Confidential Computing: A Security Overview and Future Research Directions

Alessandro Bertani[1], Danilo Caraccio[2], Stefano Zanero[1], and Mario Polino[1]

[1] Politecnico di Milano
{alessandro.bertani,stefano.zanero,mario.polino}@polimi.it
[2] Micron Technology
dcaracci@micron.com

Abstract

By performing computations within hardware-based Trusted Execution Environments (TEEs), Confidential Computing protects data in use, which has been a longstanding challenge in data security. This paper provides an overview on Confidential Computing technologies, with a focus on security implications and recent developments. We begin with an introduction to Confidential Computing, its principles, and its relevance to data security. We outline the threat model for Confidential Computing, considering in-scope and out-of-scope attack vectors. We analyze published attacks, their complexities, and mitigation approaches in the context of Confidential Computing. We analyze data security within TEEs, including encryption, access control, and memory protection mechanisms across different technologies (e.g., Intel TDX, AMD SEV, Arm CCA). Finally, we explore future research directions, including the challenges related with the integration of TEEs and emerging technologies like Compute Express Link (CXL) to further enhance data-in-use security and the use of Confidential Computing in Machine Learning applications.

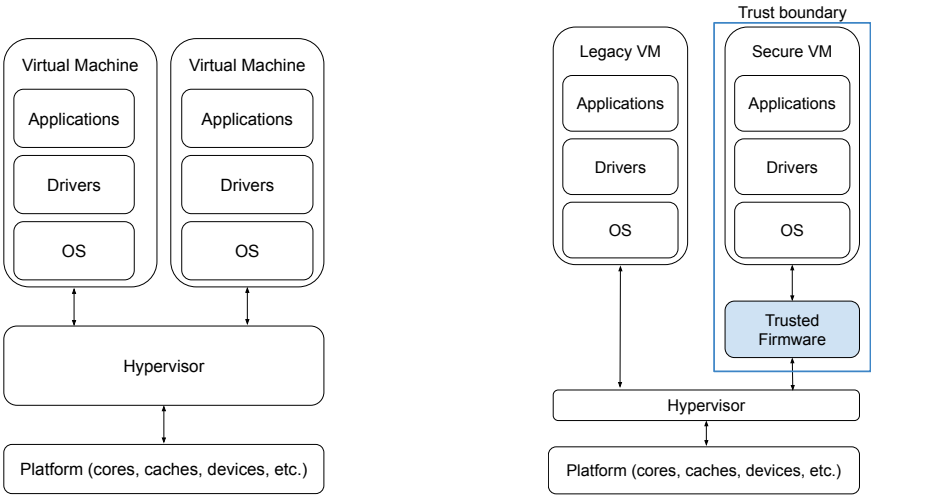
1 Introduction

Data may exist in three different states: data **in transit**, data **at rest**, and data **in use**. Data is in transit when it is traversing the network, it is at rest when it is on a storage or memory device, and it is in use while it is being processed. Protecting sensitive data in all of these states is of critical importance, and while cryptography has been successfully applied for the protection of data in transit and data at rest, the protection of data in use is still an open problem, with a few proposals that aim at solving it.

Confidential Computing [10] is the protection of data in use by performing computation on a hardware-based and attested Trusted Execution Environment (TEE). The definition specifies *hardware-based* for a valid reason: security in any layer of the computing stack could be circumvented by exploiting a vulnerability at a lower level. By providing security at the lowest possible level, it is possible to reduce the required trusted parties.

Confidential Computing is particularly relevant in the context of third-party cloud services. Confidential Computing-enabled hardware allows Cloud Service Providers (CSPs) to give users the possibility to create, deploy, and manage Virtual Machines (VMs), with guarantees on the confidentiality and integrity of the data on which they perform their computations.

In a traditional setting, CSPs give their users the possibility to create VMs that share the same hardware and are managed by a **hypervisor**. The data transferred to the VM by the users is then managed by the CSP, which is trusted with the confidentiality and integrity of this data. This setting requires trust at multiple levels: the user needs to trust both the provider of the hypervisor and the CSP not to share their data or tamper with them. In a Confidential Computing-enabled setting, the only point of trust is the hardware manufacturer, i.e., the one that provides the authenticated firmware which guarantees the confidentiality and integrity of the data in use inside the **secure** VM.



(a) Traditional setting. (b) Confidential Computing-enabled setting.

Figure 1: Differences between the traditional cloud services setting and a confidential computing-enabled setting.

Gradually, all the main players in the processor industry are designing new chips with hardware extensions to support Confidential Computing in their server CPUs. At the moment, there are mainly four technologies that can be used to provide Confidential Computing capabilities to users: Intel Trust Domain Extensions (TDX) [6], AMD Secure Encrypted Virtualization (SEV) [3], IBM Protected Execution Facilities (PEF) [13], and Arm Confidential Compute Architecture (CCA) [8]. These technologies have been gradually introduced and refined in server-side processors (e.g., Intel Xeon or AMD EPYC processors) since 2016 when AMD SEV was first introduced. They enable CSPs to give users the possibility to create and deploy confidential VMs, and have been lately adopted by the main CSPs. At the moment of writing, Microsoft Azure allows deploying confidential VMs with TDX-enabled fourth-generation Intel Xeon processors [5], while Google Cloud and Amazon Web Services use the last iteration of AMD SEV, named Secure Encrypted Virtualization with Secure Nested Paging (SEV-SNP) [1, 9]. A detailed description of the technologies mentioned above is available in Appendix A.

In this paper, we analyze the current status of existing Confidential Computing technologies and provide insights about the future research directions in the field. In particular, the contributions of this work are the following:

- A security-oriented survey on Confidential Computing technologies, which focuses existing attacks and mitigations.
- A comparison of the threat models of commercially available Confidential Computing solutions.
- Future research directions on Confidential Computing, the integration of TEEs with the CXL protocol and Machine Learning applications.

2 Threat model

As defined by the Confidential Computing Consortium (CCC), the goal of Confidential Computing is to reduce the ability for the owner, operator, or exploiter of a platform to access the private data and code inside TEEs so that it is not economically or logically viable to attack the platform during execution.

