryption which guarantees confidentiality, integrity, and replay protection.

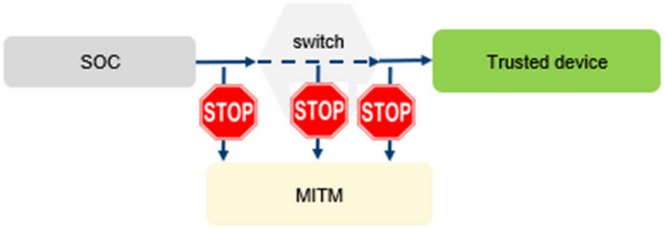

Figure 3.8: Man-In-The-Middle Attack.

3.3.1.6. *Malicious device interface programming*

This attack vector involves the VMM pre-configuring a device interface assigned to a TD in a malicious manner. Not allowing the TD an opportunity to inspect the device interface register configuration before the device interface is made operational and allowed to access TD private memories, is problematic. Malicious device interface programming attacks are addressed as follows:

- TDISP management protocol enforces that a TDI starts out in a non-operational form where it does not accept any MMIO accesses or generates DMA. The TD owner of the device interfaces is the only entity capable of generating Secure SPDM message to instruct the device to move into TDISP RUN state. TDX module exposes an API to get the TDISP message payloads and measurements required to establish trust and verify the device configuration (i.e., DEVICE_INTERFACE_REPORT) thus, the TD can first attest and verify the device configuration and only then, initiate a TDISP request to move the device into an operational (TDISP RUN) state.

3.3.1.7. *Un-Authorized TD Access*

Attacker VMM maps MMIO pages to an aggressor TD. Later, the VMM maps the same MMIO pages to the victim TD without first removing the MMIO pages from the aggressor TD or with invalidating the IOTLB. The aggressor TD can use the mapped MMIO or the stale IOTLB caches to access to TEE-IO MMIO space currently assigned to the victim TD.

Unauthorized TD access to device MMIO is addressed as follows:

- TDX module maintains MMIO assignment and mapping tracking tables in TDX protected memory
- TDX MMIO management APIs ensure MMIO pages are assigned and mapped to a single TD.
- TDX module exposes API for the TD to “accept” and verify the MMIO ranges are mapped exclusively to its private GPA space. TD must accept all the private MMIO pages as they appear in the TDISP report to ensure their exclusive assignment and mapping to the TDI and the TD private memory.
- TDX module does not allow removal of the MMIO mappings while the TDI is assigned to the TD.
- TDX module does not allow assignment of MMIO space to another TDI before all the MMIO pages from the previous TDI assignment have been removed and their associated IOTLBs were invalidated.

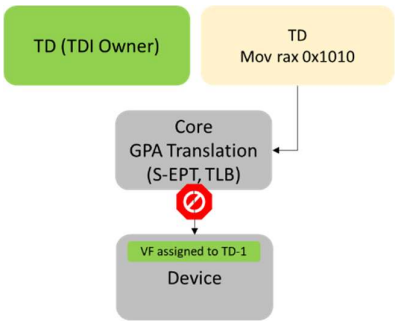

Figure 3.9: Un-Authorized TD Access.

3.3.1. Secure MMIO Management

To address the trusted MMIO security objectives, TDX module is extended with Secure MMIO management support to provide the following security properties:

1. TDI MMIO pages can only be mapped as private to a TD which is the TDI owner
2. TDI MMIO pages cannot be aliased by more than one GPA address for a TD
3. TDI MMIO page un-map can only be done after proper TLB invalidation and after the device is in TDISP CONFGI_UNLOCKED state
4. MMIO page cannot be assigned to another TDI and TD unless it was properly and securely removed from its previous TD

In addition, the TDX Module must provide the TD with a mechanism to ensure that the MMIO ranges from the TDISP report are mapped as private GPA and HPA in the TD Secure-EPT.

The CPU core is extended to carry a T bit set to 1 when a TD access to MMIO is made using a private GPA and HPA with the private TD KeyID. This T bit is carried to the RP and the IDE engine in order to differentiate between trusted and non-trusted MMIO accesses. For private MMIO access with T bit set, the RP IDE engine ensures the request or completion falls into IDE selective stream which is in secure state and leaves the SOC only as encrypted IDE TLP with T bot set in the IDE prefix.

At a high level, the following access control mechanism and rules are designed to enforce trusted MMIO access from TD to TDI private MMIO:

1. Before accepting a TDI and requesting its transition into TDISP RUN state, the TD must use TDX module APIs to ensure the TDI MMIO ranges are correctly mapped in the TD Secure-EPT by the VMM
2. TD access to private (GPA) MMIO page is translated to HPA with the TD private KeyID
3. The SOC routing fabric converts private KeyID as T bit = 1 to the RP
4. When T bit is set, the RP IDE engine identifies the transaction as trusted ensuring the TLP is selected by an IDE selective stream which is in a secure state. The IDE sets the T bit in the TLP IDE prefix
5. TEE-IO device and TDI must ensure access TEE-MMIO space is only possible when TDI is in TDISP RUN state, only using the default IDE stream and with IDE prefix T bit set.

