Narrowing the Gap between TEEs Threat Model and Deployment Strategies

Filip Rezabek[1,2], Jonathan Passerat-Palmbach[1,3], Moe Mahhouk[1], Frieder Erdmann[1], and Andrew Miller[1]

[1]Flashbots
[2]Department of Informatics, Technical University of Munich, Germany
[3]Imperial College London

***Abstract*—Confidential Virtual Machines (CVMs) provide isolation guarantees for data in use, but their threat model does not include physical level protection and side-channel attacks. Therefore, current deployments rely on trusted cloud providers to host the CVMs' underlying infrastructure. However, the TEE attestation does not provide information about the operator hosting a CVM. Without knowing whether a Trusted Execution Environment (TEE) runs within a provider's infrastructure, a user cannot accurately assess the risks of physical attacks. We observe a misalignment in the threat model, where the workloads are protected against other tenants but do not offer end-to-end security assurances to external users without reliance on cloud providers. The attestation should be extended to bind CVM with the provider. A possible solution can rely on the Protected Platform Identifier (PPID), a unique CPU identifier. However, the implementation details of various TEE manufacturers, attestation flows, and providers vary. This makes verification of attestations, migration easiness, and building applications without relying on a trusted party challenging, highlighting a key limitation that must be addressed for the adoption of CVMs. We discuss two points focusing on hardening and extensions of TEEs' attestation.**

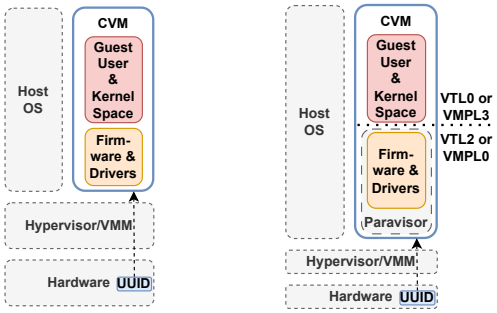
***Index Terms*—Cloud, TEE, Attestations, Threats**

1. Introduction

Many applications require safeguarding sensitive data during processing. While traditional security measures protect data at rest and in transit, data in use remains vulnerable to threats like unauthorized access and malicious insiders [1], [2]. This risk is especially concerning in cloud environments, where multiple tenants share physical hardware on (ideally) untrusted third-party infrastructure, heightening data breach chances. Confidential Computing addresses these challenges by protecting data in use through Trusted Execution Environments (TEEs). TEEs provide an isolated environment where sensitive computations can be executed without interference, even from higher-privileged software like operating systems or hypervisors. Recently, technologies such as Intel Trust Domain Extensions (TDX) or AMD Secure Encrypted Virtualization (SEV)-Secure Nested Paging (SNP) run the whole Virtual Machine (VM) in such an isolated environment. As part of their Trusted Computing Base (TCB), they include the guest OS and guest VM's privileged users, but can protect against malicious host Operating System (OS) or hypervisor [1]. External users can request a hardware-signed attestation report to verify that key components remain untampered, including code and data inside the TEE. This combination of isolation and attestation enables secure execution of sensitive workloads, even in untrusted environments. However, both AMD [3] and Intel [1] exclude memory integrity [4], side channels and sophisticated physical attacks e.g., microscope probing or fault injection, from their threat models. While most side channels can be mitigated at the application level, physical attack vectors require users to trust the physical host location of the Confidential VMs (CVMs). This is especially needed for use cases in the Web3 space, where TEEs could protect millions of dollars in value [5], [6]. The robustness of CVMs depends on ensuring they operate in a trusted environment, relying on operators who do not tamper with nodes and enforce strong access policies. Remote attestation verifies that communication terminates in a TEE on an authenticated platform but does not provide details about the operational environment.

Therefore, the attestation flow should be extended with additional information, assuring that the environment where the TEE platform runs is a trusted cloud data center, thus strengthening the relation to the provider. This closes the gap between the threat model of current TEEs and the trust in the infrastructure owner. A recently proposed solution called LooseSEAL relies on the Protected Platform Identifier (PPID) to derive keys originating from CVMs on the same machine [7]. The PPID is generated based on the Universally Unique Identifier (UUID) [8] of Intel or the CPU_ID [9] of AMD CPUs that run inside the cloud's infrastructure. As this feature is currently not implemented, the provider must enhance the attestation capabilities. Besides, each provider and ideally TEE manufacturers should implement the same flow for ease of migration, requiring industry standardization. Another option is to rely on a certification party that certifies the physical location to ensure the usage of untampered hardware and additional intrusion detection to detect physical access to the devices, as is the case of Apple [10]. Another, even more demanding, approach is to extend the threat model to include physical attacks and tampering with the chip manufacturer's supply chain. However, this is less likely as it requires new TEE designs [11].

