Poster: TGX: Secure SGX enclave management using TPM

Dhiman Chakraborty
CISPA Helmholtz Center for Information Security, Germany
Saarland University, Germany
dhiman.chakraborty@uni-saarland.de

Atul Anand Jha
CISPA Helmholtz Center for Information Security, Germany
Saarland University, Germany
s8atjhaa@stud.uni-saarland.de

Sven Bugiel
CISPA Helmholtz Center for Information Security, Germany
bugiel@cispa.saarland

***Abstract*—Intel SGX provides a trusted execution environment on commodity computing platforms. Recent micro-architectural attacks like Spectre, Meltdown, or Foreshadow, however, raise doubts about the promised isolation of SGX-protected code and data, including some of the necessary cryptographic operations and credentials, e.g., for attestation.**

In this poster we present TGX, a combination of SGX and TPM working together to provide stronger isolation of crucial cryptographic operations of SGX and a way to circumvent micro-architectural attacks against SGX. TGX enables SGX to move its signing and verification mechanism from processor to TPM making the security sensitive information never available outside TPM, removing, for instance, the possibilities of stealing them from L1 cache. In particular, TGX should motivate that SGX and TPM can form a beneficial symbiosis.

I. Introduction

Trusted computing technology has become a valuable building block for security solutions. Trusted computing components are deployed in millions of devices in different forms, such as Trusted Execution Environments (TEE) like ARM TrustZone or the Trusted Platform Module (TPM). The most recent addition to this list is Intel's Software Guard Extension (SGX). Intel SGX is a CPU-based implementation of process isolation, providing a protection layer that isolates the runtime of logical processes (enclaves). In particular, SGX enables secure computation despite the complete software stack including the host OS being malicious. SGX ensures this isolation by implementing strict policies not only on enter and exit to and from the enclaves, but also limiting state transfer between enclaves and the untrusted software stack. To allow remote parties to distinguish between legitimate hardware and untrusted software and to establish secure end-to-end channels with enclaves (e.g., to provision sensitive data), SGX also implements *remote attestation* (RA) of enclaves.

However, recently discovered micro-architectural attacks, like *Spectre* [2], *Meltdown* [3], or *Foreshadow* [1] raised doubts about the strength of SGX enclave security and protection of secrets entrusted to enclaves. For instance, Foreshadow allows potentially reading sensitive data that can be brought into the L1 cache, increasing the risk of leaking L1 cache data holding enclave secrets, such as application data or secret keys.

The nature of those attacks also raises the question if alternative solutions to the micro-architectural patches by the processor manufacturers exist. Keeping the claims of SGX in mind, one possible way of tackling the situation, which we propose here, is in the form of support by onboard TPMs. TPM is the most widely deployed form of trusted computing technology on end-users devices, even being prescribed as mandatory component on computing platforms by software vendors like Microsoft. Thus, we have a situation in which off-the-shelf devices come with TPM and SGX equipped at the same time. TPM provides a crypto co-processor, protected credentials, secure storage, NVM, monotonic counters, and the attestation of its keys or stored measurements. One of the most powerful features of TPM2.0 are its Extended Authorization Policies (EAP) that allow access control on keys (and other TPM protected) entities based on, e.g., a command issuer's privilege level (locality) or signature, or the current state of the platform (counters or stored measurements). TPM can be invoked by the CPU to perform various cryptographic operations. Information protected inside TPM is not available to the CPU or host platform. With this in mind, we propose in this poster TGX, a symbiotic combination of SGX and TPM, in general, and show this combination concretely for TPM-based SGX enclave management:

- **TPM-based enclave building**: An enclave building using the TPM as the key generator and the signing entity.
- **TPM-based enclave verification**: An enclave verification process using TPM as the primary entity to create and verify for enclave launching.
- **Attestation with TPM**: A remote attestation mechanism via TPM to ensure confidentiality of the verification data. TPM enables the user to perform a local attestation based on TPM providing less dependency on Intel's provisioning server using extended authorization policies.

II. Technical Background and Motivation

SGX heavily depends on two infrastructure enclaves, namely i) Launch Enclave (LE) and ii) Quoting Enclave (QE). LE is used to launch a user space enclave. In SGX a userspace enclave requires generation of a valid and verified token (EINITTOKEN) by LE using two information: a) measurement of the content of the enclave (MRENCLAVE) and b) a valid author of the enclave (MRSIGNER). To generate the EINITTOKEN, LE generates a 128-bit key called Launch Key (LK). After generation of EINITTOKEN and LK, einit verifies the token against LK and upon a valid verification, einit launches the enclave. On abrupt halt, the LK process lets the key stay in L1 cache and, e.g., a Foreshadow attack can retrieve the key.