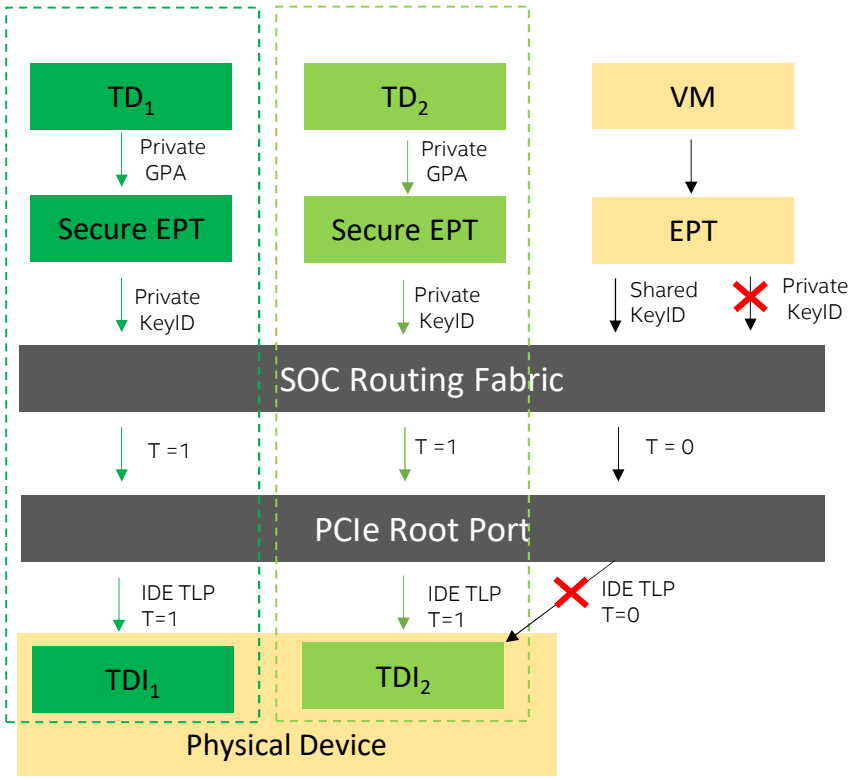

Figure 3.10: Secure MMIO access control

3.3.2. Trusted DMA Security Objectives

Intel TDX Connect enforces Secure DMA access controls to ensure a TDI may only access TD private memory after:

1. The TD has accepted the device into its TCB
2. The TDI is in TDISP CONFIG_LOCKED or RUN state (TDI is responsible not to access or accept TD memory accesses when it's not in TDISP RUN state)
3. The TD has verified the TDI report and explicitly accepted the TDI MMIO pages and trusted DMA mappings between the TDI (RID + default PASID) and the TD private memory.
4. The view of memory as seen by a TD guest must match the view of memory as seen by a device interface programmed by the TD guests.
5. The DMA translation tables used for performing the GPA/GPA to PA translations must have integrity such that they cannot be tampered with by untrusted software.

Unlike CPU-originated load/store where the CPU has at one instance of time a single TD that is executing, device-originated load/store can occur at any time, even when its TD is not executing. The right translation tables used to perform the translation are thus obtained from identities carried in the TLPs generated by the device interface itself.

Intel TDX Connect architecture enabling of trusted DMA access to TD private memory must cope with the following problems:

3.3.2.1. *ID spoofing*

Attacks using malicious devices spoof RID and/or PASID to reach TD private memory. The RID in the TLP is used to determine the context table entry and the PASID is used to determine the corresponding PASID table entry. These ID spoofing attacks are addressed as follows:

- Intel TDX Connect architecture restricts TD private memory accesses only to IDE TLPs with T bit set to 1 and when the stream-id in the IDE TLP prefix is that of a selective IDE stream. Any TLP which is not an IDE TLP or that which does not have the T bit set or where the stream on which it is received is not a selective IDE stream does not get the access rights to TD private memory.
- When an IDE TLP prefix is received by the SOC root port, the stream-id in the prefix is used to decrypt and authenticate the TLP. Once the TLP has been authenticated, if the T bit is set in the IDE TLP prefix but the stream ID on which it was received is not a selective IDE stream then TLP is rejected as an unsupported TLP. If the stream

ID is that of a selective IDE stream but source RID of the TLP does not match that stream RID association register, then the TLP is rejected as an unsupported TLP. This ensures that only devices with which a selective IDE stream has been established can generate TLP with T bit set to 1 and such devices cannot spoof a RID which is not in the RID association register used to setup that stream.

Figure 3.11: ID Spoofing Attack

3.3.2.2. *Remap attack*

This attack involves remapping GPA to PA or create different mappings seen by the TD compared to mapping seen from a device interface assigned to the TD. Remap attacks are addressed as follows:

- TDX-module enforces that the EPT used by the CPU for translating GPA is also linked to the PASID table entries for the TD assigned device interfaces.
- TDX-module enforces that any GPA to PA translation changes is accompanied by a CPU TLB, IOMMU TLB, and device TLB invalidates to address such remap attacks from being caused by a stale translation in any of these TLBs.
- TDX-module enforces that the translation caches and TLBs in the IOMMU are tagged with a unique per TD domain-ID and that the TD assigned domain-ID cannot be assigned to translation tables setup for untrusted legacy VMs or other untrusted software.
- TDX-module enforces invalidations of CPU, IOMMU, and device TLBs and translation caches before a TD assigned domain-ID can be reclaimed and used for other TDs.

