White Paper

Client Computing
Platform Security

Intel® Hardware Shield: Trustworthy SMM on the Intel vPro® Platform

Wider adoption of modern DRTM-based computing platforms necessitates dynamically attestable isolation between OS assets and System Management Mode

Authors

Kirk Brannock
Senior Principal Engineer

Amanda Ueno
Strategic Planner

Background

Data can be almost anything (e.g., a cryptographic key used by the operating system, credentials, a confidential document, wireless configuration, etc.). For data to be secure, the environment that contains the data must behave as expected and defend the data when that environment is attacked by an adversary who does not have rights to the data.

Typically, this environment consists of the software application handling the data and the entire stack beneath it. Trustworthiness of this environment may be implicit, e.g., not informed or evidence-based; or explicit, where the data owner has sufficient evidence to convince themselves that the environment is trustworthy. In either case, the decision about the platform's "trustworthiness" is made when the data owner chooses to place their data in the care of the platform. If the data owner does not trust the platform, they will not place their data there. Put another way, it is the act of placing their data on the platform that is the actual decision point regarding trust, regardless of whether that platform is ultimately trustworthy or not.

The question is, therefore, what evidence is available to convince the data owner that the platform and the software are trustworthy and will defend the data as expected? The identity and reputation of the hardware, firmware and software can be used to assess the platform's trustworthiness. A modern platform running modern software provides cryptographic evidence about the platform itself and the software environment by generating cryptographic measurements of these components. Collectively, this measurement represents what is known as the Trusted Compute Base (TCB) for the data in question. These measurements can be used in conjunction with access control mechanisms for critical assets (e.g., cryptographic keys) and for proof of the platform's integrity to an external verifier. This facilitates verified explicit evidence-based platform trust decisions instead of unverified implicit platform trust without precise evidence.

There are two classes of these measurements, which are static and dynamic.

- **Static** – Measurements rooted at platform reset in a component called the Static Root of Trust for Measurement (SRTM). The static TCB includes the SRTM and the subsequent measurement chain, including the entire boot process up to the point the resulting software environment is self-defending. This means all BIOS/UEFI, option ROMs, boot loader, and intermediate software are included in the TCB. The scope of this TCB is quite large and contains many millions of lines of code. Furthermore, there is a huge variety of platforms and operating systems used in traditional general-purpose PCs, and therefore, even identifying what is present on any given platform is effectively impossible. It does make it possible to observe if the environment changes from one boot to the next, which is useful. However, informed assessment of trustworthiness based on static measurements is impractical.

- **Dynamic** – Measurements are rooted in a hardware event known as the Dynamic Root of Trust for Measurement (DRTM). From a TCB perspective, the platform restarts after the static boot has "completed" such that a new TCB can be spawned without a full reboot of the platform. The hardware event plus the DRTM isolate the subsequent dynamically launched sequence of software from the pre- existing static boot.

The DRTM provides an independent and separable root and measurement chain that is small and consistent. As such, a dynamically launched TCB is not subject to the near-infinite variation of the static trust chain. Intel pioneered the DRTM in 2007 on the Conroe and Merom generation of Intel® Core™ 2 Duo processors with Intel® Trusted Execution Technology.

Intel provides several complementary technologies that work together under the Intel® Hardware Shield banner to further reduce the size of the attack surface:

- Intel® Trusted Execution Technology (Intel® TXT)
- Intel® Virtualization Technology (Intel® VT-x)
- Intel® Virtualization Technology for Directed I/O (Intel® VT-d)
- Intel® Runtime BIOS Resilience
- Intel® System Security Report
- Intel® System Resources Defense

This paper is organized chronologically, showing the various generational steps along the path between the Coffee Lake generation (8th Gen Intel® Core™ processors) and the Comet Lake generation (10th Gen Intel® Core™ processors). Figure 1. shows a chronological timeline of the technologies.

This paper is intended to be sufficiently technical to facilitate understanding of how these technologies operate without being so deeply technical as to make it long and unreadable; however, it does assume the reader has a general understanding of Intel TXT, System Management Mode (SMM), and BIOS/UEFI architecture. This paper should not in any way be considered a specification.

