SoK: Confidential Quartet - Comparison of Platforms for Virtualization-Based Confidential Computing

Guanciale, Roberto; Paladi, Nicolae; Vahidi, Arash

Published in:
Proceedings of the 2022 IEEE International Symposium on Secure and Private Execution Environment Design (SEED'22)

DOI:
10.1109/SEED55351.2022.00017

2022

SoK: Confidential Quartet - Comparison of Platforms for Virtualization-Based Confidential Computing

Roberto Guanciale
Royal Institute of Technology (KTH)
Stockholm, Sweden
robertog@kth.se

Nicolae Paladi
Lund University and CanaryBit
Lund and Stockholm, Sweden
nicolae.paladi@eit.lth.se

Arash Vahidi
Research Institutes of Sweden (RISE)
Lund, Sweden
arash.vahidi@ri.se

***Abstract*—Confidential computing allows processing sensitive workloads in securely isolated spaces. Following earlier adoption of process-based approaches to isolation, vendors are now enabling hardware and firmware support for virtualization-based confidential computing on several server platforms. Due to variations in the technology stack, threat model, implementation and functionality, the available solutions offer somewhat different capabilities, trade-offs and security guarantees. In this paper we review, compare and contextualize four virtualization-based confidential computing technologies for enterprise server platforms - AMD SEV, ARM CCA, IBM PEF and Intel TDX.**

***Index Terms*—Confidential Computing**

I. Introduction

Third-party cloud services allow organizations to increase computing capacity with minimal investments in hardware, infrastructure and staff. They enable IT infrastructure to scale dynamically and on short notice to meet changing business demands [6].

Cloud service providers (CSPs) commonly maintain a fleet of servers and ancillary equipment rented to customers either as dedicated or time-shared resources. This is often achieved through virtualization which isolates tenant virtual machines (VM) from each other. Historically, the CSP has been trusted with full access to customer VMs. However, to achieve true confidential computing where customers can deploy privacy sensitive workloads, additional hardware support is needed to eliminate the need for this explicit trust.

The current four leading platform vendors[1] have all published their VM-based confidential computing solutions: AMD SEV (Secure Encrypted Virtualization) [1], ARM CCA (Confidential Compute Architecture) [3], IBM PEF (Protected Execution Facility) [20] and Intel TDX (Trusted Domain Extensions) [2].

In this paper we provide the first comparison of these solutions. While these technologies aim to achieve similar goals, each does so in a slightly different manner. Note that some of these solutions are not yet available (due to missing firmware and/or hardware) and may remain so for at least another year. This rules out quantitative comparisons such as performance overhead. Still, it is important to perform a preemptive architectural analysis before hardware and software are ready, to better understand general capabilities of each platform and their functional and security differences.

[1]Due to space constraints we limit this work to the major commercial platform providers. Keystone for RISC-V [27] was excluded due to lack of commercial availability and its many similarities to IBM PEF.

A. Contribution and outline

In this work we investigate four technologies, study their differences, and evaluate how they affect function and security. We provide the following contributions: we review the cloud business model, introduce the necessary terminology, and describe the cloud threat model (Section II). After investigating important aspects of the four proposed solutions (Section III), we discuss the security guarantees provided by each solution (Section IV), analyse a sample application (Section V) and review related work (Section VI). While the main focus of this work is spatial isolation (protection of VM memory and registers), we identify several other comparison dimensions for further investigation (Section VII).

II. Background

The recent evolution of Confidential Computing is driven by the need to protect computations on shared machines operated by CSPs [38]. In a cloud environment, sensitive customer data may leak to other tenants due to software issues, misconfiguration or other vulnerabilities [41]. Furthermore, the customer and the CSP (who has full control over servers, network and storage) may have conflicting business goals. For example, Amazon provides cloud services to multiple retail companies while at the same time competing with them via its online marketplace. Further complicating matters, a CSP may also operate under foreign law incompatible with the customer's goals or national security and privacy regulations [16]. This motivates creating an improved cloud environment where security and privacy cannot be bypassed even by the CSP. Some solutions proposed by enterprise server vendors aim to enable this. Mainstream CPU architectures are introducing support for creating isolated environments with integrity and confidentiality guarantees for the deployed workloads. This is coupled with remote attestation protocols, allowing a third party obtain information that helps them assess the trustworthiness of a target isolated environment.

A. Important terminology

The following terminology is used throughout this work. We use vendor-neutral terms whenever feasible, to avoid confusion and simplify comparison.

A system *Virtual Machine* (VM) is a computer system executing in an isolated environment with limited access to some resources. It is managed by a privileged software component called *Virtual Machine Monitor* (VMM) or *Hypervisor*.

A *Trusted Execution Environment* (TEE) is an execution environment that provides confidentiality and integrity guarantees for the applications it hosts, with the help of various isolation mechanisms [39]. We call VMs hosted by a TEE as *secure* VMs. These can coexist with legacy VMs, which we refer to as *normal* VMs.

Analogously, memory protected by a TEE is considered *secure memory*, while the remaining memory is called *normal memory*. Furthermore, irrespective of type, a VM will access memory using *(guest) virtual* addresses. This may be translated to *intermediate* addresses and then to *physical* addresses according to page tables maintained by the VM and hypervisor or TEE respectively.

In our context the *Trusted Computing Base* (TCB) comprises the combination of hardware and software elements that implement the TEE on a given platform.

Attestation is a process where an *attester* provides claims to an *appraiser* (or *verifier* in IETF terminology [39]) about the properties of a target and supplies evidence supporting those claims [13], to be appraised in a form that can be trusted by a *relying party* [19]. A verifier uses attestation to collect information about the TEE to assess its trustworthiness. Attestation may rely on device unique credentials to prohibit cloning and emulation.

B. Cloud threat model

In the cloud model we consider the following actors:

- **Cloud Service Provider** (CSP) operates all infrastructure, has physical access to all machines and controls all platform software except the trusted firmware.
- **Cloud customer** deploys possibly sensitive workloads to execute in virtual machines hosted by the CSP.
- **Silicon provider** is a trusted entity providing the hardware and trusted firmware components.
- **Adversary** aims to break *confidentiality*, *integrity* and to a limited extent *availability* of a customer's workload.

The adversary may be another cloud customer on the same platform as the target cloud customer. She may attempt to access [17] or modify the target cloud customer's data [25], or affect availability by hindering the target cloud customer's VM to execute properly.

The adversary may also act with CSP privileges, while attempting to breach *confidentiality* or *integrity* by accessing internal state, memory or storage of the customer VM through higher privilege software, via other system components such as direct memory access (DMA) engines, or by physical probing. We do not consider *availability* for the malicious CSP, who could simply refuse to run the customer's software. Furthermore, we do not consider micro-architectural side-channels attacks.

A vulnerable or malicious CSP could mean a malicious hypervisor with additional memory access privileges and control of intermediate-to-physical memory mappings of each VM. This could lead to attacks against VM integrity, where the hypervisor may for example replace VM memory with older content (replay attack), or modify true physical mappings of the VM memory (e.g. aliasing and re-mapping attacks). Hence, while the hypervisor is a privileged component, in this threat model it is not part of the TCB and may in fact be malicious and attempt to compromise the TCB.

III. Virtualization-based Confidential Computing Technologies

Common components in VM-based confidential computing are illustrated in Figure 1. Normal VMs can share memory to communicate with each other (1) or with the hypervisor (2). Secure VMs are not normally accessible by the hypervisor and are instead managed by a new software layer. This software layer may also oversee other virtualization functions such as handling hypercalls and interrupts (3). Note that the VM operating system (OS) may need to be modified to support execution in this new environment. Finally, some security mechanisms protect secure memory regions from direct access by the hypervisor or other unauthorized entities (4). We next describe four vendor-specific implementations of confidential computing, noting the approaches to memory encryption, memory integrity, and attestation.

