Attestation Capabilities of Trusted Execution Environments in the Wild

Eber Christer, Filip Rezabek*
*Chair of Network Architectures and Services
School of Computation, Information and Technology, Technical University of Munich, Germany
Email: eber.christer@tum.de, frezabek@net.in.tum.de

***Abstract*—Attestation is one of the most crucial mechanisms of confidential computing, allowing trust to be established between software systems. While the underlying concept of software attestation remains consistent in all Trusted Execution Environments (TEEs), each silicon manufacturer has their own unique approach to this process – different architectures, isolation guarantees, measurements, and attestation flows. This paper aims to abstract the intricacies behind TEE technologies by providing a general overview of the underlying concepts behind TEEs, as well as present the attestation capabilities of industry-recognized TEE solutions, such as Intel SGX, Intel TDX, and AMD SEV-SNP.**

***Index Terms*—confidential computing, trusted execution environments, attestation**

1. Introduction

In an era of digital transformation, the notions of data integrity and confidentiality are becoming increasingly prevalent. While a traditional computing system is able to protect data at rest and in transit, data in use remains vulnerable [1]. Confidential computing is a technique that aims to mitigate this problem by ensuring that sensitive computations are performed inside a Trusted Execution Environment (TEE). At its core, TEE is a hardware-based mechanism that isolates software execution in a secure region within the memory and CPU, preventing unauthorized access and tampering [2], [3]. Regardless of the TEE implementations, these capabilities of ensuring data integrity and confidentiality are particularly useful when dealing with public cloud platforms, where the underlying system is considered untrusted, opaque, and uncontrollable [4].

Before sharing confidential information and executing any processes inside a TEE, the trustworthiness of said execution environment must be proven in a process called attestation. The inherent mechanisms and guarantees that can be attested by each TEE differ from one vendor to another. Therefore, this paper explores the attestation capabilities of different state-of-the-art TEEs. In Section 2, the theoretical background of TEEs and attestation methods are given, followed by the architecture and attestation methods of different state-of-the-art TEE technologies in Section 3. Finally, Section 4 concludes the paper by highlighting the key findings and outlining directions for future work and improvements.

2. Background

This section presents relevant background information on the concepts relevant to this paper, namely the different TEE models and the general attestation flows.

2.1. TEE Models

Process-Based Model. Process-based TEEs, such as Intel Secure Guard Extensions (SGX) and ARM TrustZone, provision encrypted memory areas called secured enclaves where sensitive workloads can run in isolation [5]. This process involves separating an application into two components: trusted and untrusted. When a workload is to be executed in isolation, an enclave with the necessary resources (mainly memory) is created, where the computations will take place [4]. The untrusted component serves as an interface that communicates with the OS and bridges the secure enclaves with the rest of the system through dedicated channels [6].

VM-Based Model. On the other hand, VM-based TEEs involve dynamically encrypting the memory of a confidential VM (CVM) [5]. This process isolates the whole application running inside the VM from the hypervisor itself. On top of data confidentiality, state-of-the-art VM-based TEEs such as Intel Trusted Domain Extensions (TDX) and AMD Secure Encrypted Virtualization-Secure Nested Paging (SEV-SNP), have additionally introduced integrity-preserving features [2], [3].

2.2. Attestation Types

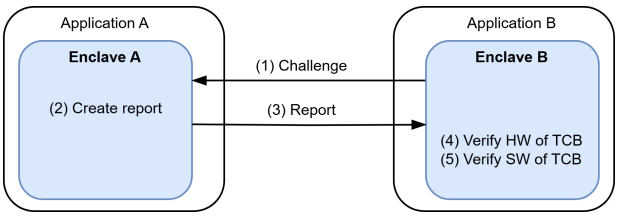

Figure 1: Generic local attestation protocol [6]

Local Attestation. As the name suggests, local attestation refers to the process in which one enclave verifies the identity and integrity of another on the same TEE-enabled CPU via a challenge-response protocol [6]. In general, since the enclaves reside in the same hardware,

 doi: 10.2313/NET-2024-09-1_10

the authenticity of the local attestation request can be verified using a Message Authentication Code (MAC)–based symmetric-key scheme [4].

