Intel® Trust Domain Extensions - SEAM Loader (SEAMLDR) Interface Specification

343755-003US
MARCH 2022

Revision History

Revision	Description	Date
343755-001	Initial release of document.	September 2020
343755-002	Updated to include Intel persistent SEAMLDR details.	May 2021
343755-003	Updated to include TD-Preserving support.	March 2022

TABLES

FIGURES

CHAPTER 1 SECURE ARBITRATION MODE (SEAM)

1.1 OVERVIEW

Secure Arbitration Mode (SEAM) is an extension to the Virtual Machines Extension (VMX) architecture to define a new, VMX root operation called SEAM VMX root and a new VMX non-root operation called SEAM VMX non-root. Collectively, the SEAM VMX root and SEAM VMX non-root execution modes are called operation in SEAM.

SEAM VMX root operation is designed to host a CPU-attested, software module called the Intel® Trust-Domain-Extensions (Intel® TDX) module to manage virtual machine (VM) guests called **Trust Domains (TD)**.

The Intel TDX module helps implement the functions to build, tear down, and start execution of TD VMs. The VMM is designed to provide the memory resources to build the TD and schedule the TD executions using the API provided by the Intel TDX module. SEAM VMX root operation is designed to additionally host a CPU-attested, software module called the Intel Persistent SEAM Loader (Intel P-SEAMLDR) to load and update Intel TDX modules.

NOTE

Currently, the Intel TDX Module is the only SEAM module that the Intel P-SEAMLDR installs. In the future, the Intel P-SEAMLDR may support the installation of other SEAM modules.

Virtual machines launched/resumed from SEAM VMX root operation are TDs, and VMs launched/resumed from legacy VMX root operation are legacy VMs. Intel TDX modules use the SEAM instruction set extensions to help protect the confidentiality and integrity of TD memory contents and CPU state from all other software, including the hosting Virtual Machine Monitor (VMM), unless explicitly shared by the TD itself.

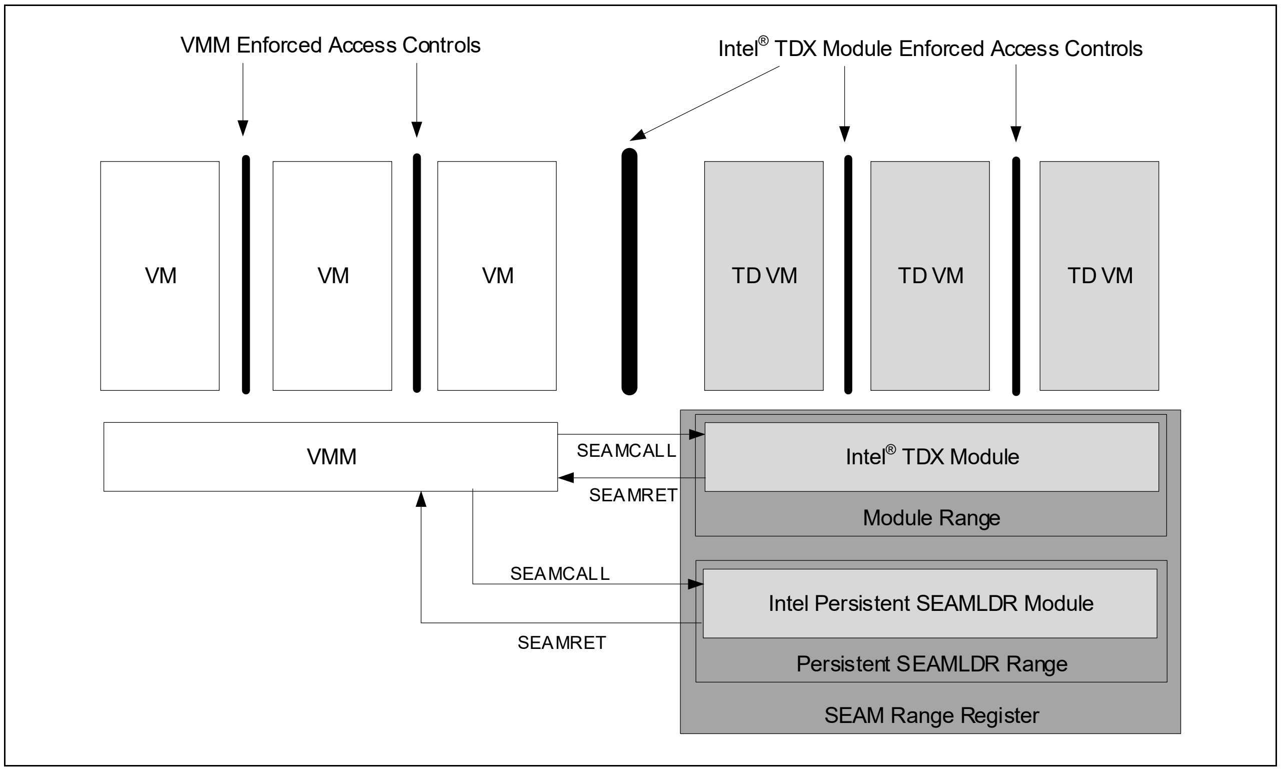

Figure 1-1. Intel® Trust Domain Extension Components

The Intel TDX module and the Intel P-SEAMLDR module, which execute in SEAM VMX-root operation, execute out of the memory range defined by the SEAM range registers (SEAMRR). The reserved range of memory specified using SEAM range registers (SEAM range) is configured by the platform owner and programmed by the BIOS.

