Position Paper: Challenges Towards Securing Hardware-assisted Execution Environments

Zhenyu Ning [1] **Fengwei Zhang** [1] Weisong Shi [1] Larry Shi [2]

[1]Wayne State University

[2]University of Houston

November 21, 2019

Overview of The Talk

- Motivation
- TEE Background
- Research Challenges
- Conclusion

Overview of The Talk

- Motivation
- TEE Background
- Research Challenges
- Conclusion

Motivation

Hardware-assited Trusted Execution Environments (TEEs) have been widely adopted in commodity systems
Examples of TEE include but not limited to:

- Intel software guard extensions (SGX) [1, 2, 3]
- AMD memory encryption technologies [4, 5, 6]
- ARM TrustZone [7]
- x86 system management mode (SMM) [8]
- Intel management engine (ME) [9]
- AMD platform security processor (PSP) [10]

Motivation

Due to the nice features provided by the TEEs, the code base in a TEE is getting large

- OS running in TrustZone [11]
- A hypervisor is deployed in SMM [12]
- Linux containers running in SGX [13]

The complexity of the software introduces vulnerabilities, and "Trusted" execution environments are not trustworthy

Motivation: Problem Statement

Trusted Execution Environments have been introduced in different platforms for securing software execution, but achieving security not only depends on technologies of execution environments themselves (e.g., small TCB, strong isolation), but also relies on the **executed code**.

State-of-the-art trusted execution environments research lacks:

- Frameworks to verify the executed code
- Defenses within the trusted environments
- Methods detecting compromised TEEs
- Mitigation plans for TEE-attack responses

Overview of The Talk

- Motivation
- TEE Background
- Research Challenges
- Conclusion

Background

We category TEEs into three types:

- Ring 3 TEEs implemented via memory encryption
  - Intel SGX
  - AMD memory encryption technologies
- Ring -2 TEEs implemented via memory restriction:
  - x86 SMM
  - ARM TrustZone
- Ring -3 TEEs implemented via co-processors:
  - Intel ME
  - AMD PSP

Background: Ring 3 TEEs via Memory Encryption

Intel SGX

- Was introduced at HASP 2013
- A set of instructions and mechanisms for memory accesses

AMD memory encryption technologies

- Was introduced at ISCA 2016
- Two new features: Secure Memory Encryption (SME) and Secure Encrypted Virtualization (SEV)

