A Shallow Dive into Attestation Mechanisms for Trusted Execution Environments

(Poster submission with short paper)

Anonymous Author(s)

Abstract

Attestation is a fundamental building block to establishing trust over a software system. When used in conjunction with trusted execution environments, it guarantees the genuineness of the code executed against powerful attackers and threats, paving the ways for adoption in several sensitive application domains. This short paper briefly reviews remote attestation principles and explains how modern trusted execution environments (such as Intel SGX, Arm TrustZone, or RISC-V architectures) leverage these mechanisms.

Keywords

Trusted execution environments, attestation, Intel SGX, ARM TrustZone, AMD SEV, RISC-V

ACM Reference Format:
Anonymous Author(s). 2022. A Shallow Dive into Attestation Mechanisms, for Trusted Execution Environments: (Poster submission with short paper). In *Proceedings of ACM SAC Conference (SAC'22).* ACM, New York, NY, USA, Article 4, 4 pages. https://doi.org/xx.xxx/xxx_x

1 Introduction

Confidentiality, integrity and availability are essential features to guarantee when building secure computer systems, particularly when the underlying system cannot be fully trusted. For example, video broadcasting software can be tampered with by end-users to circumvent digital rights management, or virtual machines are candidly open to the indiscretion of their cloud hosts. Recent evolutions in hardware such as Intel SGX, Arm TrustZone, AMD SEV and upcoming RISC-V mechanisms made it possible to establish Trusted Execution Environments (TEEs), in which a piece of software can be executed with more robust security guarantees. These evolutions offer different degrees of guarantees, all contributing to increasing the confidentiality and integrity of applications such as the examples given.

Attestation plays an important role when using TEEs in a distributed setup, as it allows verification of software authenticity and integrity. Through remote attestation, one can be sure to be communicating with a specific, trusted (attested) program. Guarantees are stronger when the attestation is provided by a TEE, because one can also be sure that the program is protected (by the TEE) against future indiscretion and tampering. We contribute a short survey of the current state-of-the-art regarding *attestation mechanisms* for TEEs, including the open ISA RISC-V.

This short paper is organized as follows. In §2, we describe the general principles of attestation, in particular highlighting the differences between local and remote attestation. In §3 we survey the existing support for attestation mechanisms in the TEE implementations currently available in commodity hardware. We conclude in §4 discussing some future directions.

2 Remote attestation

The mechanism of remote attestation allows to establish trust between devices and provide cryptographic proofs that the executing software is genuine and untampered [11]. In the remainder of this work, we adopt the terminology proposed by the IETF to describe remote attestation and related architectures [5]. Under these terms, *a relying party* wishes to establish a trusted relationship with an *attester*, with the help of a verifier. The attester provides the state of its system, indicating the hardware and the software stack that runs on its device by collecting a set of claims. An example of *claim* is the devices' application code measurement, under the form of a cryptographically secure hash. Claims are collected and cryptographically signed to form an *evidence*, that is later observed and potentially asserted by the *verifier*. Once the attester is proven genuine, the relying party can safely interact with the attested device and, for example, transfer confidential data or delegate computations.

The problem of remotely attesting software has been studied in depth, several academic solutions have been proposed, and industrial implementations already exist. Three leading families of remote attestation methods exist: software-based, hardware-based, and hybrid. Software-based remote attestation [10, 32, 37] do not depend on any particular hardware and are interesting for low-cost use cases. Hardware-based remote attestation rely on tamper-resistant hardware as a Trusted Platform Module (TPM) to ensure that the claims are trustworthy [41], or simply a Physical Unclonable Function (PUF) that prevents impersonations by using unique hardware marks produced at manufacture [15, 18]. Hybrid solutions combine hardware devices, and software implementations [8, 16, 27], in an attempt to leverage advantages from both sides. Some researchers used hardware/software co-design techniques to propose a hybrid design with a formal proof of correctness [26]. Finally, more specific implementations using TEEs include Sanctum [21], LIRA-V [35] and SRACARE [14].

2.1 Mutual attestation

Trusted applications may need stronger trust assurances by ensuring both ends of a secure channel to be attested. For example, when retrieving confidential data from a sensing IoT device (data is sensitive), the device must authenticate the remote party, while

SAC'22, April 25 –April 29, 2022, Brno, Czech Republic

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/xx.xxx/xxx_x

the latter must ensure the sensing device has not been spoofed or tampered with. For that purpose, mutual attestation protocols have been designed to appraise the trustworthiness of both end devices involved in a communication. We also report how mutual attestation has also been studied in the context of TEEs [39], as we see detail further in §3.