Figure 1 visualizes a generic flow for a local attestation. In this case, Enclave B wants to verify that Enclave A is running on genuine TEE-enabled hardware. The process is as follows [4], [6]:

1) Enclave B sends Enclave A its enclave-unique information.
2) Enclave A generates an attestation report containing a cryptographic hash of its initial state, Enclave B's identity, and a symmetric key (i.e., Diffie-Hellman Key).
3) Enclave A sends Enclave B the attestation report encrypted with a platform-specific key
4) Enclave B retrieves the platform-specific key. If the attestation report can be verified by Enclave B using the platform-specific key, then both enclaves are on the same TEE platform.
5) Enclave B verifies the report content to authenticate the software component of the TCB.

Assuming both parties have verified their security measures are as expected, a secure channel can be created using the symmetric key propagated from the attestation reports.

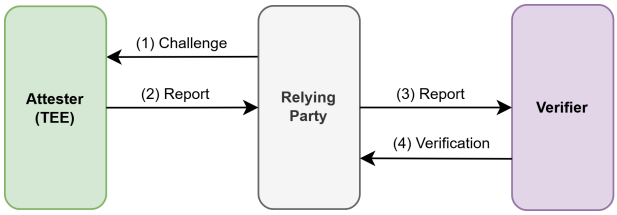

Figure 2: Generic remote attestation protocol [7]

Remote Attestation. On the contrary, remote attestation establishes trust between environments running on different hardware or platforms [4]. Generally, this process involves the following components [7]:

- **Verifier**: A tool used to authenticate the evidence provided by the TEE based on configured policy, ensuring that the certificate is genuine and that the issuer is trusted.
- **Attester**: A TEE looking for its evidence or measurements to be appraised by the verifier, such that it can be trusted to execute workloads.
- **Relying Party**: An entity that uses information about an attester to determine its trustworthiness.

Before workloads can be executed inside a TEE, trust between the *relying party* and the TEE must be established through the *verifier*. The *relying party* first sends a challenge to the *Attester* (TEE), requesting an attestation (1). The TEE submits a set of claims to prove its trustworthiness to the *relying party* through a signed attestation report. Depending on the requirements enforced by the attestation protocol, additional claims – such as roots of trust, trusted computing base (TCB), and metadata – may be requested (2). The *relying party* relays this report to the *verifier*, which will appraise the evidence by applying constraints and enforcing policies. Upon successful verification, an asymmetric key is issued to the *attester* through an attestation report, which it can use to build a safe communication channel between the *relying party* and the TEE (4). Figure 2 shows a generic remote attestation flow, summarizing the overall process [7].

2.3. Challenges and Limitations

While TEEs offer an additional layer of security to applications, a barrier impeding the wide adoption of this technology includes their proprietary nature and lack of standardization. In fact, a majority of successful attacks stemmed from specific TEE design flaws [8]. For example, early iterations of AMD-based TEEs were susceptible to unencrypted register attacks. This reliance on a vendor for updates, security patches, and support can create a single point of failure. Moreover, since there is limited transparency about the internal workings of TEEs, it becomes difficult to assess their security or trustworthiness independently.

This paper aims to identify common design patterns across different state-of-the-art TEEs. For the following analysis, we additionally assume that the silicon manufacturer can be fully trusted to provide secure and reliable TEE solutions.

3. Attestation with State-of-the-Art TEEs

This section explores state-of-the-art TEEs widely adopted by various cloud service providers (CSPs), namely Intel SGX, Intel TDX, and AMD SEV-SNP. While the inherent mechanism of TEEs remains consistent across different solutions, the enabling architecture differs. For each TEEs, we aim to study their security guarantees and attestation capabilities by answering the following questions:

- How are data **confidentiality and integrity** achieved?
- What **measurements** are collected?
- How is the **attestation** conducted?

3.1. Intel SGX

Intel SGX extends the instruction set architecture, allowing it to generate and manage process-based TEEs called enclaves.

Confidentiality and Integrity. Enclave data and code are stored in an encrypted memory region within the DRAM called the Enclave Page Cache (EPC) [6]. The system software (i.e., OS kernel or hypervisor) is responsible for allocating and freeing pages in the EPC for the enclaves, which are created through the application software [9]. The EPC is designed so that an enclave's artifacts are inaccessible to non-enclave software, including the application that created it. This restriction serves as the basis for SGX's confidentiality guarantees. In addition, SGX offers integrity guarantees of its enclaves through its local and remote attestation procedures [4].