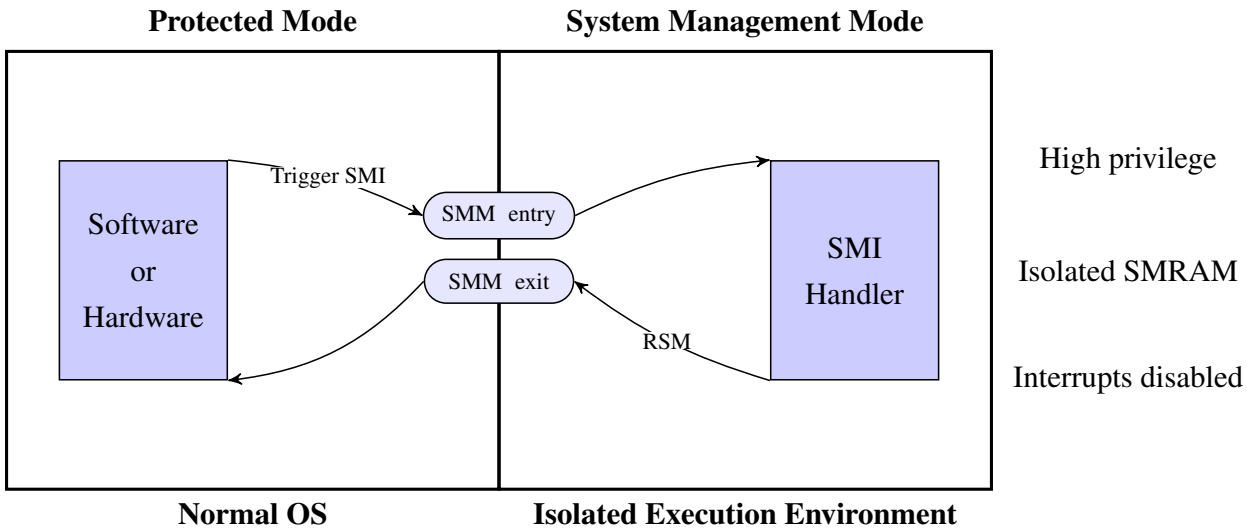
Intel SGX v.s. AMD SME/SEV

- SGX is at user-level but SEV can encrypt VMs
- Encryption performance comparison

Background: Ring -2 TEEs via Memory Restriction

x86 System Management Mode

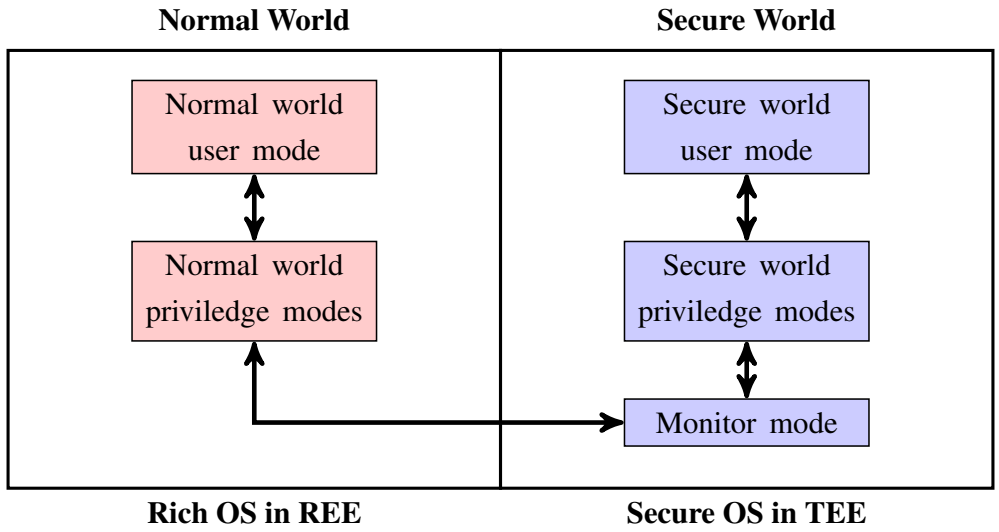
- A special CPU on x86
- Isolated System Management RAM (SMRAM) that is inaccessible from OS
- Only way to enter SMM is to trigger a System Management Interrupt (SMI)
- Executing `RSM` instruction to resume OS (Protected Mode)



Background: Ring -2 TEEs via Memory Restriction

ARM TrustZone Technology

- Similar to SMM on x86
- Two worlds: secure mode and normal mode
- Secure Monitor Call (SMC) instruction or triggering secure interrupts to switch to secure mode



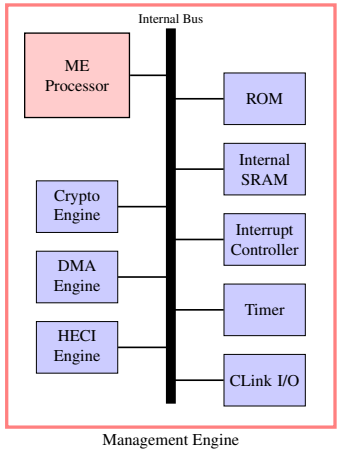
Background: Ring -3 TEEs via Co-Processors

Intel Management Engine

- Micro-computer embedded inside of all recent Intel processors
- First ME application: Active Management Technology (AMT)
- Others including enhanced privacy identification, protected audio video path, identity protection technology, and boot guard.