The SEAM range is further partitioned into two sub-ranges by the processor:

- MODULE_RANGE
- P_SEAMLDR_RANGE

The MODULE_RANGE is used by the Intel TDX module, which provides functions to build and manage TD VMs. The P_SEAMLDR_RANGE is used by the P-SEAMLDR module, which is used to measure, verify, and install Intel TDX modules into the MODULE_RANGE. By design, access to the P-SEAMLDR range is restricted to the P-SEAMLDR module only.

Intel provides an authenticated code module (ACM) called the Intel Non-Persistent SEAM Loader (Intel NP-SEAMLDR) to install an Intel P-SEAMLDR module, whose image is embedded within the NP-SEAMLDR ACM, in the P_SEAMLDR_RANGE.

This is illustrated in Figure 1-2.

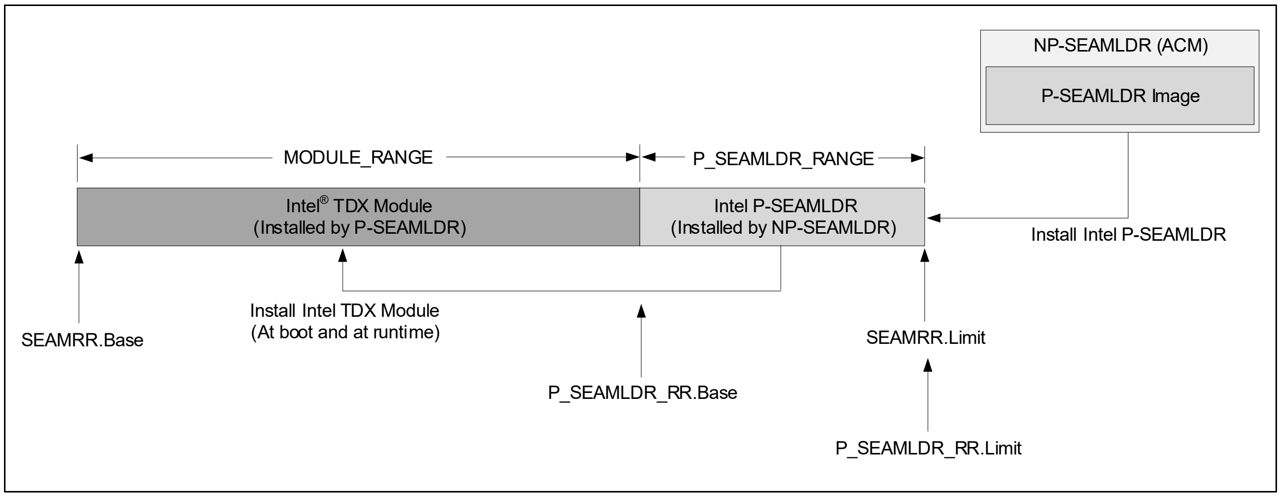

Figure 1-2. SEAM Range Register Details

The NP-SEAMLDR ACM helps with the initialization of the SEAM range, establishes the P-SEAMLDR range, sets up the SEAM transfer VMCS structure for transfers to the Intel P-SEAMLDR module, and loads the embedded Intel P-SEAMLDR module's image into the P_SEAMLDR_RANGE.

The OS can launch the SEAMLDR ACM using the GETSEC[ENTERACCS] instruction if the SEAMRR range enable bit (bit 11) of the IA32_SEAMRR_PHYS_MASK MSR is 1.

The SVN of the NP-SEAMLDR ACM itself is reported in the IA32_SGX_SVN_STATUS MSR in bits 63:56. OS/VMMs that launch an ACM such as SINIT or NP-SEAMLDR are expected to read the IA32_SGX_SVN_STATUS MSR to determine whether the ACM can be launched or if a new ACM is needed. If either the Intel® Software Guard Extensions (Intel® SGX) SVN of the ACM value in the ACM's header is greater than or equal to the value reported by the IA32_SGX_SVN_STATUS MSR, or the lock bit in the IA32_SGX_SVN_STATUS MSR is not set, then the OS/VMM can safely launch the NP-SEAMLDR ACM. Otherwise, the OS/VMM would not launch that version of the ACM. It would obtain an updated version of the ACM either from the BIOS or from an external resource.

The NP-SEAMLDR ACM is designed to follow the steps below to load or update the Intel P-SEAMLDR module into the Persistent SEAMLDR range in SEAMRR:

1. Perform basic checks on current state of platform.
2. Initialize the entire SEAM memory range.
3. Install the embedded Intel P-SEAMLDR module in the P-SEAMLDR memory range.
4. Set up data and stack regions for the Intel P-SEAMLDR module.

5. Set up a single SEAM transfer VMCS. This VMCS is used by SEAMCALL instructions when an Intel P-SEAMLDR module's API is called, and by the SEAMRET instruction when the Intel P-SEAMLDR module API returns.
6. Update the load status of the Intel P-SEAMLDR module.
7. Exit to OS using the GETSEC[EXITAC] instruction.