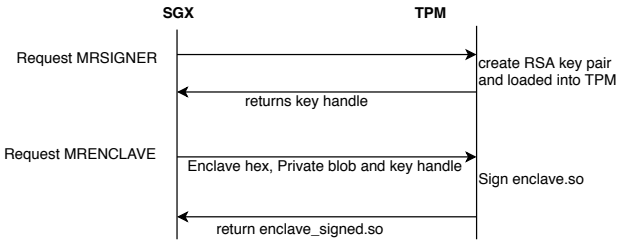


Fig. 1. Secure enclave build in SGX using TPM.

On the other hand, QE is designed to validate local attestation reports generated with an asymmetric private key that can be verified by a remote verifier. After receiving the private key from Intel, QE requests for a provisioning seal key using the same key retrieval algorithm used by LE. And in a similar way (like the attack on LE), on abrupt halting the get key operation could expose the sealing key in the L1 cache.

TPM can help to mitigate the above explained problems by keeping all the cryptographic secrets inside its protected storage and having a dedicated communication with SGX. This has historic precedent with Intel's late-launch (or DRTM) technology, where the CPU has exclusive privilege (i.e., locality level 4) to issue a reset of the TPM between power-cycles. Using late-launch as a template, with EAP on TPM2.0 and the particular features of SGX, flexible new policies can be built that allow SGX and enclaves to securely outsource credentials and cryptographic operations to the TPM. For instance, this integration could allow a custom verification in cohesion with TPM for QE, reducing dependency on Intel's provisioning service. Our prototype is currently under development in the SGX simulator. Our changes are implemented solely inside the SGX micro-code and make use of the default features of TPM.

III. Example Use-cases

A. TPM-based Enclave Build

We propose to bypass Intel's launch control policy and use TPM to generate MRSIGNER RSA key-pair for building enclaves (see Figure 1). Using the generated key, loaded into the TPM, SGX uses the TPM to sign the enclave code, yielding enclave.signed.so. This enclave can be verified and run by the SGX LE using previously generated public key. This method alleviates the problem of key availability in L1 cache, removing the possible attack surface. To prevent misuse of this TPM key by unauthorized parties, we use EAP for the key including locality. The CPU will use locality level 3 for SGX management commands, level 2 for commands from enclave code, and enforce level 0 for any non-SGX-related code (e.g., host OS and user processes). Thus, this key can be made exclusive to the SGX management code.

B. TPM-based Enclave Verification

We bypass the KEYID and EGETKEY sequence in LE and use the TPM to generate a 128-bit launch key (LK) that can be stored securely in TPM NVM (see Figure 2). This is followed by enclave measurement and verification against MRENCLAVE, MRSIGNER and enclave attribute data provided to TPM by LE. Using the data in PCRs and NVM,

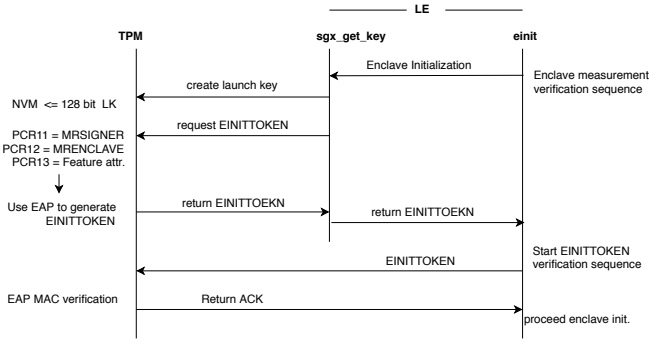


Fig. 2. TPM-based enclave verification in TGX. TPM locks down the security critical information inside its NV storage, securing it from cache based attacks.

TPM uses EAP to generate a MAC as EINITTOKEN. This EINITTOKEN can safely by forwarded to the untrusted environment and LE. On receiving EINITTOKEN, einit triggers EINITTOKEN validation using EAP MAC verification of TPM. Contrary to the SGX implementation, TPM preserves LK for the session to validate EINITTOKEN.

C. Remote attestation with TPM

Upon receiving Attestation Key (AK) from Intel Provisioning Service (IPS), Provisioning Enclave (PE) forwards the AK to TPM. TPM stores the AK into non-volatile memory and henceforth acts the signer of attestation reports on behalf of QE. When a remote verifier issues a challenge to the enclave, enclave binds the attestation report to its identity and sends it to QE, which forwards the report to TPM. TPM creates a quote to answer the attestation and returns it to QE to be forwarded to the remote verifier. Remote verifier checks the report against the public attestation key using IPS. As before, EAP can be used to protect the AK within the TPM.

