Trusted Execution Environments: A Comparative Analysis of SGX and SEV

Omar Bin Kasim Bhuian
Computer Science Department,
Huaiyin Institute of Technology,
Huai'an, Jiangsu, China
omarbinkasimsefat@gmail.com

Ki-Woong Park*
Department of Information Security and Convergence
Engineering for Intelligent Drone,
Sejong University,
Seoul 05006, South Korea
woongbak@sejong.ac.kr

***Abstract*—Trusted Execution Environments (TEEs) are the most vital security features of contemporary computing, especially in a virtualized environment. Two popular hardware-based TEEs include Intel Software Guard Extensions (SGX) and AMD Secure Encrypted Virtualization (SEV), which respectively help protect sensitive computation from several forms of attacks. This paper investigates SGX and SEV very deeply, including their architecture, memory encryption mechanisms, and the security vulnerabilities they encounter. SGX adopts an enclave-based approach to application-level isolation, whereas SEV affords VM system-wide memory encryption. We discuss the implications of such designs in cloud computing environments and proffer recommendations that will help secure attacks emanating from side-channel and rollback.**

Keywords: trusted execution environments, Intel SGX, AMD SEV, secure program execution, memory encryption.

I. Introduction

The advent of cloud computing and virtualized systems has turned the world of computing upside down by allowing resources to be shared amongst multiple tenants (e.g. users) while scaling with balance. But they also represent a security risk because attackers can exploit system software weaknesses in OS and hypervisors. To mitigate the same, Trusted Execution Environments (TEEs) provide a secure execution environment to run trusted code within an untrusted system. Trusted Execution Environments (TEEs), including Intel Software Guard Extensions (SGX) and AMD Secure Encrypted Virtualization, have been around for a while and are commonly used to protect sensitive computations if the receiving OS or hypervisor is not fully trusted [1].

With the growing need for secure code execution in cloud environments, confidential computing solutions have emerged, such as SGX or SEV. In this way, SGX protects enclaves with an application-level isolation and SEV is implemented for the encryption of the memory space of virtual machines (VMs) at the system-wide level. The paper provides a broad analysis of cryptosystem implementation techniques, isolation techniques, and various system vulnerabilities in the domain of confidential computing. We analyze them to differentiate where they are suitable for the varieties of cloud-based workloads and strive to identify potential directions on secret computing.

II. Architectural Comparison

A. Intel Software Guard Extensions (SGX)

Intel SGX also creates secure enclaves (protected memory regions) within the application address space, insulated from both privileged and unprivileged system software. While data inside enclaves resides in the application address space, the Memory Encryption Engine (MEE) encrypts this data on path outside the CPU , when pages are evicted to system DRAM [2]. Active enclave data lives in the Enclave Page Cache (EPC), which is a quantity of protected areas in the CPU itself. The most substantial value of SGX would be the highly specific isolation small sections of an application could run inside enclaves. However, this level of precision results in significant expansion of the Trusted Computing Base (TCB), since enclaved applications require many trusted components like the SGX Platform Software (PSW) and driver stacks. This dependence increases the attack surface [3].

B. AMD Secure Encrypted Virtualization (SEV)

AMD SEV encrypts the complete memory space of virtual machines via Secure Memory Encryption (SME), guaranteeing that data remains encrypted while residing in DRAM. The Platform Security Processor (PSP) oversees encryption keys, segregating virtual machines (VMs) from one another and from the hypervisor. SGX offers fine-grained memory encryption, which is good for protecting fully virtualized environments. SEV, on the other hand, doesn't have the application-level isolation feature that SGX does [4].

SEV-SNP (Secure Nested Paging) enhances SEV by incorporating memory integrity verifications, protecting against rollback attacks and hypervisor manipulation. SEV-SNP checks the integrity of virtual machine memory, stopping malicious hypervisors from changing or rolling back memory, which would be a security breach [4].

TABLE 1 COMPARISON OF SGX AND SEV FEATURES

Features	**Intel SGX**	**AMD SEV**
Memory Encryption	Enclave-specific via MEE	Entire VM memory via SME
Key Management	Managed by CPU	Managed by PSP
Attack Surface	Vulnerable to side-channel and speculative attacks.	Vulnerable to rollback and cold-boot attacks
Use Cases	Application-level isolation	Full VM protection
Granularity	Fine-grained, application-specific	Coarse-grained, VM-wide

III. Memory Encryption and Isolation

A. SGX Memory Encryption and Isolation

SGX employs enclave-specific encryption using the Memory Encryption Engine (MEE) to protect enclave data throughout memory paging processes. The CPU regulates memory access rights and retains enclave data in the Enclave Page Cache (EPC). Nonetheless, SGX's security possesses certain limits. While it encrypts data when transferred to untrusted memory, it is susceptible to specific side-channel attacks during active enclave activities [1]. The Foreshadow attack, which exploits speculative execution to extract enclave data from the processor's speculative cache, is a notable example. Furthermore, side-channel attacks like page-fault and cache timing attacks might undermine SGX's protections by scrutinizing memory access patterns [6].