Target/Attack	TDX	SEV	CCA	PEF
Confidentiality				
VM Memory	✓	✓	✓	✓
VM State	✓	✓	✓	✓
DMA Protection	✓	✓	✓	✓
Integrity				
Replay Protection	NS	✓	ID	NS
Online Data Corruption	✓	✓	ID	✓
Memory Aliasing	✓	✓	ID	NS
Memory Re-Mapping	✓	✓	ID	NS
Availability				
DoS on Hypervisor	✓	✓	ID	✓
DoS on Guest	OoS	OoS	OoS	OoS
Physical Attacks				
Architectural Side Channels	OoS	OoS	OoS	OoS
Cold DRAM Attack	✓	✗	ID	✗

Table 1: Comparison between the threat models of the main commercial Confidential Computing technologies. “ID" means Implementation-Dependent, “NS" stands for Not Specified (in the documentation), and “OoS" stands for Out of Scope.

There are various threat vectors that can be used to exploit the vulnerabilities in a system: not all of them are addressed by Confidential Computing, in fact, some are explicitly considered in scope while others are considered out of scope. In particular, the following threat vectors are considered to be in scope for Confidential Computing: ① software attacks, i.e., attacks on software and firmware installed on the host, including the OS, the hypervisor, BIOS, and so on; ② protocol attacks, i.e., attacks on protocols associated with attestation as well as workload and data transport; ③ cryptographic attacks; ④ basic physical attacks: cold DRAM extraction, bus and cache monitoring, plugging of attack devices into an existing port; ⑤ basic upstream supply-chain attack, i.e., attacks that compromise TEEs such as adding debugging ports.

There is a set of threat vectors for which the mitigations vary significantly based on the silicon implementation, and there are some grey areas (such as integrity, rollback, and replay attacks) that may be considered in scope by some vendors, and out of scope by others. Sophisticated physical attacks are out of scope, as well as availability attacks on the TEEs. A key assumption behind the guarantees provided by Confidential Computing is that there are no exploitable side-channels that the owner (or other entities with access to the system) could use to infer information about the data or execution. Any existing side-channel could allow attackers to infer information about data or operations inside a TEE by exploiting the knowledge of the architecture of the TEE itself. The CCC states that preventing side-channel attacks depends not only on the TEE manufacturers but also on third-party vendors and application developers, thus considering this class of attacks out of scope.

Differences among the threat models of commercial Confidential Computing solutions are summarized in Table 1.

2.1 Data Security and Memory Protection

The main memory, i.e., where data in use resides, is the main asset that Confidential Computing aims to protect. Different Confidential Computing solutions make different assumptions about the threats that the main memory is subject to and this, as explained in the previous section, leads to slightly different threat models between them.

There are three security requirements that all Confidential Computing technologies must meet:

- **Data confidentiality**: unauthorized entities cannot view data while it is in use within the TEE.
- **Data integrity**: unauthorized entities cannot add, remove, or alter data while it is in use within the TEE.
- **Code integrity**: unauthorized entities cannot add, remove, or alter code executing in the TEE.

Among the different technologies, the most used solution to provide confidentiality and integrity to the data residing in the main memory is a combination of encryption and access control. Every technology implements this in a different way, with different results on the guarantees.

3 Existing attacks and mitigation proposals

In the past years, researchers from both industry and academia have found vulnerabilities in Confidential Computing platforms and published attacks that exploit them. Mitigations for the previously mentioned attacks have been proposed in research papers or distributed from the manufacturers in the form of microcode updates. In this section, we describe existing attacks and mitigation proposals for Intel TDX and AMD SEV-SNP. To the best of our knowledge, no attacks have been published regarding Arm CCA.

3.1 Intel TDX

The main vulnerabilities that have been disclosed for Intel TDX come from a report published by Google Project Zero and Google Cloud Security in 2023 [7]. To the best of our knowledge, no academic work has been published regarding attacks on Intel TDX.

Exit Path Interrupt Hijacking. The attack described in this section exploits vulnerabilities in one of the Attested Code Modules (ACMs) provided by Intel, which are the TDX module, the Non-Persistent SEAM Loader (NP-SEAMLDR), and the Persistent SEAM Loader (P-SEAMLDR). These two are code modules whose function is to ultimately load the TDX module in secure memory.

Since the startup BIOS code is outside the Trusted Computing Base (TCB) for Intel TDX, Intel designed the NP-SEAMLDR to dynamically establish a root of trust on which the rest of the TDX infrastructure is loaded. The NP-SEAMLDR performs two main tasks before returning control to the hypervisor: it validates the system configuration and installs the P-SEAMLDR into the `SEAMRR` memory region. The hypervisor can then interact with the trusted P-SEAMLDR to install the signed TDX module into the `SEAMRR` memory region.

All the code outside these ACMs is outside the TCB, and can thus attack NP-SEAMLDR. For this reason, the ACM protects itself from exploitation in different ways. First of all, all external interrupts are masked and hardware breakpoints are disabled. Then, software exceptions are inhibited by setting the Interrupt Descriptor Table Register (IDTR) limit to zero, which leads to any exception causing a triple fault and system shutdown. Finally, the binary is loaded at a known virtual address and no ASLR is applied, unlike P-SEAMLDR and the TDX module.

From an attacker perspective, there are two interesting windows during the execution of the NP-SEAMLDR: shortly after ACM entry the host's Interrupt Descriptor Table (IDT) is still configured before IDTR is set to zero, and shortly before ACM exit the IDT is restored to host's. If an exception can be forced to occur within these windows, the attacker can gain control over the instruction pointer while in privileged mode. Intel fixed this vulnerability in the 1.0 release of TDX by checking that every return address in the exit path is canonical and non-malicious.

ECC Disablement Vulnerability. This vulnerability depends on the ability of the attacker to misconfigure the system. If a privileged attacker can successfully disable Error-Correcting Code (ECC), Rowhammer [14] bit flips could be more likely.

This is not an issue if TDX cryptographic integrity is enabled: the HMAC provides a protection that is similar to ECC, with respect to memory integrity attacks. However, if only TDX *logical* integrity is enabled, there is a single bit per cache line: if the attacker is able to disable ECC, then they would only need to flip a single bit in order to bypass the TDX logical integrity checks. This leads to TDX being vulnerable to Rowhammer-style attacks, where a malicious VMM tries to flip bits in memory owned by the TDX module or by Trust Domains (TDs).