The Intel P-SEAMLDR module aims to provide interfaces to the VMM, invoked using the SEAMCALL instruction, to gain information about Intel TDX, install Intel TDX modules, and shutdown itself. The installation interface is designed to follow the steps below to load or update an Intel TDX module into the MODULE_RANGE:

1. Verify input parameters, including the Intel TDX module's signature structure (SEAM_SIGSTRUCT).
2. Load the Intel TDX module image into the MODULE_RANGE, measure it and verify the measurement matches with the signature structure.
3. Set up data regions, stack regions, and page tables for all logical processors.
4. Set up SEAM transfer VMCSs for all logical processors. These VMCSs are used by SEAMCALL instructions when an Intel TDX module's API is called, and by the SEAMRET instruction when the Intel TDX module API returns.
5. Record the Intel TDX module identity into CPU measurement registers and update its load status.
6. Return to VMM using the SEAMRET instruction.

CHAPTER 2 INTEL P-SEAMLDR MODULE LOAD AND UPDATE

The OS/VMM loader is designed to perform the following actions to load or update a previously loaded Intel P-SEAMLDR module:

1. Find the NP-SEAMLDR ACM image file and copy it to memory. The memory for the NP-SEAMLDR image must be contiguous, physical memory and must be below 4 GB.
2. Ensure that the NP-SEAMLDR ACM is compatible with the platform using the same rules as other ACMs. However, NP- SEAMLDR does not have dependency on Intel® TXT-supported chipset and TPM. So, filtering by Chipset ID or TPM information tables is not required (the NP-SEAMLDR ACM will have empty Chipset ID and TPM information tables). The steps to be performed are as follows:
   - Step 1: Verify it is NP-SEAMLDR ACM by evaluating module type.
   - Step 2: Verify platform type match.
   - Step 3: Verify AC module and chipset production flags match.
   - Step 4: Match CPUID.FMS and PLATFORM_ID.
3. The NP-SEAMLDR ACM must be launched on a logical processor marked BSP (i.e., where IA32_APIC_BASE.BSP is 1). All other logical processors in the socket on which NP-SEAMLDR will be launched must be in the WAIT-FOR-SIPI state. If they are not already in the WAIT-FOR-SIPI state (for example, coming out of reset), then the OS might need to send INIT IPIs to all other logical processors to place them in the WAIT-FOR-SIPI state.
4. The NP-SEAMLDR ACM also requires that the OS/VMM loader has invoked the shutdown function provided by the Intel P-SEAMLDR module (if it was previously installed), and VMXOFF has been executed on the BSP logical processor in which NP-SEAMLDR is launched.
5. The OS/VMM loader should prepare to launch NP-SEAMLDR ACM by setting up parameters for the NP-SEAMLDR in registers as follows:
   - R9 = GDT base to be established when returning to the OS.
   - R10 = RIP where control is transferred when returning to the OS.
   - R11 = CR3 value to be established when returning to the OS.
   - R12 = IDTR base value to be established when returning to the OS.
   - EBX = NP-SEAMLDR ACM physical address base.
   - ECX = NP-SEAMLDR ACM size.
   - EAX = 2, which is the leaf number for the ENTERACCS function of the GETSEC instruction.
6. The OS/VMM loader should launch NP-SEAMLDR ACM in IA32e mode with either 4-level or 5-level paging mode.
7. The OS/VMM loader invokes GETSEC[ENTERACCS] to launch the NP-SEAMLDR ACM.
8. Before exiting, the NP-SEAMLDR ACM restores launcher's GDTR, IDTR, and CR3. Certain processor states are not restored on EXITAC, and the OS/VMM loader must re-establish such states when control returns from the NP-SEAMLDR.
9. The NP-SEAMLDR ACM returns completion status in register R9. A value of 0 indicates that the P-SEAMLDR module has been successfully installed into P_SEAMLDR_RANGE, and can be invoked by the VMM using SEAMCALL instructions.

A TXT MLE that launches the NP-SEAMLDR ACM using ENTERACCS should be aware of following:

- The NP-SEAMLDR ACM does not close the TPM locality 2 or the TXT private space if the platform is post-SENTER as indicated by LT.STS[0] == 0x01. An MLE that launches the NP-SEAMLDR ACM should be aware of this, and re-evaluate TPM locality 2 and TXT private space status following execution of the NP-SEAMLDR ACM. TPM Locality 3 is closed unconditionally.

- The GETSEC[EXITAC] function unconditionally unmasks SMI, INIT, and NMI signals. Such an MLE must launch NP-SEAMLDR ACM only after the MLE has enabled SMI (or there is no STM/PPAM, and so SMI are enabled by SINIT).

2.1 CPU STATE FOLLOWING NP-SEAMLDR ACM EXECUTION

Table 2-1 describes the intended CPU state after NP-SEAMLDR ACM execution.