8th Gen Intel® Core™ vPro® Processor (CFL): Intel® Runtime BIOS Resilience

Intel contributed SMM Paging Enhancements to Open Source

Intel Runtime BIOS Resilience was introduced in the Coffee Lake generation of the Intel vPro platform, launched in 2018. Earlier, in 2016, Intel developed and published enhancements to the UEFI SMI handler's implementation to EDK II, an open-source implementation of Unified Extensible Firmware Interface (UEFI), to take advantage of the inherent security properties memory paging can provide.

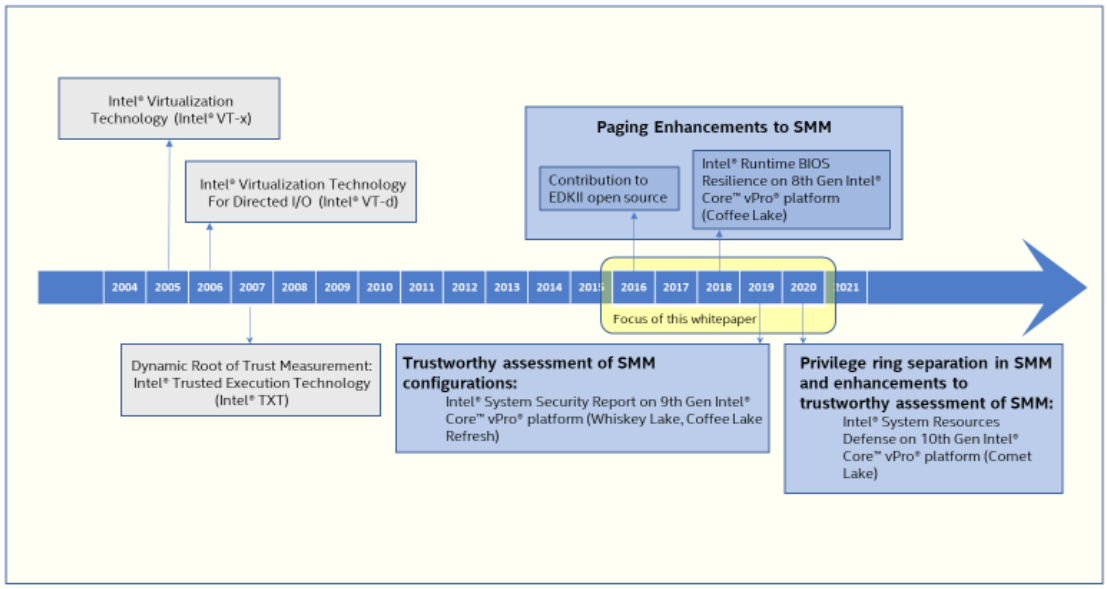

Figure 1. Intel Hardware Shield Historical Timeline

White Paper Scope

This paper focuses on the more recent additions: Intel® Runtime BIOS Resilience, Intel® System Security Report, and Intel® System Resources Defense. Together these technologies provide a hardware-based mechanism that enforces resources access policy on System Management Interrupt (SMI) handlers along with an Intel® Trusted Execution Technology (Intel® TXT) hardware rooted attestation. This makes possible a reduction of the SMI handler's privilege and provides attestation of the resulting SMI handler's properties and access privilege into the system, respectively.

With these changes, rather than allow runtime SMM page table edits as was typical in contemporary BIOS implementations, the SMM page table is constructed such that it is static and once established does not change over the course of a boot cycle. All pages are identity mapped, as is typically the case for SMM.

Furthermore, the page table is constructed defensively in order to grant no more memory privilege than is necessary, e.g.:

- Pages that reference the page table itself are marked as Read-Only as a first-order defense to prevent runtime modification of the page table.
- Code pages are marked Read-Only and are only permitted within System Management memory (SMRAM). This helps prevent SMM code injection.

- All data pages are marked with the Execute Disable (XD) bit to prevent execution of data pages. This also helps prevent SMM code injection.
- Data pages that are not intended to be modified at runtime are marked Read-Only.
- In general, memory outside of SMRAM that is not needed by the SMI handler is not mapped. Typically, all OS owned memory can be removed from SMM's view. If this is done, the potential for meaningful exploit of a latent SMM vulnerability to exploit user data is greatly reduced.