A. AMD Secure Encrypted Virtualization

AMD SEV was introduced with the Zen architecture to improve cloud security. In this work we consider SEV-SNP, which is the latest revision of a series of related AMD security technologies:

- AMD SME (Secure Memory Encryption) provided transparent encryption and decryption of off-chip memory [24].
- AMD SEV (Secure Encrypted Virtualization) extended memory encryption to virtual machines [24].
- AMD SEV-ES (Encrypted State) extended encryption to other CPU resources such as saved registers [23].
- AMD SEV-SNP (Secure Nested Paging) adds integrity protection to encrypted memory [1].

1) Memory encryption: As shown in Figure 2-(a), dedicated memory encryption hardware and the Platform Security Processor (PSP) transparently encrypt external memory accessed by application cores. The new field "C-bit" of a page table entry decides if encryption is enabled, as illustrated in Figure 2-(b). As there are two page tables in AMD-V (gPT and nPT controlled by VM and hypervisor respectively), the encryption key is decided by which C-bit is set (VM C-bit

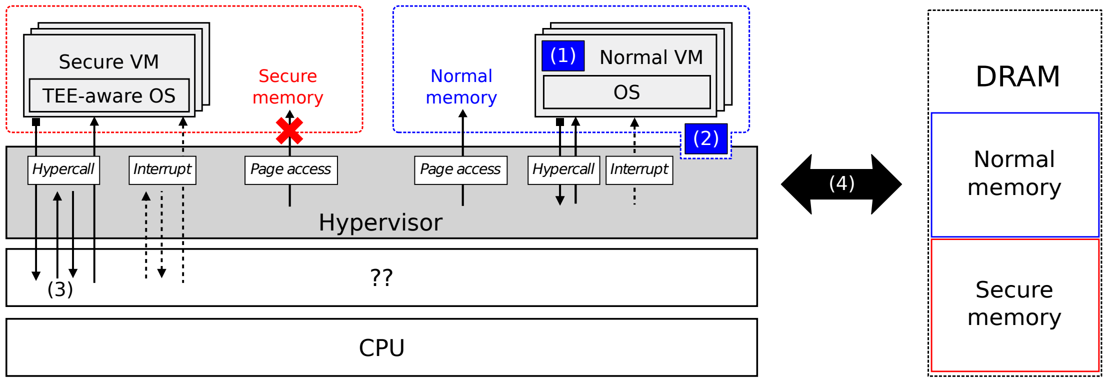


Fig. 1: Common concepts in virtualization-based confidential computing technologies

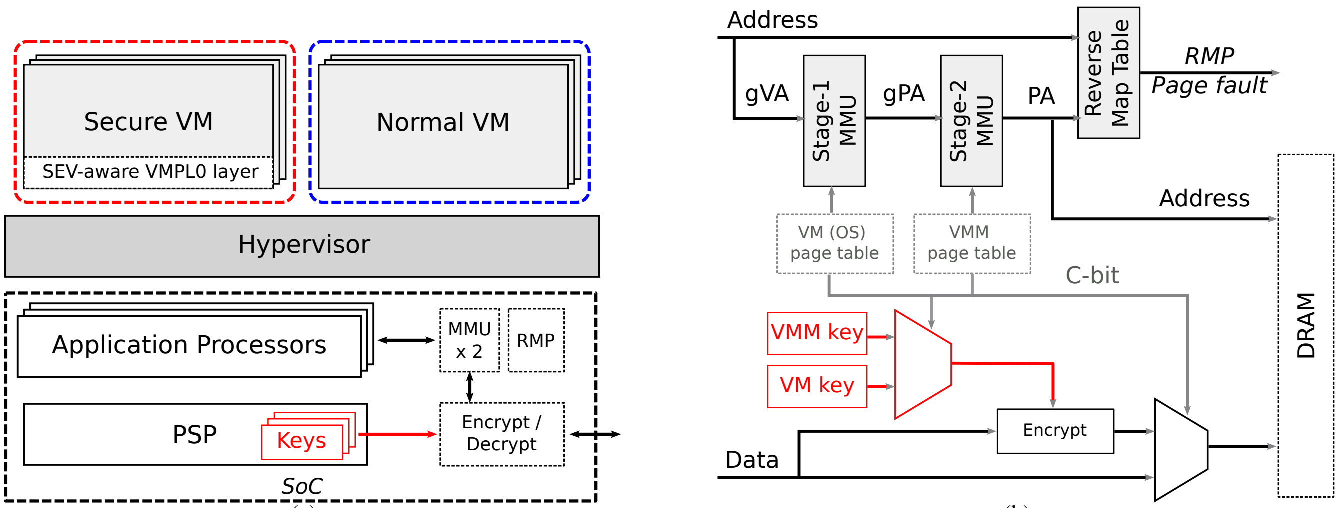


Fig. 2: Memory encryption in AMD SEV with dedicated hardware and processor for key management (a) but controlled by VM and hypervisor page tables (b).

has priority). This could for example allow sharing memory pages among VMs while still being encrypted in DRAM by the hypervisor key.

As a higher privileged component, a hypervisor would normally have access to VM internal state including various CPU registers. Since SEV-ES this is avoided by storing the VM state in encrypted memory during a context switch. If hypervisor involvement is required, the VM must contact the hypervisor and explicitly supply any required information.

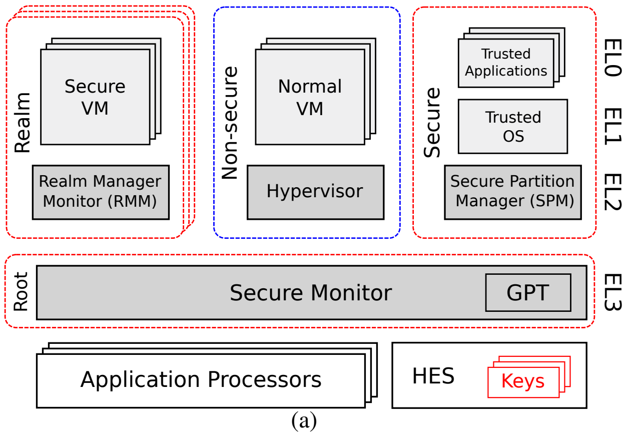
2) Memory integrity: SEV-SNP introduced Reverse Map Table (RMP) to restrict physical page ownership (write access) to one entity. Change of ownership always erases page content and notifies the owner. This achieves integrity without use of cryptography, since a memory read either returns the last written value or fails altogether. An exception to this is the guest context page (which contains various guest keys) that has additional cryptographic integrity protection. Note that the RMP does not prohibit a hypervisor from reading encrypted VM memory. Furthermore, handling re-mapping and similar hypervisor attacks requires the VM to keep track of its own physical pages.

3) Keys hierarchy: A SEV-SNP-enabled CPU contains a unique P-384 ECDSA Chip Endorsement Key (CEK), signed by an AMD SEV Signing Key (ASK). To create a key unique to device and firmware version, Versioned CEK (VCEK) is derived from CEK plus TCB firmware version. The VM owner may also provide ID and AUTH public keys (for signing and encryption), to be injected into the VM before launch. Finally, the VM Encryption Key (VEK) for memory encryption is generated by PSP.

4) Attestation: At runtime the VM can request an attestation report from the PSP, which will cover VM memory contents and layout measured before launch plus platform and VM configurations and is signed by VCEK. The attestation report will also include the owner's ID and AUTH keys, a 256-bit custom hypervisor data (provided at launch) and a 512-bit custom guest data (provided during attestation, for example to prove freshness to a remote party).

B. ARM CCA

ARM CCA is an enhanced isolation technology introduced in ARMv9-A [3]. This new architecture retains the four



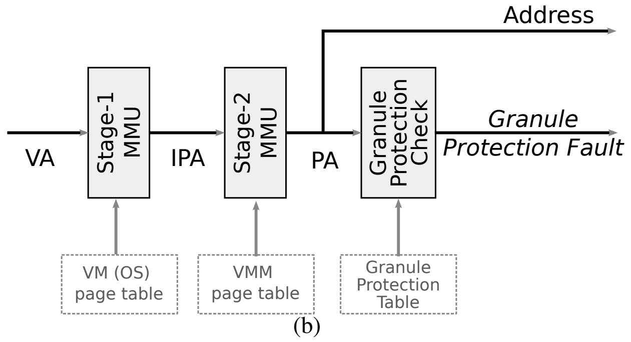


Fig. 3: Different layers in ARM CCA (a) and memory isolation mechanisms (b).

exception levels of Virtualization Extensions (from ARMv7 and v8) while extending the two security states of TrustZone (ARMv6) to four: *Non-Secure*, *Secure*, *Root* and *Realm*, as illustrated in Fig. 3-(a). Realms represent areas where a group of related VMs can execute isolated from all others. Realms are managed by a new component called the Realm Manager Monitor (RMM). Secure separation between normal (Non-Secure), Secure and the new Realm is managed by the Monitor in Root, which is the highest privileged software executing on the application processors.