Measurements. An enclave is initialized with an SGX Enclave Control Structure (SECS) within a dedicated EPC page to store its identity [9]. SECS comprises two measurement registers called `MRENCLAVE` and `MRSIGNER`

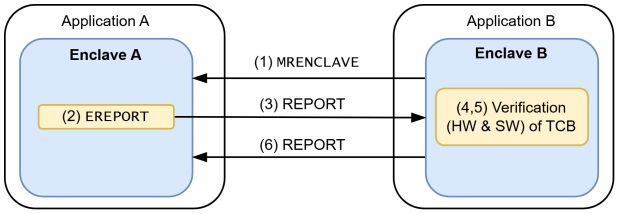

Figure 3: Intel SGX local attestation flow [6]

[10]. The former represents the enclave measurement and contains logs pertaining to the enclave's memory – this includes the contents, positions, and security measurements of the pages used by the enclave. While the latter represents the sealing identity and contains a hash of the enclave author's public key. SECS also includes other fields, such as the attributes of the enclave, product ID, and Security Version Number (SVN) of the modules [9].

Attestation. Local attestation establishes trust between enclaves residing in the same SGX platform. We describe this process as shown in Figure 3 by adapting the generic naming conventions and local attestation process used in Figure 2. The following local attestation flow is adapted from [9], [10], and the SGX developer guide [6]. After establishing a communication channel between Enclave A and B, Enclave B retrieves its `MRENCLAVE` and sends it over to Enclave A (1). Using Enclave B's `MRENCLAVE` as input, the CPU instruction `EREPORT` generates a signed attestation report (REPORT), which binds Enclave A's identity information from its SECS using a MAC tag. This MAC tag is computed using a symmetric key shared exclusively between the target enclave and the same SGX SVN (2). Enclave A sends this signed attestation report over to Enclave B (3), where the report's authenticity can be verified using the MAC tag (4). By invoking the `EGETKEY` command, Enclave B is capable of retrieving the Report Key (symmetric key) needed to recompute the MAC of the REPORT. If the MAC generated by Enclave B matches the MAC of the attestation report, then that would mean that both enclaves reside within the same platform, and the hardware component of the TCB can be verified. Enclave B proceeds to examine the content of REPORT to authenticate the software component of the TCB (5). After all components are verified, Enclave B generates a new REPORT using Enclave A's `MRENCLAVE` and sends it over to Enclave A (6). Enclave A can use this attestation report to verify that Enclave B resides on the same platform as it does.

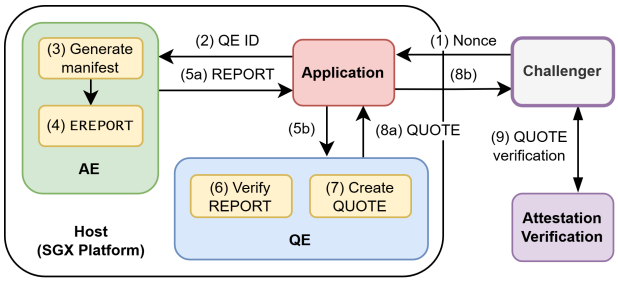

Figure 4: Intel SGX remote attestation flow [10]

Remote attestation, on the other hand, is enabled by a special enclave called the *Quoting Enclave* (QE) which replaces the MAC-bound REPORT (locally verifiable) with a signature generated by a device-specific asymmetric key, creating what is known as a QUOTE (remotely verifiable) [11]. This key is provisioned by the Intel Enhanced Privacy ID (EPID), which protects the identity of the signer [10].

Figure 4 details the process of attestation by an *Application Enclave (AE)* to a remote *Challenger*, adapted from [10] and the SGX developer guide [6]. In this case, the *Challenger* first sends a challenge (nonce) to the *Application* (1). The *Application* relays this challenge along with the *QE* 's identity to the *AE* (2). The *AE* generates a manifest containing the challenge answer and an ephemeral public key. This key is used to facilitate communication between the *Challenger* and the *AE* (3). The *AE* then invokes `EREPORT` to generate a REPORT, containing the hash of the manifest (4) and sends it to the *QE* for signing (5). The *QE* calls `EGETKEY` to obtain the Report Key used to verify the correctness of the REPORT (MAC tag recalculation process as in local attestation) [11] (6). Upon successful verification of the report, the *QE* creates a signed QUOTE from the REPORT using its EPID key (7). The *QE* then sends the QUOTE and the associated manifest to the *Challenger* for verification (8). The *Challenger* may validate the QUOTE 's signature using the EPID public key certificate or an independent attestation verification service. The *Challenger* verifies the manifest by checking its response with the initial challenge (9).

