Architectures for Security: A comparative analysis of hardware security features in Intel SGX and ARM TrustZone

Muhammad Asim Mukhtar
Information Technology University
Lahore, Pakistan
asim.mukhtar@itu.edu.pk

Muhammad Khurram Bhatti
Information Technology University
Lahore, Pakistan
khurram.bhatti@itu.edu.pk

Guy Gogniat
University of South Brittany
Lorient, France
guy.gogniat@univ-ubs.fr

***Abstract*—A variety of applications are executing on a large untrusted computing base, which includes the operating system, hypervisor, firmware, and hardware. This large computing base is becoming complex and unverifiable. This untrusted computing base problem opens a way for a malicious application to steal secrets of a security-critical application by compromising the untrusted computing base. To resolve the untrusted computing base problem, computer architectures have introduced a concept of the trusted execution environment, which aim to ensure the sensitive data to be stored and processed in an isolated environment. Existing popular trusted execution environments are relying on hardware to isolate the environments without or minimum relying on system software. However, existing hardware assisted trusted execution environments are still vulnerable to sophisticated attacks. This paper analyses popular trusted execution environments that are Intel SGX and ARM TrustZone in order to provide better insights about the intended scope of the protection. This paper illustrates the functionality, implementation and security analysis.**

***Index Terms*—Trusted Execution Environments, TEE, Memory isolation, Intel SGX, and ARM TrustZone.**

I. Introduction

Normal and security-critical applications are executing on a large untrusted computing base, which includes an operating system, hypervisor, firmware, and hardware. This large computing base is becoming complex and unverifiable. For example, an operating system such as Linux has 17 millions of lines code [2] and CVE has reported 166 vulnerabilities in it during the year of 2018 related to Denial-of-Service, overflow, unauthorized privilege gain, memory corruption, directory traversal, execute unauthorized code. Similarly, Xen is a well-known hypervisor that has 150,000 lines code [27], which has relatively small code than Linux but still has vulnerabilities, and CVE has reported 18 vulnerabilities in Xen in the year of 2018 [11]. Moreover, attacks that subvert firmware are reported [1] [25] [23]. Execution of normal and security-critical applications are executing on shared resources that controlled by untrusted computing base raises security threats. This opens the way for a malicious application to attack the vulnerabilities to gain the unauthorized privilege, and then steal secrets form security critical application's address space.

This research work is partially supported by the PHC PERIDOT Project e-health.SECURE and National Center for Cyber Security (NCCS), Pakistan.

To cope up the untrusted computing base problem, computer architectures have introduced the concept of trusted execution environments that aim to isolate security-critical applications from untrusted computing base. Trusted execution environments guarantee security by relying on less hardware and software computing base. Hardware is generally considered as the trusted base because the cost and complexity of attacks on hardware are usually high [12]. This leads the industry to develop computer architectures to develop a trusted execution environment for security-critical application maintained by hardware with no or less dependency on OS and hypervisor. These architectures includes ARM TrustZone Technology [17], Intel Software Guard eXtensions (SGX) [14] [20], AMD Memory Encryption Technologies [15], AMD Platform Secure Processor [13], x86 System Management Mode [8], and Intel Management Engine (ME) [22].

Intel SGX and ARM TrustZone are popular trusted execution environments. Both Intel SGX and ARM TrustZone are hardware assisted trusted execution environments but the mechanism behind making the trusted environment for trusted applications are different. Intel SGX creates a trusted environment for trusted applications such that it executes over existing untrusted system software. Whereas, ARM TrustZone creates a new trusted world for trusted applications that executes over trusted system software and hardware that only visible to the trusted world. These TEEs are vulnerable to a different set of attacks because of different mechanism of creating trusted execution environments. This paper analyses these trusted execution environments in order to provide better insights into the intended scope of the protection. This paper illustrates the functionality, implementation and security analysis.

II. Functionality and Implementation of ARM TrustZone and Intel SGX

A. ARM TrustZone

ARM TrustZone provides a set of intellectual property cores (IP blocks) to develop a partitioning based security framework. The way of combining these IP blocks by chip manufactures

978-1-5386-9609-5/19/$31.00 ©2019 IEEE

define the security properties. In general ARM TrustZone divides a system's resources (CPU, memory, and peripherals) into two classes referred to as secure world and normal world [17].