Our work aims to bring discussion points (DPs) about:
DP1 Unification/standardization of PPID & deployments.
DP2 Threat model extension by physical access.



(a) Bare/CVM Flow (b) Paravirtualization Flow

Figure 1: Simplified TEE Attestation Flow for various Deployments. The provider controls grey dotted boxes.

2. TEEs and Attestation Flows

We introduce relevant background information supporting the **DPs**. VM-based TEEs, such as Intel TDX [12], [13] or AMD SEV-SNP [2], [14], [15], enhance VM security through encrypting and isolating guest VMs from the hypervisor and supporting nested virtualization. Users gain confidence in a given TEEs enclave via the request of a remote attestation. For CVMs, Figure 1 presents two attestation flows varying between bare metal/native virtualization (1a) and with an additional paravirtualized layer (1b) and how is UUID available to CVM. The paravisor allows for live migration of the CVM and provides an additional layer of virtual drivers between the guest OS and underlying VM Manager (VMM). In the bare metal setup (Figure 1a), the CVM runs directly on the hypervisor, e.g., QEMU. Attestation in this scenario involves verifying the firmware, operating system, and TEE itself. On the other hand, in the paravirtualized environment (Figure 1b), the CVM additionally relies on a paravisor, e.g., OpenHCL [16] or COCONUT [17]. The paravisor implements an access mode present as a Virtual Trust Level (VTL) for Intel TDX and Virtual Machine Protection Level (VMPL) for AMD SEV-SNP. Of note, VMPL0 is the highest privilege level, and VTL0 is the lowest, hinting at other implementation approaches. The attestation report should include verification of the same components as regular deployment and the paravirtualization stack. This requires the paravisor's components to be open-source to enable reproducible builds and thus obtain the checksum to compare with the value in the attestation's fields. This is, however, not always the case, as was the case of Microsoft Azure's paravisor before OpenHCL [18]. Even when using the paravisor approach, the quote contains the PPID constructed during Intel's initial platform verification. For the case of live migration, the verifier must regularly be made aware of migration or request attestation, as the PPID is hardware-dependent,

3. TEE Attestation Extensions & Beyond

Building on top of the background information, we address the **DPs** introduced in Section 1. The **DP1** focuses on the solution's reliance on PPID and **DP2** motivates physical attacks integration to the TEE threat model. One way cloud providers can offer assurance that a given TEE is indeed cloud-based could be to leverage PPID. The PPID is a unique identifier derived uniquely for each Intel CPUs, allowing attestation reports to be linked to a known and verifiable machine or infrastructure for Intel TDX and SGX. In the case of Data Center Attestation Primitives (DCAP) attestation quotes, the Platform ID (first 16 B of user data) is either the encrypted PPID or is derived from it. The PPID is encrypted using Intel's public key of the Intel Provisioning Certification Service (PCS), with the private key being only owned by Intel. Therefore, to modify the PPID, the CPU manufacturer would have to be involved, serving as a separation of interests. Overall, the PPID is a consistent identifier that links attestation quotes to specific physical CPUs. For the solution to work, the cloud provider must keep a list of its publicly available hardware identifiers so the users can then validate the provided PPID. Such a mechanism would ensure that workloads are executed on certified hardware and within a secure infrastructure. We rely on cloud providers for physical protection, so enabling the PPID does not increase the attack surface, and as an operator of CVM, we can verify the information is correct. Of note, different CVMs on the same hardware share the same PPID. To ensure privacy, we can use a Zero-knowledge (ZK) proof of the attestation and the PPID details proving that our node runs in a particular cloud, without disclosing which. There are already ZK instantiations for DCAP [19] which can be extended for this setting. The effectiveness of this solution depends on how cloud providers implement and disclose such identifiers. Implementing PPIDs should also be unified to allow easy migration across providers to mitigate possible friction. PPID is a robust solution considering the current setting and not too demanding from the cloud provider's perspective. Nevertheless, the trust in the cloud provider does not increase, as we rely on the provider for physical attack protection.

However, several challenges must be addressed in designing and deploying such a solution. One key difficulty is accounting for hardware diversity, as different processor architectures (e.g., Intel TDX, AMD SEV, ARM Confidential Compute Architecture (CCA)) implement TEEs with varying security models and attestation mechanisms. A standardized solution must accommodate these differences while maintaining security guarantees. The key issue of physical and side-channel attacks persists. Therefore, extending the threat model to include more physical, supply-chain, and side-channels attacks will improve the potential of TEE as a technology.