Table 2-1. CPU State After NP-SEAMLDR ACM Execution

Register	CPU State After EXITAC from NP-SEAMLDR
CR0	PE=1, PG=1, NE=1; other bits cleared.
CR3	As provided by OS in R11 with PCD and PWT bits cleared.
CR4	PAE=1, SMXE=1; PGE and LA57 are unchanged (as it was when NP-SEAMLDR was launched); other bits cleared.
RFLAGS	Reset value (0x2).
IA32_EFER	LME=1, LMA=1, NXE=1; other bits cleared.
IA32_PAT	Reset value (0x00070406_00070406).
RIP	As provided by OS in R10.
RAX	EXITAC leaf number (0x3).
RBX	As Provided by OS in R10.
R9	Load status returned by NP-SEAMLDR: success (0x0) or error code.
Other GPRs, including RSP	0x0
CS	Selector value bits 11:3 unchanged (as it was when NP-SEAMLDR was launched); selector bits 15:12 and 2:0 cleared; base/limit denote flat segment.
DS	Selector value bits 11:3 same as original ES (as ES was when NP-SEAMLDR was launched); selector bits 15:12 and 2:0 cleared; base/limit denote flat segment.
ES	Selector value bits 11:3 unchanged (as it was when NP-SEAMLDR was launched); selector bits 15:12 and 2:0 cleared; base/limit denote flat segment.
FS	Selector value bits 11:3 unchanged (as it was when NP-SEAMLDR was launched); selector bits 15:12 and 2:0 cleared; base/limit denote flat segment.
GS	Selector value bits 11:3 unchanged (as it was when NP-SEAMLDR was launched); selector bits 15:12 and 2:0 cleared; base/limit denote flat segment.
SS	Selector value bits 11:3 unchanged (as it was when NP-SEAMLDR was launched); selector bits 15:12 and 2:0 cleared; base/limit denote flat segment.
GDTR	Base = As provided by OS in R9; Limit unchanged.
IDTR	Base = As provided by OS in R12; Limit unchanged.
DR7	0x400
IA32_DEBUGCTL	0x0
IA32_PERF_GLOBAL_CTRL	0x0
IA32_PEBS_ENABLE	0x0
IA32_RTIT_CTRL	TraceEn = 0, other bits are unchanged.
IA32_LBR_CTRL	0x0
IA32_MISC_ENABLES	0x8 (only thermal throttling is enabled)
Other registers (including MSRs, XCR0, XSAVES-able registers, DRs)	Unchanged

2.2 ERROR HANDLING

The NP-SEAMLDR ACM returns error codes in the format 0x80000000_cccceeee, where the value cccc specifies the error class, and the value eeee specifies the error code within that class.

The error classes are described in Table 2-2.

Table 2-2. Error Classes

Error Class	Error Class Name	Description
0000	ECPARAM	Parameter validation errors. These errors are usually indicative of errors in the software that invokes the NP-SEAMLDR ACM.
0001	ECPLAT	Platform configuration errors. These are usually indicative of misconfiguration of the platform. This might be due to BIOS errors or unsupported hardware configurations.

The list of errors codes returned by the NP-SEAMLDR are described in Table 2-3.

Table 2-3. NP-SEAMLDR Error Codes

Error Code	Error Name	Description
0x8000 0000 0000 0001	EMODBUSY	The P-SEAMLDR module has not been shut down.
0x8000 0000 0000 0002	ELDINPROG	Software attempted to invoke the NP-SEAMLDR ACM simultaneously on multiple logical processors.
0x8000 0000 0001 0000	EBADRANGE	The SEAM range is not valid.
0x8000 0000 0001 0001	EBADPLATF	Unsupported platform configuration.
0x8000 0000 0001 0002	ENOMEM	The P-SEAMLDR module does not fit within the P_SEAMLDR_RANGE constraints.
0x8000 0000 0001 0003	EUNSPECERR	Unspecified error.

If the class of error is ECPARAM or ECPLAT, these are usually indicative of BIOS or OS software errors. Retrying the load will likely not succeed, and thus these errors can be fatal to the load process.

CHAPTER 3 SEAMLDR DATA STRUCTURES

3.1 SEAM_SIGSTRUCT: INTEL® TDX MODULE SIGNATURE STRUCTURE

Each Intel TDX module is accompanied by a signature structure, which provides metadata information about the module and allows verification.

All fields in the SEAM_SIGSTRUCT (e.g., MODULUS, SIGNATURE, etc.) are in little-endian form.

Table 3-1. SEAM_SIGSTRUCT: Intel® TDX Module Signature Structure

Field Name	Offset	Size	Description	Signed?
SIGNATURE STRUCTURE HEADER				
HEADER_TYPE	0	4	Module type; must be 6. Generic FW.	Y
HEADER_LENGTH	4	4	Header Length; must be 0xE1 (900 bytes). The length includes the following: ▪ Header: 128 B ▪ MODULUS: 384 B ▪ EXPONENT: 4 B ▪ SIGNATURE: 384 B The bytes following this Header (offset 900 onwards) are the body of the signed module.	Y
HEADER_VERSION	8	4	Structure Version Bits 31:16 - Major Version Bits 15:0 - Minor Version Must be 0x00010000.	Y
MODULE_TYPE	12	4	Module Type Bits 30:0 must be 0. Bit 31: Debug/Production Signed ▪ 0: Production ▪ 1: Debug	Y
MODULE_VENDOR	16	4	Module Vendor Must be 8086h (Intel module).	Y
DATE	20	4	Build date: yyyymmdd format (yyyy: four-digit year, mm: 1 to 12, dd: 1 to 31)	Y
SIZE	24	4	Size of entire module (i.e., SEAM_SIGSTRUCT) in DWORDS; must be 0x200.	Y
KEY_SIZE	28	4	RSA public key size; must be 0x60.	Y
MODULUS_SIZE	32	4	RSA public key modulus size; must be 0x60.	Y
EXPONENT_SIZE	36	4	RSA public key exponent size; must be 1.	Y
RESERVED	40	88	Must be 0x0.	Y
MODULUS AND SIGNATURE				
MODULUS	128	384	Module public key (3072 bits) represented as a byte string of length 384, with the most significant byte at offset 511, i.e., in little-endian format.	N
EXPONENT	512	4	RSA exponent. Must be 2^16+1.	N

