Intel® Trust Domain Extensions (Intel® TDX) Module Application Binary Interface (ABI) Incompatibilities between TDX 1.0 and TDX 1.4/1.5

354808-002US

January 2023

.

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.

Table of Contents

Intel TDX ABI 1.5-1.0 Incompatibilities

1. About this Document

1.1. Scope of this Document

This document describes incompatibilities between the Application Binary Interface (ABI) of the Intel® Trust Domain Extensions (Intel® TDX) module, as defined for TDX 1.0 and for TDX 1.5. Such incompatibilities may require the host VMM s/w and/or guest TD s/w that has been written for TDX 1.0 to be updated for use with TDX 1.5.

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

Table 1.1: TDX Module Architecture Specification Set

Document Name	Reference	Description
TDX Module **Base Architecture Specification**	[TDX Module Base Spec]	Base TDX module architecture overview and specification, covering key management, TD lifecycle management, memory management, virtualization, measurement and attestation, service TDs, debug aspects etc.
TDX Module **TD Migration Architecture Specification**	[TD Migration Spec]	Architecture overview and specification for TD migration
TDX Module **TD Partitioning Architecture Specification**	[TD Partitioning Spec]	Architecture overview and specification for TD Partitioning
TDX Module **ABI Reference Specification**	[TDX Module ABI Spec]	Detailed TDX module Application Binary Interface (ABI) reference specification, covering the entire TDX module architecture
→ **TDX Module ABI Incompatibilities between TDX 1.0 and TDX 1.4/1.5**	[TDX Module ABI Incompatibilities]	Description of the incompatibilities between TDX 1.0 and TDX 1.4/1.5 that may impact the host VMM and/or guest TDs

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. Glossary

See the [TDX Module Base Spec].

1.3. Notation

See the [TDX Module Base Spec].

1.4. References

See the [TDX Module Base Spec].

2. TDX Module Enumeration

TDX Module 1.5 feature enumeration is forward compatible with TDX Module 1.0.

2.1. Host-Side TDX Module Enumeration

In TDX 1.0, once the TDX module has been initialized by TDH.SYS.INIT and at least one LP has been initialized by TDH.SYS.LP.INIT, TDH.SYS.INFO could be called to return a fixed-format TDSYSINFO_STRUCT, which enumerated various TDX module features and configuration.

A host VMM that is not aware of TDX 1.5 can still use the above enumeration method with TDX module version 1.5 or later.

However, In TDX 1.5, the fixed format of TDSYSINFO_STRUCT prevented adding all the new information required for TDX 1.5 features. Thus, a host VMM that is aware of TDX 1.5 should use the new interface functions TDH.SYS.RD and TDH.SYS.RDALL. These functions use the metadata interface convention and can be used to read one or more global metadata fields. A single field has been added to TDSYSINFO_STRUCT to indicate the availability of the new functions.

a host VMM that is aware of TDX 1.5 can use the following method to enumerate the TDX module:

1. Execute TDH.SYS.INIT
2. Execute TDH.SYS.LP.INIT
3. If the host VMM suspects that the TDX module version may be 1.0:
   3.1. Execute TDH.SYS.INFO to read the TDSYSINFO_STRUCT.
   3.2. If TDSYSINFO_STRUCT.SYS_RD is 0, this is a TDX 1.0 module. Else, it's a TDX 1.5 or newer module.
4. If the TDX module version is 1.5 or newer:
   4.1. Execute TDH.SYS.RD or TDH.SYS.RDALL to read the TDX module's functionality and configuration.

The host VMM should not rely on the TDX module version information provided in TDSYSINFO_STRUCT. Any update to the TDX module from the TDX 1.0 baseline is enumerated by the global fields read by TDH.SYS.RD*.

For further details, see the [TDX Module Base Spec] section on TDX module enumeration.

2.2. Guest-Side TDX Environment Enumeration

In TDX 1.0, a TD could call TDG.VP.INFO to enumerate a few configuration variables that were not available via the architectural means (CPUID, MSRs).

A guest TD that is not aware of TDX 1.5 can still use the above enumeration method with TDX module version 1.5 or later.