3 Attestation for TEEs

Several solutions exist to implement hardware support for trusted computing, and Trusted Execution Environments (TEEs) are particularly promising. Typically, a TEE consists of isolating key components of the system, *e.g.*, portions of the memory, denying access to more privileged but untrusted systems, such as kernel and machine modes. Depending on the implementation, it guarantees the confidentiality and the integrity of the code and data of trusted applications, thanks to the assistance of CPU security features. Prominent processor designers and vendors offer TEEs nowadays for commodity or server-grade processors, such as Intel with Intel SGX[12], AMD SEV[1], or Arm TrustZone[28].

More recently, RISC-V, an open ISA with multiple open-source core implementations, ratified the physical memory protection (PMP) instructions, offering similar capabilities to memory protection offered by aforementioned technologies [29]. Many emerging and academic and proprietary frameworks capitalised on the standard RISC-V primitives to provide system isolation, including Sanctum [13], TIMBER-V [40], Keystone [22], Hex-Five MultiZone [17], HECTOR-V [25] and more [24].

The attestation of software and hardware components requires an environment to issue an evidence securely. In practice, this role is usually assigned to some software or hardware mechanism that cannot be tampered with. These environments rely on measuring the executed software and combining that output with cryptographical values derived from the hardware, such as a root of trust fused in the die or a physical unclonable function. We analyzed today's practices for the leading processor vendors for issuing cryptographically signed evidences.

Figure 1 illustrates the generic workflow TEE developers usually follow for the deployment of trusted applications. Initially, the application is compiled and measured on the developers' premise. It is later transferred to an untrusted system, which executes in the TEE facility. Once the trusted application is loaded and required to receive sensitive data, it communicates with a verifier to establish a trusted channel. The TEE environment must facilitate this transaction by exposing an evidence to the trusted application, which adds key material to bootstrap a secure channel from the TEE, thus preventing an attacker from eavesdropping on the communication. The verifier asserts the evidence, which maintains a list of reference values to identify a genuine instance of the trusted application. If recognised as trustworthy, the verifier can proceed to the data exchange.

3.1 Intel SGX

Intel Secure Guard Extensions (SGX) [12] introduced TEEs for mass-market processors in 2015. Specifically, Intel's Skylake architecture introduced a new set of processor instructions to create encrypted regions of memory, called *enclaves*. A trusted application executing in an enclave may establish a local attestation with another enclave running on the same hardware. This principle is further extended

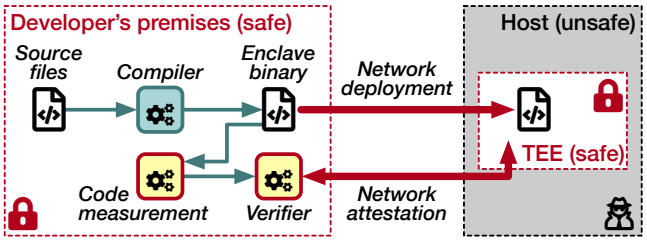

Figure 1: The generic workflow of deployment and attestation of TEEs.

by Intel thanks to the usage of the built-in *quoting enclave*. A trusted application can hence receive a cryptographically signed evidence, *i.e.*, a *quote*, which may be enhanced by additional information, such as a public key for channel establishment. This quote can then be forwarded to a relying party and verified remotely using the Intel attestation service [4, 6] or a dedicated public key infrastructure [31].

Intel designed their remote attestation protocol based on the SIGMA protocol [19]. While the quoting enclave is closed-source and the microcode of Intel SGX not disclosed, recent work analyzed the TEE and its attestation mechanism formally [30, 38]. MAGE [9] further extended the remote attestation scheme of Intel SGX by offering mutual attestation for a group of enclaves without trusted third parties.

3.2 Arm TrustZone architectures

Arm TrustZone [28] provides the hardware elements to establish a single TEE per system. Broadly adopted by commodity devices (including mobile devices, IoT edge nodes, *etc.*), TrustZone splits the processor into two states: the secure world (TEE) and the normal world (untrusted environment). A secure monitor instruction performs the switching of worlds. The two worlds operate within their own user and kernel space: the former uses a trusted operating system (*e.g.*, OP-TEE [23]) and runs trusted applications as isolated processes, while the latter uses a traditional operating system.