Intel implemented and contributed this portion of Intel Runtime BIOS Resilience functionality to the UEFI open source, driving improvement of SMM security for the whole industry.

Hardware Augmentation with Intel Core vPro Processors

While the paging implementation described above is a big step in the right direction, the protections it enables could theoretically be defeated if an SMM vulnerability allowed a bypass of these paging properties. E.g., loading a different page table in place of the one provided by the BIOS, or relocating the SMI handler base somewhere else in memory, etc.

To reduce the risk of such a bypass, Intel Runtime BIOS Resilience also includes new SMM specific hardware capabilities in the Intel Core vPro processors. These hardware enhancements include several new locks and related changes to architectural behavior when the locks are used, e.g., CR0 lock, CR3 lock, SMBASE lock, etc. This hardware augmentation eliminates several otherwise permitted CPL0 operations while in SMM to help resist exploitation even if there exists a vulnerability in SMM CPL0 code. This "deprivileging" of SMM CPL0 hardens the software portion of the Intel Runtime BIOS Resilience solution without incurring performance or functional deficits in SMM operation and doesn't require rewriting large pieces of SMM code.

Microsoft Windows v1809: Windows Defender System Guard

In sync with the capabilities added to the Coffee Lake Intel vPro platform, Microsoft added requirements for Windows Defender System Guard in Windows 10 version 1809, including DRTM support using Intel TXT along with specific SMM page table properties and SMM page tables are locked on every SMM entry that Intel Runtime BIOS Resilience supports; though, at this point there existed no software mechanism to verify that Intel Runtime BIOS Resilience implementation was configured as per Microsoft Windows Defender System Guard requirements.

9th Gen Intel® Core™ vPro® Processor (WHL): Intel System Security Report

The improvements made to SMM security in the Coffee Lake generation of the Intel vPro platform (2018) were impossible to objectively assess, since they were completely contained in SMRAM and not visible to the OS. This meant that there was no practical opportunity to factor the presence of Intel Runtime BIOS Resilience into OS or IT policies, e.g., network conditional access policy. Given the theoretical possibility of a subverted/rogue SMI handler, it is not particularly useful to directly ask the SMI handler about itself. This information needs to be made available in a verifiable way that is not easily tampered, even in the presence of a subverted/rogue SMI handler. Therefore, a separate mechanism is required.

Intel System Security Report was introduced in the Whisky Lake and Coffee Lake Refresh generation (9th Gen Intel Core vPro processors) of the Intel vPro platform to satisfy these attestation requirements. To facilitate robust and correct attestation, some minor changes were made to the implementation of Intel Runtime BIOS Resilience. They are:

1. The SMM entry code was delivered as an Intel signed binary rather than as source code to be compiled as part of the platform's BIOS build. The SMM entry code is the first bit of code in the SMI handler that runs when an SMI occurs. It is responsible for enabling Intel Runtime BIOS Resilience, e.g., loading the page table (which was constructed by BIOS POST code) and configuring Intel Runtime BIOS Resilience hardware enhancements. The integrity of the SMM entry code must be provable and attestable as it is within the TCB of Intel Runtime BIOS Resilience.
2. A component designed to assess and report the security properties of the SMI handler (including the entry code) is known as the Platform Properties Assessment Module (PPAM). The PPAM resides in the BIOS System Management RAM (SMRAM). The PPAM can inspect the SMM entry code to verify its integrity and inspect the page table to determine the SMM memory access properties implemented on the platform.
3. A PKCS7 certificate providing a digital signature for the PPAM. This is used by an Intel TXT-based measured launched environment (MLE) to verify the integrity of the PPAM.