B. SEV Memory Encryption and Isolation

SEV's Secure Memory Encryption (SME) scheme offers comprehensive protection for VM memory by encrypting it entirely. This protects against threats emanating from the hypervisor. The Platform Security CPU (PSP), a specialized CPU, oversees encryption keys for each virtual machine (VM). SEV's coarse-grained encryption safeguards the operating system and user data within the VM from unauthorized access [7].

SEV-SNP mitigates SEV's susceptibility to rollback attacks, wherein a malevolent hypervisor may return the virtual machine to a prior state to exploit inadequate encryption. SEV-SNP offers strong defenses against such attacks by incorporating integrity protection into VM memory and verifying the encryption state through attestation [8].

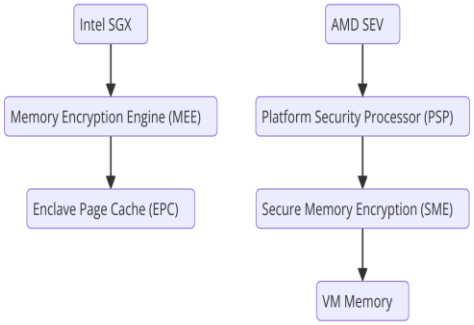

Fig 1. Diagram illustrating the memory encryption process in SGX vs SEV

IV. Threat Models and Security Vulnerabilities

A. Intel SGX Threat Model

Since the operating system and hypervisor could be hacked, SGX operates under the premise that enclaves would still be shielded from such privileged attacks. SGX is susceptible to microarchitectural side-channel attacks, such as Foreshadow and Meltdown, which exploit speculative execution to leak sensitive enclave information [5]. Foreshadow: Attackers exploit speculative execution vulnerabilities to leak data stored in the EPC despite using SGX's memory protection capabilities.

Foreshadow breaks the confidentiality of the EPC and thus compromises SGX's memory protection on speculative execution, as attackers can leak any data that is loaded into memory protected by a page encryption key maintained in the EPC. SGX design is vulnerable to side-channel attacks, such as cache timing, page-fault side channels. On the other hand, cache timing attacks exploit slight variations in timing between different access(es) to a cached resource to leak selectively sensitive data and page-fault attacks monitor page accesses to deduce behaviors of an enclave [6] . These assaults draw attention to the inherent hazards in depending just on hardware isolation, including later SGX versions' mitigating measures.

B. AMD SEV Threat Model

SEV presumes the hypervisor could be malevolent; hence, it encrypts VM memory to safeguard against unwanted access. SEV is susceptible to rollback attacks when a compromised hypervisor restores the VM to a prior encrypted state, enabling the attacker to change the VM without decrypting it [8].

SEV's dependence on the hypervisor for the management of encryption keys presents supplementary dangers. An assailant with access to the hypervisor may seize control of

the PSP and alter encryption keys, circumventing SEV's safeguards. Cold-boot attacks represent a considerable risk, as an assailant with physical access to a powered-off system can get encryption keys from memory [9].

TABLE 2: SUMMARY OF VULNERABILITIES IN SGX AND SEV

Vulnerability Type	**Intel SGX**	**AMD SEV**
Side-Channel Attacks	Cache timing, page fault, speculative execution	Limited to hypervisor or cold-boot attacks
Rollback Attacks	Not applicable	Hypervisor-based rollback is a concern
Key Management Attacks	Secure CPU-based key management	Hypervisor and PSP-managed, susceptible to compromise

V. Performance Considerations

A. Intel SGX Performance

The performance overhead of SGX mostly comes from the costs of creating an enclave, context switching between enclaves during enclaving & mode transitions and enclave paging. Applications that are in regular communication with enclaves, or applications managing lots of data suffer from a lot of performance degradation. Memory-intensive applications can experience up to a 3x performance slowdown under SGX vs. regular (unprotected) environments [1], [2].

Also, the performance of using SGX is inherently difficult because enclave transitions need to save and restore the enclaves state, which means every time there is some overhead added. Literature has suggested a number of optimizations to address this challenge, including decreasing the frequency at which these transitions are invoked and using hardware-rooted approaches tailored for security-sensitive operations, but enclosure management remains complex and continues to be a major scalability bottleneck with respect to SGX deployment.

B. AMD SEV Performance

SEV typically has a lower performance overhead than SGX since it encrypts the entire VM and so does not require switching back and forth between trusted and untrusted execution contexts. However, memory encryption and decryption mechanisms result in latency during context switching; the impact is especially problematic in multi-tenant cloud systems (where virtual machines frequently access shared resources) [9].