Despite the commercial success of this technology, TrustZone lacks attestation mechanisms, preventing relying parties to validate and trusting the state of the TrustZone TEE remotely. Researchers proposed several variants of one-way remote attestation protocol for Arm TrustZone [3, 42], as well as mutual remote attestation [2]. All of these propositions require hardware primitives to be available on the system-on-chip (SoC): *(i)* a root of trust only available in the secure world, *(ii)* secure source of randomness for cryptographical operations, and *(iii)* a secure boot mechanism, ensuring the bootloader can either be authenticated or measured.

In addition to the aforementioned remote attestation mechanism, Shepherd et al. [34] proposed a protocol for sensing devices running on Arm processors, establishing mutually trusted channels for bi-directional attestation.

3.3 AMD SEV

AMD SEV [1] allows isolating virtualized environments (VMs) from untrusted hypervisors. It exploits a closed ARM Cortex-v5 processor as a secure co-processor. At its core, it leverages a chip endorsement key (CEK) issued by AMD for its attestation mechanism. However, this was recently proven unsecure [7], exposing the system to roll-back attacks and allowing a malicious cloud provider with physical

access to SEV machines to easily install malicious firmware and be able to read in clear the (otherwise protected) system. Future iterations of this technology, *i.e.*, AMD SEV-SNP [33], plan to overcome these limitations, typically by means of in-silico redesigns.

3.4 RISC-V architectures

RISC-V has seen numerous TEE propositions based on its PMP instructions and provided remote attestation mechanisms inspired by the previous protocols highlighted in the paper.

Sanctum [13] has been the first proposition with support for attesting trusted applications. It offers similar promises to Intel's SGX by providing provable and robust software isolation, running in enclaves. The authors replaced Intel's opaque microcode with a secure open-source monitor as a means to provide verifiable protection. This solution added hardware at the interfaces between generic building blocks, with a minimal performance impact. A remote attestation protocol is proposed, as well as a comprehensive design for deriving trust from a root of trust. The enclaves are signed thanks to a signing enclave, similarly to Intel SGX. The same authors have later published an extension for Sanctum [20], establishing a secure boot mechanism and an alternative method for remote attestation by deriving a cryptographic identity from manufacturing variation using PUF, which is useful when a root of trust is not present.

TIMBER-V [40] achieved the isolation of execution on small embedded processors thanks to hardware-assisted memory tagging. Tagged memory transparently associates blocks of memory with additional metadata. Unlike Sanctum, they aim to bring enclaves to smaller RISC-V featuring only limited physical memory. Similarly to TrustZone, the user mode (U-mode) and the supervisor mode (S-mode) are split into a secure and normal world. The secure supervisor mode runs a trust manager, called *TagRoot*, which manages the tagging of the memory. The secure user mode improves the model of TrustZone, as it can handle multiple concurrent enclaves, which are isolated from each other. The trust manager exposes an API for the enclaves to get cryptographic key material issued based on the enclave identify and a secret platform key. While TIMBER-V does not specify a remote attestation mechanism, the key material can be used to create a secure channel of communication with a remote relying party.

Keystone [22] is a modular framework that provides the building blocks to create trusted execution environments, rather than providing an all-in-one solution that is inflexible and is another fixed design point. Instead, they advocate that hardware should provide security primitives instead of point-wise solutions. Keystone implements a secure monitor at machine mode (M-mode) and relies on the RISC-V PMP instructions to provide isolated execution and, therefore, does not require any hardware change. Since Keystone leverages features composition, the framework users can select their own set of security primitives, *e.g.*, memory encryption, dynamic memory management and cache partitioning. Each trusted application executes in user mode (U-mode) and embeds a runtime that executes in supervisor mode (S-mode). The runtime decouples the infrastructure aspect of the TEE (*e.g.*, memory management, scheduling) from the security aspect handled by the secure monitor. As such, Keystone programmers can roll their custom runtime to fine-grained control of the computer resources without managing the TEE's security. Remote attestation is a first-class citizen in Keystone, where the secure monitor exposes a range of API to query a signed evidence, based on the code measurement. While the authors consider key distribution and infrastructure orthogonal, they provide a case study with a trusted application that is remotely attested.