IV. Conclusion

TGX is a combination of SGX and TPM, which by working in unison can help SGX to securely outsource cryptographic operations and secure storage to the TPM, which offers a dedicated chip with stronger isolation. Using EAP and the unique access of SGX to certain TPM localities together with measurements created by SGX, such TPM-managed credentials can be further protected. In this poster we show three use-cases of TPM-based enclave management. Further use-cases can be envisioned, e.g., using the TPM's monotonic counters for replay protection of enclave data stored in TPM-protected, persistent storage or using EAP for user authentication to unlock enclave's credentials protected by TPM.

References

[1] J. V. Bulck, M. Minkin, O. Weisse, D. Genkin, B. Kasikci, F. Piessens, M. Silberstein, T. F. Wenisch, Y. Yarom, and R. Strackx, "Foreshadow: Extracting the keys to the intel SGX kingdom with transient out-of-order execution," in *27th USENIX Security Symposium*, 2018.

[2] P. Kocher, J. Horn, A. Fogh, , D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz, and Y. Yarom, "Spectre attacks: Exploiting speculative execution," in *40th IEEE Symposium on Security and Privacy (S&P'19)*, 2019.

[3] M. Lipp, M. Schwarz, D. Gruss, T. Prescher, W. Haas, A. Fogh, J. Horn, S. Mangard, P. Kocher, D. Genkin *et al.*, "Meltdown: Reading kernel memory from user space," in *27th USENIX Security Symposium*, 2018.

TGX: Secure SGX Enclave Management using TPM

Dhiman Chakraborty, Atul Anand Jha, Sven Bugiel

1. Introduction

SGX in a Nutshell

- Hardware based isolation in CPU.
- Isolates runtime for secure execution using enclaves.
- Widely deployed with Intel CPUs.

Problems: Side-channel attacks

- Discovery of Spectre [1] and Meltdown [2].
- Discovery of Foreshadow [3].
- Cryptographic secrets can be stolen from L1 cache.

Possible solutions

- Fix of micro-architectural attacks,
  - → Requires Intel to patch micro code.
- Attaching other Secure Elements such as TPM together with SGX.

TPM in a Nutshell

- Most widely available Trusted Computing Technology.
- Works as a trusted third party on the system.
- Offers:
  - → Crypto co-processor.
  - → Protected credentials.
  - → Secure storage.
  - → Extended authorization policy.

2. Motivation

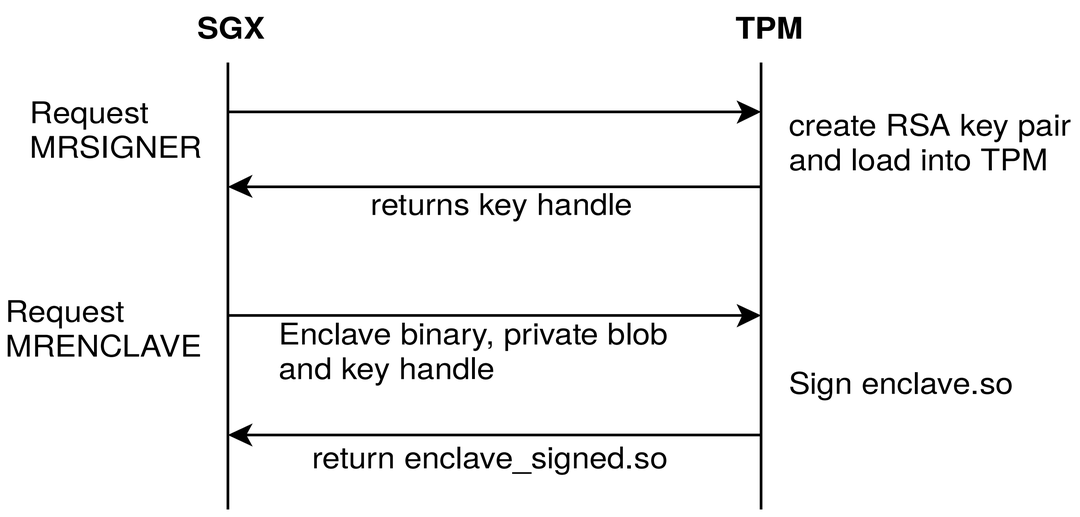
SGX dependency on cryptography for

- Launch enclave (LE).
- Quoting enclave (QE).

Our proposal → TGX (Trusted Guard Extension)

- A cohesion between SGX and TPM,
  - → Secure enclave building.
  - → Secure enclave verification.
  - → Attestation with TPM.

