Kick-off: Trusted Execution Environments

Chair for IT Security / I20
Prof. Dr. Claudia Eckert
Technical University of Munich

Christian Epple
christian.epple@aisec.fraunhofer.de

Hendrik Meyer zum Felde
hendrik.meyerzumfelde@aisec.fraunhofer.de

13 07 2020

Outline

1. Organization
2. Requirements
3. Grading
4. Time Table
5. Examples of TEEs
6. Topics
7. Introduction to Scientific Writing
8. Next Steps

Organization

The seminar will be organized as a scientific conference:

1. Familiarization phase (2 Weeks)
2. Writing phase (7 Weeks)
3. Review phase (2 Weeks)
4. Improvement phase (8 Weeks)
5. Talk preparation (1 Week)
6. Talk and Discussion

Requirements

- Report
  - Written report in the form of a scientific paper
  - Mandatory length of 8 pages (references don't count)
  - Usage of LaTeX is mandatory
  - Formatting with the provided LaTeX-Style (IEEE 2-column)
- Review
  - Every Student creates two anonymous reviews
  - Review template will be provided
  - Approximately 1/2 page
  - Every Student writes a rebuttal
- Presentation
  - Presentation with slides
  - 30 minutes presentation
  - 15 minutes discussion

Grading

Grading considers all contributions to this seminar:

1. Report (50%)
   - Contents, Accuracy, Style, Effort, Grasp
2. Presentation (30%)
   - Slides, Execution, Contents, Understandability
3. Reviews (15%)
   - Written Reviews and Rebuttal
4. Participation and discussion (5%)

Time Table

13.07.20	Kick-off meeting (today) *Register for this seminar from 16.07. until 21.07.2020. TUM-link*
30.07.2020	Start of topic assignments (once matching is finished in TUMonline)
11.10.2020	Submit your first version (outline finished, 80% of content)
15.10.2020	**Meeting**: Joint intermediate review and discussion
29.11.2020	Submit your paper
01.12.2020	Receive papers for review
10.12.2020	Submit your reviews
20.12.2020	Submit your rebuttal + “camera-ready” version + presentation
13.+14.01.2021	**Meeting**: Presentations and discussion

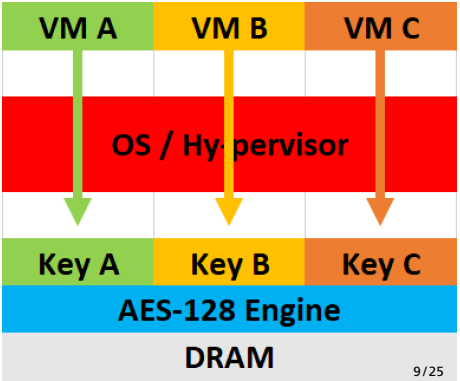
Before we go on....

... any questions so far?

- A Trusted Execution Environment (TEE) is an isolated environment which aims to protect executions within against high privileged adversaries.
- Software TEEs solely rely on software mechanisms for protection, while hardware TEEs use additional hardware mechanisms to protect the confidentiality and integrity of code and data within the environment.
- Most famous ones are
  - Arm TrustZone
  - AMD SEV and
  - Intel SGX

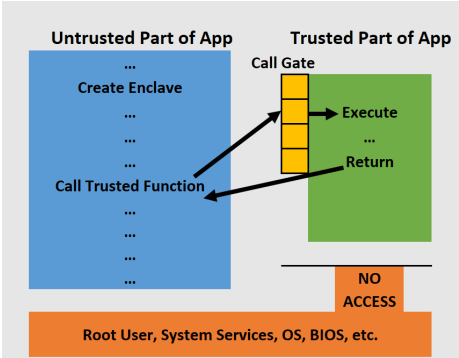
AMD SEV's memory encryption

- AMD Secure Memory Encryption (SME) allows to encrypt memory content before writing in to RAM
- Prevents an attacker from physical RAM reading attacks
- AMD Secure Encrypted Virtualization (SEV) is based on SME uses a different key for each virtual machine.
- This prevents a malicious hypervisor from reading a VM's memory content.
- Moving ciphertext between memory location is prevented by encryption of physical memory address
- Problem caused: swapping or VM migration not possible