Lastly, LIRA-V [36] drafted a mutual remote attestation for constrained edge devices. While this solution does not enable the execution of arbitrary code in a TEE, it introduces a comprehensive remote attestation mechanism that leverages PMP for code protection of the attesting environment and the availability of a root of trust to issue cryptographically signed evidences. The proposed protocol relies exclusively on machine mode (M-mode) or machine and user mode (M-mode and U-mode). It has been formally verified and enables the creation of a trusted channel of communication, which Keystone does not strictly cover.

4 Conclusion

This work presents state-of-the-art remote attestation schemes, which leverage hardware-assisted TEEs, helpful for deploying and running trusted applications from commodity devices to cloud providers. TEE-based remote attestation has not yet been extensively studied and seems to remain an industrial challenge. This survey highlights four different architectural extensions: Intel SGX, Arm TrustZone, ARM SEV, and upcoming RISC-V TEEs. Intel SGX provides a remote attestation protocol built-in, with an incomplete yet fixed hardware design. Arm TrustZone does not provide attestation for software running in the trusted environment, relying on the community to develop software-based alternatives. ARM SEV is designed for virtualized environments, shielding VMs from untrusted hypervisors. It comes with a built-in, however severely flawed, attestation mechanism. RISC-V TEEs (*e.g.*, Keystone) advocate that manufacturers must expose security primitives (*e.g.*, PMP), so TEE programmers may compose them in a versatile solution, with scales better for future extensions. Whether provided by the manufacturers or developed by third parties, remote attestation remains an essential part of the design of trusted computing solutions. They are the foundation of trust for remote computing where the target environments are not fully trusted.

References

[1] Advanced Micro Devices. 2019. *Secure Encrypted Virtualization API: Technical Preview.* Technical Report 55766. Advanced Micro Devices.

[2] Jaehwan Ahn, Il-Gu Lee, and Myungchul Kim. 2020. Design and implementation of hardware-based remote attestation for a secure internet of things. *Wireless Personal Communications* 114, 1 (2020), 295–327.

[3] Jae-Young Ahn, Il-Gu Lee, and Myungchul Kim. 2020. Design and Implementation of Hardware-Based Remote Attestation for a Secure Internet of Things. *Wireless Personal Communications* (2020), 1–33. https://doi.org/10.1007/s11277-020-07364-5

[4] Ittai Anati, Shay Gueron, Simon Johnson, and Vincent Scarlata. 2013. Innovative technology for CPU based attestation and sealing. In *Proceedings of the 2nd international workshop on hardware and architectural support for security and privacy*, Vol. 13. Citeseer, 7.

[5] Henk Birkholz, Dave Thaler, Michael Richardson, Ned Smith, and Wei Pan. 2021. *Remote Attestation Procedures Architecture.* Internet-Draft draft-ietf-rats-architecture-12. Internet Engineering Task Force. https://datatracker.ietf.org/doc/html/draft-ietf-rats-architecture-12 Work in Progress.

[6] Ernie Brickell and Jiangtao Li. 2007. Enhanced privacy ID: A direct anonymous attestation scheme with enhanced revocation capabilities. In *Proceedings of the 2007 ACM workshop on Privacy in electronic society*. 21–30.

[7] Robert Buhren, Christian Werling, and Jean-Pierre Seifert. 2019. Insecure until proven updated: analyzing AMD SEV's remote attestation. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. 1087–1099.

[8] Xavier Carpent, Norrathep Rattanavipanon, and Gene Tsudik. 2018. Remote attestation of iot devices via smarm: Shuffled measurements against roving malware. In *2018 IEEE international symposium on hardware oriented security and trust (HOST)*. IEEE, 9–16.

[9] Guoxing Chen and Yinqian Zhang. 2020. Mage: Mutual attestation for a group of enclaves without trusted third parties. *arXiv preprint arXiv:2008.09501* (2020).

[10] Young-Geun Choi, Jeonil Kang, and DaeHun Nyang. 2007. Proactive Code Verification Protocol in Wireless Sensor Network. In *Computational Science and Its Applications – ICCSA 2007*, Osvaldo Gervasi and Marina L. Gavrilova (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 1085–1096.

[11] George Coker, Joshua Guttman, Peter Loscocco, Amy Herzog, Jonathan Millen, Brian O'Hanlon, John Ramsdell, Ariel Segall, Justin Sheehy, and Brian Sniffen. 2011. Principles of remote attestation. *International Journal of Information Security* 10, 2 (2011), 63–81.