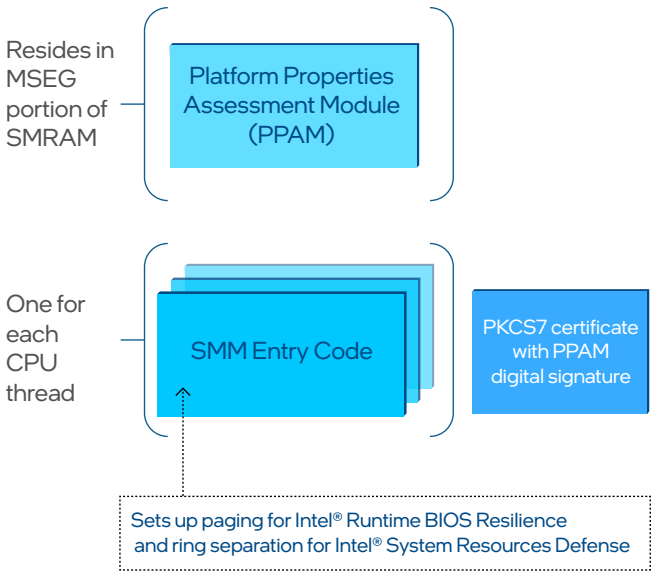

Figure 2. Intel System Security Report components

SMM Entry Code Binary

The SMM Entry Code is the first bit of code in the SMI handler in response to an SMI. It is responsible for enabling Intel Runtime BIOS Resilience, e.g., loading the page table and configuring Intel Runtime BIOS Resilience hardware enhancements.

Before the inclusion of Intel System Security Report, this code at the entry to the SMI handler was assembled from the source as part of the platform's BIOS build. For systems supporting Intel System Security Report, Intel supplies this SMM entry code to platform manufacturers as a binary.

The BIOS boot process overlays this binary at the SMI entry point in SMRAM for each CPU. This early code within the SMI handler is within the TCB of Intel Runtime BIOS Resilience (and Intel System Resources Defense, which was introduced in later platforms and described on page 5). By including the SMM entry code as a binary, it is possible to normalize its cryptographic hash of this bit of the TCB such that it can be verified by the PPAM (more on this in Trust Relationship: PKCS7 PPAM Certificate section below).

When an SMI occurs, each CPU hardware thread enters SMM at the entry point established by BIOS during POST. This entry point is, by convention, set within SMRAM to make sure the SMI handler is isolated from other software. The SMM Entry code will transition to protected mode, turn on paging, and transition to long mode (64-bit mode). It will then configure the Intel Runtime BIOS Resilience hardware feature before passing control to the platform manufacturer's BIOS's SMI rendezvous code where each hardware thread is coordinated and SMIs are processed.

Intel TXT handling of SMM

Before exploring the PPAM in detail, it is necessary to understand how Intel TXT handles SMM, and the inclusion/exclusion of the SMI handler in the Intel TXT Measured and Launched Environment (MLE). Because SMM code is within the trust boundary of an Intel TXT launched software environment, a dynamically attestable mechanism is required to describe SMM and its access privilege reach into the platform. The Intel TXT architecture for SMM was designed to satisfy three basic requirements:

1. SMM bounding policy establishment/negotiation between BIOS and MLE
2. Enforcement of this policy at runtime
3. Attestation of both the policy enforcement mechanism and the policy itself

Because of the complexity involved, a dynamically attested platform opt in/out for these capabilities was also included.

History: SMI Transfer Monitor

Client Intel® TXT[2] and Intel® VT-x architectures were originally designed to allow for encapsulation and virtualization of the platform SMI handler using an SMRAM resident peer monitor. This peer monitor was known as an SMI Transfer Monitor (STM). This is part of the instruction set that implements the Intel VT-x architecture, which is known as VMX.

Measurement of the STM was to be done by Intel TXT SINIT Authenticated Code Module (ACM). Subsequent configuration of the STM was to be done by an Intel TXT launched MLE using special behavior of the VMX VMCALL instruction when executed from the hypervisor root mode. This configuration process would be done with SMI masked and would establish bounds for the BIOS SMI guest that the STM would enforce once the STM hypervisor was "switched on." Once fully configured, the STM would intercept all SMI events and transfer control to the SMI handler code running within a VT guest, enforcing established bounding policies accordingly.

While architecturally complete regarding the three requirements listed above, the STM faced many implementation and deployment challenges stemming from its inherent complexity.