Intel resolved this issue in fourth-generation Intel Xeon Scalable CPUs so that the control registers that contain configuration values for ECC are locked before `MCHECK` runs. Then, `MCHECK` validates that their values are configured properly before enabling TDX.

3.2 AMD SEV, SEV-ES, SEV-SNP

Since the introduction of AMD SEV in 2016, several attacks against this new architectural extension have been published. Some of these attacks have been mitigated with the introduction of Secure Encrypted Virtualization with Encrypted State (SEV-ES) and SEV-SNP, while others may still be applicable under the right conditions. Google Project Zero and Google Cloud Security published a security report on AMD SEV-SNP as well [2].

Crossline. When a secure VM is created, it is assigned an Address Space Identifier (ASID) by the hypervisor, which then notifies the AMD Secure Processor (SP) that a new secure VM has been created. The AMD SP creates the ephemeral encryption key associated with the newly assigned ASID, which is used to look up the key whenever a private memory page belonging to the VM needs to be decrypted. The ASIDs are not authenticated: this is the logic flaw behind Crossline [17], a class of attacks that rely on the ability of a malicious VM to change its ASID into the one of the victim VM, thus being able to decrypt its memory. There are no assumptions of the adversary's knowledge about the contents of the VM: the only assumption is that they control the hypervisor. This assumption is in line with the threat model of SEV.

Two versions of Crossline attacks have been proposed. Crosslinev1 explores the use of nested page table walks to decrypt the victim's memory, while Crosslinev2 is a more powerful variant that allows the attacker VM to execute an instruction inside the encrypted memory of the victim VM. The huge advantage of these attacks is that they're stealthy: they rely on modifying the state of the attacker VM alone, and these changes are not propagated to the victim VM's state, so there is no way for it to notice the ongoing attack. Moreover, it is even possible for the attacker to rewind the state of their VM to eliminate any trace of the attack.

The introduction of SEV-ES, which also encrypts the control structures of the VMs, increased the difficulty of successfully executing these attacks. In fact, while version 1 is still applicable with some further steps, the impossibility of manipulating the values in specific registers makes version 2 unfeasible. With the introduction of SEV-SNP, specifically aimed at preventing attacks against memory integrity, both versions of this attack have been rendered unfeasible due to the new Reverse Map Table (RMP) table walk mechanism.

Ciphertext Side-Channel Attacks. As explained in Appendix A.2, the RMP is used to perform different checks depending on who is requesting access to a memory page. Specifically, if the hypervisor is requesting read access to a memory page, there will be no RMP table walk, even if the requested page belongs to a secure VM.

This behavior opens the door to Cipherleaks [16]: this attack, called "ciphertext side-channel attack", allows the privileged hypervisor to monitor the changes of the ciphertext blocks on the

guest VMs memory pages and exfiltrate secrets from the guest. This is also possible thanks to the mode of operation used by SEV's memory encryption: XOR-Encrypt-XOR (XEX) encrypts each 16-byte memory block independently and preserves the one-to-one mapping between the plaintext and ciphertext pairs for each physical address.

In particular, the CIPHERLEAKS attack monitors the ciphertext of the VM Save Area (VMSA) area during `VMEXIT`s, then by comparing the ciphertext blocks with the ones collected during previous `VMEXIT`s the adversary can learn that the corresponding register values have changed, and infer the execution state of the guest VM. Moreover, by looking up a dictionary of plaintext-ciphertext pairs collected during the VM startup, the adversary is able to recover some selected values of the registers. Due to the severity of this attack, AMD released a microcode patch to mitigate it. This patch enables the third-generation AMD EPYC processors to include a nonce into the encryption of the VMSA area, thus breaking the link between the plaintext and the ciphertext. However, even if this patch is enough to make CIPHERLEAKS unfeasible, it is not enough to prevent other ciphertext side channel attacks that exploit memory leakage from any other memory page than the VMSA. In fact, it was demonstrated that ciphertext side-channel attacks can be successfully performed by targeting any memory region [15].

As a defense against generalized ciphertext side-channel attacks, the CIPHERFIX framework [23] has been recently proposed. This solution is based on binary instrumentation and tracks secret data to identify critical memory accesses, that are then safeguarded by randomizing observable write patterns. This way, the resulting binary does not leak information through the ciphertext side-channel. Being CIPHERFIX a software-based defense mechanism, there is a trade-off to be made in terms of performance: in the worst possible case, the slowdown is up to 40 times the original runtime.

RMP Degradation Attack. An *unchecked write* is defined as a memory write that does not go through the RMP access control. Such an operation can be achieved by exploiting programming errors in the code of the AMD trusted firmware: researchers from Google Project Zero and Google Cloud Security found a bug that allowed writes in a 2 MiB reserved memory area when only 1 MiB of this memory area was actually used by firmware.

The RMP contains self-protecting entries, i.e., entries that contain the address where the RMP resides, marked as belonging to the trusted firmware so that the hypervisor cannot modify them. An unchecked write could be leveraged modify these entries to be marked as belonging to the hypervisor. In this state, a malicious hypervisor can transition any page to any state simply by writing to the RMP: all SEV-SNP security features are lost. This bug has been fixed by modifying the size of the memory region initialized by the firmware from 1 MiB to 2 MiB. However, any possible bug that allows to perform unchecked writes could, under the right conditions, be leveraged to perform an RMP degradation attack.

Microarchitectural Side Channel Attacks. A common technique when attacking TEEs, especially Intel Software Guard Extensions (SGX), is called *single-stepping*. This technique involves using the system's APIC timer to interrupt the enclave after the execution of each instruction, to increase the temporal resolution of microarchitectural attacks. The same technique has been successfully applied to AMD SEV VMs, with SEV-Step [24]. This is a framework that allows to perform single-stepping inside SEV VMs, and gives access to common attack primitives like page fault tracking and cache attacks against SEV. Since side-channel attacks are out of scope, with respect to SEV, so there is no specific countermeasure in place.