Figure 3.12: Remap Attack

3.3.2.3. *Man-In-The-Middle (MITM)*

Eavesdropping, replaying, tampering with device-2-memory transactions. The MITM attack is addressed as follows:

- Intel TDX Connect architecture does not include switches in the trust boundary and setup selective IDE streams that are a pass through for the switches. This avoids having to trust any configurations in the switches or security mechanism around observation interfaces (e.g., debug ports) in the switches. The IDE TLPs are confidentiality, integrity, and replay protected end-to-end between the SOC root port and the Intel TDX Connect capable device. The IDE extensions implement a mechanism to detect violation of PCIe producer-consumer ordering rules.

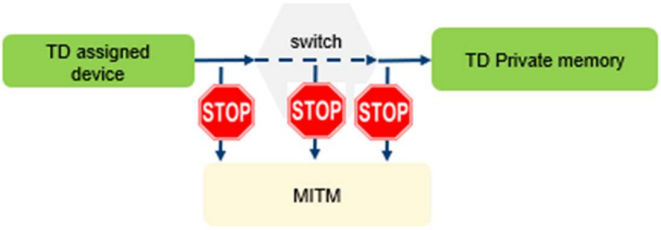

Figure 3.13: Man-In-The-Middle Attack

3.3.2.4. *Confused deputy access attacks*

A trusted device shared by two TDs and one TD instructs the device to access memory mapped to the other TD. The confused deputy attack is addressed as follows:

- The security model assumes that a device trusted by two TDs is in the trust boundary of both TDs and the TDs rely on the device to not use the incorrect Source ID when generating DMA transactions.
- TDX-module ensures that each device identified by RID (BDF) is exclusively assigned to a single TD and can be mapped in the trusted DMA table scalable-mode context entry and into a single VMM assigned PASID table entry only to the TD owner of the assigned device.
- TDX-module implements device assignment to ensure all DMA mappings of a TDI (identified by BDF from Function ID) are removed and the required IOTLB invalidations have been correctly done by the VMM before a TDI can be reclaimed from a TD and reassigned.

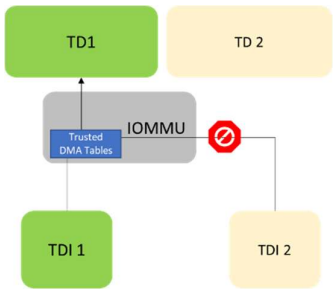

Figure 3.14: Confused Deputy Attack

3.3.3. Trusted DMA Translation

To meet the trusted DMA access control security objectives, trusted IOMMU hardware extensions provide the following capabilities:

1. Trusted DMA translation root table – a second root table address register that is restricted to SEAM SAI. The TDX module programs this root table address register with the root of the TDX module managed DMA translation tables for TD assigned devices. This allows the TDX module to enforce integrity on DMA translations for TD assigned devices.

2. Trusted invalidation queue – a second invalidation queue address register along with a second head and tail register that are restricted to SEAM SAI. The TDX-module programs this invalidation queue address register with the address of a TDX module managed invalidation queue. This allows the TDX module to manage the IOMMU and device TLB invalidations in a trusted manner.

3. IOTLB, PASID cache, and context entry cache are tagged to identify if they were cached from the trusted DMA translation tables programmed by the TDX-module.

4. Enforcing domain ID partitioning where-in the highest half of the domain-IDs are restricted to be used only in the trusted DMA translation tables when TDX mode is enabled. First and second level paging structure cache entries created from trusted DMA translation tables are thus differentiated from first and second level paging structure cache entries from the VMM managed untrusted DMA translation tables.

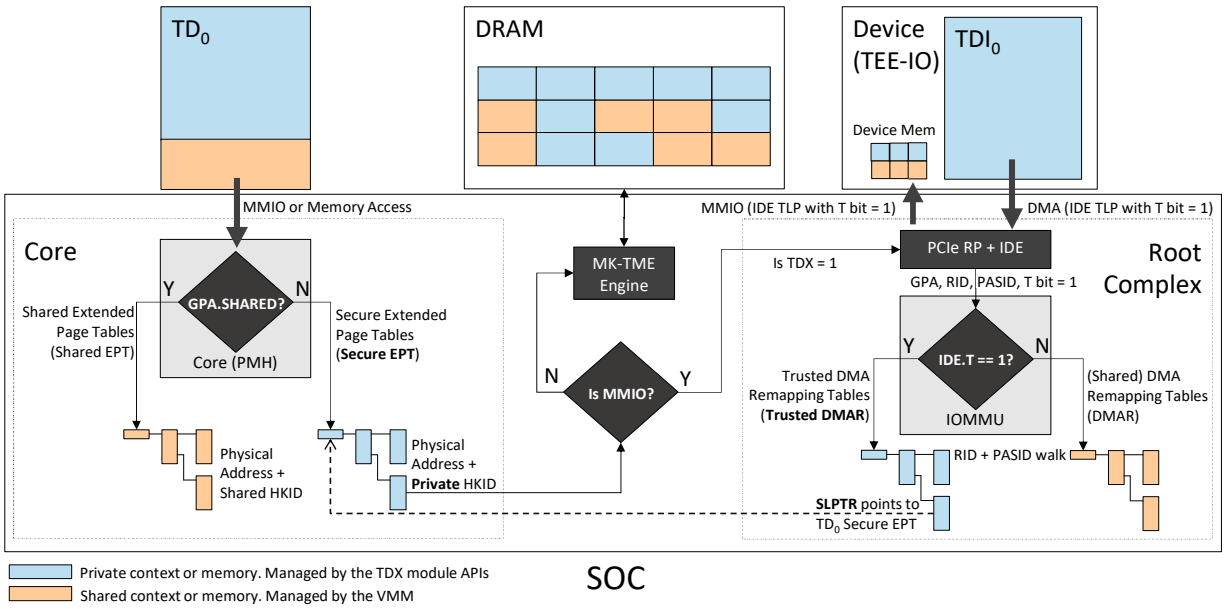

Figure 3.15: Trusted DMA and Secure-EPT

The TDX module manages the DMA trusted page tables separately, while the VMM manages the shared (legacy) DMA page tables. The trusted and shared page tables are isolated from each other with separate root table address registers.