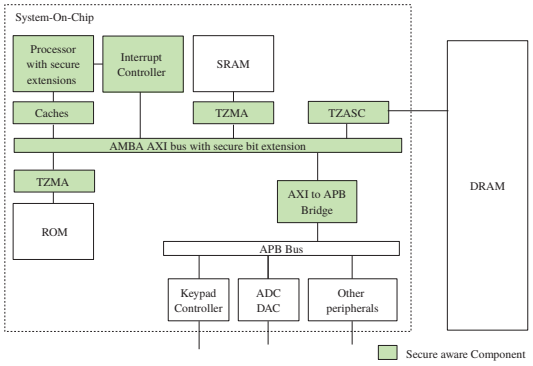

Fig. 1. ARM TrustZone Implementation

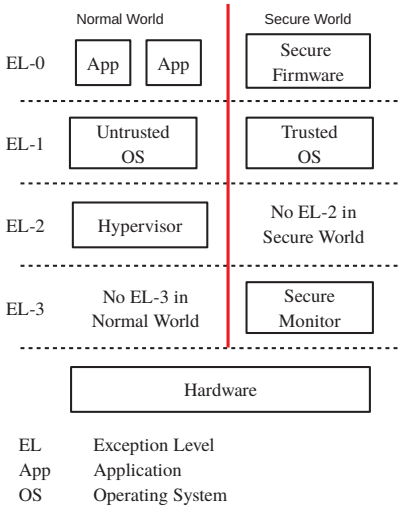

Fig. 2. ARM TrustZone software stack.

ARM TrustZone introduced a new execution environment in processor referred as secure world along with the old execution environment referred as normal world. The secure world also has multiple privilege levels same as normal world. Therefore, whole trusted software stack can be developed from user-level to system-level except hypervisor in secure-world, as shown in Figure 2. Each world has its own operating system and manages the resources for application belonging to its world's space. One world's software stack executes at a time on a processor. The context switching between secure and normal world is handled by monitor mode, which is the highest privilege-level of the secure world and can access both worlds system's resources. To reflect the current processor's world to other system's resources, bus is extended with a new bit called non-secure bit (NS). Cache lines are also extended with NS bit, which specifies the security state of the cache line. Extension of NS bit in each cache line eliminates the need of flushing the cache lines while context switches between a secure world and normal world, which results in low context switching overhead. The allocation of cache lines depends on the demand of each world and can evict the cache line of another world. TrustZone processor provides separate address translation units for the secure and normal worlds. This is achieved by implementing two page-table base registers, which are used by page walker according to the processor's current world. The physical addresses in the page-table entries are also extended to include the values of the NS bit to be issued on the AXI bus, as shown in Figure 3. The addition of secure bit in the address tag for each cache line effectively creates completely different views of the memory space to the software executing in different worlds.

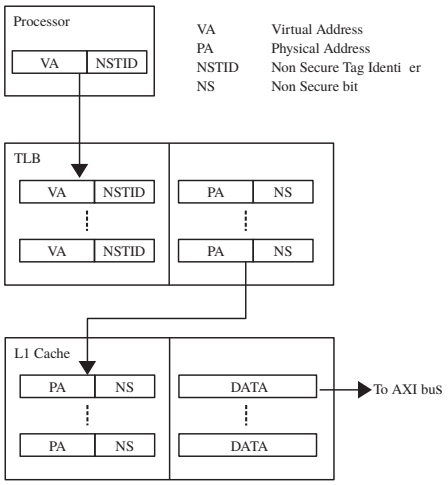

Fig. 3. ARM TrustZone TLB and cache isolation.

The memory modules like DRAM, SRAM, and ROM that are not designed according to extended AXI bus can be connected using adapters as shown in Figure 1. The TrustZone Memory Adapter (TZMA) can be used to partition an on-chip ROM or SRAM into a secure region and a normal region, and the TrustZone Address Space Controller (TZASC) partitions the memory space provided by a DRAM controller into secure and normal regions. A TrustZone-aware DMA controller rejects DMA transfers from the normal world that reference secure world addresses.