4 Future Research Directions

Confidential Computing technologies are rapidly evolving, with revisions and new iterations being developed year after year. The proliferation of Confidential Computing technologies has introduced

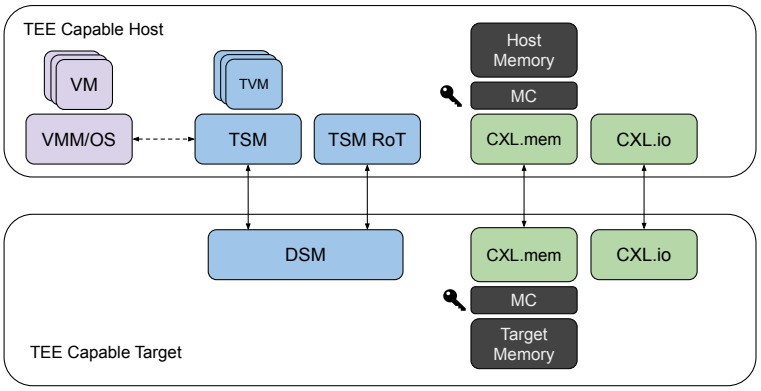


Figure 2: TEE Security Protocol Reference Architecture

a demand for interoperability across a wide variety of devices spanning CPUs, GPUs, accelerators, and memory, which requires the introduction of standards to allow these devices to be compatible with all Confidential Computing solutions. In this section, we propose and analyze two research directions related with Confidential Computing: the integration of TEEs with the Compute eXpress Link standard and the use of Confidential Computing in Machine Learning applications.

4.1 Compute eXpress Link and TEEs

Compute eXpress Link (CXL) is a multi-protocol technology designed to support accelerators and memory devices over the PCIe protocol [11]. The purpose of CXL is sharing computing or memory resources in datacenters. Since its introduction in 2019, several revisions of the standard have been released: the CXL 3.1 specification, released in 2023, introduced the support of Confidential Computing technologies in CXL.

TEE Security Protocol. The TEE Security Protocol defines the architecture to support workload confidentiality in a CXL system. Its scope is limited to directly connected CXL Type 3 (memory expansion device) Single Logical Devices (SLDs) or Multi-Headed SLDs (MH-SLDs) that might support dynamic capacity features for memory pooling architectures.

The diagram in Figure 2 outlines the components of a confidential computing architecture in a CXL system. The TEE Security Manager (TSM) and the TEE Security Manager Root-of-Trust (TSM RoT) are responsible for the authentication and attestation of the device and for the exchange of security protocol transactions in order to discover security properties or to configure and securely lock the device. These operations are performed via the SPDM protocol that guarantees the confidentiality of the messages exchanged. The transactions on the device are processed by the Device Security Manager (DSM). TSM and TSM RoT are the agents that can assign security properties (such as memory encryption and access control) on a VM-basis distinguishing between secure VMs and legacy VMs.

Memory Encryption. The memory encryption feature is introduced to support data-in-use confidentiality. Two modes are defined: ① *initiator-based* encryption, where data are encrypted through the CXL host and exchanged encrypted with the target, and ② *target-based* encryption, where data are exchanged in clear text and encrypted/decrypted by the device. The recommended encryption algorithm is AES-XTS 256 but the standard is open to other types or future vendor-specific solutions. The main requirement of the encryption engine is to minimize the impact on the memory access latency, power, and costs.

TEE Exclusive State tracking and Access Control. To meet integrity protection requirements,

TEE Exclusive State (TE State) is used to indicate whether the content of the memory is for TEE or non-TEE data. Initiators that generate memory accesses shall determine the TEE status of each memory transaction (TE Intent). TEEs are permitted to access both exclusive and non-exclusive memory, while non-TEE entities are permitted to access only non-TEE memory.

Threats to mitigate are related to integrity attacks: changing data of a VM in memory (even with encryption) and replay attacks. Integrity violations impact software running on trusted VMs in an unpredictable way. The basic principle of integrity protection is that if a trusted VM can read a private (encrypted) page of memory, it must always read the last value it wrote.

Access control is the verification of TE Intent against TE State in the memory being accessed and the resulting behavior if the verification fails. Access control can be on read and/or write. The device can advertise the supported access control types. The host can enumerate and enable access control types. TE State can be changed with different methods. One method called Implicit TE State Change, uses memory write operations to change the TE State at the host cache line level (64 bytes). A second method is based on specific commands that can have larger granularities (typically 4 kilobytes). The main architectural challenges on the CXL Type 3 device related to TE State Tracking and Access control are the impact on the memory space to store TE state on a 64 bytes basis and the performance degradation due to the storage of the TE state and to the checks to be performed when the device is accessed.

4.2 Confidential Computing in Machine Learning Applications

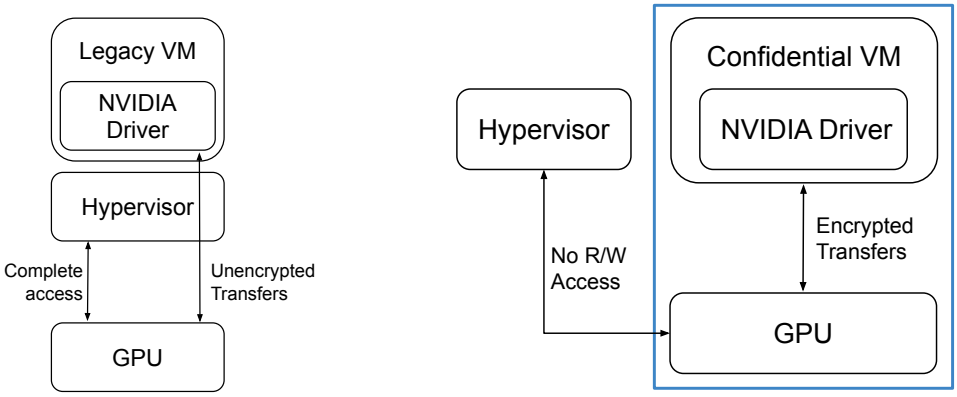
Machine learning and data-driven technologies have been subject to rapid and pervasive development in the last few years. Machine learning models may be trained using sensitive information and, as the use of cloud-based machine learning platforms has increased, robust privacy and security guarantees have become a necessity.

The security of training and inference processes of machine learning models in cloud environments is a relevant topic [18] because there are many different parties involved in the process: the data owners, the model owners, the result receivers, and the host of the ML computation. This implies that there are different entities that need to be trusted not to divulge, tamper with, or steal data, and therefore the attack surface is quite large.