An important difference between ARM CCA and the other solutions is that ARM no longer manufactures CPUs. Instead, their designs are licensed to other companies who may manufacture them after some modifications. Furthermore, some key aspects of ARM CCA are currently undefined or undocumented. For these reasons, based on ARM recommendations and development plans, a number of assumptions have been made about CCA: (1) Hardware Enforced Security (HES) is available as a dedicated security CPU, (2) memory encryption is provided by dedicated hardware using an address-tweak mode, (3) and each Realm is assigned a dedicated key upon initialization.

1) Memory protection: Each security state manages its own address space using the two-stage MMU. The Granule Protection Check (GPC) provides a third stage that instead of performing address translation validates access to physical memory regions for different security states, see Fig. 3-(b).

2) Memory confidentiality and integrity: Secure, Root and Realm security domains reside in encrypted memory. The encryption keys are generated during boot and are stored in HES (which is accessible only from Monitor). Each Realm instance is given a different key, thus despite memory encryption VMs residing in the same Realm will be able to communicate using shared memory. A normal virtual machine (in EL0/1) can be migrated to a Realm (R_EL0/1), which encrypts the VM memory and removes it from direct control of the hypervisor (EL2).

Memory integrity is provided by the GPC which enforces memory separation between different security states. To improve security, only Monitor can interact with GPC and the GPC configuration (GPT) must reside in Root address space. In addition, Monitor always executes from secure on-chip memory.

3) Attestation: At runtime, a Realm can request an attestation report that will contain two main components: a Realm measurement provided by the RMM and the platform measurement provided by Monitor. Authors assume that an attestation key derived from a device secret will be used to sign this report.

C. IBM Protected Execution Facility

IBM introduced the Protected Execution Facility in the POWER 9 architecture to enable isolated execution of workloads in cloud deployments [20][2]. The PEF design relies on four components: secure mode (a new privilege state), new highest privilege firmware (called *ultravisor*), access control and secure memory, see Figure 4-(a).

1) Memory Protection: The PEF mechanism supports partitioning memory into secure and normal memory, and the amount of secure memory can be configured at boot time. Each location in main storage has an associated Secure Memory property, mem_{SM}, where memory with $mem_{SM} = 1$ is defined as “secure memory”, while memory with $mem_{SM} = 0$ is regular memory. Methods and granularity of mapping the Secure Memory property to main storage are implementation specific. The Secure Memory property is cached in the TLB and in implementation-specific lookaside buffers [22].

To support secure memory, PEF introduces a new most-privileged state labelled secure mode, which is defined by a bit in the Machine State Register, where $MSR_S = 1$. In normal state either the hypervisor or the VM are executing, while in secure state it is either the ultravisor or the secure VM.

Hardware enforces access control by only allowing processes running in the secure state to access secure memory, as illustrated in Figure 4-(b). During context switches, the

[2]New hardware features announced in POWER10 enable additional security functionality, but lack software support and are not considered in this work.

ultravisor uses secure memory to store a VM's registers, which are now made unavailable to the hypervisor.

2) Memory Confidentiality and Integrity: Whenever software not executing in secure state (such as the hypervisor) attempts to dump VM memory pages, the ultravisor encrypts data and copies it to normal memory. The ultravisor uses a symmetric key derived from a seed provided with the VM image. The seed is wrapped in a public key associated with a private key in the TPM of the target system, such that only the authorized platform TPM can extract the seed and communicate it to the ultravisor. The ultravisor encrypts data using AES GCM (Galois-Counter Mode) [34] on a page-level granularity. VMs operate on 64KB pages while ultravisor page size is 2MB, allowing the ultravisor to store authentication tags later used to detect hypervisor tampering.

3) Attestation: PEF supports attestation of both VM and platform integrity. Tenants use the platform TPM to verify that the firmware state - recorded in a TPM register - is in a *correct state*, i.e. corresponding to an expected reference value. The integrity of a secure VM is locally verified by the ultravisor prior to launch, using information from the Enter-Secure-Mode (ESM) operand inserted in the VM image at build time.

D. Intel TDX

Intel TDX extends and reuses Intel multi-key total memory encryption (MKTME) and a CPU attested software module to provide isolated execution of secure VMs. The Intel TDX module is a digitally signed software module that is executed in a new processor mode, called SEAM. The TDX module acts as a separation kernel and provides an interface to the hypervisor to schedule, create, and manage secure VMs, which are called Trusted Domains, as illustrated in Figure 5-(a). The Intel TDX module is implemented and distributed by Intel, the hypervisor is provided by the cloud service provider, and each VM is provided by a different cloud customer.

1) Memory protection: TDX enforces isolation of secure memory among secure VMs and the hypervisor. The hypervisor can request TDX to add memory pages to the secure memory of a VM, which must explicitly accept the page. When a page is accepted by a VM, the TDX module keeps its state and unique ownership in the PAMTs data structures and zeroes the memory page. A VM is free to set up the first stage memory translation using page tables in secure memory. The highest bit of intermediate addresses is used to identify secure memory. Second stage translation of non-secure memory is directly managed by the hypervisor. Configuration of second stage translation for secure memory is mediated by the TDX module. The TDX module uses PAMTs to prohibit multiple owners, prohibit intermediate address aliasing of the same physical page, and enforce VM isolation. Accessing a secure page sets the SEC bit of the ECC memory, which is used to prevent hypervisor accesses to the secure memory content, see Figure 5-(b). This mechanism is also used to protect Intel TDX data structures and code, secure second stage page tables, and VM registers during context switching, making them inaccessible to the hypervisor.

2) Memory confidentiality and integrity: Intel TDX uses MKTME to transparently encrypt secure memory of VMs. When a secure VM is created, MKTME assigns a unique key to encrypt the VM's secure memory. Neither the VM nor the hypervisor has access to the private key. Additionally, MKTME is used to save in the ECC memory a SHA-3 MAC truncated to 28 bits for each cache line. When a VM accesses a cache line with an incorrect MAC it receives an exception. Furthermore, the memory interface is designed to return dummy values whenever memory with ECC SEC bit set is read by the hypervisor.

3) Attestation: When a secure VM is created, Intel TDX initializes the measurement registers by measuring the TDX TCB. The TDX module extends these measurements when the hypervisor requests to add a set of initial pages to the VM. The module also provides special registers that the VM can extend with measurements of additional code and data at runtime. Challenger requests are processed in two steps: (1) the VM requests the TDX module for the VM's report, which is generated by the CPU and is integrity protected using a MAC; (2) the VM requests the hypervisor to convert the report to a remote attestation, which is performed by an SGX quoting enclave that verifies the report MAC.

IV. Comparison

Despite many similarities, we observe some key differences among the presented technologies. These manifest in how a threat is met, implementation details, or simply as a consequence of existing architectural differences. In this section we investigate these differences from a security perspective.

A. Secret Deployment

Secure VMs may need to execute confidential software or process confidential data, which requires a method to deploy secrets. This is often done by first deploying a normal VM and later injecting the confidential components. The solutions we reviewed generally adopt one of the following approaches: set up a secure communication channel and manually transfer secret components, or *seal* secret components with a key available only to a correctly verified VM.

IBM PEF supports a slightly different mechanism where secrets are placed in an ESM operand at image build time. The ESM header contains a seed sealed to the target TPM. The ultravisor reconstructs the secret component and injects it into the VM image before launch.