Table 3-1. SEAM_SIGSTRUCT: Intel® TDX Module Signature Structure

Field Name	Offset	Size	Description	Signed?
SIGNATURE	516	384	RSA signature represented as a byte string of length 384, with the most significant byte at offset 899, i.e., in little-endian format. Signature hash covers (Header (128B) || Body) in that order.	N
Intel TDX MODULE CONFIGURATION PARAMETERS (Body)				
SEAMHASH	900	48	Intel TDX module SHA384 hash.	Y
SEAMSVN	948	2	Intel TDX module SVN.	Y
ATTRIBUTES	950	8	TDX module attributes (non-Intel). Must be 0x0.	Y
RIP_OFFSET	958	4	Offset of Intel TDX module entry point.	Y
NUM_STACK_PAGES	962	1	Stack size per thread in units of (# of 4K pages), minus 1.	Y
NUM_TLS_PAGES	963	1	TLS size per thread in units of (# of 4K pages), minus 1.	Y
NUM_KEYHOLE_PAGES	964	2	Keyhole pages in units of (# of 4K pages), minus 1.	Y
NUM_GLOBAL_DATA_PAGES	966	2	Global data in units of (# of 4KB pages), minus 1. This number does not include # of handoff data pages.	Y
MAX_TDMRS	968	2	Max number of TDMRs. 0 = 64 TDMRs. N > 0 = N TDMRs.	Y
MAX_RESERVED_PER_TDMR	970	2	Max number of reserved areas per TDMR. 0 = 16 reserved areas. N > 0 = N reserved areas.	Y
PAMT_ENTRY_SIZE_4K	972	2	PAMT entry size for 4KB pages, in bytes. 0 = 16 bytes. N > 0 = N bytes.	Y
PAMT_ENTRY_SIZE_2M	974	2	PAMT entry size for 2MB pages, in bytes. 0 = 16 bytes. N > 0 = N bytes.	Y
PAMT_ENTRY_SIZE_1G	976	2	PAMT entry size for 1GB pages, in bytes. 0 = 16 bytes. N > 0 = N bytes.	Y
RESERVED	978	6	Must be 0.	Y
MODULE_HV	984	2	The handoff version that this Intel TDX module natively supports.	Y
MIN_UPDATE_HV	986	2	The minimum handoff version that this Intel TDX module supports.	Y
NO_DOWNGRADE	988	1	A boolean flag that indicates whether this Intel TDX module cannot be replaced by an Intel TDX module whose handoff version is smaller than MODULE_HV. 0 = Downgrade allowed. 1 = Downgrade disallowed.	Y
RESERVED	989	1	Must be 0.	Y
NUM_HANDOFF_PAGES	990	2	Handoff data size in units of (# of 4KB pages) - 1.	Y

Table 3-1. SEAM_SIGSTRUCT: Intel® TDX Module Signature Structure

Field Name	Offset	Size	Description	Signed?
RESERVED	992	32	Must be 0.	Y
CPUID_TABLE_SIZE	1024	4	Number of entries in CPUID table. Valid values are 0 through 255. Setting to 0 matches all CPUs.	Y
CPUID_TABLE	1028	1020	Table of supported CPU version numbers as returned by CPUID.1.EAX. Stepping number is ignored for the match.	Y

3.2 SEAMLDR_PARAMS

The SEAMLDR_PARAMS structure is used by the VMM to provide the Intel P-SEAMLDR module with information about the Intel TDX module to load or update. The SEAMLDR_PARAMS structure contains a pointer to the SEAM_SIGSTRUCT structure that describes the Intel TDX module, a list of pointers to Intel TDX module image pages, and the requested update/load scenario.

A SEAMLDR_PARAMS structure with version = 0 is 4K bytes in size and formatted as described in Table 3-2.

Table 3-2. SEAMLDR_PARAMS Structure

Field Name	Offset (Bytes)	Size (Bytes)	Description
VERSION	0	4	Structure version; must be 0.
SCENARIO	4	4	Scenario for which SEAMLDR was invoked: ▪ 0: LOAD. Load Intel TDX module. ▪ 1: UPDATE. Update previously loaded Intel TDX module to the same or another Intel TDX module. Other values are reserved.
SIGSTRUCT_PA	8	8	A 4-KByte aligned physical address of the Intel TDX module's SEAM_SIGSTRUCT.
RESERVED	16	104	Reserved; must be 0.
NUM_MODULE_PAGES	120	8	Intel TDX module size in number of 4-KByte pages. Valid range 1 to 496 for version 0.
MOD_PAGES_PA_LIST	128	8 * 496	Array of 4-KByte aligned physical addresses to the Intel TDX module's executable pages.