In ARM system most peripherals are connected to the APB bus, which is a low power bus than the main AXI bus. The APB protocol does not carry the NS bit. To defeat software attacks that use peripheral to extract information, and ARM TrustZone introduced the security handling feature AXI-to-APB Bridge, which interfaces the high-speed AXI domain to the low-power APB domain. The bridge contains an address decoder that selects the APB peripheral based on the incoming AXI transaction. The bridge takes a single bit input for each peripheral that is located on the bus to determine whether the peripheral is configured as secure or non-secure. The AXI-to-APB bridge will reject non-secure requests to secure peripheral address ranges using NS bit information of AXI bus.

There are two types of hardware interrupts in ARM TrustZone: Fast Interrupt Request (FIQ) and Interrupt Request (IRQ). Both of the interrupts can be configured as secure-interrupt by configuring the IRQ bit and FIQ bit in secure-configuration register (SCR). The secure interrupt is directly trapped into monitor mode (EL-3 level) and then forwarded to the normal world if required. ARM recommends that the IRQ should be used as the interrupt source of the normal world and the FIQ should be used as a secure-interrupt because most commonly used interrupt source most of the operating environments is IRQ, so the use of FIQ as the secure interrupt will require the fewest modifications to existing software.

B. Intel Software Guard Extensions (SGX)

Intel has provided the general purpose hardware-assisted TEE referred as Intel SGX. Intel SGX is an extension of x86 architecture with new set of security-related instructions [6] [8] [7]. These instructions are used by the security-critical applications to build hardware-assisted trusted environment referred to as an enclave [19]. Intel SGX enclave ensures the confidentiality using memory access checks with hardware maintained data structure and integrity by encrypting of data and code when goes outside the CPU package [16]. Intel SGX is a centralized security model, trusted computing base TCB is considered to be the CPU package.

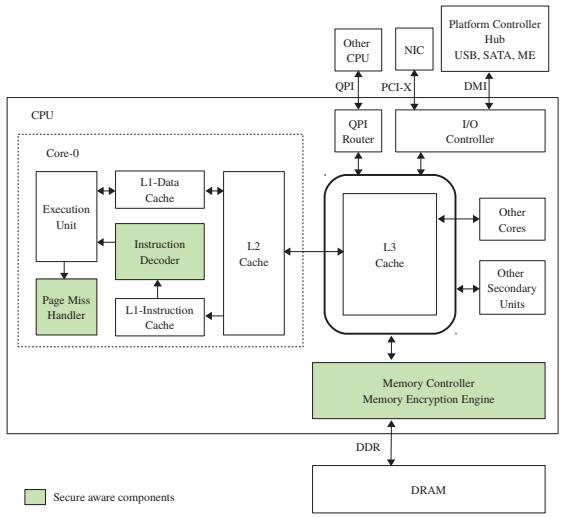

Fig. 4. Intel SGX implementation.

SGX processor does not provide orthogonal privileged levels to secure application as in the TrustZone, as shown in Figure 5. The application executes on the same untrusted operating system but security is achieved by matching with hardware managed meta-data, which OS cannot read or write. The reason behind achieving security over conventional software stack is to minimize the effort required to modify application code to benefit from SGX. History suggests this is a wise decision, as a large factor in the continued dominance of the Intel architecture is its ability to maintain backward compatibility.

Conceptually, Intel SGX offers a new type of address space to an application referred to as processor reserved memory (PRM) that have special security properties as shown in Figure 6. PRM address space cannot read or write by high privileged software such as operating system and hypervisor although it contains the code/data of low privileged level. Inversely, the code/data in PRM doesn't have high privilege than the system software because the code in it cannot access system software address space. Moreover, processor checks and overrides the memory mapping decision taken by OS if processor finds an inconsistency with meta-data that is managed by the processor without the assistance of OS, which is called enclave page cache map (EPCM). EPCM contains expected virtual address used to access enclave page and access permissions i.e. read, write and execute of each enclave page.

Processor encrypts the pages in PRM when goes out of the CPU chip, which guarantees security against bus tapping attacks. The processor also measures signature before loading into cache to ensure the integrity of page. This captures the malicious write, read or replacing with other pages by OS but cannot restrict the OS form doing such malicious acts.