AMD Platform Security Processor

- Similar technology to Intel ME
- A co-processor on AMD CPUs



Overview of The Talk

- Motivation
- TEE Background
- Research Challenges
- Conclusion

Research Challenges

This paper considers the following Research Challenges (RCs) for further securing trusted execution environments

- Hunting Bugs in TEEs code
- Protecting Mechanisms in the TEEs
- Detecting a Compromised TEE
- Patching and Rejuvenation of TEEs

RC1: Hunting Bugs in TEEs code

- The software running in a TEE can contain text-book vulnerabilities such as buffer overflows since this software can be written by careless programmers or third-party developers
  - "Millions of BIOSes" are easy to be compromised because the known vulnerabilities of SMM code are never patched in the BIOS [14]
  - Vulnerabilities that can be used to execute arbitrarily code in TrustZone [15]
  - A series of SGX-based applications have been developed such as SCONE [13], Haven [16], VC3 [17], etc.
- Existing solutions of bug hunting can not be applied directly because the TEEs code requires particular environments
- Open source code v.s. binary images (SMM, TrustZone, and ME)

RC2: Protecting Mechanisms in TEEs

RC1 failed. It is impractical to have perfect code running in the TEEs, and the attackers will eventually found a vulnerability and exploit it at some point; however, existing TEEs lack of defense mechanisms in the execution environments

- The code running in SMM shares a single address space and paging is disabled
- Applications running in the SGX enclaves do not have protection mechanisms such as ASLR
- Defense mechanisms such as non-executable stack, data execution prevention, and address space randomization are missing in TEEs

Since we consider these environments are more secure than the normal OS, these basic system defense mechanisms are needed for securing the environment

RC3: Detecting a Compromised TEE

RC1&2 failed. A TEE has been compromised due to the vulnerabilities in the code. How to detect a compromised TEE? TEEs run at a high-privilege memory space that inaccessible from the system software or use encrypted memory that their contents are mysterious without the key

- Intel SGX encrypts its code and data in enclaves
- SMM and TrustZone code is not accessible by the system software (e.g., OS)

Because of these "security protection" features, a TEE can achieve a strong security guarantee. However, after compromising a TEE, attackers can implement undetectable advanced rootkits in it

- SMM keyloggers [18] and SMM rootkits developed by NSA [19]
- Ring -3 rootkits [20]

RC4: Patching and Rejuvenation of TEEs

RC1&2 failed and RC3 succeed. We know that a TEE has been compromised. How to mitigate attackers from a compromised TEE and patch it to a good state?

- Using System software to update the compromised TEE. How about ring -2 TEEs (SMM and TrustZone) and ring -3 TEEs (Intel ME)?
- If the TEE is compromised, it is likely that the system software is malicious as well
- Using a TEE to update another TEE (Intel ME to patch BIOS/SMM)?

Ensuring the restore process is secure, we need a Trust Base

Overview of The Talk

- Motivation
- TEE Background
- Research Challenges
- Conclusion

Conclusion

Existing trusted execution environments focus on removing trust from system software and achieving strong isolation by leveraging hardware support. We present four research challenges towards securing TEEs

- Hunting Bugs in TEEs code
- Protecting Mechanisms in the TEEs
- Detecting a Compromised TEE
- Patching and Rejuvenation of TEEs

The goal of this position paper is to draw the attention to the system security community about the challenges for achieving a more secure execution environment

References I

[1] F. Mckeen, I. Alexandrovich, A. Berenzon, C. Rozas, H. Shafi, V. Shanbhogue, and U. Savagaonkar, "Innovative Instructions and Software Model for Isolated Execution," in *Proceedings of the 2nd Workshop on Hardware and Architectural Support for Security and Privacy (HASP'13)*, 2013.

[2] I. Anati, S. Gueron, S. P. Johnson, and V. R. Scarlata, "Innovative Technology for CPU Based Attestation and Sealing," in *Proceedings of the 2nd Workshop on Hardware and Architectural Support for Security and Privacy (HASP'13)*, 2013.

[3] M. Hoekstra, R. Lal, P. Pappachan, C. Rozas, V. Phegade, and J. del Cuvillo, "Using Innovative Instructions to Create Trustworthy Software Solutions," in *Proceedings of the 2nd Workshop on Hardware and Architectural Support for Security and Privacy (HASP'13)*, 2013.