4. Next Steps for TEEs

We highlight the need to extend TEEs' threat model to include physical access attacks. Current VM-based TEEs implicitly trust the cloud provider, which is misaligned with attestation flows that do not bind the provider to the report. Available solutions such as PPID can improve TEEs adoption. Strengthening the threat model and closing the attestation gap requires collaboration among manufacturers, service providers, and researchers.

Acknowledgments

We thank the anonymous reviewers for their valuable feedback and our shepherd for his valuable guidance.

References

[1] D. Kuvaiskii *et al.*, "Gramine-tdx: A lightweight os kernel for confidential vms," in *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '24, Salt Lake City, UT, USA: Association for Computing Machinery, 2024, pp. 4598–4612. [Online]. Available: https://doi.org/10.1145/3658644.3690323.

[2] J. Ménétrey *et al.*, *An exploratory study of attestation mechanisms for trusted execution environments*, 2022. arXiv: 2204.06790 [cs.CR].

[3] M. Li *et al.*, "A systematic look at ciphertext side channels on amd sev-snp," in *2022 IEEE Symposium on Security and Privacy (SP)*, 2022, pp. 337–351.

[4] D. Lee *et al.*, "An Off-Chip attack on hardware enclaves via the memory bus," in *29th USENIX Security Symposium (USENIX Security 20)*, USENIX Association, Aug. 2020, pp. 487–504. [Online]. Available: https://www.usenix.org/conference/usenixsecurity20/presentation/lee-dayeol.

[5] R. Cheng *et al.*, "Ekiden: A platform for confidentiality-preserving, trustworthy, and performant smart contracts," in *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*, 2019, pp. 185–200.

[6] K. Rabimba *et al.*, "Lessons learned from blockchain applications of trusted execution environments and implications for future research," in *Workshop on Hardware and Architectural Support for Security and Privacy*, ACM, Oct. 2021, pp. 1–8. [Online]. Available: http://dx.doi.org/10.1145/3505253.3505259.

[7] M. Mahhouk, *Loose SEAL: Enabling Crash-Tolerant TDX Applications by Utilizing SGX Sealing Provider Sidecar - TEE - Trusted Execution Environment - The Flashbots Collective*, [Online; accessed 14. Feb. 2025], Dec. 2024. [Online]. Available: https://collective.flashbots.net/t/loose-seal-enabling-crash-tolerant-tdx-applications-by-utilizing-sgx-sealing-provider-sidecar/4243/1.

[8] Intel, *Intel tdx dcap: Quote generation library and quote verification library*, [Online; accessed 14. Feb. 2025], Sep. 2024. [Online]. Available: https://download.01.org/intel-sgx/latest/dcap-latest/linux/docs/Intel_TDX_DCAP_Quoting_Library_API.pdf.

[9] AMD, *Sev secure nested paging firmware abi specification*, [Online; accessed 14. Feb. 2025], Jan. 2025. [Online]. Available: https://www.amd.com/content/dam/amd/en/documents/epyc-technical-docs/specifications/56860.pdf.

[10] Apple Inc., *Hardware integrity in private cloud compute*, https://security.apple.com/documentation/private-cloud-compute/hardwareintegrity, Accessed: 2025-04-01.

[11] Q. Kilbourn, *Zero Trust Execution Environments - TEE - Trusted Execution Environment / Trustless TEEs - The Flashbots Collective*, [Online; accessed 15. Feb. 2025], Oct. 2024. [Online]. Available: https://collective.flashbots.net/t/zero-trust-execution-environments/3966.

[12] Intel. "Intel/tdx-module." (Accessed on 05/10/2024). (2024).

[13] M. U. Sardar, S. Musaev, and C. Fetzer, "Demystifying attestation in intel trust domain extensions via formal verification," *IEEE Access*, vol. 9, pp. 83 067–83 079, 2021.

[14] AMD. "Github - amdese/amdsev: Amd secure encrypted virtualization." (Accessed on 10/15/2023). (2022).

[15] R. Li *et al.*, *Sok: Tee-assisted confidential smart contract*, 2022. arXiv: 2203.08548 [cs.CR]. [Online]. Available: https://arxiv.org/abs/2203.08548.

[16] Microsoft, *openvmm*, [Online; accessed 14. Feb. 2025], Feb. 2025. [Online]. Available: https://github.com/microsoft/openvmm.

[17] SUSE, *svsm*, [Online; accessed 14. Feb. 2025], Feb. 2025. [Online]. Available: https://github.com/coconut-svsm/svsm.

[18] C. Perezvargas, *Confidential vms on azure*, 2023. [Online]. Available: %5Curl%7Bhttps://techcommunity.microsoft.com/blog/windowsosplatform/confidential-vms-on-azure/3836282%7D.

[19] A. Network, *Automata dcap attestation*, Accessed: 2025-04-03, 2025. [Online]. Available: https://github.com/automata-network/automata-dcap-attestation.