However, In TDX 1.5, the fixed format of TDG.VP.INFO prevented adding all the required information. Instead, new interface functions TDG.SYS.RD and TDG.SYS.RDALL have been added. These functions use the metadata interface convention and can be used to read one or more global metadata fields. A single field has been added to TDG.VP.INFO's return values to indicate the availability of the new functions.

a guest TD that is aware of TDX 1.5 can use the following method to enumerate its run time environment:

1. Execute TDG.VP.INFO
   1.1. If R10.SYS_RD (bit 0) is 0, the TD is running in a TDX 1.0 environment. Else, the TD is running in a TDX 1.5 or newer environment.
2. If the TD is running in a TDX 1.5 or newer environment:
   2.1. Execute TDG.SYS.RD or TDG.SYS.RDALL to read the TDX module's functionality and configuration.

Any update from the TDX 1.0 baseline is enumerated by the global fields read by TDG.SYS.RD*.

Examples of new guest-side features that can be enumerated:

- A Migration TD uses the Service TD guest-side API. It should also determine if TD migration is supported.
- Guest TD should understand if CPUID virtualization guest control is available, if it wants to use that feature.
- Guest TD should understand if TD partitioning is supported, if it wants to use that feature.
- Guest TD should understand if local attestation is supported, if it wants to use that feature.

For further details, see the [TDX Module Base Spec] section on guest TD run time environment enumeration.

3. TD Configuration

3.1. CPUID Configuration

TDX 1.0 Background

In TDX 1.0, the virtualization of CPUID fields was configurable based on the TD configuration, as provided to TDH.MNG.INIT in the TD_PARAMS structure. CPUID field could be configured based on the TD's ATTRIBUTES and XFAM setting, or it could be based on direct configuration provided by the host VMM in TD_PARAMS.CPUID_CONFIG.

The list of directly configurable CPUID bits is enumerated to the host VMM in TDX 1.0 in the TDSYSINFO_STRUCT output of TDH.SYS.INFO. Even with TDX 1.0 **the host VMM is required to consult this list in order to properly configure a TD** – this list is subject to change and will change between TDX 1.0 minor releases.

TDX 1.5 Updates

To support TD migration, TDX 1.5 added the option for some CPUID bit fields to be configurable based on both CPUID_CONFIG and either XFAM or ATTRIBUTES sections of TD_PARAMS. This is intended to support **fine-grained virtualization of sub-features of extended features**. For example:

- The host VMM can configure the TDX module to virtualize some AVX512 as available, but to virtualize other AVX512 instructions as unavailable.
- The host VMM can configure the TDX module to virtualize the Perfmon architectural events support.

This is useful for TD migration, as it allows the host VMM to configure a common subset of supported sub-features.

From the host VMM's perspective, the update means there are more configurable CPUID bits. The list returned by TDH.DYD.INFO, or the equivalent CPUID_CONFIG_LEAVES and CPUID_CONFIG_VALUES read by TDH.SYS.RD*, enumerates those bits.

For details, see the [TDX Module Base Spec] section on CPUID configuration by the host VMM.

Backward Compatibility Impact

A properly written VMM should be able to handle the fact that more CPUID bits become configurable.

The host VMM should always consult the list of directly configurable CPUID leaves and sub-leaves, as enumerated by TDH.SYS.RD/RDALL or TDH.SYS.INFO. If a CPUID bit is enumerated as configurable, and the VMM was not designed to configure that bit, the VMM should set the configuration for that bit to 1.

If the host VMM neglects to configure CPUID bits that are configurable, their virtual value (as seen by guest TDs) will be 0.

4. SEAMCALL and TDCALL Completion Status

4.1. Background

Completion status codes are returned in RAX by all SEAMCALL and TDCALL functions.