Confidential Computing has emerged as a promising approach to achieve secure and trustworthy machine learning, mainly because of its high performance coming from hardware-based TEEs. Most of the computation in the context of machine learning models' training is performed on external devices, like Graphics Processing Units (GPUs) or other accelerators. For this reason, in the last few years, there have been efforts to provide solutions for the integration of external devices in Confidential Computing-enabled environments both in academia and in industry. Even if the research interest on this topic converges on machine learning applications because of their diffusion, any application that offloads its computation to GPUs or other external devices could also benefit from the application of these solutions.

NVIDIA Confidential Computing. NVIDIA has recently announced its new H100 GPU, based on the Hopper architecture, which is the first commercial GPU solution introducing hardware-based confidential computing capabilities. In July 2023, they released a whitepaper explaining what are the goals of NVIDIA Confidential Computing, which features are enabled on the H100 GPU, and how it integrates into the TEEs created by Confidential Computing-enabled CPUs. NVIDIA specifically requires that the CPU used together with their H100 needs to be from Intel, AMD, or Arm, and must support TDX, SEV-SNP or CCA, respectively. With NVIDIA Confidential Computing enabled, all the data transferred between the secure VM and the GPU will be fully encrypted, with no possibility from the hypervisor to read or write the data. This behavior is represented in Figure 3.

The main goals set by NVIDIA for the H100 GPU are to provide data and code confidentiality,



(a) NVIDIA CC Off. (b) NVIDIA CC On.

Figure 3: Comparison between disabled and enabled NVIDIA Confidential Computing.

data and code integrity, and to provide protection against basic physical attacks so that interposers on buses such as PCIe and DDR memory cannot leak data or code.

NVIDIA outlines three main modes of delivering GPUs to a VM: assigning an entire confidential GPU to a single trusted VM (mainly used for inference, HPC or lightweight training), assigning multiple confidential GPUs to a single trusted VM (with NVLink support and multiple possible topologies, typically used for training) and, finally, assigning each confidential GPU to multiple tenants. The threat model of NVIDIA Confidential Computing is essentially the same as for TDX, CCA and SEV-SNP. Also in this scenario, sophisticated hardware attacks are out of scope, as well as denial of service attacks.

Other Solutions. The NVIDIA H100 GPU is the only available commercial solution that provides Confidential Computing capabilities on GPUs. However, researchers from both academia and industry published several proposals for the extension of confidential computing to GPUs or, more generally, accelerators.

Graviton [22] is a joint work from Microsoft and the University of Lisbon, and is one of the first proposals of a Confidential Computing extension to accelerators. Graviton is an architecture that supports TEEs on GPUs. It enables applications to offload security-sensitive kernels and data to a GPU and execute them in isolation from other code running on the GPU and on the host, including the device driver that communicates with the GPU, the operating system, and the hypervisor. In Graviton, a TEE is a set of GPU resources that are cryptographically bound to a public/private key pair and isolated from untrusted software on the host and all other GPU contexts. Graviton then guarantees that once a secure context has been created, its resources can only be accessed by a user application in possession of the corresponding private key. Graviton works by modifying the interface between the GPU driver and the hardware: the driver can no longer access security-sensitive resources (e.g., page tables, page directories, and memory in general) because Graviton forces all the resource allocation requests coming from the driver to pass through the GPU's command processor. This component tracks ownership of resources and ensures that no resource owned by a secure context can be accessed by other entities. This design has low hardware complexity and low performance overheads, requiring minimum changes for it to be integrated into an existing GPU architecture.

ACAI [20] is a CCA-based solution that allows confidential VMs to use accelerators while relying on hardware-based memory protection to preserve security. There are three modes in which VMs can access accelerators on ACAI. If the SoC has integrated accelerators, then ACAI uses existing CCA primitives to enable VM access. For PCIe devices, ACAI supports encrypted access. This mode creates redundant data copies (at least three). For this reason, the third mode is the *protected* mode, which

reduces the number of copies to one by allowing accelerators to directly access the VM memory.s This requires careful consideration, as in the original CCA specification external accelerators connected over PCIe cannot access realm memory. ACAI addresses this by modifying the granule protection mechanism to disallow any other software from accessing the normal world shared memory: the VM and accelerator can communicate over a shared memory area in normal world memory. ACAI also establishes mutual attestation between accelerators and VMs leveraging existing attestation mechanisms from PCIe5.

IPU Trusted Extensions (ITX) [21] is another proposal made by researchers from several companies (Microsoft, Meta, Graphcore, and others) that presents a set of hardware extensions to enable TEEs in Graphcore's GC200 Intelligence Process Unit (IPU), a state-of-the-art AI accelerator. ITX isolates workloads from untrusted hosts and ensures their data and models remain encrypted at all times (except for when they're inside the accelerator's chip). Trust in ITX is rooted in the *Confidential Compute Unit* (CCU), a new hardware root of trust on the IPU board: the CCU provides each device with a unique identity based on a hardware secret. The new execution mode that ITX introduces, called *trusted* mode, guarantees that all security-sensitive information is isolated from a potentially malicious host. Once the IPU enters this mode, its configuration registers and tile memory can only be accessed by the CCU and the IPU Control Unit (ICU). In the paper, the authors present a specific use case which they call "offline mode": in this mode, ITX requires *no CPU-based TEE*. Suppose that there are multiple parties: the model provider, the data providers, and the untrusted cloud provider. In trusted offline mode, the model provider and the data providers upload the encrypted model and data, verify the attestation report coming from the CCU, and provide their encryption key to the CCU, encrypting them with the CCU's public key. Then, they can be offline while the training of the model goes on, with strong guarantees on the security of their model and data.

All these models have different approaches in including external devices inside the CPU TEE's trust boundary. However, devices implementing these approaches are not commercially available at the time of writing, and, while they are surely promising, it's impossible for us to say whether these approaches are taken into account in the development of the new generation of Confidential Computing-enabled AI accelerators.

5 Conclusions

Confidential Computing technologies are rapidly evolving. New use cases and scenarios, like the integration with the CXL standard and the use of Confidential Computing with Machine Learning applications, need careful threat modeling and security analyses to avoid that the introduction of external devices into the Trust Boundary of Confidential Computing systems cause their security guarantees to be weakened. In this paper we have provided a comparison of the threat models of commercially available Confidential Computing technologies, as well as a detailed analysis of their inner workings. We have provided an overview on the attacks and mitigations for these systems, and an analysis on future research directions in this field, identifying the integration with the CXL standard and Machine Learning applications as the most interesting and promising.