B. Encryption

CPU access to external memory may occur in cache line sized chunks. Hence, the encryption scheme must securely handle small blocks accessed in a random order. It must furthermore be position dependent to prohibit attacks where ciphertext from one region is replaced with another [33]. To achieve this with good performance, Intel TDX and AMD SEV use AES-128 in a *tweakable* XEX (Xor-Encrypt-Xor) encryption mode [42]:

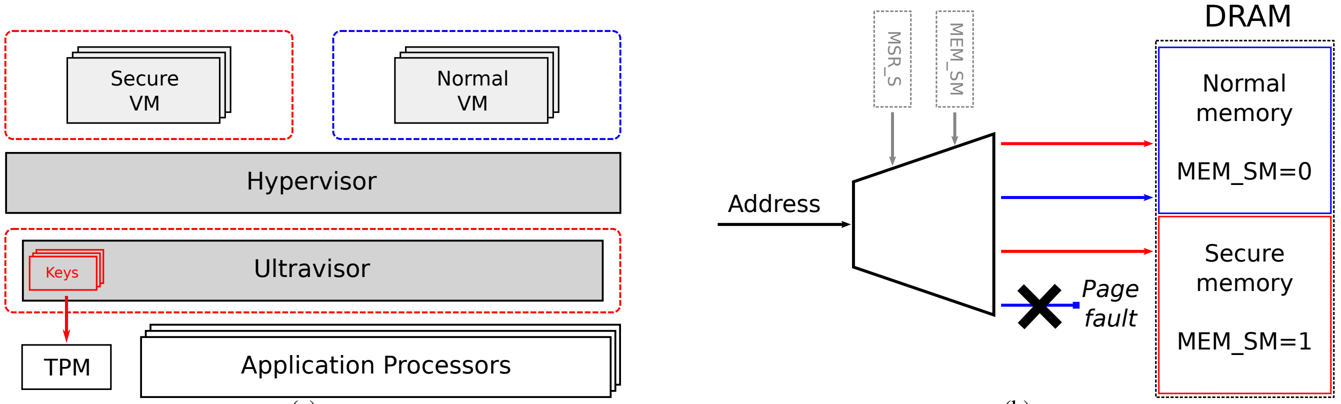


Fig. 4: IBM PEF structure (a) and memory isolation mechanisms for preventing normal VMs accessing secure memory (b).

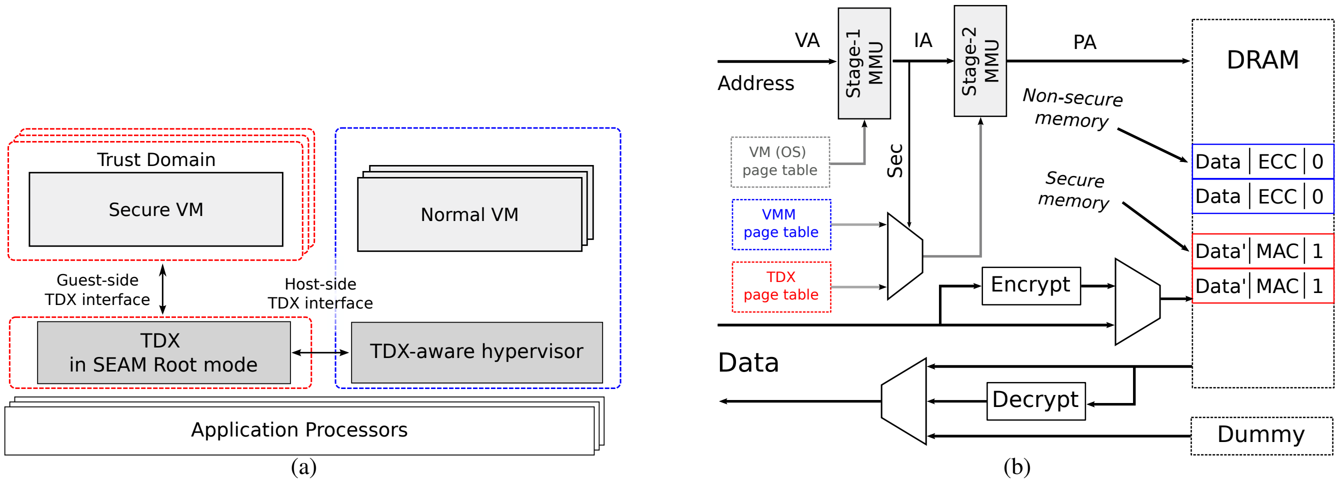


Fig. 5: Intel TDX overview (a) and the memory isolation mechanism (b).

$$C = AES_k(M \oplus f(PA)) \oplus f(PA)$$

Given a physical address, $f(PA)$ generates a 128-bit tweak parameter (SEV uses a 32-bit value repeated four times) [49]. Intel TDX additionally uses a SHA-3 based MAC truncated to 28 bits for integrity protection. Note that alternative design choices have led to severe security attacks [37], [29]. ARM CCA recommends use of address tweak but does not specify the encryption algorithm and mode[3]. IBM PEF currently does not implement memory encryption due to the lack of hardware support [20].

C. Protection from other VMs

VM isolation is one of the core objectives of a hypervisor and is provided by existing technologies, for example by employing a 2-stage MMU. The four solutions additionally guarantee VM isolation by mediating allocations of secure pages by the hypervisor to VMs, and forbidding multiple ownership. In all reviewed solutions, resource management and scheduling are handled by the hypervisor (or other privileged components), which can prevent denial of service attacks from malicious VMs.

[3]This allows silicon vendors to select one based on their performance, power and area requirements

D. Protection from Hypervisor attacks

The four solutions use different mechanisms to prevent the hypervisor from accessing secure memory. In IBM PEF, secure memory regions are statically defined during boot and cannot change at runtime, which allows easy identification of forbidden hypervisor access. Both AMD SEV and ARM CCA extend page walks using a secure data structure (RMP and GPT respectively) which enables a third MMU/MPU stage to identify authorized memory access. While this is a more flexible approach, it does add some performance overhead as well as additional complexity which can introduce new security vulnerabilities.

Intel TDX uses a substantially different approach by exploiting bits of ECC memory to save metadata at cache line size granularity. Intel TDX is the only solution that allows the hypervisor to modify VM memory. However, these modifications will invalidate the MAC and are detected when read back by the VM. Note that this only provides 28-bits of truncated MAC, which may not be sufficiently secure for some applications. Furthermore, when the hypervisor reads from secure memory the hardware returns a dummy value. AMD SEV on the other hand allows the hypervisor to read encrypted VM memory. This could allow CPU-side attacks by observing memory updates.

AMD SEV and Intel TDX provide protection against replay, re-mapping and aliasing attacks, by ensuring that secure memory is cleared after remapping or ownership changes and that physical pages have at most one owner. We assume a similar functionality could be implemented in ARM CCA within RMM or Monitor. Finally, IBM PEF does not provide any protection against these attacks.

A hypervisor may also try to affect the execution flow, for example by modifying the program counter or the status flags during a context switch. A hypervisor might also extract sensitive information such as keys from general purpose registers. For this reason all solutions (except the original SEV) protect the VM state during a context switch, for example by encrypting the VM state, requiring the VM to manually control transfer of state data or delegating context switch to a trusted component outside the hypervisor.

Finally, no solution guarantees availability as a hypervisor can starve a VM or remove its memory. While there are some mechanisms to prevent hypervisor interference with some operations such as interrupt delivery, these are often non-security functions (e.g. performance improvement such as XIVE in POWER9 [8]) that can be disabled by the hypervisor.

E. Protection against DMA, physical, and offline attacks

Peripheral access to secure memory is normally prohibited to avoid indirect unauthorized access. However ARM CCA introduced a new RME-aware System-MMU which can perform granule page checks for secure access.

A malicious CSP could physically access server memory to extract DRAM contents. This is only possible if the memory is not encrypted, which all solutions except IBM PEF currently support[4]. Physical access enables the adversary to launch several attacks against encryption mechanisms [29], [26]. Likewise, a malicious CSP could physically access server memory to modify DRAM contents. Currently IBM PEF provides no protection against such attacks. Furthermore, as AMD SEV and ARM CCA achieve memory integrity by blocking unauthorized memory writes, they can neither detect nor prohibit outside memory manipulation. In contrast, Intel TDX utilizes a MAC which can help the CPU to detect physical memory manipulation. Furthermore, the tweakable memory encryption will protect against memory pages being physically reordered. No solution provides protection against replay attacks or other types of physical attacks, such as electromagnetic and power analysis, or fault injection. Such attacks may be used to extract device secrets [11].