TDX 1.0 Background

The TDX 1.0 spec specifies the structure of status codes as follows:

Table 4.1: TDX 1.0 Interface Functions Completion Status (Returned in RAX)

Bits	Name	Description
63	ERROR	Instruction aborted due to error. 0: Indicates that the function completed successfully – possibly with some warnings. 1: Indicates that the function aborted due to some error.
62	NON_RECOVERABLE	Recoverability hint – applicable only when ERROR is 1. 0: Indicates that the function may possibly be retried after some conditions have been corrected. 1: Indicates that the error is probably not recoverable.
61:48	RESERVED	Reserved – set to 0
47:40	CLASS	Class of the function completion status
39:32	DETAILS_L1	Details of the function completion status
31:0	DETAILS_L2	Additional details of the function completion status – e.g., includes: • Implicit or explicit operand identifier • CPUID leaf or sub-leaf • MSR index • VMCS field code • VM exit reason • CMR index • TDMR index

Properly written host VMMs and guest TDs should use the ERROR and NON_RECOVERABLE indications bits and the CLASS/DETAILS_L1/DETAILS_L2 fields. They should ignore the RESERVED bits.

TDX 1.4/1.5 Updates

Two of the previously reserved bits are now used, as follows:

Table 4.2: New Bits in TDX 1.5 Interface Functions Completion Status (Returned in RAX)

Bits	Name	Description
61	FATAL	Fatality hint – applicable only for SEAMCALL. 0: Indicates that the TD can continue its normal lifecycle. 1: Indicates that the TD entered a state where it can only be torn down. E.g., when an import has failed and the TD's OP_STATE is FAILED_IMPORT.

Intel TDX ABI 1.5-1.0 Incompatibilities

Bits	Name	Description
60	HOST_RECOVERABILITY_HINT	As a TDH.VP.ENTER output, indicates a TDCALL that resulted in a trap-like TD exit for which the host VMM needs to provide a recoverability hint in the following TD entry. On the following TDH.VP.ENTER, the host VMM provides a hint to the guest TD, which is the output of the TDCALL: 0: The host VMM hints that the guest-side function may possibly be retried (e.g., the host may have corrected some conditions). 1: The host VMM hints that the error is probably not recoverable.

There are many new status codes (CLASS/DETAILS_L1/DETAILS_L2 combinations).

Backward Compatibility Impact

Properly written host VMMs and guest TDs should ignore the reserved bit; thus, they should not be confused by the newly defined bits 61 and 60.

There could be compatibility issues in cases where interface functions return different status (CLASS/DETAILS_L1/DETAILS_L2 combinations) in TDX 1.5 than in TDX 1.0. These are discussed below.

4.2. TD Operation State (OP_STATE)

To support TD migration, the operation state (OP_STATE) of the TD in TDX 1.5 is much more complicated than it was in TDX 1.0. As a result, we no longer implement specific state check logic for most SEAMCALL interface functions, and do not provide a specific error code for most failure cases. Instead, we have a generic table-driven check, and provide a generic TDX_OP_STATE_INCORRECT status in case of failure.

We expect the host VMM to not care about the failure reason. For debug, it is possible to query the TD's OP_STATE using TDH.MNG.RD.

4.3. Secure EPT Entry State

To support TD migration, the Secure EPT entry state in TDX 1.5 is much more complicated than it was in TDX 1.0. As a result, we no longer implement specific state check logic for most interface functions, and do not provide a specific error code for most failure cases. Instead, we have a generic table-driven check, and provide a generic TDX_EPT_ENTRY_STATE_INCORRECT status in case of failure.

As in TDX 1.0, TDX 1.5 also provides extended error information, which includes the Secure EPT entry architectural content, Secure EPT level and entry state of the Secure EPT entry where the error was detected.

4.4. TDH.VP.ENTER Completion Status

TDH.VP.ENTER has some new TD exit cases. See Ch. 5 for details.

4.5. Other Completion Status Incompatibilities

There are few other cases where error codes returned by TDX module interface functions have been enhanced or modified. There error codes are not expected in normal operation. They are only used in error cases and are intended for (human) debuggability.

Specifically, TDX 1.4/1.5 support of TD metadata access has been greatly enhanced over TDX 1.0. One size effect is that some error codes returned by TDH.MNG.RD/WR and TDH.VP.RD/WR have been modified and became more elaborate.