A Confidential Computing Technologies

Even if commercially available Confidential Computing solutions are similar to one another, in this appendix we highlight their differences. We also add details with respect to other works, as well as a description of RISC-V AP-TEE, which was not included in the work of Guanciale et al. [12].

A.1 Intel Trust Domain Extensions

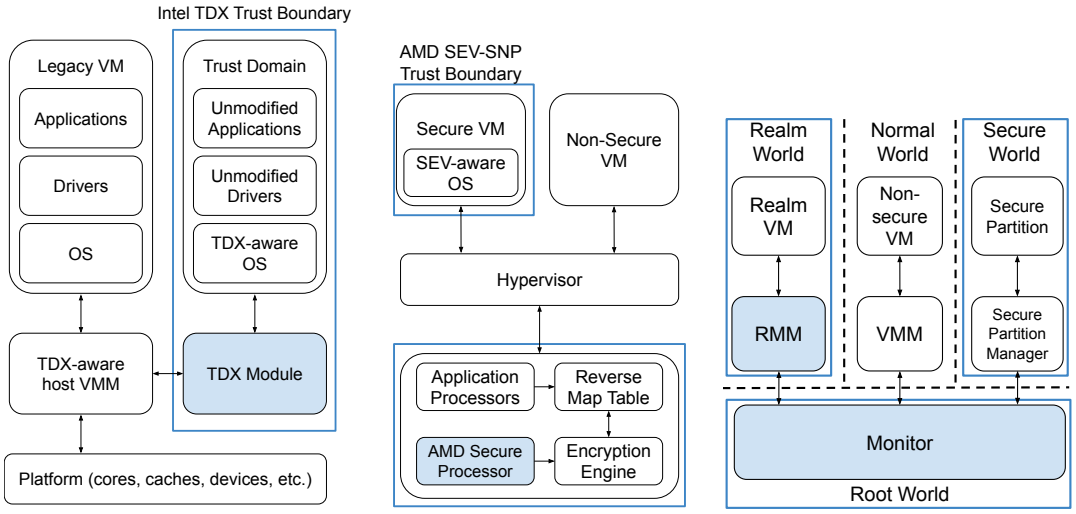
Intel TDX is a set of architectural extensions that enable the creation and management of hardware-isolated, secure VMs called TDs. TDX is designed to isolate VMs from every non-TD component, including the hypervisor.

Intel TDX uses a CPU-attested software module (the TDX module), Intel Virtual Machine Extensions (VMX) and Intel Multi-Key Total Memory Encryption (MK-TME), which is used to create and manage a different private key for each TD, that is then used to encrypt the memory pages of each TD and their control structures.

Intel TDX introduces a new CPU mode that helps enforce the security policies for the TDs, called Secure Arbitration Mode (SEAM), that is the CPU mode in which the privileged and trusted TDX module is executed. Control transfers between the hypervisor and the TDX module happen when a `SEAMCALL` instruction is executed. This instruction causes the CPU mode to switch to SEAM, transferring the control to the TDX module.

To guarantee memory isolation between TDs and from the hypervisor, a Guest Physical Address (GPA) can be *private* or *shared*, depending on the `SHARED` bit of the GPA. The CPU translates shared GPAs using the *shared* Extended Page Table (EPT), which resides unencrypted in host VMM memory and is directly managed by the VMM. The CPU translates private GPAs using the *secure* EPT, which is unique per TD and is encrypted with the private key of its associated TD. The secure EPT is designed not to be directly accessible by any other software than the TDX module, nor by any devices.

During TD launch, its initial contents and configuration are recorded by the TDX module. At runtime, the Intel TDX architecture reuses the Intel SGX attestation infrastructure to support attest-



(a) TDX Architecture. (b) SEV-SNP Architecture. (c) CCA Architecture.

Figure 4: Comparison of the architectures of Intel TDX, AMD SEV-SNP and Arm CCA.

ing to these measurements. Software running inside a TD can request the TDX module to generate an integrity-protected `TDREPORT` structure that includes TD's measurements and an asymmetric key that is used to establish a secure channel with the software running inside the TD. An SGX *Quoting Enclave* can be used to check the integrity of the report produced in the previous step. If integrity is successfully verified, the Quoting Enclave can insert the guest TD's measurements in a quote, which is crucial in establishing trust between the user and the platform.

Memory Confidentiality and Integrity. TDX uses MK-TME to enable cache-line-level memory encryption. The TDX module assigns each TD a unique, private `KeyID` which corresponds to an AES-128 bit cryptographic key managed by the memory controller. The keys saved into the memory controller are not accessible by software or by using external interfaces to an SoC.

TDX also provides two memory integrity modes: *cryptographic* integrity and *logical integrity*. When cryptographic integrity is enabled, each cache line is protected with a 28-bit MAC (obtained by truncating the output of a SHA-3-256-based MAC generation function), in addition to AES-XTS-128 encryption. Moreover, a 1-bit TD ownership tag is maintained with each cache line to identify if the line is associated with a memory page assigned to a TD. When cryptographic integrity is enabled, the ownership tag is included in the computation of the MAC. When only logical integrity is enabled, the TD ownership tag is maintained but there is no MAC computation.

A.2 AMD Secure Encrypted Virtualization

AMD SEV aims at isolating VMs and the hypervisor from one another. It uses one key per virtual machine to isolate guests and the hypervisor from one another. The keys are managed by the AMD Secure Processor (SP) and are used to encrypt the memory pages owned by each guest. SEV-ES is an improvement over SEV: it encrypts all CPU register contents when a VM stops running, thus preventing the leakage of information in CPU registers to untrusted components, and can detect malicious modifications to a CPU register state. SEV-SNP is the last iteration of this technology and the focus of this section. It adds strong memory integrity protection to help prevent malicious hypervisor-based attacks like data replay, memory re-mapping, and more. The architecture of SEV-SNP is depicted in Figure 4b. The basic principle of SEV-SNP integrity is that if a VM is able to read a private (i.e., encrypted) page of memory, it must always read the last value it wrote: if this is not possible, it should get an exception indicating that the value could not be read. This principle is enforced by a component called RMP.