3.3 SEAMLDR_INFO

This data structure is used to return SEAMLDR information to the VMM. Some of the SEAMLDR_INFO fields describe the NP-SEAMLDR ACM that installed this P-SEAMDLR instance.

A SEAMLDR_INFO structure with version = 0 is 256 bytes in size and formatted as described in Table 3-3.

Table 3-3. SEAMLDR_INFO Structure

Field Name	Offset (Bytes)	Size (Bytes)	Description
VERSION	0	4	Structure Version; returns 0.
ATTRIBUTES	4	4	Bitmap of attributes: ▪ Bit 31 - Production-worthy (0) or debug (1). ▪ Bits 30:0 - Reserved 0.
VENDOR_ID	8	4	Vendor ID: Value is fixed to 0x8086 (Intel P-SEAMLDR module).
BUILD_DATE	12	4	Build date, in yyyy.mm.dd BCD format (from NP-SEAMLDR ACM header).
BUILD_NUM	16	2	Build number.
MINOR	18	2	Minor version number.
MAJOR	20	2	Major version number.
RESEREVED	22	2	Reserved 0.
ACM_X2APICID	24	4	The X2APICID of the logical processor on which NP-SEAMLDR ACM was launched.
NUM_REMAINING_UPDATES	28	4	Number of remaining Intel TDX module updates.
SEAM_INFO	32	128	The SEAM information of the Intel TDX module currently loaded in MODULE_RANGE (0 if no Intel TDX module is currently loaded). The SEAMINFO structure provides information about the currently loaded Intel TDX module. It's the lower 128 bytes of the TEE_TCB_INFO structure, as defined in the Intel® Trust Domain CPU Architectural Extensions Specification, Table 2-3.
SEAM_READY	160	1	A boolean flag that indicates, when set to 0x1, that the Intel TDX module is ready for SEAMCALL.
SEAM_DEBUG	161	1	A boolean flag that indicates, when set to 0x1, that a debuggable Intel TDX module (i.e., an Intel TDX module with SEAM_SIGSTRUCT.MODULE_TYPE[bit 31] = 1) can be loaded.
P_SEAMLDR_READY	162	1	A boolean flag that indicates, when set to 0x1, that the Intel P-SEAMLDR module is ready for SEAMCALLs (always 0x1).
RESEREVED	168	88	Reserved 0.

3.4 SEAMLDR_SEAMINFO

This data structure is used to return SEAM system information to the VMM.

A SEAMLDR_SEAMINFO structure with version = 0 is 2K bytes in size and formatted as described in Table 3-4.

Table 3-4. SEAMLDR_SEAMINFO Structure

Field Name	Offset (Bytes)	Size (Bytes)	Description
VERSION	0	8	Structure Version - Set to 0.
TOT_NUM_LPS	8	4	Total number of logical processors in platform.
TOT_NUM_SOCKETS	12	4	Total number of sockets in platform. For single socket platforms this will report 1. The maximum number of sockets supported is 8.
SOCKET_CPUID_TABLE	16	32	A table of 8 family/model/stepping (FMS) values, as reported by CPUID leaf 1 in the EAX register, from all sockets. If TOT_NUM_SOCKETS < 8, entries TOT_NUM_SOCKETS ... 7 return 0.
P_SEAMLDR_RANGE_BASE	48	8	Physical base address of P_SEAMLDR_RANGE.
P_SEAMLDR_RANGE_SIZE	56	8	Size of P_SEAMLDR_RANGE, in bytes.
SKIP_SMRR2_CHECK	64	1	A boolean flag that indicates whether the Intel TDX module can skip checking of SMRR2.
TDX_AC	65	1	A boolean flag that indicates whether TDX memory is protected by access control (AC) mechanism, without memory integrity.
RESERVED	66	62	Must be 0.
CMR_0_BASE	128	8	A 4K-aligned physical base address of Convertible Memory Region 0 (CMR0).
CMR_0_SIZE	136	8	Size of CMR0, in bytes (a multiple of 4K). Size = 0 indicates that CMR0 is not valid.
...			
CMR_31_BASE	624	8	A 4K-aligned physical base address of CMR31.
CMR_31_SIZE	632	8	Size of CMR31, in bytes (a multiple of 4K). Size = 0 indicates that CMR31 is not valid.
RESERVED	640	1408	Must be 0.

CHAPTER 4 P-SEAMLDR FUNCTIONS

The Intel P-SEAMLDR module provides several APIs for the VMM. The ABI can be invoked after the Intel P-SEAMLDR module has been successfully loaded by the NP-SEAMDLR ACM into the P_SEAMLDR_RANGE within the SEAM range, and until the Intel P-SEAMLDR module's shutdown session ended, or a fatal SEAM shutdown event occurred.

The Intel P-SEAMLDR module's ABI is invoked by executing the SEAMCALL instruction with RAX bit 63 set to '1. A successful SEAMCALL invocation transitions to the Intel P-SEAMLDR module unless another call to the Intel P-SEAMLDR module is already in progress.

The Intel P-SEAMLDR module's ABI supports the following functions:

- Leaf 0 = SEAMDLR.INFO: Retrieve information about the Intel P-SEAMLDR module and the current Intel TDX module.
- Leaf 1 = SEAMLDR.INSTALL: Load or update an Intel TDX module.
- Leaf 2 = SEAMLDR.SHUTDOWN: Shutdown the Intel P-SEAMLDR module in preparation for updating it.
- Leaf 3 = SEAMLDR.SEAMINFO: Get SEAM information.