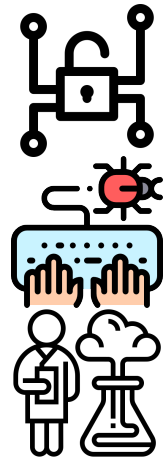
Intel SGX Enclaves

- Shipped within every Intel Skylake CPU of the 6th generation or newer
- Allows code parts to be executed in hardware separated enclave
- Limit 128-256MB RAM simultaneous usage of SGX enclaves
- Possible to attest that code is running inside an SGX enclave
  - TLS session can be terminated directly in enclave
  - Typical monitoring capabilities
  - Intense bug fixing history
  - Enclave guarantee's correct code execution despite malicious OS



Topics

- Trusted Execution Environments in general
  - Survey on the Effect of Plundervolt on TEEs
  - Comparison of existing Trusted Execution Environments (AMD SEV, ARM TrustZone, Intel SGX, ...)
  - Survey on SDKs for Trusted Execution Environments
  - Intel (MK)TME as competitor to AMD SME/SEV
  - Keystone Enclaves on Risc-V
- Intel Software Guard Extensions (SGX)
  - Limitations and Possibilities of Intel SGX Enclaves and TPM interaction
  - Exploiting Speculative Execution
  - Emulating TPMs Using Secure Enclave Technologies
- AMD Secure Encrypted Virtualization (SEV)
  - Exploiting AMD SEV's missing integrity protection
  - Attacking AMD SEV's memory encryption



Survey on the Effect of Plundervolt on TEEs

- **Intro**: A new attack scenario was recently proposed called Plundervolt, which allows breaking the security features of some Trusted Execution Environments (TEE). The attack vector uses a decrease of the voltage of a computer system in a precise way to allow manageable bit flips in certain logic components which allows breaking the security. This ultimately results in variable or pointer manipulation in order to break, for instance, the confidentiality of the TEE.
- **Goal:**
  - Understand the basic concept of Plundervolting
  - Understand the basic concepts of the affected TEEs (Intel SGX technology, AMD-SEV, ARM Trustzone etc.)
  - Gather an overview and analysis of work-arounds and solution attempts to the issue
- **References:**
  - General information on Plundervolt plundervolt.com
  - Costan V. and Devadas, S. Intel SGX explained (link)

Comparison of existing Trusted Execution Environments

- **Intro**: Multiple hardware TEEs have been proposed, the most famous and practically working ones are Arm TrustZone, AMD SEV and Intel SGX. Additional TEEs are Intel TXT, IBM Secure Execution, MultiZone Security Trusted Execution Environment, Open-TEE, OPTEE, Trusty TEE, SierraTEE, etc.
- **Goal:**
  - Understand ARM TrustZone + AMD SEV + Intel SGX + additional hardware TEEs
  - Analyse their differences and compare:
    - The root of trust
    - The isolation techniques
    - TEE privileges
    - What can be executed inside the TEEs - use cases
    - The attack model that has been dealt with
    - Which standards are implemented
- **References:**
  - Costan V. and Devadas, S. Intel SGX explained (link)
  - David Kaplan, Jeremy Powell, Tom Woller: AMD Memory Encryption (link)
  - Sandro Pinto and Nuno, Santos: Demystifying Arm TrustZone: A Comprehensive Survey

SDKs for Trusted Execution Environments

- **Intro**: A Trusted Execution Environment (TEE) is an isolated environment which aims to protect executions within against high privileged adversaries. Interestingly, multiple frameworks exists that claim to work with different types of TEEs.
- **Goal:**
  - Understand AMD SEV
  - Understand Intel SGX
  - Understand Google Asylo
  - Research other existing TEE frameworks
  - Research uniform capabilities and interfaces for SEV & SGX
- **References:**
  - Google Asylo: (link)
  - Costan V. and Devadas, S. Intel SGX explained (link)
  - David Kaplan, Jeremy Powell, Tom Woller: AMD Memory Encryption (link)