3.2. Intel TDX

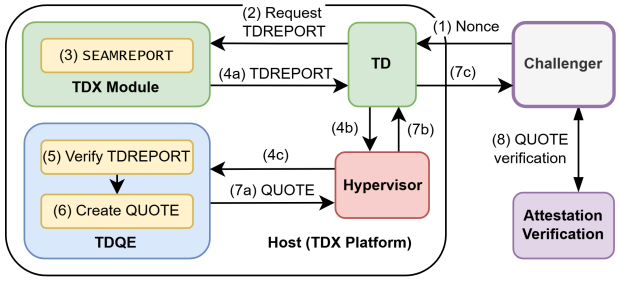

Figure 5: Intel TDX attestation flow [2], [11]

Intel TDX is a VM-based TEE solution responsible for creating and managing confidential VMs called Trusted Domains (TDs). This multi-component system introduces a new CPU mode that extends the functionality of VMX, called Secure-Arbitration Mode (SEAM) [2]. Intel TDX also uses special Intel SGX enclaves called the TD-Quoting Enclave (TDQE) to generate remote attestations for TDs. Behind all this mechanism is the new Intel TDX Module, an Intel-signed software module that interfaces the hypervisor and the TDs [11].

Confidentiality and Integrity. Memory confidentiality is primarily achieved through Intel's Multi-Key Total Memory Encryption (MKTME) technology, which offers encryption at cache line granularity using TD-specific keys [2]. Additionally, in the case of unauthorized attempts to access the cache lines, a fixed bit pattern will be returned to prevent cyphertext analysis. TDX also offers an

option for cryptographic integrity protection, which will prematurely terminate a TD in case of any write attempts on secure memory [11]. Furthermore, TDX uses two extended page tables (EPTs) to ensure address-translation and memory-layout integrity, as well as allow TDs to communicate with untrusted entities while maintaining isolating TD's private memory [2], [11].

Measurements. The TDX architecture provides a TD with two types of measurement registers: TD Measurement Register (TDMR) for buildtime and Runtime Measurement Register (RTMR) for runtime of the TD [2]. During the creation of a TD, the TDX Module extends the TDMR with the measurements and metadata of the initially allocated pages for the TD. RTMRs, on the other hand, are used for code and data measurements at runtime [11].

Attestation. TDX supports an explicit remote attestation, enabled through an implicit local attestation [11]. For the following, we go through each TD attestation step as detailed in Figure 5, which is adapted from the TDX white paper [2] and [11]. A remote *Challenger* first sends an attestation request to a TD by transmitting a nonce (1). Upon receiving this request, the TD requests a local report called TDREPORT from the *TDX Module* (2). The *TDX Module* then invokes the `SEAMREPORT` operation to request the CPU to generate an HMAC-protected report containing TD measurements, TD attributes, TD identities, and the TCB SVNs (3). The generated TDREPORT is then handed over to the *TDQE* for further processing (4). The *TDQE* uses the `EVERIFYREPORT2` instruction to check whether the header information, TCB SVNs, and the computed MAC match their expected values (5). If the verification is successful, the *TDQE* replaces the MAC in the TDREPORT with a digital signature generated by an asymmetric-attestation key to form what is known as a QUOTE (6). The generated QUOTE is then sent back to the TD and is relayed back to the remote *Challenger* (7). On receiving the QUOTE, the *Challenger* verifies the signature of the QUOTE, as well as the MRTD and RTMRs of the TD (8). After successful verification, the *Challenger* can trust the *TD* for further communication and computation.