[4]IBM POWER9 does not support memory encryption. IBM POWER10 allegedly introduces support for transparent memory encryption [21]; however, we did not find any published documentation to support this claim, neither references to transparent memory encryption functionality in the latest version of the POWER ISA [22]

F. Communication

For efficient communications, a VM may need to share memory with other components: i.e., the hypervisor (e.g., for VirtIO [43]) or DMA enabled peripherals.

Also, VMs hosted on the same platform usually communicate via some sort of network interface that is virtualized by the hypervisor. However, the cloud provider may provide the infrastructure for more efficient VM-to-VM communication that is based on shared memory or temporary memory grants, thus creating performance benefits for certain classes of applications [5]. All solutions allow secure VMs to share memory with other components. The hypervisor can protect this memory from attacks coming from other VMs or DMA using legacy mechanisms. However, the four solutions provide different types of guarantees in case of hypervisor or physical attacks.

All solutions prevent DMA peripherals and the hypervisor from accessing secure memory, therefore communications with a secure VM must be done by sharing non-secure memory. This memory is accessible to the hypervisor and can be protected from physical attacks if the platform supports memory encryption with hypervisor-controlled keys (i.e., all reviewed solutions except IBM PEF).

In Intel TDX and IBM PEF, the hypervisor can access memory that is shared among VMs, since they can only share non-secure memory. AMD SEV allows launching secure VMs using the same VEK, which permits inter-VM communications to be protected from hypervisor attacks (even if the hypervisor can observe the encrypted memory and therefore access side channels). ARM CCA allows sharing pages between secure VMs that are protected from the hypervisor via the third stage page tables. Hence, VM intercommunication can be protected from the hypervisor independently of encryption.

Signaling is a further form of communication. All technologies allow a VM to notify the hypevisor via hypercalls. The opposite signaling (i.e., hypervisor to VM) is usually achieved by injecting virtual interrupts and exceptions. Inter VM signaling is implemented by hypervisor, which therefore can act as a man in the middle adversary. Signaling between devices and VMs is usually mediated by the hypervisor, which can control the device and can inject virtual interrupts in the VMs. Some solutions, such as POWER9 XIVE [8], can minimize hypervisor involvement in forwarding interrupts to improve performance.

G. Software Support

Lacking sufficient hardware support for virtualization, early consumer processors primarily implemented virtualization in software. This was realised by removing, replacing and emulating privileged instructions, or even demoting the OS to a lower privilege level and running the hypervisor in its place [15]. Modern processors include hardware support to reduce this effort and can often execute an unmodified OS with minimal performance loss. The current generation of confidential computing offerings takes a small step in the opposite direction and delegates some virtualizations tasks to

Platform	Required software support
AMD SEV	SEV-SNP firmware, **Hypervisor**, *VM TEE-aware OS*, **Attestation Validation Service**, *Attestation Validation Client*
ARM CCA	RMM, SPM, Monitor, **Trusted OS**, **Hypervisor**, *VM TEE-aware OS*, **Attestation Validation Service**, *Attestation Validation Client*
IBM PEF	Ultravisor, **TPM Software Stack**, **Hypervisor**, *VM TEE-aware OS*, *Wrapping the key seed*, **Attestation Validation Service**, *Attestation Validation Client*
Intel TDX	TDX module, Seam module, SGX quoting enclave, **Hypervisor**, *VM TEE-aware OS*, **Attestation Validation Service**, *Attestation Validation Client*

TABLE I: Required software support for confidential computing.

the OS, requiring changes in guest VMs to enable confidential computing support.

Table I reports the software support for confidential computing. In order to simplify adoption of their platform, the Silicon Provider usually implements software support, for example by providing software modules (e.g. IBM ultravisor and Intel TDX module) or patches to existing kernels and hypervisors. However, for operation, the Cloud Provider and Cloud Customer are free to choose alternative implementations of components in **bold** and *italic*.

Required guest VM modifications include adding support for page management (for example deciding if a memory page is private or public, in Intel TDX); support for using PSP functionality (in AMD SEV); support for generating OS image integrity information or a modified bootloader to support a custom boot process (in IBM PEF). Such support may be added through extensions in the mainline kernel or discrete kernel modules, through configuration changes, or by creating a new component more privileged than the kernel (in AMD SEV). Moreover some part of the kernel may not be resilient against attacks coming from the hypervisor, which was considered trusted until the advent of TEE. For example, today device virtualization mainly uses VirtIO [43], which implements a virtual device interface between the VM and the hypervisor. This interface mimics a DMA controller, whose buffer descriptors are stored in shared memory. The hypervisor may inject wrong pointers in these data structures, which requires the kernel drivers to validate the inputs received by the hypervisor.

As hypervisor privileges are stripped away to align with the threat model, hypervisor changes are necessary to support confidential computing. In some cases this can be enabled through kernel extensions.

Considering operating aspects, CSPs need to maintain extensive additional tooling to enable and support confidential computing. For example, CSPs need to extend the deployment pipeline and maintain a separate line of VM images; maintain infrastructure for attestation mechanisms; and further develop or update deployment, monitoring and migration tooling. Enabling confidential computing support has performance impacts and CSPs may opt for disabling it.

H. Migration

Migration is the process of moving a VM from one physical host to another. This can be orchestrated by the VM itself, but a more interesting type of migration is *live migration*, where the hypervisor performs migration with minimal interruption.

Encrypted memory complicates live migration in a secure VM. In AMD SEV an optional TCB component called the *Migration Agent (MA)* is needed to support migration. The source MA re-encrypts the VM content with a key also accessible to the target MA. This allows migration without revealing VM content to the hypervisors or the network. In Intel TDX migration occurs via a similar mechanism, where a Migration TD (MigTD) is a customer software component checking if the source and target platforms meet the customer migration policy and negotiating the transportation key.

IBM PEF currently does not implement live VM migration. ARM CCA currently does not define a migration scheme, although this may be added later.

I. Attestation

Remote attestation is a fundamental functionality of TEEs, allowing a relying party to judge the trustworthiness of a TEE instance and the workload deployed in it [13]. A common approach - adopted by AMD SEV, ARM CCA and Intel TDX - is to support *dynamic attestation*. This allows a relying party to request trustworthiness evidence at any point during the execution of the VM.

In contrast, IBM PEF implements *cached certificate-style* attestation [19] without support for dynamic attestation of VMs. IBM PEF implements a two level trust model: a relying party attests the platform and authorizes deployment of the VM on that platform. The platform firmware subsequently verifies the VM's integrity and authenticity before launching it in secure memory.

V. An example application

To better understand the differences between the four solutions, we consider a hypothetical application with three interacting principals, illustrated in Figure 6. A *data controller* operates on a set of confidential audio recordings and intends to automatically transcribe and index them. A *model owner* provides proprietary AI models for voice recognition. For efficiency, both execute their workloads on the same platform operated by a *cloud service provider*.

The principals each have an asset to protect (recordings, model and cloud system). They do however not trust each other, for example the data controller suspects a malicious model owner or CSP could try to steal or manipulate sensitive recordings. We assume however that principals do not collude, expecting only one malicious party.

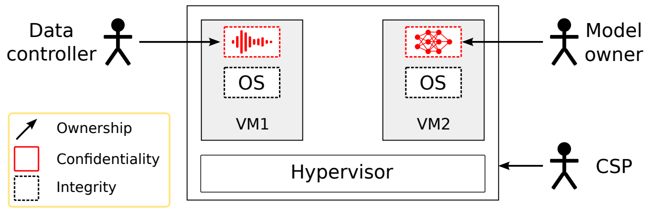


Fig. 6: Application scenario.

A. Analysis

Let us consider how differences between the four solutions will manifest in an actual implementation.