Intel (MK)TME as competitor to AMD SME/SEV

- **Intro**: Intel plans to follow the lead of AMD with a technology which encrypts the complete RAM of a computer system. The concept is called Total Memory Encryption (TME). Another concept takes it one step further and allows the decryption of the memory using multiple sets of keys, called Multi-Key Total Memory Encryption (MKTME).
- **Goal:**
  - Understand Intel's Multi-Key-Total-Memory-Encryption
  - Understand AMD SME/SEV
  - Research the state-of-the-art of the technologies
  - Compare the technologies
- **References:**
  - Intel MK-TME spec link
  - David Kaplan, Jeremy Powell, Tom Woller: AMD Memory Encryption (link)
  - WikiChip link

Keystone Enclaves on Risc-V

- **Intro**: The Keystone framework provides an open source construction environment for TEEs which are based on the RISC-V architecture. The project has three main goals: (1) provide transparency for TEEs (2) keep the trusted computing base low and (3) be highly optimized for specific applications.
- **Goal:**
  - Understand Keystone
  - Understand Risc-V
  - Analysis of the security architecture of Keystone Enclaves for different threat models
  - Systematic Analysis of potential attack surface in comparison with other TEE architectures
- **References:**
  - Keystone, an open framework for architecting TEEs https://keystone-enclave.org/
  - D. Lee et al., Keystone: an Open Framework for Architecting Trusted Execution Environments, EuroSys'20
  - Jo Van Bulck, David Oswald, Eduard Marin, et.al.: A Tale of Two Worlds: Assessing the Vulnerability of Enclave Shielding Runtimes.
  - (to compare to TZ) Cerdeira, David et al. SoK: Understanding the Prevailing Security Vulnerabilities in TrustZone-Assisted TEE Systems.

Analysis: Intel SGX Enclaves and TPM interaction

- **Intro:** Intel SGX is a TEE. However, relying on one piece of technology is often a major disadvantage. Attempts were made to combine the benefits of a Trusted Platform Module (TPM) with the benefits of SGX-enclaves. Fundamental design limitations such as security and feature restrictions make the interaction of the two technologies complicated.
- **Goal:**
  - Understand Intel SGX
  - Understand Trusted Platform Modules
  - Reserach the limitation of current interaction between TPM modules and SGX enclaves
  - Gather an overview of research which has attempted a combination of both
  - Sum up advantages and disadvantages
- **References:**
  - Costan V. and Devadas, S. Intel SGX explained (link)
  - Paul Georg Wagner, Pascal Birnstill, Jürgen Beyerer: Distributed Usage Control Enforcement through Trusted Platform Modules and SGX Enclaves link
  - TPM Wikipedia link

Exploiting Speculative Execution

- **Goal:** Meltdown and Spectre are two classes of vulnerabilities which are based on speculative execution in mostly Intel CPUs. The vulnerabilities are capable of reading protected memory, potentially including memory protected by a Trusted Execution Environment.
- **Goal:**
  - Understand Spectre + Meltdown
  - Research speculative execution attacks on a TEE of your choice (for instance Intel SGX, ARM TrustZone)
  - Survey the solution attempts available (or even propose your own)
- **References:**
  - Spectre Attacks: Exploiting Speculative Execution link
  - Meltdown: Reading Kernel Memory from User Space link
  - (for interest in ARM TrustZone) Le Guan, Peng Liu, Xinyu Xing, Xinyang Ge, Shengzhi Zhang, Meng Yu, Trent Jaeger: TrustShadow: Secure Execution of Unmodified Applications with ARM TrustZone
  - (for interest in Intel SGX) Jo Van Bulck, Marina Minkin, Ofir Weisse, Daniel Genkin, Baris Kasikci, Frank Piessens, Mark Silberstein, Thomas F. Wenisch, Yuval Yarom, and Raoul Strackx: FORESHADOW: Extracting the Keys to the Intel SGX Kingdom with Transient Out-of-Order Execution