Memory integrity systems that perform memory health checks by validating encrypted memory are another source of overhead added via SEV-SNP. Incorporate these verifications with high-availability cloud setups or complicated work, and the hold-up might aggravate. The security/effectiveness trade-off is generally considered OK for SEV-SNP and its stronger protections.

Fig 2. Performance Overhead: SGX vs SEV Across Workloads

VI. Future Directions in TEE Development

Intel SGX and AMD SEV both have introducing limitations that must be addressed subsequent versions. SGX is facing the challenge of mitigating side-channel attacks with increased performance scalability. However, although initiatives like SGXv2 improve enclave sizes and leverage dynamic memory management as a means to mitigate certain performance constraints, despite the existence of mitigations such as Specter and Meltdown, SGX remains insecure against speculative execution attacks.

For the future, we would like to see improvements that will reduce dependency on hypervisors and grow key management protocols throughout SEV. You also could limit the dangers from hypervisor hacking with multi-key encryption as well as hardware-based key attestation. It will also be more relevant in the cloud if SEV supports secret computing frameworks (such as multi-party computation or federated learning).

VII. Conclusion

This work reveals that Intel SGX and AMD SEV are similar in terms of their ability to protect computations in untrusted environments. SGX has strong application-level isolation through enclaves, where though SGX does better against privilege attacks but exposes secure enclaves to side-channels and eventually will be significantly slowed down. By extension, in our evaluation, SEV has broad memory

encryption for VMs, and the security scheme of this measure relies on HPA's integrity, while it can be improved by tackling the weakness of rollback attacks.

SGX, the poster child of enclave-based security solutions plagued by performance issues, is a common choice when you need strong, application-level security. On the other hand, SEV offers benefits in virtualized environments since it virtually eliminates any setup requirements and degrades performance much less. While using SGX or SEV to create strong confinement is crucial, this decision should be made on a workload basis, and threat models should not place fundamental reliance on them to maintain good security in any case of workloads that requires the instantiation.

ACKNOWLEDGEMENT

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Ministry of Science and ICT (Project No. RS-2022-00165794, 50%; RS-2024-00438551, 30%), and the Ministry of Science and ICT grant through the Information Technology Research Center (ITRC) Program (Project No. RS-2023-00228996, 20%)

REFERENCES

[1] "Costan, V., Devadas, S.: Intel SGX Explained. Cryptology ePrint Archive, 2016." Accessed: Sep. 14, 2024. [Online]. Available: https://eprint.iacr.org/2016/086.pdf

[2] I. A. I. Frank McKeen, "Innovative instructions and software model for isolated execution |." Accessed: Sep. 14, 2024. [Online]. Available: https://www.researchgate.net/publication/266654240_Innovative_instructions_and_software_model_for_isolated_execution

[3] J. Götzfried, M. Eckert, S. Schinzel, and T. Müller, "Cache Attacks on Intel SGX," Apr. 2017, pp. 1–6. doi: 10.1145/3065913.3065915.

[4] P. Paradžik, A. Derek, and M. Horvat, "Formal Security Analysis of the AMD SEV-SNP Software Interface," Jul. 23, 2024, arXiv: arXiv:2403.10296. doi: 10.48550/arXiv.2403.10296.

[5] J. Van Bulck et al., "Foreshadow: extracting the keys to the intel SGX kingdom with transient out-of-order execution," in Proceedings of the 27th USENIX Conference on Security Symposium, in SEC'18. USA: USENIX Association, Aug. 2018, pp. 991–1008.

[6] Y. Xu, W. Cui, and M. Peinado, "Controlled-Channel Attacks: Deterministic Side Channels for Untrusted Operating Systems," in 2015 IEEE Symposium on Security and Privacy, San Jose, CA: IEEE, May 2015, pp. 640–656. doi: 10.1109/SP.2015.45.

[7] "AMD Secure Encrypted Virtualization (SEV)," AMD. Accessed: Sep. 16, 2024. [Online]. Available: https://www.amd.com/en/developer/sev.html

[8] L. Wilke, J. Wichelmann, M. Morbitzer, and T. Eisenbarth, "SEVurity: No Security Without Integrity -- Breaking Integrity-Free Memory Encryption with Minimal Assumptions," in 2020 IEEE Symposium on Security and Privacy (SP), May 2020, pp. 1483–1496. doi: 10.1109/SP40000.2020.00080.

[9] M. Morbitzer, M. Huber, J. Horsch, and S. Wessel, "SEVered: Subverting AMD's Virtual Machine Encryption," in Proceedings of the 11th European Workshop on Systems Security, Apr. 2018, pp. 1–6. doi: 10.1145/3193111.3193112.