3.3. AMD SEV-SNP

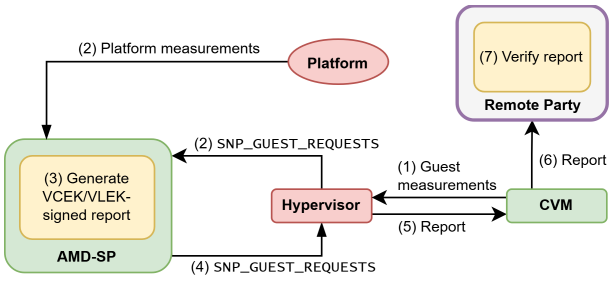

Figure 6: AMD SEV-SNP attestation flow [3], [12]

AMD SEV-SNP is AMD's third generation VM-based TEE solution, which builds upon the previous SEV technologies, specifically [13]:

1) **SEV**: Assigns each VM with a unique encryption key to protect their in-use data, providing guest memory isolation and confidentiality.
2) **SEV-Encrypted State (ES)**: Encrypts the VM register state and obfuscates data being used by the VM from the hypervisor, preventing information leakage to untrusted components.

AMD SEV-SNP introduces new hardware-based security protection with strong memory integrity guarantees through the AMD Secure Processor (AMD-SP) [3].

Confidentiality and Integrity. Similar to TDX, confidentiality is primarily achieved through memory encryption. Each CVM is initialized with a unique ephemeral encryption key, which ensures that specific data is only accessible to the associated SEV-SNP guest [13]. In addition to confidentiality, SEV-SNP architecture enforces data integrity by ensuring that a CVM must be able to read the value it last wrote if it can read a private encrypted page [3]. This guarantee is realized through a structure called the Reverse Map Table (RMP) and an updated nested page table walk, which enforces proper access control to memory pages in the system.

Measurements. The measurements included in the attestation report can be broken down into two parts: platform and guest. Platform measurements ensure that the platform running the SEV-SNP guests is running the latest firmware and microcode by collecting information regarding the TCB SVN [13], SVN threshold, unique chip ID, and specific platform properties [14]. On the other hand, the guest measurements collect various information about a specific guest during its initialization lifecycle [12]. Measurements of the pages and metadata associated with the guest are stored inside a launch digest, which is then compared against the expected guest measurements [14].

Attestation. SEV-SNP only supports remote attestation. In contrast to TDX's multi-module approach, *CVMs* only need to communicate with the *AMD-SP* during the attestation process, as shown in Figure 6. At the guest creation process, a set of private communication keys is created by the *AMD-SP* to facilitate secure direct communication between the *CVM* and the *AMD-SP* [13]. These guests can request an attestation report to the *AMD-SP* at any time through the *hypervisor* by first constructing a `MSG_REPORT_REQ` message, which includes the guest-specific measurements [12] (1). The hypervisor then invokes `SNP_GUEST_REQUESTS`, which wraps the message and sends it to the *AMD-SP*. It is important to note that the *hypervisor* cannot access (read or write) the messages without detection [12] (2). *AMD-SP* then generates the attestation report containing the measurements, optional public keys for communication, and an arbitrary report data field [13]. This report is signed with either the Versioned Chip Endorsement Key (VCEK) or the Versioned Loaded Endorsement Key (VLEK). While both keys are provisioned by the AMD Key Distribution Service (KDS) and the current TCB SVN, VCEK uses chip-unique seed, whereas VLEK uses CSP-unique seed maintained by the KDS [12] (3). The signed report is returned to the *CVM* embedded within the `MSG_REPORT_RSP` message, which is again sent via the `SNP_GUEST_REQUESTS` command [12]

(4,5). This report can be sent to a *remote party* looking to establish a secure communication channel for confidential computing [3] (6). Before trust can be established, the *remote party* verifies the software component of the TCB by checking the report content. Additionally, verifying the VCEK/VLEK-signed report proves the platform's authenticity, thus verifying the hardware component of the TCB (7).

4. Conclusion and Future Work

In this paper, we abstracted intricate concepts revolving around modern TEEs by giving a generic overview of the different TEE models and attestation flows. To observe how these concepts fit into state-of-the-art TEEs, we explored Intel SGX, Intel TDX, and AMD SEV-SNP. Specifically, we investigated how these TEE solutions guarantee confidentiality and integrity, what measurements are being attested, and how their attestations are carried out.

Our findings highlight the complex attestation capabilities of different TEEs. While various studies have conducted benchmarks measuring the impact TEEs have on the performance of a system, there is a lack of literature exploring the performance overhead incurred by generating attestation reports. Moving forward, conducting more practical research to explore such technicalities is beneficial as it also plays a significant role in gauging the scalability of using TEEs.