To achieve trusted storage SGX modifies the CPU components only in system resources, as shown in Figure 4. This is because every memory and I/O access requests transferred through the processor, so checks at CPU is sufficient for achieving security. In SGX the major changes are in three components: instruction decoder, page miss handler, and memory controller. Firstly, 18 new instructions are introduced in instruction decoder, which contains 5 user instructions that used by an application to initialize and build enclave and 13 supervisor instructions that are used by OS to manage enclave page table. Also, microcode related to memory access checks are added which are triggered by page miss handler [10]. Secondly, PMH hardware is modified to develop an ability to trigger the microcode assist for all address translations when a logical processor is in enclave mode, or when the physical address produced by the page walker machine matches the PRM range. Lastly, new register referred as processor reserved memory range registers (PRMRR) is introduced in the memory controller, which defines the size of PRM. Also, the memory controller is integrated with a memory encryption engine, which uses non-standard cryptographic primitives that consists of slightly modified AES operating mode [21] and a Carter-Wegman MAC construction [4] [24].

III. ARM TrustZone and Intel SGX Protection Against Vulnerabilities

Trusted execution environments aim is to ensure secure data to be stored and processed in an isolated, trusted environment. Trusted execution environments reduce the computing base so as to limit the links of security-critical application with the potentially malicious applications. Direct or indirect critical links between attacker and security-critical applications can be broadly categorized into three classes: Attacker makes link to security critical application through privileged software (Figure 7, referred to as Vulnerability-1), Attacker links to security critical application through micro-architectural events of hardware (Figure 8, referred to as Vulnerability-2), and

Attacker makes link to security critical application through directly probing hardware (Figure 9, referred to as Vulnerability-3).

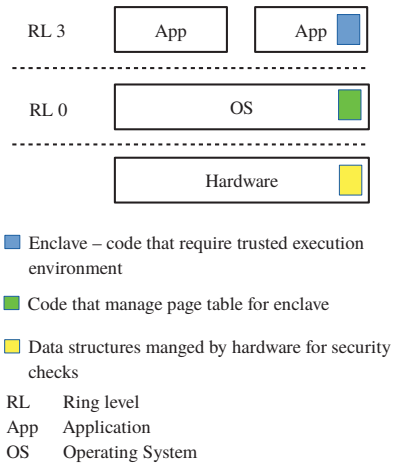

Fig. 5. Intel SGX software stack.

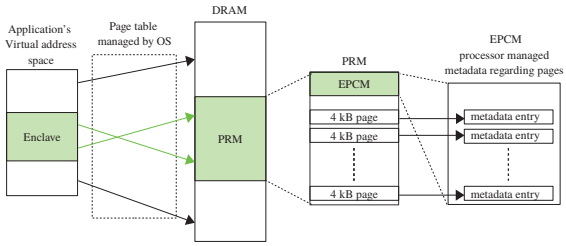

Fig. 6. Intel SGX trusted storage.

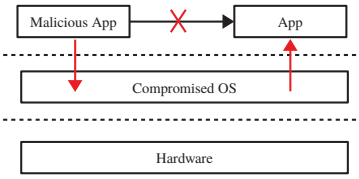

Fig. 7. Vulnerability 1: Malicious application makes link to other application's secure data through privileged software.

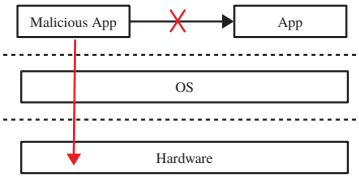

Fig. 8. Vulnerability 2: Malicious application makes link to other application's secure data through micro-architecture events of hardware

A. Vulnerability -1

The attacker uses system software privilege to make a link to secure data of an application. This privilege the attacker to modify the page tables and TLBs, unauthorized DMA transfer and Denial of service (DoS). The attacker cannot

