Towards TEEs with Large Secure Memory and Integrity Protection Against HW Attacks

Pierre-Louis Aublin  
IIJ Innovation Institute  
Tokyo, Japan

Mohammad Mahhouk  
TU Braunschweig  
Braunschweig, Germany

Rüdiger Kapitza  
TU Braunschweig  
Braunschweig, Germany

ABSTRACT

Providing integrity protection against physical attacks on a large memory region is a difficult problem for Trusted Execution Environment (TEE) designers. This is due to the space and computational cost to maintain the memory integrity tree.

Instead of trying to optimize the integrity tree, we propose a novel approach that consists of combining two classes of TEEs: one with a large secure memory but no integrity protection, and another one with integrity protection but a small secure memory. We briefly describe several use-cases, challenges, and the implementation of a proof-of-concept over Intel SGX.

CCS CONCEPTS

• Security and privacy → Distributed systems security.

KEYWORDS

Trusted Execution Environment, Intel SGX, Integrity

ACM Reference Format:


1 INTRODUCTION

Trusted Execution Environments (TEE) such as Intel SGX [5] or ARM TrustZone [12] give security guarantees to applications running in an untrusted environment where attackers have privileged access to the software stack (including the OS) and/or the hardware.

Unfortunately, providing integrity guarantees against physical attacks over large amounts of memory is prohibitive, which limits the applicability of TEEs. For example, Intel dropped the integrity protection in favor of a large secure memory in its latest SGX hardware [6].

In this research statement, we explore how one can combine different TEE technologies that offer complementary protection. This approach poses different challenges: data partitioning granularity, integrity scheme, communication between the two TEEs, and attestation.

We detail a proof-of-concept based on Intel SGX hardware that combines a Scalable SGX-capable processor [6] (for its large secure memory area) with the PCIe Intel VCA2 card [3] which embeds three Client SGX processors (each providing integrity guarantees).

<table>
<thead>
<tr>
<th>TEE</th>
<th>Arch.</th>
<th>Conf.</th>
<th>Int.</th>
<th>Secure mem. size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client SGX</td>
<td>Intel</td>
<td>✓</td>
<td>✓</td>
<td>256 MB</td>
</tr>
<tr>
<td>Scalable SGX</td>
<td>Intel</td>
<td>✓</td>
<td>X</td>
<td>512 GB</td>
</tr>
<tr>
<td>SEV-SNP</td>
<td>AMD</td>
<td>✓</td>
<td>X</td>
<td>All mem.</td>
</tr>
<tr>
<td>TrustZone</td>
<td>ARM</td>
<td>✓</td>
<td>X</td>
<td>All mem.</td>
</tr>
<tr>
<td>Keystone</td>
<td>RISC-V</td>
<td>✓</td>
<td>on-chip</td>
<td>All mem.</td>
</tr>
<tr>
<td>PENGLAI</td>
<td>RISC-V</td>
<td>✓</td>
<td>FPGA</td>
<td>512 GB (*)</td>
</tr>
</tbody>
</table>

Table 1: Comparison of various TEE implementations in terms of Architecture, protection against hardware attacks targeting Confidentiality or Integrity, and Secure memory size. (*)PENGLAI has been evaluated with only up to 600 MB of secure memory.

2 TEES AND PROBLEM STATEMENT

2.1 Trusted Execution Environments

Trusted Execution Environments (TEEs) are secure areas of a processor that provide integrity and confidentiality guarantees on the code and data placed inside even in the presence of a powerful attacker who has privileged access to the software stack (including the OS) and/or the hardware (excluding the processor package).

Over the past years, different TEE implementations has been proposed, both from industry: Intel SGX [5, 6], AMD SEV-SNP [16], ARM TrustZone [12]; as well as from academia: Keystone [9] or PENGLAI [4].

While each implementation has its specificities, they all provide a secure execution mode commonly called an enclave. The enclave can access a specific memory area whose size depends on the TEE implementation, from hundreds of MBs up to the entire system memory. To provide confidentiality, the hardware either transparently encrypts memory when it leaves the CPU die, or carefully manages memory accesses. To provide integrity guarantees, TEEs maintain an integrity tree.

TEEs provide an attestation mechanism to prove to a third-party the authenticity of the secure component and the hardware on which it runs [7, 9, 16]. Attestation can be either local, to attest several instances of the same TEE between each others, or remote, to give remote clients the guarantee that they are communicating with a secure service.

2.2 Problem Statement

As detailed in Tab. 1, the existing TEEs exhibit different security guarantees and secure memory size. Among the commercially available TEEs, Client SGX is the only one that provides integrity guarantees against a physical attacker, but at the cost of a small secure memory: at most 256 MB shared between all
the enclaves of the system. Intel Scalable SGX, AMD SEV-SNP and ARM TrustZone trade protection against physical attacks for a larger secure memory area. Among research-oriented TEEs, Keystone defers integrity protection to a special hardware chip that has not been implemented yet; while PENGLAI protects a larger memory (up to 512 GB in theory), but requires CPU changes and has been implemented and evaluated on an FPGA with only 600 MB of secure memory.