Secret deployment: All of the solutions provide mechanisms for *secret deployment* into freshly created VMs. Moreover, IBM PEF provides a slightly more convenient mechanism allowing to deploy encrypted components (the audio recording and the model) into encrypted VM images prior to attestation. This may be especially convenient for the model owner, which may distribute the encrypted virtual machine containing the trained model that can be deployed later by the data controller. This may free the model owner from the need of a remote attestation infrastructure, which is instead needed if a clean VM must establish a secure channel with the model owner to get the models. This also allows the model VM to be functional in an environment (e.g., for testing) where it has no remote connectivity.

Protection from hypervisor: Assume the data controller uses the transcription to count the occurrences of a secret message. In the case of AMD SEV, the hypervisor can monitor memory changes, allowing it to obtain information about the counter. Intel TDX has the opposite problem where, with low probability (due to MAC checks) and blindly (due to encryption), the hypervisor can modify VM memory. This can for example be used to corrupt the model and affect transcription or to blindly corrupt the transcription counter. In IBM PEF, the page import/export functionality can be used in a replay attack to replace the transcription results with previous values.

Protection from peripherals, offline and physical attacks: In case the adversary obtains physical access to the platforms, several types of attacks may be possible. For IBM PEF, the adversary can read and modify at will the audio recording, model and resulting transcript. Due to memory encryption, in other solutions the attacker can only launch a replay attack, e.g. replace transcription results with previous values. Moreover, in AMD SEV and ARM CCA the attacker could blindly modify VM memory, for example corrupting the model and thus affecting transcription accuracy. This is also possible with Intel TDX but with a low probability of success due to the 28-bit MAC.

ARM CCA is the only solution that can support secure peripherals, such as computation accelerators for improving audio transcription efficiency.

Communication: Our application has one inter-VM channel, which is used to transfer audio and transcripts. This channel may use shared memory and should be protected from hypervisor attacks. ARM CCA supports the channel by placing both VMs inside a Realm and prevents hypervisor attacks via memory protection. AMD SEV can achieve a similar effect by using the same memory encryption key (VEK) for both VMs, but this must be specified at the time of creation. Intel TDX and IBM PEF lack this kind of protection; the VMs must explicitly establish a session key, encrypt and authenticate their communications.

To prevent the model owner VM from leaking the audio recordings, as soon as the VM receives the sensitive data, all its channels (with the exception of the channel with the data controller) should be disabled. No solution can prevent a VM from communicating with other components. Therefore the data controller must trust the hypervisor to disable the model owner channels and send the audio recording only after the hypervisor has disabled them. This conforms with our attacker model, which assumes that CSP and customers do not collude.

Software support: Enabling support for confidential computing requires various actions from the CSP, which includes both changes to the hypervisor and updates to the VM image build pipeline, the extent of which can vary significantly depending on the solution. When it comes to customers using these systems, the transition to confidential computing mainly requires additional steps to check the validity of the deployment through remote attestation.

Migration: In case the applications need to be moved to more powerful machines or relocated for other reasons, live migration can be used to achieve this. This is supported by AMD SEV and Intel TDX, but not IBM PEF and is currently undefined for ARM CCA.

Attestation: As supported by all four solutions, customers may use attestation to verify VM integrity before deployment, or to ensure that the produced transcript has not tampered with. Attestation can also be used to support key agreement; however this is not possible with IBM PEF due to lack of dynamic attestation support. On the other hand, IBM PEF supports a local (on-platform) attestation model, thus offering an alternative to network-based remote attestation.

VI. Related work

In this work we exclusively consider virtualization-based confidential computing technologies for enterprise server platforms. These technologies build on earlier or on-going approaches to implementing trusted execution environments that have been extensively studied and compared.

A. Related technologies

Intel SGX provides *enclaves* for process-based confidential computing [35]. SGX applications are normally split into a minimal trusted component executing inside an enclave and the rest outside. Communication between the two halves can be slow and affect performance, although some mitigations have been proposed to address this [47]. Library OSs simplify porting of legacy applications to SGX by providing a rich execution environment [7], [48], [46].

ARM TrustZone partitions the processor into *normal* and *secure* worlds, where the former is prohibited from accessing resources in the latter [40]. A *monitor* executes in secure world with additional privileges and mediates switching between the two worlds. Trusted application can be deployed into the secure world with some constraints.

IBM Secure Execution enables workload protection in VMs and containers on IBM's z15 platforms in cloud settings [10], however without support for remote attestation. We did not include z15 in our comparison because it assumes a different trust model where the CSP is trusted.

Keystone [27] is an open-source framework for building customized TEEs, using hardware abstractions such as memory isolation and a programmable layer underneath untrusted components. Keystone uses a highest privilege security monitor (akin to the IBM PEF ultravisor) for memory isolation between the untrusted OS and the TEEs. Keystone-based TEEs can be built to run on unmodified RISC-V hardware. We did not include Keystone in our comparison because we are not aware of known or announced implementations of Keystone-based enterprise server platforms. For this reason, we focused on the current four leading platform vendors.

NVIDIA announced support for confidential computing in the H100 Tensor Core GPU Architecture [4], which can be used together with a secure VM. This technology allows to create a TEE to protect the confidentiality and integrity of data and code deployed on the GPU, as well as to establish a secure channel between a secure VM and the GPU.

Solutions relying on confidential computing technology are available to enable a layered approach to platform security for various cloud and edge computing use cases [9].

B. Earlier evaluations

Earlier comparisons and evaluations examined various TEE implementation aspects. TEE approaches were individually studied in depth, as is the case of ARM TrustZone [40] and Intel SGX [14]. Intel SGX and AMD SEV have been closely studied and compared, both when it comes to memory protection [36] and in a wider scope including performance and energy trade-offs [18]. Note that Intel SGX and AMD SEV rely on different approaches to TEE implementation (process-based and VM-based respectively), making in-depth comparisons of their security mechanisms challenging. In this work we chose to only compare virtualization-based TEE implementations. Our work complements earlier security analysis of individual technologies [44], [28].

VII. Open research questions

During our analysis we identified several other aspects that are worthwhile further research.

Making use of confidential computing capabilities requires new tooling that is not available today. Current tools for monitoring VMs (for instance to detect execution of covert binaries [32]) may need to be re-thought to work with secure VMs. Correct implementation of these and other related tools is essential to enable trust in that workloads are effectively protected.

The new attacker model, where the cloud provider and hypervisor are not trusted, requires to review the existing software stack used by the VMs, since it could lack sufficient validation of input coming from the hypervisor. For example, the drivers of VirtIO, which use data structures shared between the VM and the hypervisor, must validate that the the buffer descriptors do no point to secure memory.

An important open issue is protection against side channel attacks. While existing defenses can be employed by VMs, the hypervisor is a powerful attacker that is in charge of scheduling the VMs and can therefore perform trace driven attacks with arbitrary sampling intervals. Moreover, all proposed solutions require extending existing memory protection mechanisms. The possibility of bypassing these extensions using transient executions (e.g. Meltdown [31]) should be investigated. Additional side channels are possible when the hypervisor can intercept signaling between the VM and other components (i.e. other VMs and peripherals), or even when the hypervisor can read encrypted memory of the guest VM [30]. To the best of our knowledge, the existing technologies do not implement specific security mechanisms to avoid these channels.

Another observation is that the functionality of confidential computing mechanisms often depends on platform hardware support [20] with significantly slower release cycles compared to the frequency and breadth of newly discovered micro-architectural side-channel attacks [12]. On the other hand, platform attestation enables customers to obtain information about the trusted computing base and help assess the execution platform's trustworthiness. As comprehensive solutions against micro-architectural attacks emerge [50], customers may rely on attestation results to choose platforms that incorporate such solutions. Finally, it should be noted that execution of TCB operations such as key management on application cores and in normal memory may be more susceptible to side-channel attacks compared to solutions that use a dedicated security processor, dedicated secure memory, or both.