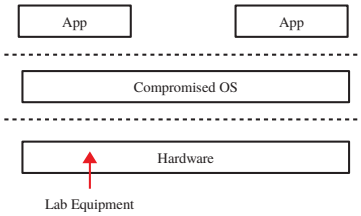

Fig. 9. Vulnerability 3: Malicious application makes link to other application's secure data by directly probing the hardware

able to modify page tables and TLBs in TrustZone and SGX. TrustZone page tables and TLBs are managed by secure world operating systems and stored in the secure memory region, which is not accessed by untrusted system software. Moreover, TrustZone has also divided the TLBs for secure and normal world. Intel SGX protects the modification of page tables by encrypting it and placing it in PRM, which cannot be accessed by system software. Intel SGX flushes TLB on exiting from enclave and applies security checks before storing address translation into TLB. These both reasons make the untrusted modification in TLB difficult. Direct memory accesses are also bounced back in TrustZone and SGX, which makes these architectures secure from DAM based attacks. DoS in SGX can easily be launched as compared to TrustZone because secure application page table and TLB management is the responsibility of untrusted operating system whereas in TrustZone trusted operating system manages the page tables and TLB.

B. Vulnerability-2

Intel SGX and ARM TrustZone architecture are vulnerable to cache-based side-channel attacks. The success of the cache-based side-channel attacks depends on the contention between the attacker and victim for getting cache lines.

Intel SGX is using the same cache architecture as in the non-secure architectures, which is vulnerable to cache-based side channel attacks. Schwarz et al. attacks the SGX enclave via cache side channels and demonstrates that the private key in the RSA implementation of mbedTLS can be extracted within five minutes. Other than RSA decryption, Ferdinand also demonstrates a more efficient attack on the human genome indexing via SGX cache-based information leakage.

In ARM TrustZone each cache lines are extended with a non-secure bit to distinguish the belonging to cache line to secure or non-secure world but these cache lines are allocated on the bases of memory access demand form the world at runtime and contend for the cache line, which makes the ARM TrustZone still vulnerable to cache-based. ARMageddon [18] uses Prime+Probe cache attack to extract the information from secure world to normal world and mentioned that Prime+Probe attack on the ARM TrustZone is not much different from a Prime+Probe attack on any application in the normal world. TruSpy [26] uses Prime+Probe to extract the key of AES encryption running in secure-world in 2.5 second form compromised OS of the normal world. Moreover, authors of [26] showed that the AES key can be extracted

TABLE I
TRUSTED COMPUTING BASE

Trusted Computing Base	ARM TrsutZone	Intel SGX
Software	Application Operating System Firmware	Application Page Table management part of OS
Hardware	System on Chip package	CPU chip package

TABLE II
RESILIENCE OF INTEL SGX AND ARM TRUSTZONE TO SET OF ATTACKS

Vulnerabilities		ARM TrustZone	Intel SGX
Vulnerability-1	Memory Mapping Passive Attacks	Secure because secure world's page-tables and TLBs,which is divided into two in TrustZone, are managed by trusted system software	Not secure because enclave's page-tables and TLBs are managed by untrusted system software
	Memory Mapping Active Attacks based on Page tables	Secure because secure world's page tables are managed by trusted system software	Secure because enclaves page tables are encrypted and detected if modified by untrusted system software
	Memory Mapping Active Attacks based on TLB	Secure because of TLB managed by trusted system software	Secure because of flushing mechanism of TLB
	DMA based Attacks	Secure because DMA accesses are rejected in secure world	Secure because DMA accesses are rejected in PRM
	Firmware based Attacks	Secure because firmware is a part of secure world	Secure because firmware is subject to TLB access checks
	Denial of Service Attack	Weakly secured because secure world's system software is separated from normal world's system software but monitor mode still has link with untrusted system software	Not secure because enclave is executing over untrusted system software
Vulnerability-2	Cache Based Side Channel Attacks	Not secure because of attacker can evict the desired cache line	Not secure because of attacker can evict the desired cache line
	Row-hammer Attack	Security depends on the developer if developer includes integrity check mechanism then it is safe	Secure because SGX can detect the unexpected modification of data because of integrity check mechanism
Vulnerability-3	Port Attacks	Safe as soon as designer disables debug ports	Safe because debug ports are usually disabled
	Bus Tapping Attacks	Safe as soon as secure data is limited to memory with in the SoC	Safe because CPU encrypts data when transferred on bus.
	Chip Attacks	Not secure	Not secure
	Power Analysis Attacks	Not secure	Not secure

in a couple of minutes without compromising the OS of the normal world. Recently, Prime+Count-based crossword covert channel is presented in [5], which can transfer information with bandwidth as high as 27 KB/s under the single-core scenario and 95 B/s under the cross-core scenario.