As happens for Intel TDX, the guarantees of memory confidentiality and integrity are enforced by a trusted firmware component, provided by AMD, that runs on the AMD SP. The AMD SP, the SoC hardware, and the secure VM itself are the only trusted components in this technology's threat model.

While SEV and SEV-ES only supported attestation during the launch of a guest VM, SEV-SNP is more flexible: a guest VM can request an attestation report from the AMD AP at any time. Attestation reports contain system information and a block of arbitrary data supplied by the guest VM as part of the request and are signed by the AMD SP. Attestation reports enable third parties, e.g., the guest owner, to validate that specific data came from a specific VM

Memory Confidentiality and Integrity. SEV-SNP uses Multi Key Secure Memory Encryption to provide memory confidentiality to secure VMs. At boot, the keys for VMs are randomly generated and stored in the AMD SP. SEV-SNP also uses AES-128 with XEX as mode of operation.

Many of the integrity guarantees of SEV-SNP are enforced through the RMP, a single data structure, shared across the system, which contains one entry for every page of DRAM that may be used by VMs. The purpose of the RMP is to track the owner of each page of memory: the hypervisor, a specific VM, or the AMD SP. Memory accesses are controlled in a way that only the owner of the page can write it. The RMP is only checked when the hypervisor is performing write accesses to memory pages: since SEV-SNP encrypts all the memory pages belonging to secure VMs, the hypervisor being

(a) Native RMP table-walk.

(b) Nested RMP table-walk.

Figure 5: Differences in RMP table-walk when the system is executing in non-VM mode and when it is running an SEV-SNP VM.

able to read the encrypted content of a memory page is not considered as a threat. Both read and write accesses *inside* an SEV-SNP VM require RMP checks. Figure 5 shows the differences in RMP table-walks when the hypervisor and an SEV-SNP VM request write access to a memory page.

A.3 Arm Confidential Compute Architecture

Arm CCA allows to deploy VMs while preventing access by other software components, like the hypervisor. CCA allows the hypervisor to control the VM but removes any right to access code, register states, or data used by the VM. This separation is enabled by protected VM execution spaces called *Realms*. A Realm is completely isolated from a "normal" execution environment in terms of code execution and data access. The separation is achieved through a combination of hardware extensions and trusted firmware. The architecture of Arm CCA is depicted in Figure 4c.

Armv8-A already introduced the concept of *world*, i.e., a combination of a security state of a processing element and physical address space. The security state a processing element is executing in determines which physical address it can access. The Arm CCA introduces the *Realm Management Extension* (RME) [4], which adds two new worlds: ① the *Root* world is the world with the highest privilege (the Monitor runs in Root world), ② the *Realm* world is composed of the Realm security state and the Realm Physical Address range.

The RME is composed of a set of hardware extensions that are required by the architecture to allow isolated Realm VM execution, while the software component that is used to manage the Realm VMs is called *Realm Management Monitor* (RMM), which is part of the TCB of Arm CCA, is the trusted component that executes in the Root world and is in charge of ensuring the isolation among Realm, Normal, and Secure worlds.

CCA remote attestation allows the user of a service provided by a Realm to determine the trustworthiness of the Realm and of the implementation of the CCA platform. The protocols that should be used for attestation are implementation-specific and are not discussed in the guidelines provided by Arm. However, the desired outcome of successful attestation is a secure point-to-point connection between an attested endpoint in the Realm, and the reliant party (the user).

Memory Confidentiality and Integrity. In the case of Arm CCA, the guarantees on the security of the data is implementation-dependent. Arm provides some rules that must be strictly followed, and

some suggestions. Arm only suggests using encryption algorithms with *address tweaking*, leaving the choice of the algorithm and mode of operation to the hardware manufacturer. This allows the selection of an algorithm depending on the specific power and area requirements. Memory integrity should also be provided as a guarantee to the Realm owner, but the details are implementation-dependent.

A.4 Other Architectures

There are other Confidential Computing architectures that have been proposed, both in academia and in industry. IBM PEF and RISC-V Application Platform - TEE (AP-TEE) are two examples of these proposals. The former doesn't have publicly available information, apart from the paper published by IBM in which they describe the architectural extension at a high level. The latter is the result of the effort of several contributors, whose aim is to provide Confidential Computing capabilities to the RISC-V open-source platform.

IBM Protected Execution Facility. In 2021, IBM published the description of its own implementation of a Confidential Computing extension to the OpenPOWER architecture [13] and, to the best of our knowledge, this paper is the only documentation available on this TEE.

The goal of PEF is to enable users to create and manage secure VMs, guaranteeing the confidentiality and integrity of their memory. To do so, PEF utilizes a Trusted Platform Module (TPM), and a new trusted firmware called the **Protected Execution Ultravisor** (or just **ultravisor**).

PEF achieves isolation between secure VMs and the outside through hardware-enforced access control policies, and memory confidentiality and integrity with the use of cryptography. It also introduces a new CPU state, called the *secure state*, which is managed by the Ultravisor: this firmware component manages all security-related hardware features in the processor, and is the only component that can do so.

The access control mechanism in PEF is based on the assignment of VMs to security domains: each secure VM has its own security domain in secure memory, while the hypervisor is in another security domain in normal memory. This approach ensures that the secure VMs are protected from the hypervisor and one another and that the hypervisor's security domain is protected from all the secure VMs.

The Ultravisor protects the confidentiality of the secure VM when the hypervisor is paging or dumping it. When data from secure memory are made available to software that is not running in secure memory, the Ultravisor performs encryption with integrity using Galois Counter Mode as the mode of operation, prior to allowing a page to be moved to normal memory. When a page is accessible to the hypervisor, it is not accessible to the secure VM: when the latter wants to access the page, the Ultravisor performs an integrity check and, if it is successful, decrypts the page and allows access from the secure VM.

RISC-V Application Platform - TEE. This section describes the first proposal of a Confidential Computing extension to the RISC-V architecture, called **Confidential Virtual Machine Extension** or **CoVE** [19].

As for previous architectures, CoVE relies on the presence of a trusted software module called the TEE Security Manager, or TSM, which manages security properties for workload assets to protect against access from the OS or the hypervisor. The isolation of the TSM from the host is supported by Instruction Set Architecture (ISA) extensions.