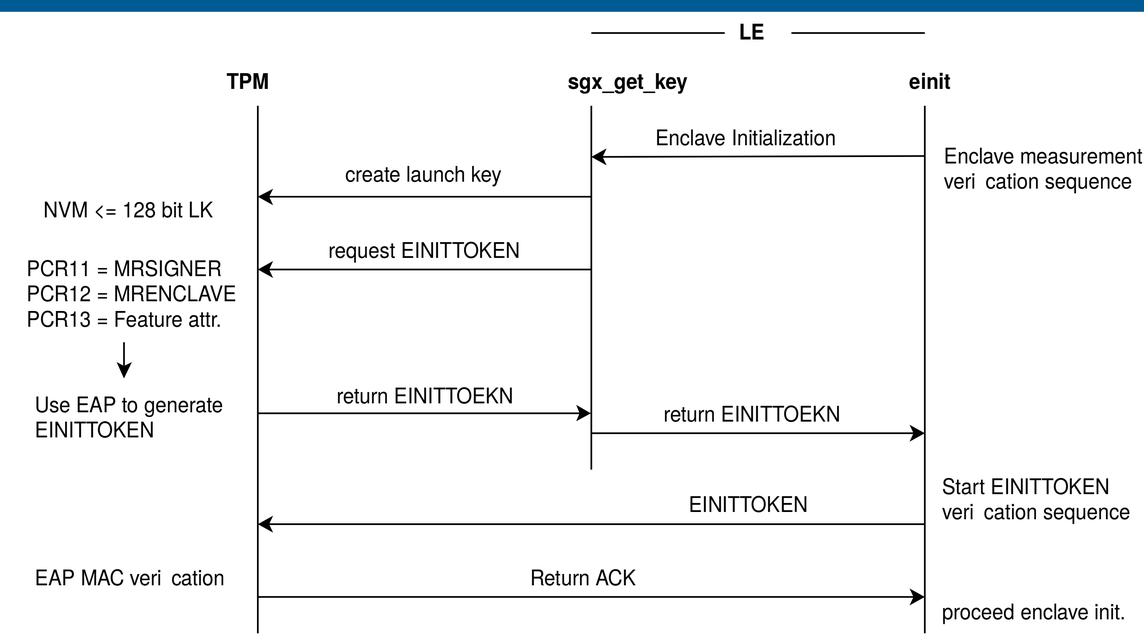
3. TPM based enclave build workflow



- SGX uses TPM to create MRSIGNER key pair.
- SGX invokes TPM to sign enclave using signer private key.

➡ Malicious agents cannot obtain MRSIGNER private key.

4. TPM based enclave launch workflow



- TPM creates LK on behalf of SGX-LE and stores it in NVM.
- LE requests EINITTOKEN for MRSIGNER, MRENCLAVE.
- TPM creates EINITTOKEN usign LK and returns to LE.
- LE - einint invokes EINITTOKEN verification in TPM.
- TPM performs verification using session LK.

➡ Malicious agents cannot obtain LK and forge EINITTOKEN.

5. Attestation using TPM

Efficient and secure attestation protocol with TPM

- TPM stores the attestation key (AK) from Intel provisioning service.
- Works as report issuer for QE.
- Plays challenge-response game with remote verifier through QE.

➡ TPM works as a signer on behalf of QE and by safe-keeping of AK, TPM makes the remote attestation process safer.

6. Roadblock for the project

Roadblock and workaround

- Cannot change the SGX hardware code.
- TPM cannot be changed.
- Current development is in emulator, easy to deploy in hardware through micro-code patching.

7. Gains using TPM

TPM provides certain gains

- Completely separate processing and storage of cryptographic data.
- Stores SGX secrets like keys securely in NV-memory.
- Custom verification procedure, removing dependency from Intel provisioning service.

➡ SGX together with TPM 2.0 (a la DRTM) allows advanced use-cases based on locality, EAP and CPU-supplied measurements.

8. Conclusion

Adding TPM in the signing and verification protocol of SGX will not only strengthen SGX but also protects it from micro-architectural attacks. Together with TPM, SGX will open more possibilities for trusted and secure computing such as custom verification protocol.

References:
[1] P. Kocher, J. Horn, A. Fogh, , D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz, and Y. Yarom,“Spectre attacks: Exploiting speculative execution,” in 40th IEEE Symposium on Security and Privacy (S&P’19), 2019.
[2] M. Lipp, M. Schwarz, D. Gruss, T. Prescher, W. Haas, A. Fogh, J. Horn, S. Mangard, P. Kocher, D. Genkin et al., “Meltdown: Reading kernel memory from user space,” in 27th USENIX Security Symposium (USENIX Security ’18), 2018.
[3] J. V. Bulck, M. Minkin, O. Weisse, D. Genkin, B. Kasikci, F. Piessens, M. Silberstein, T. F. Wenisch, Y. Yarom, and R. Strackx, “Foreshadow: Extracting the keys to the intel SGX kingdom with transient out-of-order execution,” in 27th USENIX Security Symposium, 2018.

CISPA
HELMHOLTZ CENTER FOR INFORMATION SECURITY