Intro to Intel SGX

Lianke Qin

Pictures credited to SGX Manual, SGX tutorial(ISCA 2015 CCS 2017), SGX 101, SGX Explained and SGX-Perf

Preliminary Knowledge

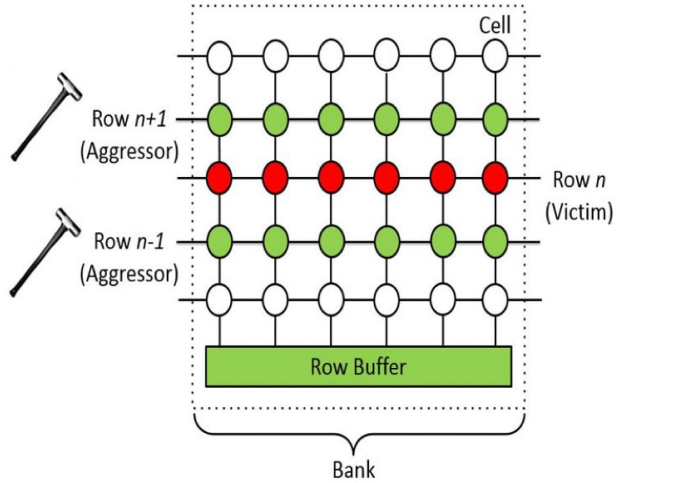
- Cryptography
  - Encryption(confidentiality)
  - Hash, MAC & Signature(integrity)
  - Nonce(freshness)
  - Key exchange protocol(DHKE)
- OS
  - Memory hierarchy and management
  - Process & Threads
  - Privilege levels

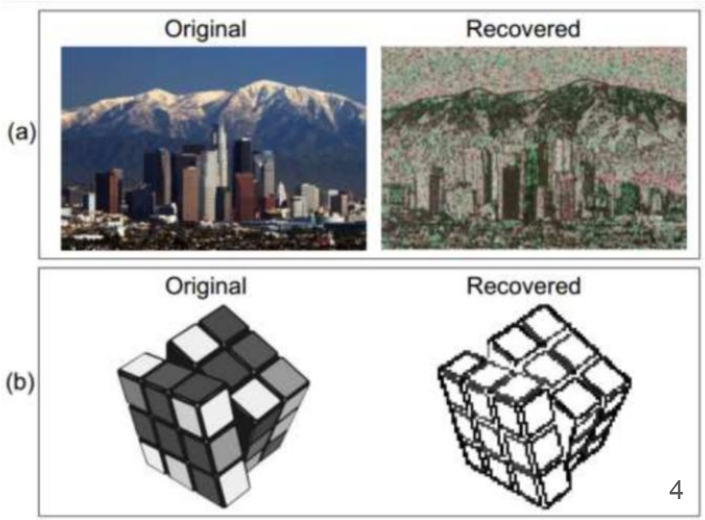
Outline

- Vulnerabilities
- SGX Pros & Cons
- Enclave
- Attestation
- SGX Application(with demo)
- SGX Related Paper

Vulnerabilities

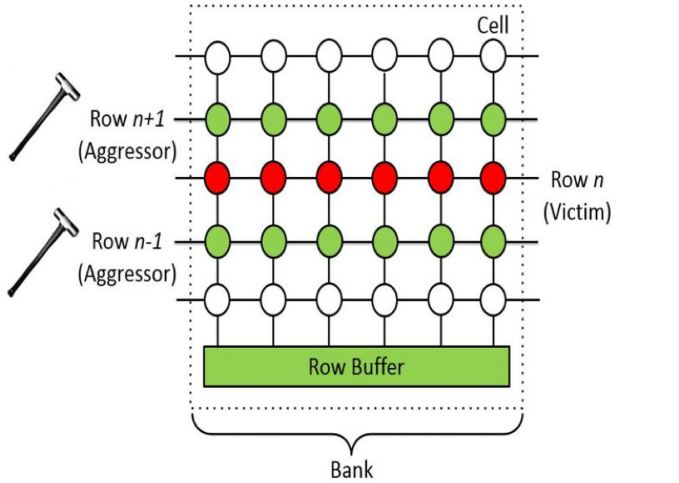
- Physical attacks
  - Chip attacks[1]
- Software attacks on peripherals
  - Rowhammer DRAM attack[2]
- Address translation attacks
  - Page table(gain higher privilege level)[3]
- Cache timing attack[4, 5]
- ……