The First-stage tables must be in TDX private memory, whereas the second-stage tables (EPT) may either be in TD private memory or TD shared memory. Secure EPT are managed by the TDX-module in TD private memory, whereas shared EPT is managed by the VMM in TD shared memory. Secure EPT are accessed with a private GPA, whereas shared EPT are accessed with a Shared GPA.

For nested translation, the FS table entries must store Private GPAs, and GPA translation must be assigned to SS table entries in Secure EPT.

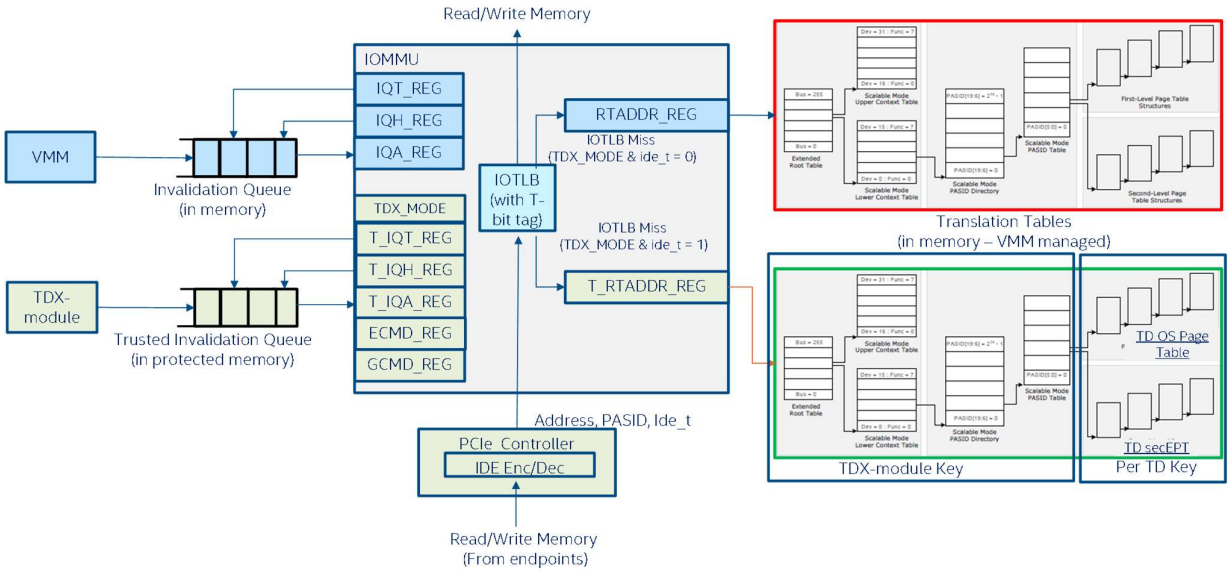

Figure 3-16: IOMMU Trusted DMA Translations

At a high level, the following access control mechanism and rules are designed to enforce trusted DMA access from TDI to its owner TD private memory:

1. TDX module exposes new APIs for trusted DMA management
2. TD must accept the device into its TCB and accept DMA mapping to its Secure-EPT

3. TEE-IO enforces TDI DMA access only allowed in RUN state using the default selective IDE stream with T bit = 1
4. Host RP IDE engine decrypts and authenticates the TLP passing T bit to IOMMU
5. The T bit is set in the request, the IOMMU use trusted DMA translation walking the DMA tables managed by the TDX module
6. SOC routing logic carries the T bit using it to ensure that access to memory with TD private KeyID is only allowed when T bit is set