Platform Properties Assessment Module (PPAM)

While the PPAM superficially resembles a traditional STM, it is different in function and is conceptually a subset of an STM.

A PPAM and an STM share the same binary header format and VMCALL invocation mechanism. Additionally, when the platform and MLE have both opted into PPAM (or STM) support, Intel® TXT will launch the MLE with the SMI masked to facilitate the evaluation of SMI access policy without interference from SMM.

However, unlike an STM, a PPAM does not function as a peer monitor or hypervisor within SMM. Instead, it is entirely transient around the VMCALL functions and plays no role in policy establishment or enforcement. Policy establishment is performed by the BIOS POST, and policy enforcement by the SMM Entry Code. SMI delivery and handling are executed using legacy SMI delivery mechanisms once the SMI event is re-enabled by the MLE.

Like the SMM entry code binary, the Platform Properties Assessment Module (PPAM) is a binary that is embedded into the BIOS SMRAM; specifically, in the MSEG portion of SMRAM originally designed to contain an STM. The purpose of the PPAM is to report the integrity of the SMM entry code as well as the details of the platform defined SMM access policies to an Intel TXT launched MLE.

A PPAM obtains the platform's SMI handler required access policy directly from inspection of the SMM Entry Code. Like an STM, a PPAM provides a report describing this access policy to the MLE via VMCALL. However, unlike an STM, the PPAM plays no direct role in the enforcement of this policy, nor does it ever install a virtualization layer into SMM. With a PPAM, the BIOS SMI handler runs without virtualization, just as it always has.

Enforcement of SMI Access Policy

Enforcement of the SMI access policy is done using the SMM Entry Code. The SMM Entry Code stands up all the enforcement needed to support Intel Runtime BIOS Resilience based memory policy at each SMI entry. The SMM entry code is designed to be self-contained and has no external references before it completes and hands control to the BIOS SMI handler. Code within the SMI handler that runs after the SMM Entry Code is constrained by the environment set up by the SMM Entry Code and enforced by Intel Runtime BIOS Resilience hardware extensions.

PPAM Reporting Function

Like an STM, the PPAM supports reporting of the SMI access policy via a VMCALL mapped function. The MLE must invoke the PPAM on each thread following the Intel TXT launch when SMI is masked. This must be done on every hardware thread to ensure consistency between threads. The PPAM will compute the hash of the SMM entry code and compare it to a reference hash

embedded in the PPAM image itself. By way of this cryptographic binding, the SMM entry code is effectively an extension of the PPAM.

After verifying that the correct SMM entry code is installed, the PPAM parses out certain configuration data embedded within the entry code that has been fixed up by BIOS POST code, e.g., the physical address of the SMM page table. Using this data, it can perform consistency checks on them and then derive the totality of the policy established by the SMM entry code. These configuration data are normalized when calculating the entry code hash, facilitating a common SMM entry code solution for all platforms of the same generation.

After verifying the SMM entry code hash, the PPAM then uses the embedded configuration data within the entry code to derive the rest of the security properties of the SMI handler. For example, the PPAM uses the CR3 base address within the entry code to locate and then walk the SMM page table. The footprint and access rules for all memory visible to SMM are therefore, known to the PPAM. This memory map is reported back to the calling MLE as a set of memory descriptors where further OS policy decisions can be made based on the contents of the PPAM report.

Trust Relationship: PKCS7 PPAM Certificate

An Intel TXT launched MLE trust chain begins with the CPU hardware via execution of the GETSEC[SENTER] instruction. The CPU first verifies a cryptographic signature over the SINIT ACM. It then then initializes the DRTM PCRs in the TPM and begins the dynamic measurement chain by extending the hash of the SINIT ACM. SINIT operations verify relevant platform configuration parameters and perform a set of prescribed extend operations to dynamic PCRs.

During the Intel TXT launch process, the SINIT module will extend the hash of the PPAM to PCR17. On Intel vPro platform-based systems[1], the SINIT ACM ensures that the PPAM is configured identically across all hardware CPU threads, so there is no risk of PPAM differing between threads. The MLE can verify the TPM log with the final PCR value to determine the "as measured" PPAM hash.