[4] J. P. David Kaplan and T. Woller, "AMD Memory Encryption, White Paper," http://amd-dev.wpengine.netdna-cdn.com/wordpress/media/2013/12/AMD_Memory_Encryption_Whitepaper_v7-Public.pdf, April 2016.

[5] D. Kaplan, T. Woller, and J. Powell, "AMD Memory Encryption Tutorial, ISCA 2016," https://sites.google.com/site/metisca2016/, 2016.

[6] D. Kaplan, "AMD x86 Memory Encryption Technologies, USENIX Security Tutorial 2016," https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/kaplan, 2016.

[7] ARM, "ARM Security Technology - Building a Secure System using TrustZone Technology," http://infocenter.arm.com/help/topic/com.arm.doc.prd29-genc-009492c/PRD29-GENC-009492C_trustzone_security_whitepaper.pdf, 2009.

[8] Intel, "64 and IA-32 Architectures Software Developer's Manual: Chapter 34," 2014.

[9] X. Ruan, *Platform Embedded Security Technology Revealed: Safeguarding the Future of Computing with Intel Embedded Security and Management Engine*. Apress, 2014.

[10] 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.

References II

[11] ARM, "ARM Trusted Firmware," https://github.com/ARM-software/arm-trusted-firmware, 2016.

[12] A. M. Azab, P. Ning, and X. Zhang, "SICE: A Hardware-level Strongly Isolated Computing Environment for x86 Multi-core Platforms," in *Proceedings of the 18th ACM Conference on Computer and Communications Security (CCS'11)*, 2011.

[13] S. Arnautov, B. Trach, F. Gregor, T. Knauth, A. Martin, C. Priebe, J. Lind, D. Muthukumaran, D. O'Keeffe, M. L. Stillwell, D. Goltzsche, D. Eyers, R. Kapitza, P. Pietzuch, and C. Fetzer, "SCONE: Secure Linux Containers with Intel SGX," in *Proceedings of The 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI'16)*, 2016.

[14] C. Kallenberg and X. Kovah, "How Many Million BIOSes Would you Like to Infect?" http://conference.hitb.org/hitbsecconf2015ams/wp-content/uploads/2015/02/D1T1-Xeno-Kovah-and-Corey-Kallenberg-How-Many-Million-BIOSes-Would-You-Like-to-Infect.pdf, 2015.

[15] D. Shen, "Exploiting Trustzone on Android," Black Hat USA Briefings, https://www.blackhat.com/docs/us-15/materials/us-15-Shen-Attacking-Your-Trusted-Core-Exploiting-Trustzone-On-Android-wp.pdf.

[16] A. Baumann, M. Peinado, and G. Hunt, "Shielding Applications from an Untrusted Cloud with Haven," in *Proceedings of the 11th USENIX Symposium on Operating Systems Design and Implementation (OSDI'14)*, 2014.

[17] F. Schuster, M. Costa, C. Fournet, C. Gkantsidis, M. Peinado, G. Mainar-Ruiz, and M. Russinovich, "VC3: Trustworthy Data Analytics in the Cloud," in *Proceedings of the 36th IEEE Symposium on Security and Privacy (S&P'15)*, 2015.

[18] S. Embleton, S. Sparks, and C. Zou, "SMM rootkits: A New Breed of OS Independent Malware," in *Proceedings of the 4th International Conference on Security and Privacy in Communication Networks (SecureComm'08)*, 2008.

[19] "NSA's ANT Division Catalog of Exploits for Nearly Every Major Software/Hardware/Firmware," http://Leaksource.wordpress.com, 2014.

References III

[20] A. Tereshkin and R. Wojtczuk, "Introducing Ring -3 Rootkits," http://invisiblethingslab.com/itl/Resources.html, 2009.

Thank you!

Email: fengwei@wayne.edu
Homepage: `http://fengwei.me`

Questions?