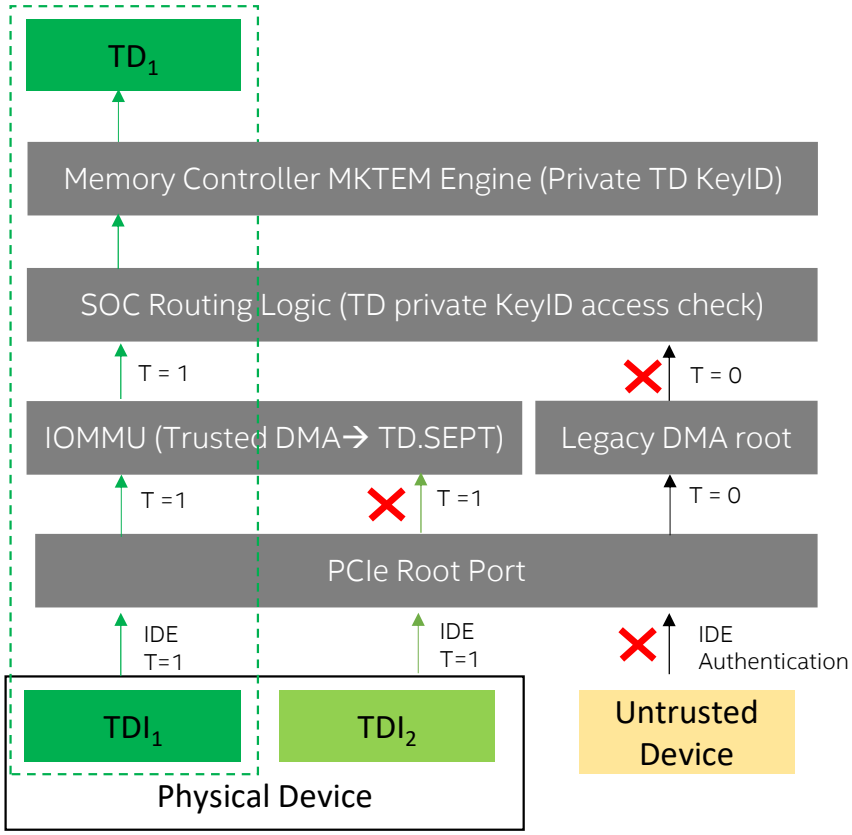

Figure 3.17: Trusted DMA access control

Note, the following capabilities are not supported by the current Intel TDX Connect specification and hardware and may be supported in future Intel TDX Connect extensions:

1. Address Translation Caching (ATC) – No support for Device-TLBs and caching of translations into device TLBs.
2. Nested Translation: Trusted DMA access may only use second level translation pointers. Nested translation (first and second level) may be supported as part of future Intel TDX Connect extensions.
3. Shared Virtual Memory (SVM) – No support for trusted page request queue.

3.4. TDX Module Extensions for TDX Connect and TEE-IO Support

The TDX module extensions for TDX Connect and TEE-IO support fall into three main categories:

1. Discovery and enabling of TDX Connect on TDX host platform
2. TEE-IO device management of SPDM session and IDE setup and teardown
3. TEE-IO Device Interface (TDI) life cycle management of TDI to TD direct assignment and removal

This document provides high-level overview of the following TDX component extensions required to support TDX

Connect and TEE-IO:

- BIOS and MCHECK extensions for TDX Connect platform configuration
- TDX module architecture and ABI extensions (for details refer to [TDX Module TDX Connect Spec])
- TDX Connect TEE-IO Provisioning Agent (for details refer to [TPA Spec])

3.4.1. Discovery and Enabling of TDX Connect on TDX host platform

The following diagram and sequence describe how TDX Connect is discovered and enabled by the platform BIOS and by the host VMM using TDX Module (and MCHECK) extensions:

1. BIOS uses Intel root complex and IDE enumeration to discover TDX connect as defined by the [Intel RC-IDE Guide]. Using the BIOS-MCHECK interface, the BIOS can decide (per IO Hub) which Root Ports shall be enabled with TDX Connect
2. MCHECK will checks the platform configuration is supported and can be secured with TDX Connect enabled and deliver information about TDX Connect enabling to the TDX module in the SEAM range protected memory
3. VMM uses intel root complex and IDE enumeration to discover TDX connect as defined by the [Intel RC-IDE Guide]. The VMM installs the TDX Module and uses TDX module extended feature enumeration to discover TDX module support for TDX Connect as defined by [TDX Module ABI Spec]
4. VMM enables TDX Connect per IO Hub (IOMMU and Root Ports) using TDX module IOMMU configuration API as defined in the [TDX Module TDX Connect Spec] and the [TDX Module ABI Spec]. This operation moves VTd, IDE and RP Type 1 registers under TDX control and management and enables TDX mode in the IO Hub.