Row hammer attack flips a bit in DRAM by accessing the nearby row of the victim row in DRAM. This can modify the data in DRAM, which may lead to privilege change or corruption of secure data. SGX store hash of secure data with it in DRAM, so modification can easily be detected by measuring the hash again before using. TrustZone is safe from such attack if all secure data and code are in the memory of SoC. in case of external DRAM, the designer has to include software or hardware based hash mechanism for the integrity of secure data.

C. Vulnerability-3

The attacker directly probes computer hardware using external debuggers, chip imaging and electrical characteristic measuring equipment to extract secrets such as keys in e-fuses. debug ports are used in chip manufacturing process to evaluate the condition of a chip and these ports are disabled

before shipping. There is a debug port in Intel architecture called as Generic Debug eXternal Connection (GDXC), which collects data from CPU chip, and transfers it to an external debugger. These ports are not intended to be used by software developers, therefore, the Intel manual [8] [6] [7] [9] does not share details about the working of such debug ports that how GDXC interact with enclave. In ARMTrustZone specific platform Zynq-7000 SoC there is a general purpose port (GP port) that connect the ARM TrustZone SoC with custom build platform, which is on FPGA. [3] shows the vulnerabilities related to this port that NS bit can be overridden in custom build platform connected to ARMTrustZone SoC. TrustZone's and SGX'x threat model excludes chip imaging and power analysis attacks. Chip imaging attacks are dependent on the cost of attack because it requires costly equipment like ion-beam microscopy, especially in the case smaller feature size. However, the keys stored in efuse, which has a large feature size, can be vulnerable. Moreover, Power attacks cannot be addressed at the architectural level. It follows that defending against chip and power analysis attacks have a very high cost-to-benefit ratio.

IV. Conclusion

The main aim of developing TEE is to ensure sensitive data is stored and processed in an isolated, trusted environment. SGX and TrustZone are successful in providing trusted storage. However, both SGX and TrustZone are failed to ensure isolation while processing because of software-based side channel attacks. Mitigating software based side channel attacks using existing isolation features incurs a high-performance cost. There must be hardware based mechanism to mitigate software based side channel attacks because these attacks are targeting the microarchitectural events that are either transparent to software stack or coarse grain controlled from software stack. As side channel vulnerabilities exploit an inherent feature of the hardware resource. As Caches are added to reduce memory access latency, the attacker is exploiting this timing difference. So, it is difficult to eliminate the timing difference. To limit such vulnerability the hardware-assisted locking, partitioning or memory-to-cache mapping randomization based mechanism is needed.

References

[1] O. Bazhaniuk J. Loucaides A. Matrosov A. Furtak, Y. Bulygin and M. Gorobets. Bios and secure boot attacks uncovered. *The 10th ekoparty Security Conference*, 2014.

[2] Sebastian Anthony. Who actually develops linux? the answer might surprise you. http://www.extremetech.com/computing/175919-who-actually-develops-linux, 2014.

[3] E. M. Benhani, C. Marchand, A. Aubert, and L. Bossuet. On the security evaluation of the arm trustzone extension in a heterogeneous soc. In *2017 30th IEEE International System-on-Chip Conference (SOCC)*, pages 108–113, Sep. 2017.

[4] J. Lawrence Carter and Mark N. Wegman. Universal classes of hash functions (extended abstract). In *Proceedings of the Ninth Annual ACM Symposium on Theory of Computing*, STOC '77, pages 106–112, New York, NY, USA, 1977. ACM.

[5] Haehyun Cho, Penghui Zhang, Donguk Kim, Jinbum Park, Choong-Hoon Lee, Ziming Zhao, Adam Doupé, and Gail-Joon Ahn. Prime+count: Novel cross-world covert channels on arm trustzone. In *Proceedings of the 34th Annual Computer Security Applications Conference*, ACSAC '18, pages 441–452, New York, NY, USA, 2018. ACM.

[6] Intel Corporation. Software guard extensions programming reference. 2013.