References

[1] Confidential Computing Consortium, "Confidential Computing: Hardware-Based Trusted Execution for Applications and Data," Nov. 2022, (Accessed: Jun. 16, 2024). [Online]. Available: https://confidentialcomputing.io

[2] Intel, "Intel Trust Domain Extensions (Intel TDX)," *White Paper*, Feb. 2023, (Accessed: Jun. 16, 2024). [Online]. Available: https://www.intel.com/content/www/us/en/developer/tools/trust-domain-extensions/documentation.html

[3] AMD, "Strengthening VM isolation with integrity protection and more," *AMD White Paper*, Jan. 2020, (Accessed: Apr. 06, 2024). [Online]. Available: https://www.amd.com/system/files/TechDocs/SEV-SNP-strengthening-vm-isolation-with-integrity-protection-and-more.pdf

[4] J. Ménétrey, C. Göttel, M. Pasin, P. Felber, and V. Schiavoni, "An Exploratory Study of Attestation Mechanisms for Trusted Execution Environments," Mar. 2022, (Accessed: Jun. 16, 2024). [Online]. Available: https://doi.org/10.48550/arXiv.2204.06790

[5] T. Geppert, S. Deml, D. Sturzenegger, and N. Ebert, "Trusted Execution Environments: Applications and Organizational Challenges," *Frontiers in Computer Science*, vol. 4, 2022, (Accessed: Jun. 16, 2024). [Online]. Available: https://www.frontiersin.org/articles/10.3389/fcomp.2022.930741

[6] Intel, "Intel Software Guard Extensions (Intel SGX) Developer Guide," *Developer Guide*, Mar. 2020, (Accessed: Jun. 16, 2024). [Online]. Available: https://www.intel.com/content/www/us/en/content-details/671581/intel-sgx-developer-guide-for-windows.html

[7] H. Birkholz, D. Thaler, M. Richardson, N. Smith, and W. Pan, "Remote ATtestation procedureS (RATS) Architecture," RFC 9334, Jan. 2023, (Accessed: Jun. 16, 2024). [Online]. Available: https://www.rfc-editor.org/info/rfc9334

[8] M. Li, Y. Yang, G. Chen, M. Yan, and Y. Zhang, "SoK: Understanding Design Choices and Pitfalls of Trusted Execution Environments," in *Proceedings of the 19th ACM Asia Conference on Computer and Communications Security*, ser. ASIA CCS '24. New York, NY, USA: Association for Computing Machinery, 2024, p. 1600–1616. [Online]. Available: https://doi.org/10.1145/3634737.3644993

[9] V. Costan and S. Devadas, "Intel SGX Explained," *IACR Cryptol. ePrint Arch.*, vol. 2016, p. 86, 2016, (Accessed: Jun. 16, 2024). [Online]. Available: https://api.semanticscholar.org/CorpusID:28642809

[10] I. Anati, S. Gueron, S. Johnson, and V. Scarlata, "Innovative technology for cpu based attestation and sealing," 2013, (Accessed: Jun. 16, 2024). [Online]. Available: https://api.semanticscholar.org/CorpusID:14218854

[11] P.-C. Cheng, W. Ozga, E. Valdez, S. Ahmed, Z. Gu, H. Jamjoom, H. Franke, and J. Bottomley, "Intel TDX Demystified: A Top-Down Approach," 2023, (Accessed: Jun. 16, 2024). [Online]. Available: https://doi.org/10.48550/arXiv.2303.15540

[12] AMD. (2023, Sep.) SEV Secure Nested Paging Firmware ABI Specification . (Accessed: Jun. 16, 2024). [Online]. Available: https://www.amd.com/content/dam/amd/en/documents/epyc-technical-docs/specifications/56860.pdf

[13] D. Kaplan, "Hardware VM Isolation in the Cloud: Enabling confidential computing with AMD SEV-SNP technology," *Queue*, vol. 21, no. 4, p. 49–67, sep 2023, (Accessed: Jun. 16, 2024). [Online]. Available: https://doi.org/10.1145/3623392

[14] AMD. (2022, Sep.) AMD SEV-SNP Attestation: Establishing Trust in Guests. (Accessed: Jun. 16, 2024). [Online]. Available: https://www.amd.com/content/dam/amd/en/documents/developer/lss-snp-attestation.pdf