[12] Victor Costan and Srinivas Devadas. 2016. Intel SGX Explained. Cryptology ePrint Archive, Report 2016/086. https://ia.cr/2016/086

[13] Victor Costan, Ilia Lebedev, and Srinivas Devadas. 2016. Sanctum: Minimal Hardware Extensions for Strong Software Isolation. In *25th USENIX Security Symposium (USENIX Security 16)*. USENIX Association, Austin, TX, 857–874. https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/costan

[14] Avani Dave, Nilanjan Banerjee, and Chintan Patel. 2020. SRACARE: Secure Remote Attestation with Code Authentication and Resilience Engine. In *2020 IEEE International Conference on Embedded Software and Systems (ICESS)*. 1–8. https://doi.org/10.1109/ICESS49830.2020.9301516

[15] Wei Feng, Yu Qin, Shijun Zhao, and Dengguo Feng. 2018. AAoT: Lightweight attestation and authentication of low-resource things in IoT and CPS. *Computer Networks* 134 (2018), 167–182.

[16] Aurélien Francillon, Quan Nguyen, Kasper B. Rasmussen, and Gene Tsudik. 2014. A minimalist approach to Remote Attestation. In *2014 Design, Automation Test in Europe Conference Exhibition (DATE)*. 1–6. https://doi.org/10.7873/DATE.2014.257

[17] Cesare Garlati and Sandro Pinto. 2020. A Clean Slate Approach to Linux Security RISC-V Enclaves. In *Proceedings of the Embedded World Conference, Nuremberg, Germany*.

[18] Joonho Kong, Farinaz Koushanfar, Praveen K. Pendyala, Ahmad-Reza Sadeghi, and Christian Wachsmann. 2014. PUFatt: Embedded platform attestation based on novel processor-based PUFs. In *2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC)*. 1–6. https://doi.org/10.1145/2593069.2593192

[19] Hugo Krawczyk. 2003. SIGMA: The 'SIGn-and-MAc' Approach to Authenticated Diffie-Hellman and Its Use in the IKE Protocols. In *Advances in Cryptology - CRYPTO 2003*, Dan Boneh (Ed.). Springer Berlin Heidelberg, 400–425. https://doi.org/10.1007/978-3-540-45146-4_24

[20] Ilia Lebedev, Kyle Hogan, and Srinivas Devadas. 2018. Invited Paper: Secure Boot and Remote Attestation in the Sanctum Processor. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*. 46–60. https://doi.org/10.1109/CSF.2018.00011

[21] Ilia Lebedev, Kyle Hogan, and Srinivas Devadas. 2018. Secure boot and remote attestation in the sanctum processor. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*. IEEE, 46–60.

[22] Dayeol Lee, David Kohlbrenner, Shweta Shinde, Krste Asanović, and Dawn Song. 2020. Keystone: An Open Framework for Architecting Trusted Execution Environments. In *Proceedings of the Fifteenth European Conference on Computer Systems (EuroSys '20)*. Association for Computing Machinery, New York, NY, USA, Article 38, 16 pages. https://doi.org/10.1145/3342195.3387532

[23] Linaro. 2021. OP-TEE: Open Source Trusted Execution Environment. https://www.op-tee.org

[24] Samuel Lindemer, Gustav Midéus, and Shahid Raza. 2020. Real-time Thread Isolation and Trusted Execution on Embedded RISC-V. In *First International Workshop on Secure RISC-V Architecture Design Exploration (SECRISC-V'20)*.

[25] Pascal Nasahl, Robert Schilling, Mario Werner, and Stefan Mangard. 2021. HECTOR-V: A Heterogeneous CPU Architecture for a Secure RISC-V Execution Environment. In *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security (ASIA CCS '21)*. Association for Computing Machinery, New York, NY, USA, 187–199. https://doi.org/10.1145/3433210.3453112

[26] Ivan De Oliveira Nunes, Karim Eldefrawy, Norrathep Rattanavipanon, Michael Steiner, and Gene Tsudik. 2019. VRASED: A Verified Hardware/Software Co-Design for Remote Attestation. In *28th USENIX Security Symposium (USENIX Security 19)*. USENIX Association, Santa Clara, CA, 1429–1446. https://www.usenix.org/conference/usenixsecurity19/presentation/de-oliveira-nunes

[27] Ivan De Oliveira Nunes, Sashidhar Jakkamsetti, Norrathep Rattanavipanon, and Gene Tsudik. 2020. On the TOCTOU Problem in Remote Attestation. *CoRR* abs/2005.03873 (2020). arXiv:2005.03873 https://arxiv.org/abs/2005.03873

[28] Sandro Pinto and Nuno Santos. 2019. Demystifying arm trustzone: A comprehensive survey. *ACM Computing Surveys (CSUR)* 51, 6 (2019), 1–36.