[7] Intel Corporation. Intel r 64 and ia-32 architectures optimization reference manual. 2014.

[8] Intel Corporation. Intel r 64 and ia-32 architectures software developers manual. 2015.

[9] Intel Corporation. Intel r software guard extensions (intel r sgx). 2015.

[10] Victor Costan and Srinivas Devadas. Intel sgx explained. *IACR Cryptology ePrint Archive*, 2016:86, 2016.

[11] CVE Details. Xen : Vulnerability statistics. https://www.cvedetails.com/vendor/6276/XEN.html, 2018.

[12] J. Grand. Advanced hardware hacking techniques, 2004.

[13] AMD TATS BIOS Development Group. Amd security and server innovation. http://www.uefi.org/sites/default/files/resources/UEFI-PlugFest-AMD-Security-and-Server-innovation-AMD-March-2013.pdf, 2013.

[14] Matthew Hoekstra, Reshma Lal, Pradeep Pappachan, Vinay Phegade, and Juan Del Cuvillo. Using innovative instructions to create trustworthy software solutions. In *Proceedings of the 2Nd International Workshop on Hardware and Architectural Support for Security and Privacy*, HASP '13, pages 11:1–11:1, New York, NY, USA, 2013. ACM.

[15] D. Kaplan J. Powell and T. Woller. Amd memory encryption, white pape. http://amd-dev.wpengine.netdna-cdn.com/wordpress/media/2013/12/AMD-Memory-Encryption-Whitepaper$_v7 - Public.pdf$, 2016.

[16] Simon P Johnson, Uday R Savagaonkar, Vincent R Scarlata, Francis X McKeen, and Carlos V Rozas. Technique for supporting multiple secure enclaves, 2015. US Patent 8,972,746.

[17] ARM Limited. Arm security technology building a secure system using trustzone technology. *IACR Cryptology ePrint Archive*, 2019.

[18] Moritz Lipp, Daniel Gruss, Raphael Spreitzer, Clémentine Maurice, and Stefan Mangard. Armageddon: Cache attacks on mobile devices. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 549–564, Austin, TX, 2016. USENIX Association.

[19] Francis X McKeen, Carlos V Rozas, Uday R Savagaonkar, Simon P Johnson, Vincent Scarlata, Michael A Goldsmith, Ernie Brickell, Jiang Tao Li, Howard C Herbert, Prashant Dewan, et al. Method and apparatus to provide secure application execution, 2015. US Patent 9,087,200.

[20] Frank McKeen, Ilya Alexandrovich, Alex Berenzon, Carlos V. Rozas, Hisham Shafi, Vedvyas Shanbhogue, and Uday R. Savagaonkar. Innovative instructions and software model for isolated execution. In *Proceedings of the 2Nd International Workshop on Hardware and Architectural Support for Security and Privacy*, HASP '13, pages 10:1–10:1, New York, NY, USA, 2013. ACM.

[21] Mahesh S Natu, Sham Datta, Jeff Wiedemeier, James R Vash, Sailesh Kottapalli, Scott P Bobholz, and Allen Baum. Supporting advanced ras features in a secured computing system, October 30 2012. US Patent 8,301,907.

[22] X. Ruan. Platform embedded security technology revealed: Safeguarding the future of computing with intel embedded security and management engine, 2014.

[23] A. Tereshkin and R. Wojtczuk. Introducing ring-3 rootkits. Master's Thesis, 2010.

[24] Mark N Wegman and J Lawrence Carter. New hash functions and their use in authentication and set equality. *Journal of computer and system sciences*, 22(3):265–279, 1981.

[25] R. Wojtczuk and A. Tereshkin. Attacking intel bios. *Invisible Things Lab*, 2010.

[26] Ning Zhang, Kun Sun, Deborah Shands, Wenjing Lou, and Yiwei Thomas Hou. Truspy: Cache side-channel information leakage from the secure world on arm devices. *IACR Cryptology ePrint Archive*, 2016:980, 2016.

[27] Xiantao Zhang and Yaozu Dong. Optimizing xen vmm based on intel virtualization technology. *2008 International Conference on Internet Computing in Science and Engineering*, pages 367–374, 2008.