The different designs can also result in significant different overheads. For example ARM CCA memory protection requires a further translation stage, which introduces additional page table walks. The static memory configuration of IBM PEF and the ownership bit saved in the ECC memory by Intel TDX avoid this overhead. Performance evaluation of realistic applications will be needed when silicons will be widely available. The adequacy of 28-bit truncated MAC for Intel TDX must be evaluated depending on the application context. A further research direction is a detailed comparison of the guarantees and security models of the different attestation technologies. Finally, the possibility of abusing of the TEE requires further investigation. For instance, a compromised secure VM can be used to deploy malware that is hidden to the hypervisor and that could harm the system, similarly to what has been demonstrated for Intel SGX [45]. Moreover, it is unclear if CSPs can protect themselves from TEE malfunctions.

VIII. Concluding remarks

The four reviewed solutions are designed to achieve similar protection, but our analysis has identified some key distinctions. IBM PEF is the only solution that statically defines secure memory at boot. It does not support transparent memory encryption and therefore provides no protection against physical attacks. It also does not prevent hypervisor replay attacks. Part of ARM CCA is currently undocumented or sometimes left up to silicon providers. It is the only solution that forbids any hypervisor access to secure memory. AMD SEV allows the hypervisor to read the memory ciphertext (but not write to it), which enables memory update side channels at cache line granularity. AMD SEV and ARM CCA are the only two solutions that provide protection against hypervisor attacks to memory that is shared among VMs. Intel TDX is the only solution that uses ECC bits to save metadata of secure memory and MAC to protect against hypervisor memory corruptions.

IX. Acknowledgements

We would like to thank our shepherd and the anonymous reviewers for the helpful comments. This work was supported in part by the VINNOVA grant 2021-01690 for the project CEST (Confidential Evaluation of Software Trustworthiness) and by the Swedish Foundation for Strategic Research, grant RIT17-0035.

References

[1] AMD SEV-SNP: Strengthening VM Isolation with Integrity Protection and More. White paper, Advanced Micro Devices (January 2020)

[2] Intel Trust Domain Extensions. White paper, Intel (August 2021)

[3] Introducing Arm Confidential Compute Architecture. White paper DEN0125 issue 2, Arm Limited (August 2021)

[4] Nvidia H100 tensor core gpu architecture. White paper Version 1.01, NVIDIA corporation (April 2022)

[5] Ahmad, A., Kim, J., Seo, J., Shin, I., Fonseca, P., Lee, B.: CHANCEL: Efficient Multi-client Isolation Under Adversarial Programs. In: 28th Annual Network and Distributed System Security Symposium (NDSS). INTERNET SOC (2021)

[6] Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., Zaharia, M.: A View of Cloud Computing. Commun. ACM **53**(4), 50–58 (apr 2010). https://doi.org/10.1145/1721654.1721672

[7] Arnautov, S., Trach, B., Gregor, F., Knauth, T., Martin, A., Priebe, C., Lind, J., Muthukumaran, D., O'Keeffe, D., Stillwell, M.L., Goltzsche, D., Eyers, D., Kapitza, R., Pietzuch, P., Fetzer, C.: SCONE: Secure Linux Containers with Intel SGX. In: 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16). pp. 689–703. USENIX Association, Savannah, GA (Nov 2016), https://www.usenix.org/conference/osdi16/technical-sessions/presentation/arnautov

[8] Auernhammer, F., Arndt, R.L.: XIVE: External interrupt virtualization for the cloud infrastructure. IBM Journal of Research and Development **62**(4/5), 5:1–5:10 (2018). https://doi.org/10.1147/JRD.2018.2845599

[9] Bartock, M., Souppaya, M., Savino, R., Knoll, T., Shetty, U., Cherfaoui, M., Yeluri, R., Malhotra, A., Banks, D., Jordan, M., Pendarakis, D., Rao, J.R., Romness, P., Scarfone, K.: Hardware-Enabled Security: Enabling a Layered Approach to Platform Security for Cloud and Edge Computing Use Cases. Tech. Report NIST IR 8320, NIST - US National Institute of Standards and Technology (May 2022), https://doi.org/10.6028/NIST.IR.8320

[10] Bornträger, C., Bradbury, J.D., Bundgen, R., Busaba, F., Heller, L.C., Mihajlovski, V.: Secure your cloud workloads with IBM Secure Execution for Linux on IBM z15 and LinuxONE III. IBM Journal of Research and Development **64**(5/6), 2:1–2:11 (2020). https://doi.org/10.1147/JRD.2020.3008109

[11] Buhren, R., Jacob, H.N., Krachenfels, T., Seifert, J.P.: One Glitch to Rule Them All: Fault Injection Attacks Against AMD's Secure Encrypted Virtualization. Proc. of the 2021 ACM SIGSAC Conference on Computer and Communications Security (2021)

[12] Canella, C., Van Bulck, J., Schwarz, M., Lipp, M., Von Berg, B., Ortner, P., Piessens, F., Evtyushkin, D., Gruss, D.: A Systematic Evaluation of Transient Execution Attacks and Defenses. In: Proc. of the 28th USENIX Conference on Security Symposium. p. 249–266. SEC'19, USENIX Association, USA (2019)

[13] Coker, G., Guttman, J., Loscocco, P., Herzog, A., Millen, J., O'Hanlon, B., Ramsdell, J., Segall, A., Sheehy, J., Sniffen, B.: Principles of remote attestation. International Journal of Information Security **10**(2), 63–81 (2011). https://doi.org/10.1007/s10207-011-0124-7

[14] Costan, V., Devadas, S.: Intel SGX explained. Cryptology ePrint Archive (2016)

[15] Douglas, H.: Thin Hypervisor-Based Security Architectures for Embedded Platforms. Master's thesis, The Royal Institute of Technology, Stockholm, Sweden (2010), https://books.google.se/books?id=eJuxDAEACAAJ

[16] European Parliament Committee on Civil Liberties, Justice and Home Affairs: Initial legal assessment of the impact of the U.S. CLOUD Act on the EU legal framework for the protection of personal data and the negotiations of an EU-US agreement on cross-border access to electronic evidence. https://edpb.europa.eu/sites/edpb/files/files/file2/edpb_edps_joint_response_us_cloudact_annex.pdf (July 2019)

[17] Gruss, D., Maurice, C., Wagner, K., Mangard, S.: Flush+flush: A fast and stealthy cache attack. In: Caballero, J., Zurutuza, U., Rodríguez, R.J. (eds.) Detection of Intrusions and Malware, and Vulnerability Assessment. pp. 279–299. Springer International Publishing, Cham (2016)

[18] Göttel, C., Pires, R., Rocha, I., Vaucher, S., Felber, P., Pasin, M., Schiavoni, V.: Security, Performance and Energy Trade-Offs of Hardware-Assisted Memory Protection Mechanisms. In: 2018 IEEE 37th Symposium on Reliable Distributed Systems (SRDS). pp. 133–142 (2018). https://doi.org/10.1109/SRDS.2018.00024

[19] Helble, S.C., Kretz, I.D., Loscocco, P.A., Ramsdell, J.D., Rowe, P.D., Alexander, P.: Flexible mechanisms for remote attestation. ACM Trans. Priv. Secur. **24**(4) (sep 2021). https://doi.org/10.1145/3470535

[20] Hunt, G.D.H., Pai, R., Le, M.V., Jamjoom, H., Bhattiprolu, S., Boivie, R., Dufour, L., Frey, B., Kapur, M., Goldman, K.A., Grimm, R., Janakirman, J., Ludden, J.M., Mackerras, P., May, C., Palmer, E.R., Rao, B.B., Roy, L., Starke, W.A., Stuecheli, J., Valdez, E., Voigt, W.: Confidential Computing for OpenPOWER. p. 294–310. EuroSys '21, Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3447786.3456243

[21] IBM: IBM Power Systems Announces POWER10 Processor (2021), https://www.ibm.com/blogs/systems/ibm-power-systems-announces-power10-processor/

[22] IBM: Power ISA Version 3.1B. Tech. report, IBM (September 2021), https://ibm.ent.box.com/v/powerisapub31b

[23] Kaplan, D.: Protecting VM register state with SEV-ES. White paper, Advanced Micro Devices (February 2017)

[24] Kaplan, D., Powell, J., Woller, T.: AMD memory encryption. White paper, Advanced Micro Devices (April 2016)