Without protection against hardware attacks, an attacker can easily mount a replay attack [18], replacing memory content (data and/or code) with a previous version, without being detected. As an example, an attacker could revert a security hot-patch to bring back a vulnerability that will allow him to extract enclave secrets [21].

Providing integrity guarantees on a large secure memory is difficult: the size of the integrity tree grows linearly with the amount of memory to protect. Given that it needs to be stored in a memory area protected from physical attacks, increasing its size is prohibitive. While several systems have tried to optimize the integrity tree [4, 18], its poor scalability remains a fundamental problem.

3 PROPOSED APPROACH

In this research statement, we are exploring an alternative approach to providing both integrity guarantees against a physical attacker as well as a large secure memory. The basic idea of our approach is to combine two different TEE hardware: one with a small secure memory but integrity protection — we call it Integrity-Protected TEE, or IP-TEE for short — and another one with a large secure memory but lacking integrity protection — called Large-Memory TEE, or LM-TEE for short.

Several challenges need to be addressed:

- **Application partitioning.** Code and data can be partitioned at different granularities, each offering different trade-offs: IP-TEE could execute integrity checks over LM-TEE at a page or memory object granularity, or directly store data in its secure memory; IP-TEE could execute only integrity checking logic, or could also execute part of the application logic; etc.

- **Integrity check scheme.** Integrity can be checked either (i) every time a particular data is read, which leads to more CPU usage but immediate violation detection; or periodically, which is less CPU intensive but detects violations later after they happen. This changes the scope of the scheme from immediate attack prevention to post-compromise attack detection, which is for example the approach chosen by the LibSEAL [1] system.

- **Communication between the two TEEs.** The communication between IP-TEE and LM-TEE needs to be secure. However, to the best of our knowledge, TEEs do not provide a mechanism to establish secure communication channel with another TEE (secure communication channels between applications running on the same TEE is possible, e.g., on AMD SEV-SNP). We thus need to send encrypted messages over untrusted shared memory, which incurs a substantial performance cost.

- **Attestation.** Both IP-TEE and LM-TEE need to be attested to the application users as well as between each others, to guarantee the security of the application. However, each TEE implements its own attestation mechanism. Thus, we need to provide a generic attestation mechanism valid across multiple TEE implementations. While there exist frameworks for remote attestation across multiple TEEs [11], to the best of our knowledge this is not the case for attestation across different TEEs running on the same physical host.

Not all applications can benefit from our approach. We target applications that: (i) need integrity protection of part of the code and/or data against hardware attacks; (ii) use several GBs of memory; and (iii) do not require integrity-protection for all computations. This includes key-value stores [2, 8, 10], databases [13, 17, 20], or data analytics systems [14, 15].

In this project, we will first address the problem of query result freshness in a large secure database [22]. Databases store a vast amount of data beyond what is available in IP-TEE. They also require integrity protection to prevent an attacker from dropping content by reverting the datastore to a previous version. This can happen if the attacker wants to save memory space or computation. Under our approach, IP-TEE inspects clients queries and maintains a replies log to ensure that LM-TEE, which executes the queries, does not send an out-of-date reply computed over old content. As an optimization, IP-TEE can also cache the most frequent query replies. The cache and log sizes need to be carefully controlled to avoid running out of secure memory.

4 PROOF-OF-CONCEPT

Our idea can be implemented in different ways, e.g., on an FPGA, a RISC-V board, or trying to bend commercial TEEs to our needs. Due to hardware availability, we propose to implement our system on the commercially available Intel SGX and AMD SEV-SNP TEEs: IP-TEE is implemented as Client SGX and LM-TEE is implemented as Scalable SGX or AMD SEV-SNP. To bring these two TEEs in the same physical machine, we propose to use an Intel VCA2 PCIe card and install it on a machine running a Scalable SGX or AMD SEV-SNP capable processor. The VCA2 card, also called SGX card [3], embeds three Client SGX-capable processors, each having its own RAM and 128MB of secure memory.

If each request coming from the network triggers rounds of communication between Client and Scalable SGX for integrity checks, the PCIe bus might become a bottleneck. To alleviate this problem we could make use of a smartNIC as shown by Lynx [19] the smartNIC would directly send the network requests to the VCA2 card, bypassing the host.

5 CONCLUSION AND FUTURE WORK

We have presented a new approach to protect large secure memory against hardware attacks aimed at jeopardizing the system integrity: combining two different TEE implementations, each having their own complementary characteristics. We are currently working on the implementation of our proof of concept.

ACKNOWLEDGMENTS

We thank our anonymous reviewers for their helpful comments. This work was supported by JSPS KAKENHI Grant Number JP21K17726.
REFERENCES