[29] RISC-V. 2019. RISC-V Foundation Announces Ratification of the RISC-V Base ISA and Privileged Architecture Specifications. https://riscv.org/announcements/2019/07/risc-v-foundation-announces-ratification-of-the-risc-v-base-isa-and-privileged-architecture-specifications/

[30] Muhammad Usama Sardar, Do Le Quoc, and Christof Fetzer. 2020. Towards Formalization of Enhanced Privacy ID (EPID)-based Remote Attestation in Intel SGX. In *2020 23rd Euromicro Conference on Digital System Design (DSD)*. 604–607. https://doi.org/10.1109/DSD51259.2020.00099

[31] Vinnie Scarlata, Simon Johnson, James Beaney, and Piotr Zmijewski. 2018. Supporting third party attestation for Intel SGX with Intel data center attestation primitives. *White paper* (2018).

[32] Arvind Seshadri, Mark Luk, Elaine Shi, Adrian Perrig, Leendert Van Doorn, and Pradeep Khosla. 2005. Pioneer: Verifying integrity and guaranteeing execution of code on legacy platforms. In *Proceedings of ACM Symposium on Operating Systems Principles (SOSP)*, Vol. 173. 10–1145.

[33] AMD SEV-SNP. 2020. Strengthening VM isolation with integrity protection and more. *White Paper, January* (2020).

[34] Carlton Shepherd, Raja Naeem Akram, and Konstantinos Markantonakis. 2017. Establishing Mutually Trusted Channels for Remote Sensing Devices with Trusted Execution Environments. In *Proceedings of the 12th International Conference on Availability, Reliability and Security (ARES '17)*. Association for Computing Machinery, New York, NY, USA, Article 7, 10 pages. https://doi.org/10.1145/3098954.3098971

[35] Carlton Shepherd, Konstantinos Markantonakis, and Georges-Axel Jaloyan. 2021. LIRA-V: Lightweight Remote Attestation for Constrained RISC-V Devices. *arXiv preprint arXiv:2102.08804* (2021).

[36] Carlton Shepherd, Konstantinos Markantonakis, and Georges-Axel Jaloyan. 2021. LIRA-V: Lightweight Remote Attestation for Constrained RISC-V Devices. In *2021 IEEE Security and Privacy Workshops (SPW)*. 221–227. https://doi.org/10.1109/SPW53761.2021.00036

[37] Rodrigo Vieira Steiner and Emil Lupu. 2019. Towards more practical software-based attestation. *Computer Networks* 149 (2019), 43–55.

[38] Pramod Subramanyan, Rohit Sinha, Ilia Lebedev, Srinivas Devadas, and Sanjit A. Seshia. 2017. A Formal Foundation for Secure Remote Execution of Enclaves. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS '17)*. Association for Computing Machinery, New York, NY, USA, 2435–2450. https://doi.org/10.1145/3133956.3134098

[39] Furkan Turan and Ingrid Verbauwhede. 2019. Propagating Trusted Execution through Mutual Attestation. In *Proceedings of the 4th Workshop on System Software for Trusted Execution (SysTEX '19)*. Association for Computing Machinery, New York, NY, USA, Article 2, 6 pages. https://doi.org/10.1145/3342559.3365334

[40] Samuel Weiser, Mario Werner, Ferdinand Brasser, Maja Malenko, Stefan Mangard, and Ahmad-Reza Sadeghi. 2019. TIMBER-V: Tag-Isolated Memory Bringing Fine-grained Enclaves to RISC-V. In *NDSS*.

[41] Wenjuan Xu, Xinwen Zhang, Hongxin Hu, Gail-Joon Ahn, and Jean-Pierre Seifert. 2012. Remote Attestation with Domain-Based Integrity Model and Policy Analysis. *IEEE Transactions on Dependable and Secure Computing* 9, 3 (2012), 429–442. https://doi.org/10.1109/TDSC.2011.61

[42] Shijun Zhao, Qianying Zhang, Yu Qin, Wei Feng, and Dengguo Feng. 2019. SecTEE: A Software-Based Approach to Secure Enclave Architecture Using TEE. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS '19)*. Association for Computing Machinery, New York, NY, USA, 1723–1740. https://doi.org/10.1145/3319535.3363205