[25] Kim, Y., Daly, R., Kim, J., Fallin, C., Lee, J.H., Lee, D., Wilkerson, C., Lai, K., Mutlu, O.: Flipping Bits in Memory without Accessing Them: An Experimental Study of DRAM Disturbance Errors. In: Proc. of the 41st Annual International Symposium on Computer Architecuture. p. 361–372. ISCA '14, IEEE Press (2014)

[26] Lee, D., Jung, D., Fang, I.T., che Tsai, C., Popa, R.A.: An Off-Chip attack on hardware enclaves via the memory bus. In: 29th USENIX Security Symposium (USENIX Security 20). pp. 487–504. USENIX Association (Aug 2020), https://www.usenix.org/conference/usenixsecurity20/presentation/lee-dayeol

[27] Lee, D., Kohlbrenner, D., Shinde, S., Asanović, K., Song, D.: Keystone: An open framework for architecting trusted execution environments. In: Proceedings of the Fifteenth European Conference on Computer Systems. EuroSys '20, Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3342195.3387532, https://doi.org/10.1145/3342195.3387532

[28] Li, M., Wilke, L., Wichelmann, J., Eisenbarth, T., Teodorescu, R., Zhang, Y.: A Systematic Look at Ciphertext Side Channels on AMD SEV-SNP. In: 2022 2022 IEEE Symposium on Security and Privacy (SP) (SP). pp. 1541–1541. IEEE Computer Society, Los Alamitos,

CA, USA (May 2022). https://doi.org/10.1109/SP46214.2022.00112, https://doi.ieeecomputersociety.org/10.1109/SP46214.2022.00112

[29] Li, M., Zhang, Y., Lin, Z., Solihin, Y.: Exploiting unprotected I/O operations in AMD's secure encrypted virtualization. In: 28th USENIX Security Symposium (USENIX Security 19). pp. 1257–1272. USENIX Association, Santa Clara, CA (Aug 2019), https://www.usenix.org/conference/usenixsecurity19/presentation/li-mengyuan

[30] Li, M., Zhang, Y., Wang, H., Li, K., Cheng, Y.: CIPHERLEAKS: Breaking constant-time cryptography on AMD SEV via the ciphertext side channel. In: 30th USENIX Security Symposium (USENIX Security 21). pp. 717–732. USENIX Association (Aug 2021), https://www.usenix.org/conference/usenixsecurity21/presentation/li-mengyuan

[31] Lipp, M., Schwarz, M., Gruss, D., Prescher, T., Haas, W., Fogh, A., Horn, J., Mangard, S., Kocher, P., Genkin, D., Yarom, Y., Hamburg, M.: Meltdown: Reading Kernel Memory from User Space. In: 27th USENIX Security Symposium (USENIX Security 18). pp. 973–990. USENIX Association, Baltimore, MD (Aug 2018), https://www.usenix.org/conference/usenixsecurity18/presentation/lipp

[32] Litty, L., Lagar-Cavilla, H.A., Lie, D.: Hypervisor support for identifying covertly executing binaries. In: USENIX Security Symposium. vol. 22, p. 70 (2008)

[33] Martin, L.: XTS: A mode of AES for encrypting hard disks. IEEE Security & Privacy **8**(3), 68–69 (2010)

[34] McGrew, D., Viega, J.: The Galois/counter mode of operation (GCM). submission to NIST Modes of Operation Process **20**, 0278–0070 (2004)

[35] McKeen, F., Alexandrovich, I., Berenzon, A., Rozas, C.V., Shafi, H., Shanbhogue, V., Savagaonkar, U.R.: Innovative Instructions and Software Model for Isolated Execution. In: Proc. of the 2Nd International Workshop on Hardware and Architectural Support for Security and Privacy. pp. 10:1–10:1. HASP '13, ACM, New York, NY, USA (2013). https://doi.org/10.1145/2487726.2488368

[36] Mofrad, S., Zhang, F., Lu, S., Shi, W.: A comparison study of intel sgx and amd memory encryption technology. In: Proc. of the 7th International Workshop on Hardware and Architectural Support for Security and Privacy. HASP '18, Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3214292.3214301

[37] Morbitzer, M., Huber, M., Horsch, J., Wessel, S.: Severed: Subverting amd's virtual machine encryption. In: Proceedings of the 11th European Workshop on Systems Security. EuroSec'18, Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3193111.3193112

[38] Mulligan, D.P., Petri, G., Spinale, N., Stockwell, G., Vincent, H.J.M.: Confidential Computing—a brave new world. In: 2021 International Symposium on Secure and Private Execution Environment Design (SEED). pp. 132–138 (2021). https://doi.org/10.1109/SEED51797.2021.00025

[39] Pei, M., Tschofenig, H., Wheeler, D., Atyeo, A., Liu, D.: Trusted execution environment provisioning (teep) architecture. Tech. rep., Internet-Draft draft-ietf-teep-architecture-03, Internet Engineering Task Force (2019)

[40] Pinto, S., Santos, N.: Demystifying Arm TrustZone: A Comprehensive Survey. ACM Comput. Surv. **51**(6) (jan 2019). https://doi.org/10.1145/3291047

[41] Ristenpart, T., Tromer, E., Shacham, H., Savage, S.: Hey, you, get off of my cloud: Exploring information leakage in third-party compute clouds. In: Proc. of the 16th ACM Conference on Computer and Communications Security. p. 199–212. CCS '09, Association for Computing Machinery, New York, NY, USA (2009). https://doi.org/10.1145/1653662.1653687

[42] Rogaway, P.: Efficient instantiations of tweakable blockciphers and refinements to modes ocb and pmac. In: Lee, P.J. (ed.) Advances in Cryptology - ASIACRYPT 2004. pp. 16–31. Springer Berlin Heidelberg, Berlin, Heidelberg (2004)

[43] Russell, R.: virtio: towards a de-facto standard for virtual i/o devices. ACM SIGOPS Operating Systems Review **42**(5), 95–103 (2008)

[44] Sahita, R., Caspi, D., Huntley, B., Scarlata, V., Chaikin, B., Chhabra, S., Aharon, A., Ouziel, I.: Security analysis of confidential-compute instruction set architecture for virtualized workloads. In: 2021 International Symposium on Secure and Private Execution Environment Design (SEED). pp. 121–131 (2021). https://doi.org/10.1109/SEED51797.2021.00024

[45] Schwarz, M., Weiser, S., Gruss, D.: Practical enclave malware with Intel SGX. In: International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment. pp. 177–196. Springer (2019)

[46] Shen, Y., Tian, H., Chen, Y., Chen, K., Wang, R., Xu, Y., Xia, Y., Yan, S.: Occlum: Secure and Efficient Multitasking Inside a Single Enclave of Intel SGX, p. 955–970. Association for Computing Machinery, New York, NY, USA (2020), https://doi.org/10.1145/3373376.3378469

[47] Svenningsson, J., Paladi, N., Vahidi, A.: Faster enclave transitions for io-intensive network applications. In: Proc. of the ACM SIGCOMM 2021 Workshop on Secure Programmable Network INfrastructure. p. 1–8. SPIN '21, Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3472873.3472879

[48] che Tsai, C., Porter, D.E., Vij, M.: Graphene-SGX: A practical library OS for unmodified applications on SGX. In: 2017 USENIX Annual Technical Conference (USENIX ATC 17). pp. 645–658. USENIX Association, Santa Clara, CA (Jul 2017), https://www.usenix.org/conference/atc17/technical-sessions/presentation/tsai

[49] Wilke, L., Wichelmann, J., Morbitzer, M., Eisenbarth, T.: Sevurity: No security without integrity : Breaking integrity-free memory encryption with minimal assumptions. pp. 1483–1496. IEEE, San Francisco, CA, USA (2020). https://doi.org/10.1109/SP40000.2020.00080

[50] Yu, J., Yan, M., Khyzha, A., Morrison, A., Torrellas, J., Fletcher, C.W.: Speculative Taint Tracking (STT): A Comprehensive Protection for Speculatively Accessed Data. Commun. ACM **64**(12), 105–112 (nov 2021). https://doi.org/10.1145/3491201