If the input RAX contains an unsupported API index, then the P-SEAMLDR module returns an EBADPARAM error.

4.1 SEAMLDR.INFO

This function allows the VMM to query about the status of the Intel P-SEAMLDR and Intel TDX modules.

Usage:

- Can be invoked on any logical processor.

Inputs:

- RAX = 0x80000000.00000000
- RCX = A 64-bit physical address of an output buffer of type SEAMLDR_INFO. This address must satisfy the following rules:
  - No reserved bits.
  - No TDX-private Key ID.
  - No overlap with SEAM range defined by SEAMRR.
  - Aligned to a 256-byte boundary.

Operation:

- Check input RCX.
- Fill the given buffer with SEAMLDR_INFO.

Outputs:

- RAX = Success (0), or one of the following error codes:
  - EBADPARAM
    - If the physical address in input RCX is illegal.

4.2 SEAMLDR.INSTALL

This function allows the VMM to install a new Intel TDX module into the MODULE_RANGE within the SEAM range.

Usage:

- The API should be called on all logical processors in the platform, serially. When invoked on the first logical processor, it starts the "installation session". When invoked on the last logical processor, it loads or updates the MODULE_RANGE with the given Intel TDX module, and ends the installation session.
- During the installation session, the old Intel TDX module, if any, cannot be invoked (i.e., SEAMCALL instructions with RAX bit 63 set to '0, return VMFailInvalid flags).
- The update session may end with one of the following outcomes:
  - **Success.** This outcome is indicated by output RAX == 0. On success, the newly installed Intel TDX module can be invoked.
  - **Recoverable Error.** This outcome is indicated by output RAX != 0 and output RDX == 0. On a recoverable installation error, the old Intel TDX module can be invoked. This outcome is possible only in "update" scenarios.
  - **Non-Recoverable Error.** This outcome is indicated by output RAX != 0 and output RDX != 0. On a non-recoverable installation error, the old Intel TDX module, if any, cannot be invoked.

Inputs:

- RAX = 0x80000000.00000001
- RCX = A 64-bit physical address of an input buffer of type SEAMLDR_PARAMS. This address must satisfy the following rules:
  - No reserved bits.
  - No TDX-private Key ID.
  - No overlap with SEAM range.
  - Aligned on 4K-byte boundary.

Operation:

- Return a recoverable error if a shutdown session is already in progress or this API was already called in this logical processor in the current installation session.
- Start an installation session if an installation session is not yet in progress.
- If this is not the last logical processor on which this API was called, return success.
- Else // last logical processor in the installation session.
  - Load or update the new Intel TDX module into the MODULE_RANGE, according to the requested installation scenario (as specified in SEAMLDR_PARAMS.SCENARIO). This is done in the following steps:
    1. Check input SEAMLDR_PARAMS address (in RCX).
    2. Read SEAMLDR_PARAMS (pointed by RCX) and check the structure's contents.
    3. Read SEAM_SIGSTRUCT (pointed by SEAMLDR_PARAMS.SEAM_SIGSTRUCT_PA), check the structure's contents, and authenticate it.
    4. If the installation scenario is "update", check that the SEAMSVN of the new Intel TDX module is not less than the SEAMSVN of the old Intel TDX module.
    5. On CPUs that support TD-Preserving module updates: check that the handoff data left by the old Intel TDX module in MODULE_RANGE is valid, and that its version matches with the handoff version that the new Intel TDX module can handle (as specified in the MODULE_HV, MIN_UPDATE_HV and NO_DOWNGRADE fields of SEAM_SIGSTRUCT).
    6. Check that the new Intel TDX module fits within the MODULE_RANGE.
    7. Clear the MODULE_RANGE (on CPUs that support TD-Preserving module updates, if the installation scenario is "update", the handoff data pages are not cleared).
       a. Note: From this point, all failures are reported as unrecoverable.
    8. Copy the new Intel TDX module's image pages (pointed by the SEAMLDR_PARAMS.MOD_PAGES_PA_LIST array) into the MODULE_RANGE.

9. Measure the image and verify it's equal to the expected measurement (in SEAM_SIGSTRUCT.MRSEAM).
10. Create mappings for the new Intel TDX module's code, data, and stacks.
11. Create transfer VMCSs for the new Intel TDX module.
12. Pass system information in a dedicated MODULE_RANGE page to the new Intel TDX module.
13. Record the SEAMINFO of the new Intel TDX module.
14. End the installation session.
15. Return success.

— If any error was detected during the above process, then:
  - End the installation session.
  - If the error was detected before memory cleanup and the requested scenario was "update", then return a recoverable error indication.
  - Else return an unrecoverable error indication.