Emulating TPMs Using Secure Enclave Technologies

- **Intro:** Secure enclaves are a subcategory of TEEs. Trusted Platform Modules (TPMs) and Secure Enclave Technologies have some but not all capabilities in common. The question is at hand, whether Secure Enclave Technologies can completely emulate the behaviour of a TPM. An obvious difference appears during the booting process of a system. A TPM can store hash-values of system configuration during boot up and runtime, whereas Enclave Environments like Intel SGX allow to attest the enclave's state at runtime.
- **Goal:**
  - Understand the concept of TPMs, TEEs and Secure Enclaves
  - Research approaches for a TPM emulation in Trusted Execution Environments
  - Find and analyse solutions to the issue of using a TPM emulated by a Secure Enclave during booting phase of a system.
- **References:**
  - Stefan Berger, Ramón Cáceres, Kenneth A. Goldman et al: vTPM: Virtualizing the Trusted Platform Module
  - Himanshu Raj, Stefan Saroiu et al: fTPM: A Software-Only Implementation of a TPM Chip
  - Dave (Jing) Tian, Joseph I. Choi et al: A Practical Intel SGX Setting for Linux Containers in the Cloud

Exploiting AMD SEV's missing integrity protection

- **Intro:** AMD Secure Memory Encryption (SME) allows to encrypt memory content before writing in to RAM. This prevents an attacker from executing physical attacks in which RAM content is extracted via DMA. AMD Secure Encrypted Virtualization (SEV) is based on SME, but uses a different key for each virtual machine. This prevents a malicious hypervisor from reading a VM's memory content. However, multiple attacks have been published that circumvent SEV and allow a malicious hypervisor to read a VM's memory content. Most of those attacks make use of the lacking integrity protection of AMD SEV.
- **Goal:**
  - Understand AMD SEV + remapping attacks
  - Understand cipherblock moving attacks
  - Research the SOTA on integrity and replay protection mechanisms for AMD SEV
- **References:**
  - David Kaplan, Jeremy Powell, Tom Woller: AMD Memory Encryption link
  - Hetzelt and Buhren. 2017. Security Analysis of Encrypted Virtual Machines.
  - Morbitzer, Huber et al. SEVered: Subverting AMD's virtual machine encryption.
  - Li, Zhang et al: Exploiting Unprotected I/O Operations in AMD's Secure Encrypted Virtualization.

Attacking AMD SEV's memory encryption

- **Intro:** AMD Secure Memory Encryption (SME) allows to encrypt memory content before writing it to RAM. This prevents an attacker from executing physical attacks in which RAM content is extracted via DMA. AMD Secure Encrypted Virtualization (SEV) is based on SME but uses a different key for each virtual machine. This prevents a malicious hypervisor from reading a VM's memory content. To prevent an attacker from moving ciphertext between memory locations, the physical memory address is included in the encryption. However, this also disables functionality that requires the hypervisor to move a VM's memory content in memory, such as swapping, or VM migration.
- **Goal:**
  - Understand AMD SEV + cipherblock moving attacks.
  - Research the SOTA on solutions (or propose your own solutions) which allow the hypervisor to move cipherblocks in AMD SEV's encryption mechanism if required.
- **References:**
  - David Kaplan, Jeremy Powell, Tom Woller: AMD Memory Encryption (link)
  - Mengyuan Li, Yinqian Zhang, Zhiqiang Lin and Yan Solihin: Exploiting Unprotected I/O Operations in AMD's Secure Encrypted Virtualization. In Proceedings of the 28th USENIX Security Symposium.

Topic assignment

After matching phase:

- We'll ask you to send your 3 top choices via email
- You may add a letter of motivation to emphasize your top choice
- We'll assign topics to students with your input
- You'll receive information and initial literature from your supervisor

Introduction to Scientific Writing

Introduction to Scientific Writing

- ??.09.2020 (date, time and location will be announced on website)
- Crash course in scientific writing
- Not mandatory
- You may need scientific writing for
  - this seminar, obviously
  - BA or MA thesis
  - any scientific paper writing

Next Steps

- Matching and Topic assignment
  - Matching concludes 30.07.2020. After that we'll get in touch with the participants
  - If you want to deregister
    - do so until 06.08.2020 without penalty
    - or brace yourself for a 5,0
  - Participants send top 3 topics via email, we'll assign the topics
- Familiarization phase
  - Literature research
  - Get an overview of your topic
  - Create report structure
- Introduction to Scientific Writing (??.09.2020)
- Writing phase

Q&A

Q&A ?