5. TD Entry and Exit

5.1. TD Exits without VMX Exit Information

TDX 1.0 Background

In TDX 1.0, all TD exits (termination of TDH.VP.ENTER) returned an architectural VM Exit Reason in the lower 32 bits of RAX. In most cases except synchronous TD exits, other GPRs returned additional VM exit information.

TDX 1.4/1.5 Updates

TDX 1.4/1.5 adds multiple cases where TD exit (i.e., completion of TDH.VP.ENTER) does not return a VM Exit Reason or any other VMX exit information.

Table 5.1: TD Exits without VMX Exit Information

TDH.VP.ENTER Completion Status (in RAX)	
TDX_NON_RECOVERABLE_TD_CORRUPTED_MD	This error code is only expected if TD-preserving update (which is only used with VMMs that are aware of TDX 1.5) is used.
TDX_HOST_PRIORITY_BUSY_TIMEOUT	See the discussion in 6.1.
TDX_CROSS_TD_FAULT	Applicable to Service TDs. A VMM that is not aware of TDX 1.5 will not bind service TDs.
TDX_CROSS_TD_TRAP	Applicable to Service TDs. A VMM that is not aware of TDX 1.5 will not bind service TDs.

The specific cases are details in the [TDX Module ABI Spec Spec] TDH.VP.ENTER sections.

5.2. Host Recoverability Hint

TDX 1.4/1.5 adds the option of trap-like TD exits to set bit 60 of the TDH.VP.ENTER completion status in RAX as a host-recoverability indicator. This is used in cases where a TDCALL would return some error status to the guest TD, e.g., TDX_HOST_PRIORITY_BUSY_TIMEOUT. The host VMM can impact the value of this bit, as will be seen by the guest TD, by setting RCX bit 52 on the next call to TDH.VP.ENTER.

Host recoverability is applicable for the following TDH.VP.ENTER completion status values:

- TDX_CROSS_TD_TRAP: Not expected to happen, see above.
- TDX_HOST_PRIORITY_BUSY_TIMEOUT: See the discussion in 6.1.

6. Concurrency Enforcement

6.1. Host-Priority Locks

6.1.1. Background

TDX 1.4/1.5 added a new host-priority locks mechanism to prevent DOS by guest TDs. This is a variant on explicit concurrency restrictions, where the host VMM side is given priority over guest TD side. A new HOST_PRIORITY flag is added to locks protecting resources that may be accessed by the host VMM and a guest TD. Both mutexes and shared/exclusive locks can be enhanced with host priority.

For details, see the [TDX Module Base Spec] section on concurrency restrictions with host priority.

6.1.2. Host-Side Operation

The host VMM is expected to recognize the TDX_OPERAND_BUSY_HOST_PRIORITY status returned by SEAMCALL operations; the VMM should retry the operation until successful. Failing to do so will prevent the guest TD from acquiring access to the busy resource, since it remains marked as "host priority".

Note: Beginning with the next TDX module release, the status code returned to the host VMM will be TDX_OPERAND_BUSY. This change will be done to preserve backward compatibility with TDX 1.0.

As a bug detection mechanism, if the guest TD tries to acquire a resource that has been held as "host priority" for more than a pre-configured time, a TD exit will happen. The returned status in this case is TDX_HOST_PRIORITY_BUSY_TIMEOUT.

6.1.3. Guest-Side Operation

The guest TD is expected to recognize the TDX_OPERAND_BUSY_HOST_PRIORITY status returned by TDCALL operations; the TD should retry the operation until successful.

Note: Beginning with the next TDX module release, the status code returned to the guest TD will be TDX_OPERAND_BUSY. This change will be done to preserve backward compatibility with TDX 1.0.

6.1.4. Backward Compatibility Impact

- The host VMM may not recognize the TDX_OPERAND_BUSY_HOST_PRIORITY status, and not retry the operation.
- The host VMM may not recognize TD exit with TDX_HOST_PRIORITY_BUSY_TIMEOUT.
- The guest TD may not recognize the TDX_OPERAND_BUSY_HOST_PRIORITY status, and not retry the operation.

Note: Beginning with the next TDX module release, the status code returned will be TDX_OPERAND_BUSY. This change will be done to preserve backward compatibility with TDX 1.0. The host VMM and guest TD should retry the operation.