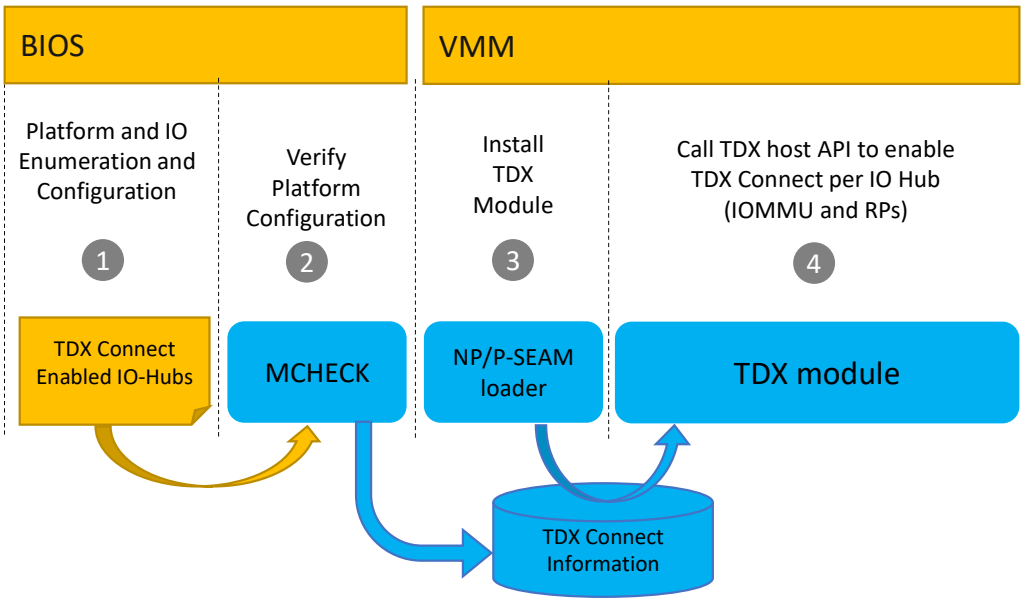

Figure 3.18: Discovery and enabling of TDX Connect

3.4.2. SPDM Management and the Intel TDX Connect TEE-IO provisioning agent (TPA) TD

Intel TDX Connect architecture requires to setup an SPDM session with the TEE-IO device and SPDM secure messages (as secure transport) for protecting the IDE_KM messages (see [PCI-SIG, IDE]) and the TDISP messages (see [PCI-SIG, TDISP]) used manage the device interface. The TDISP is protected by the Secure Protocol and Data Model (SPDM) secure session.

SPDM 1.2 key exchange is a computationally intensive function and uses RSA or ECDSA based SIGMA protocol to negotiate the management session key. The TDX-module by design is not re-entrant and not interruptive. Performing such long latency operations in the TDX-module is thus not desirable.

To avoid performing these long latency operations, the TDX-module delegates this function to a purpose-built TD called the Intel TDX Connect TEE-IO provisioning agent (TPA). The TPA TD is differentiated from other TDs by a TPA attribute bit set when such TDs are created and by its build time measurement. The TPA TD is launched by the VMM like other TDs and there is no difference in that process.

The TDX-module carries the SHA-384 hash of the TDINFO structure of the TPA TD and verifies that the TDINFO of the TPA TD matches the expected TDINFO as part of allowing the TPA TD to configure SPDM session keys. Thus, the TPA TD is in TCB of all TDs using Intel TDX Connect but is not explicitly reported in their TCB. It is implicitly in the TCB by virtue of the TPA TD measurements being authenticated by the TDX-module.

TPA TD uses the SPDM 1.2 protocol for key exchange with the device to negotiate a SPDM 1.2 session key with the device. On completion of the SPDM session setup, the TPA TD hands the SPDM session keys and context in addition to TDISP version and capabilities of the device to the TDX-module.

The TPA TD also records the SHA384 hash of a device TEE info structure of the device with which it did the key exchange. The TEE info structure is passed by the TPA to the VMM which then passes it to the TD for the attestation of the TEE-IO device. Since the VMM is not in the TCB, the TDs needs to verify the device TEE info by calculating its hash and then calling a TDX-module API function to verify the hash matches the associated device interface context structure.

The following diagram and sequence describe how SPDM session is established with the TEE-IO device by the VMM using the TPA TD:

1. VMM calls TDX module API to create the SPDM session context structure
2. VMM calls the TPA TD to requesting it to start the SPDM and TDISP negotiation process
3. The TPA TD negotiation steps include:
   - SPDM negotiation
   - SPDM identification
   - Secure SPDM Session establishment and key exchange
   - TDISP negotiation

   On each step SPDM or TDISP protocol message are generated and delivered as follows:

   - TPA TD generates protocol request and uses TD-VMCALL exit to deliver it to the VMM
   - VMM delivers the message to the DSM using DOE mailbox
   - DSM response is picked up by the VMM and delivered back to the TPA TD

4. The TPA record the SPDM session context, info hash and the TDISP negotiation version and capabilities in the TDX module SPDM context structure

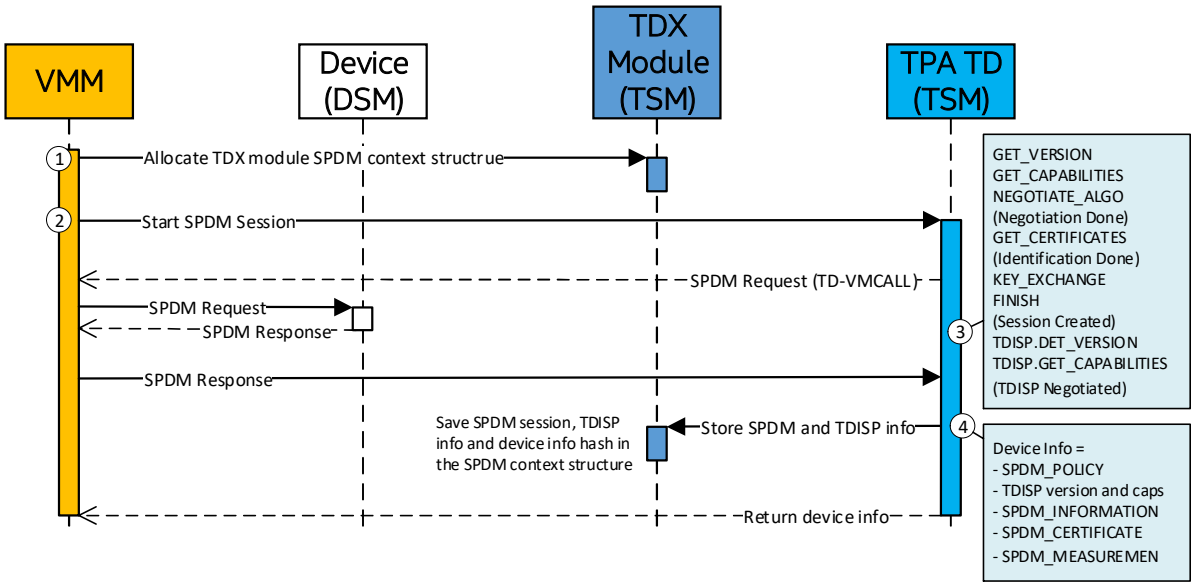

Figure 3.19: SPDM setup and TDISP negotiation sequence

5 For more details refer to the [TDX Module TDX Connect Spec] and the [TPA Spec].

3.4.3. IDE Setup

The following diagram and sequence describe the initial IDE streams setup done by the VMM using TDX module extension which follows the IDE_KM protocol for enabling and setting up the TEE-IO device EP IDE stream:

1. VMM calls TDX module API to configure the host RP IDE stream control registers (without enable the IDE yet)
2. Per IDE stream Tx/Rx directions sub-streams (PR, NPR, CPL), the VMM calls TDX module IDE_KM request API to generate and program the host RP IDE keys returning the corresponding IDE_KM KEY_PROG message. The VMM delivers the KEY_PROG message to the device using the DOE mailbox and then, the KP_ACK message from the device back to the TDX module using the IDE_KM response API
3. Per IDE stream Tx/Rx directions sub-streams (PR, NPR, CPL), the VMM calls TDX module IDE_KM request API to activate the host RP IDE keys returning the corresponding IDE_KM KEY_SET_GO message. The VMM delivers the KEY_SET_GO message to the device using the DOE mailbox and then, the K_GOSTOP_ACK message from the device back to the TDX module using the IDE_KM response API
4. When the VMM receives the last K_GOSTOP_ACK message from the device, it needs to setup the IDE-ECAP controls on the device, set the IDE stream Enable bit and check the status bit
5. When VMM feeds the last K_GOSTOP_ACK from the device as described in step 3., the TDX module will implicitly set the RP IDE stream Enable bit

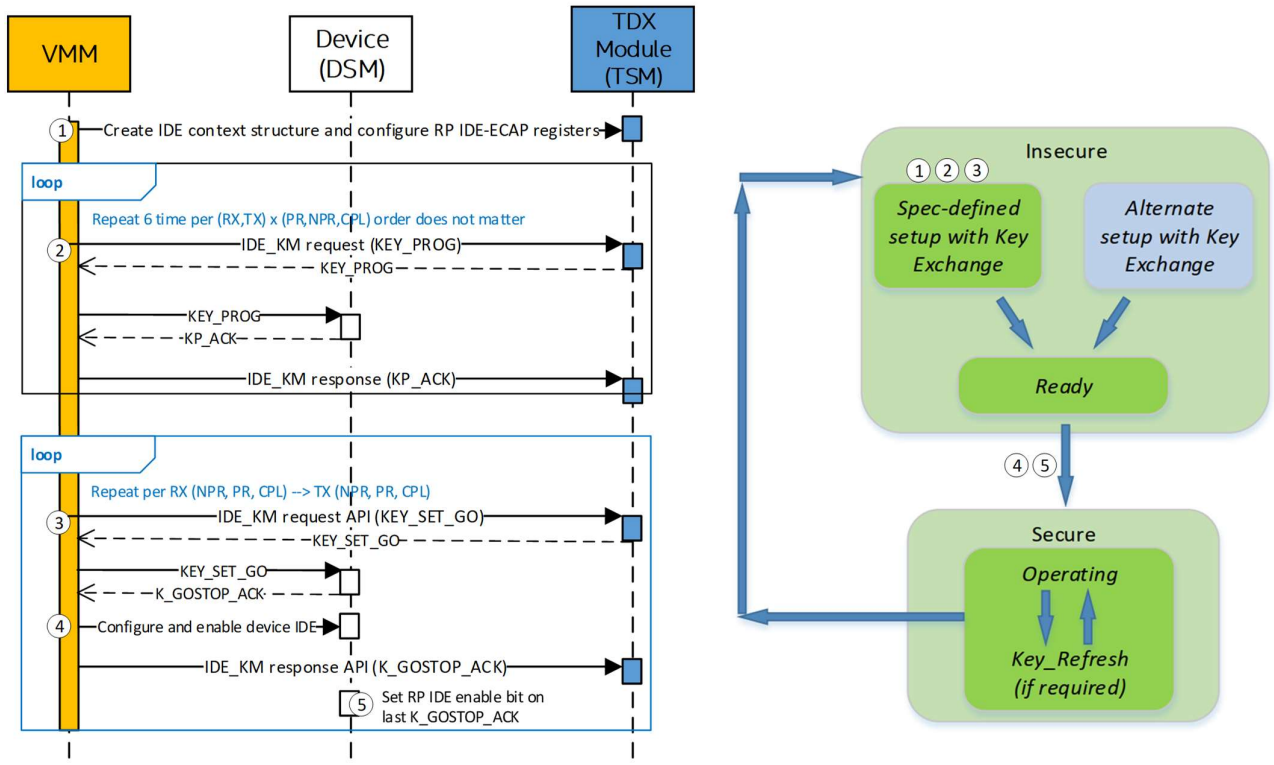

Figure 3.20: IDE setup sequence

3.4.4. TDI Assignment to TD

The following diagram and sequence describe the sequence of TDI assignment to a TD done by the VMM using and accepted by the TD using TDX module extension:

1. VMM configures the TDI on the TEE-IO device
2. VMM calls a TDX module API to assign the TDI FUNCTION_ID to a TD allocating memory for the TDI context structure
3. VMM calls the TDX module APIs for building and mapping the trusted DMA page tables into the TD Secure-EPT
4. VMM calls TDX module APIs for mapping the TDI MMIO ranges into the TD Secure-EPT as PENDING using Private GPA space and private HPA (TD Private KeyID)
5. VMM calls TDX module API for generating TDISP LOCK_INTERFACE_REQUEST. The VMM delivers the request to the device using the DOE mailbox and feeds back the device LOCK_INTERFACE_REQUEST using TDX API for processing and authenticating TDISP responses.
6. VMM delivers the TPA device info to the TD
7. TD checks the device information to decide if the device is trust-worthy or not. Then if it chooses to trust the device, the TD calculates the hash of the (VMM delivered) device information. The TD then invokes a TDX module API to verify the hash calculated over the device information matches the one recoded by the TPA TD during the SPDM session establishment with the TEE-IO device.
8. TD initiates a TDISP request for DEVICE_INTERFACE_REPORT, the request is fulfilled by the VMM call to TDX Module API to generate TD initiated TDISP request. The VMM delivers the request to the device using the DOE mailbox and feeds back the device LOCK_INTERFACE_REPORT message using the TDX module API for processing and authenticating TDISP responses. Finally, the TD reads the report using TDX module TDISP response API and verifies the device report matches the expected configuration.
9. The TD reads the MMIO ranges from the report and call TDX module API to verify and accept private MMIO mappings created by the VMM in step 4.
10. The TD accepts the PENDING trusted DMA mapping created by the VMM in step 3.
11. The TD initiates a TDISP START_INTERFACE_REQUEST (using same APIs and sequence described in step 8.)

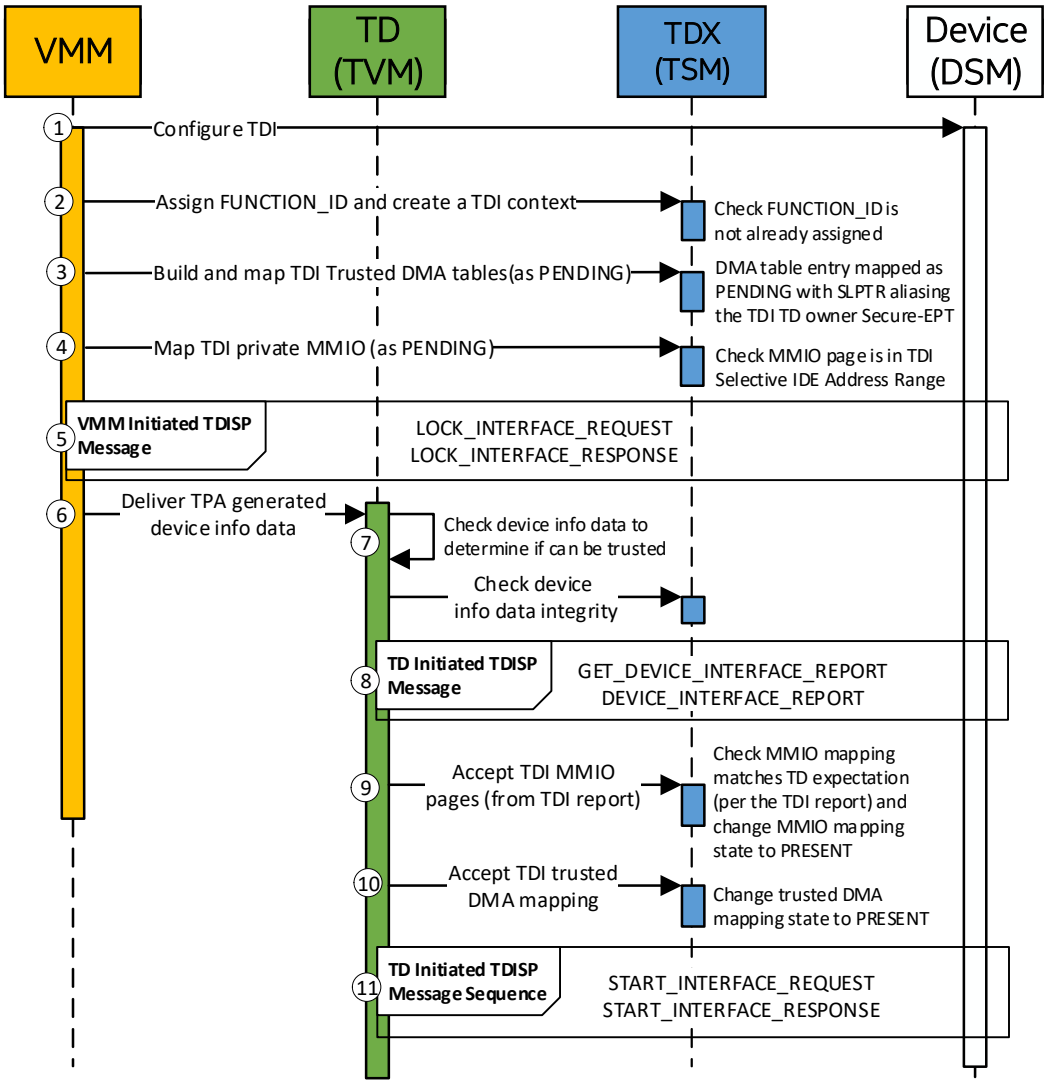

Figure 3.21: TDI assignment to TD sequence