Outputs:

- RAX = Success (0), or one of the following error codes:
  — EBADPARAM
    - The input RCX is illegal.
    - Any TMP_PARAMS field is invalid.
    - An "update" scenario was requested but there's no Intel TDX module in MODULE_RANGE.
  — EUNSPECERR
    - Unexpected error.
  — ENOENTROPY
    - Random number could not be generated.
  — EBADCALL
    - Shutdown session started.
    - This API already called on this logical processor.
    - Too many updates.
  — EBADSIG
    - SEAM_SIGSTRUCT field is invalid.
    - Unknown public key in SEAM_SIGSTRUCT.
    - The SEAM_SIGSTRUCT signature not authenticated.
    - SEAM_SIGSTRUCT is debug-signed but SEAM_UNDER_DEBUG is 0.
    - Attempt to update to a different Intel TDX module vendor.
    - Attempt to update to a smaller SEAM SVN.
  — ENOMEM
    - Module range doesn't suffice for the new module's code and data.
  — EBADHASH
    - Module's hash in SEAM_SIGSTRUCT doesn't match the actual hash of the new module's image.
  — EBADHANDOFF
    - The old module's handoff data in MODULE_RANGE is not valid, or is of a version that the new Intel TDX module cannot support.

4.3 SEAMLDR.SHUTDOWN

This API allows the VMM to shut down the P-SEAMLDR module, in order to update it (by launching another NP-SEAMLDR ACM).

Usage:

- The API should be called on all logical processors in the platform, serially, such that the last logical processor on which this API is called is the same logical processor on which the NP-SEAMLDR ACM that installed the P-SEAMLDR module had been launched.
- When invoked on the first logical processor, it starts the “shutdown session”. When invoked on the last logical processor, the shutdown session ends and the P-SEAMLDR becomes non-executable.
- After the shutdown session starts, the old Intel TDX module, if any, cannot be invoked anymore.

Inputs:

- RAX = 0x80000000.00000002

Operation:

- Start a shutdown session if a shutdown session is not yet in progress.
- If this is not the last logical processor on which this API was called, return success.
- Check that this is the logical processor on which the NP-SEAMLDR ACM that installed P-SEAMLDR was launched.
- End the shutdown session and return success.

Outputs:

- RAX = Success (0), or one of the following error codes:
  - EUNSPECERR
    - Unspecified error.
  - EBADCALL
    - This logical processor is not the logical processor on which NP-SEAMLDR ACM was launched.

4.4 SEAMLDR.SEAMINFO

This function allows the VMM to query SEAM information.

Usage:

- Can be invoked on any logical processor.

Inputs:

- RAX = 0x80000000.00000003
- RCX = A 64-bit physical address of an output buffer of type SEAMLDR_SEAMINFO. This address must satisfy the following rules:
  - No reserved bits.
  - No TDX-private Key ID.
  - No overlap with SEAM range defined by SEAMRR.
  - Aligned to a 2K-byte boundary.

Operation:

- Check input RCX.
- Fill the given buffer with SEAMLDR_SEAMINFO.

Outputs:

- RAX = Success (0), or one of the following error codes:
  - EBADPARAM
    - If the physical address in input RCX is illegal.

4.5 ERROR HANDLING

The Intel P-SEAMLDR module returns error codes in the format 0x80000000_ccccеeee, where the value cccc specifies the error class, and the value eeee specifies the error code within that class.

The error classes are described in Table 4-1.

Table 4-1. Settings to Initialize Intel® TDX Module Signature Structure

Error Class	Error Class Name	Description
0000	ECPARAM	Parameter validation errors. These errors are usually indicative of errors in the software that invokes the NP-SEAMLDR ACM.
0001	ECPLAT	Platform configuration errors. These are usually indicative of misconfiguration of the platform. This might be due to BIOS errors or unsupported hardware configurations.
0002	ECIMG	Module image verification errors. These are usually indicative of corruptions in the module image leading to errors like signature verification error etc.
0003	ECPROG	Progress status.

The list of errors codes returned by the Intel P-SEAMLDR module ABI are described in Table 4-2.

Table 4-2. Error Codes

Error Code	Error Name	Description
0x8000 0000 0000 0000	EBADPARAM	Bad input parameter.
0x8000 0000 0000 0003	EBADCALL	P-SEAMLDR has already been called on the LP.
0x8000 0000 0000 0004	EBADHANDOFF	Update failure due to invalid or unsupported handoff data.
0x8000 0000 0001 0002	ENOMEM	The new SEAM module does not fit within the SEAM range constraints.
0x8000 0000 0001 0003	EUNSPECERR	Unspecified platform configuration error.
0x8000 0000 0001 0004	EUNSUPCPU	The new SEAM module cannot be loaded on one (or more) CPUs in the platform.
0x8000 0000 0002 0000	EBADSIG	Bad SEAM module signature (malformed, or signature verification failed).
0x8000 0000 0002 0001	EBADHASH	The Intel TDX module's image hash verification failed.
0x8000 0000 0003 0000	EINTERRUPT	The function was interrupted.
0x8000 0000 0003 0001	ENOENTROPY	Insufficient entropy for generating random numbers.

4.6 SOFTWARE CONSIDERATIONS

The Intel P-SEAMLDR module's functions preserve the CPU register state, except the RIP register, which is incremented to point at the instruction that follows SEAMCALL.

Upon return from an Intel P-SEAMLDR module's function, the VMX working pointer, if any, is invalidated. The VMM can restore its VMX working pointer using the VMPTRLD instruction.

The Intel P-SEAMLDR module's APIs are not interruptible by external events (NMI, SMI, INIT, interrupts). In particular, calls to the SEAMLDR.INSTALL API on the last logical processor in the Intel TDX module's installation session may be a lengthy operation.