The CoVE TCB consists of the TSM that acts as the TCB intermediary between TEE and non-TEE components, and of hardware elements that enforce confidentiality and integrity properties for workload data-in-use. As for all other technologies, the hypervisor is untrusted and manages the resources for all workloads, both confidential and non-confidential.

References

[1] Amazon ec2 now supports amd sev-snp. https://aws.amazon.com/about-aws/whats-new/2023/04/amazon-ec2-amd-sev-snp/ [Accessed: 29/02/2024].

[2] Amd secure processor for confidential computing - security review. https://storage.googleapis.com/gweb-uniblog-publish-prod/documents/AMD_GPZ-Technical_Report_FINAL_05_2022.pdf [Accessed: 29/02/2024].

[3] Amd sev-snp. https://www.amd.com/system/files/TechDocs/SEV-SNP-strengthening-vm-isolation-with-integrity-protection-and-more.pdf [Accessed: 29/02/2024].

[4] Arm realm management extensions system architecture. https://developer.arm.com/documentation/den0129/latest/ [Accessed: 29/02/2024].

[5] Azure confidential computing on 4th gen intel xeon scalable processors with intel tdx. https://azure.microsoft.com/en-us/blog/azure-confidential-computing-on-4th-gen-intel-xeon-\scalable-processors-with-intel-tdx/ [Accessed: 29/02/2024].

[6] Intel trust domain extensions (TDX). https://cdrdv2.intel.com/v1/dl/getContent/690419 [Accessed: 29/02/2024].

[7] Intel trust domain extensions (tdx) security review. https://services.google.com/fh/files/misc/intel_tdx_-_full_report_041423.pdf [Accessed: 29/02/2024].

[8] Introducing arm confidential compute architecture. https://developer.arm.com/documentation/den0125/latest [Accessed: 29/02/2024].

[9] Oh snp! vms get even more confidential. https://cloud.google.com/blog/products/identity-security/rsa-snp-vm-more-confidential [Accessed: 29/02/2024].

[10] A technical analysis of confidential computing. https://confidentialcomputing.io/wp-content/uploads/sites/10/2023/03/CCC-A-Technical-Analysis-of-Confidential-Computing-v1.3_unlocked.pdf [Accessed: 29/02/2024].

[11] Compute eXpress Link Consortium. Compute eXpress Link consortium. https://computeexpresslink.org/ [Accessed: 29/02/2024].

[12] Roberto Guanciale, Nicolae Paladi, and Arash Vahidi. SoK: Confidential Quartet - Comparison of Platforms for Virtualization-Based Confidential Computing. In *2022 IEEE International Symposium on Secure and Private Execution Environment Design (SEED)*, pages 109–120, September 2022.

[13] Guerney D. H. Hunt, Ramachandra Pai, Michael V. Le, Hani Jamjoom, Sukadev Bhattiprolu, Rick Boivie, Laurent Dufour, Brad Frey, Mohit Kapur, Kenneth A. Goldman, Ryan Grimm, Janani Janakirman, John M. Ludden, Paul Mackerras, Cathy May, Elaine R. Palmer, Bharata Bhasker Rao, Lawrence Roy, William A. Starke, Jeff Stuecheli, Enriquillo Valdez, and Wendel Voigt. Confidential computing for OpenPOWER. In *Proceedings of the Sixteenth European Conference on Computer Systems*, EuroSys '21, pages 294–310, New York, NY, USA, April 2021. Association for Computing Machinery.

[14] Yoongu Kim, Ross Daly, Jeremie Kim, Chris Fallin, Ji Hye Lee, Donghyuk Lee, Chris Wilkerson, Konrad Lai, and Onur Mutlu. Flipping bits in memory without accessing them: an experimental study of DRAM disturbance errors. *ACM SIGARCH Computer Architecture News*, 42(3):361–372, October 2014.

[15] Mengyuan Li, Luca Wilke, Jan Wichelmann, Thomas Eisenbarth, Radu Teodorescu, and Yinqian Zhang. A Systematic Look at Ciphertext Side Channels on AMD SEV-SNP. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 337–351, May 2022. ISSN: 2375-1207.

[16] Mengyuan Li, Yinqian Zhang, Kang Li, Yueqiang Cheng, and Huibo Wang. CIPHERLEAKS: Breaking Constant-time Cryptography on AMD SEV via the Ciphertext Side Channel.

[17] Mengyuan Li, Yinqian Zhang, and Zhiqiang Lin. CrossLine: Breaking "Security-by-Crash" based Memory Isolation in AMD SEV. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 2937–2950, Virtual Event Republic of Korea, November 2021. ACM.

[18] Fan Mo, Zahra Tarkhani, and Hamed Haddadi. Machine Learning with Confidential Computing: A Systematization of Knowledge, April 2023. arXiv:2208.10134 [cs].

[19] Ravi Sahita, Atish Patra, Vedvyas Shanbhogue, Samuel Ortiz, Andrew Bresticker, Dylan Reid, Atul Khare, and Rajnesh Kanwal. CoVE: Towards Confidential Computing on RISC-V Platforms, April 2023. arXiv:2304.06167 [cs].

[20] Supraja Sridhara, Andrin Bertschi, Benedict Schlüter, Mark Kuhne, Fabio Aliberti, and Shweta Shinde. ACAI: Extending Arm Confidential Computing Architecture Protection from CPUs to Accelerators, May 2023. arXiv:2305.15986 [cs].

[21] Kapil Vaswani, Stavros Volos, Cédric Fournet, Antonio Nino Diaz, Ken Gordon, Balaji Vembu, Sam Webster, David Chisnall, Saurabh Kulkarni, Graham Cunningham, Richard Osborne, and Daniel Wilkinson. Confidential Computing within an AI Accelerator.

[22] Stavros Volos, Kapil Vaswani, and Rodrigo Bruno. Graviton: Trusted Execution Environments on GPUs.

[23] Jan Wichelmann, Anna Pätschke, Luca Wilke, and Thomas Eisenbarth. Cipherfix: Mitigating Ciphertext Side-Channel Attacks in Software.

[24] Luca Wilke, Jan Wichelmann, Anja Rabich, and Thomas Eisenbarth. SEV-Step: A Single-Stepping Framework for AMD-SEV, July 2023. arXiv:2307.14757 [cs].