The reference measurement for this "as measured" PPAM hash is provided by the platform via a PKCS7 certificate with a well-known signing key. This certificate provides the cryptographic anchor for the PPAM. Since the PPAM image contains the hash of the SMM entry code, the hash of the SMM Entry code is part of the PPAM measurement in PCR17. Therefore, the SMM entry code is rooted to the PKCS7 certificate also.

The MLE can establish trust in the PPAM based on trust in the PPAM PKCS7 certificate. This trust can therefore be extended to the report generated by the PPAM regarding the SMM entry code properties and the platform SMM access policy.

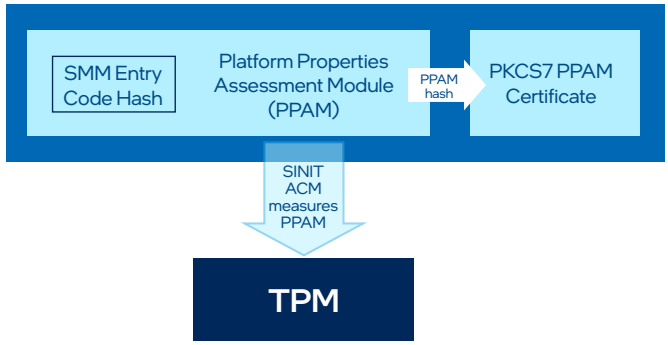

Figure 3. Trust relationship: PKCS7 Certification

Microsoft Windows v1903: Windows Defender System Guard

In step with the capabilities that Intel System Security Report added to the platform, Windows 10 version 1903 updated Windows Defender System Guard support to include SMM Firmware Measurement.

For the first time, visibility of the SMI handler's reach (and its limits) was made possible facilitating a more informed understanding of risk posed by the scope of the TCB.

10th Gen Intel® Core™ vPro® Processor (CML): Intel System Resources Defense

While maintaining all the properties of Intel Runtime BIOS Resilience, the Comet Lake Intel vPro platform (launched 2020) added privilege ring separation between the SMM entry code and the SMI handler code. The SMM entry code is extended to become a CPL0[2] access policy enforcement shim, and the rest of the SMI handler is run at CPL3. This ring separation within SMM provides the necessary mechanism to create and enforce policies for MSRs, I/O ports, and other registers, in addition to the memory restrictions provided by the paging described above.

Continuing the theme of running SMI handlers with "least privilege," with the ring separation in place, the page table remains immutable, even to the CPL0 code. Like the memory policy contained within the page table, the new SMM access policies included in Intel System Resources Defense are configured during BIOS POST and locked in place using the immutable page table. Since CPL0 cannot modify the page table, neither can it modify the declared policies.

Therefore, the overall architecture and trust model remains the same as it was in the Whiskey Lake (9th Gen Intel Core vPro processors) platform. As logically follows, Intel System Security Report was extended to include the new resource types supported by Intel System Resources Defense in its report of SMI access policy. PPAM simply checks a few more things and includes more information in its report.

Microsoft Windows v2004: Windows Defender System Guard

Given the additional platform security capabilities and visibility in the Comet Lake Intel vPro platform, commensurate changes were made to the OS requirements for Windows Defender System Guard in Windows 10 version 2004.

Concluding Observations

Over the course of three platform generations, by refactoring SMM implementation and adding Intel-specific hardware features, the Intel vPro platform has reduced the exploitation potential of SMM, even if the SMI handler contains vulnerable code.

Unique to Intel, these changes are fully attestable via an Intel TXT DRTM launch in a manner that is designed to be free from interference from potentially exploitable SMM or pre-boot firmware. These technologies enable the OS to explicitly verify the isolation between OS assets and SMM rather than implicitly trusting the platform without strong evidence.

The net consequence of this evolution is that the contemporary Intel vPro platform delivers state-of-the-art platform security and enhanced isolation of user data from SMM while preserving the required platform utility which SMM provides.

[1] Might not apply to all workstation systems using server-based CPUs.
[2] CPL is the current privilege level of the currently executing task on the processor. CPL0 is CPL at Level 0, the most privileged level and CPL3 is at CPL Level 3, the least privileged level.

0521/KBAU/INTEL/PDF