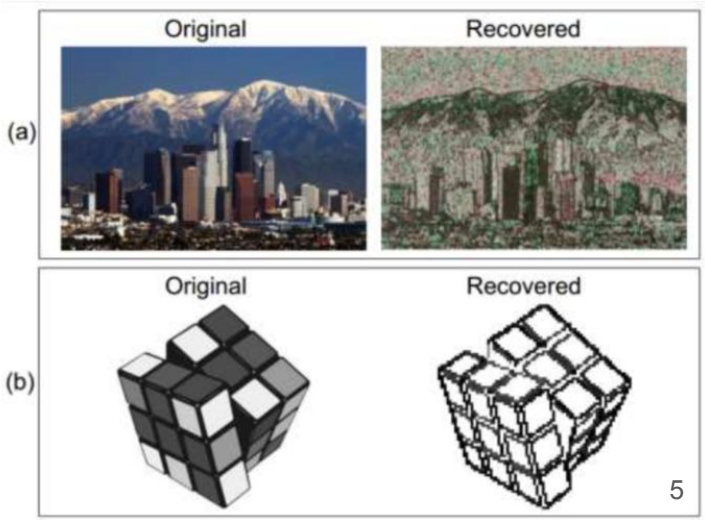




Vulnerabilities

- Physical attacks
  - Chip attacks[1]
- Software attacks on peripherals
  - Rowhammer DRAM attack[2]
- Address translation attacks
  - Page table(gain higher privilege level)[3]
- Cache timing attack[4, 5]
- ……





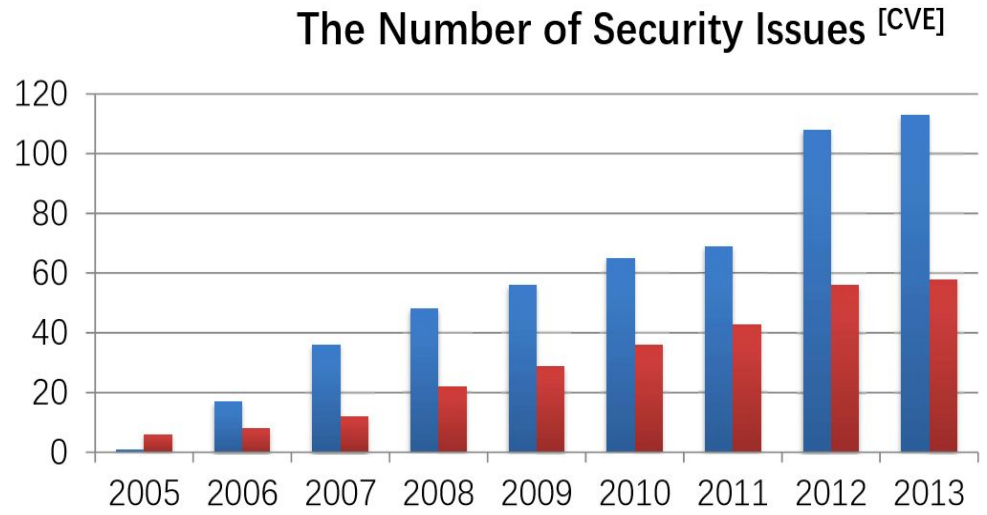
Attack Surfaces

Attack Surface Without Enclaves

Top 50 Products By Total Number Of "Di

Go to year: 1999 2000 2001 2002 2003 2004 2005 200

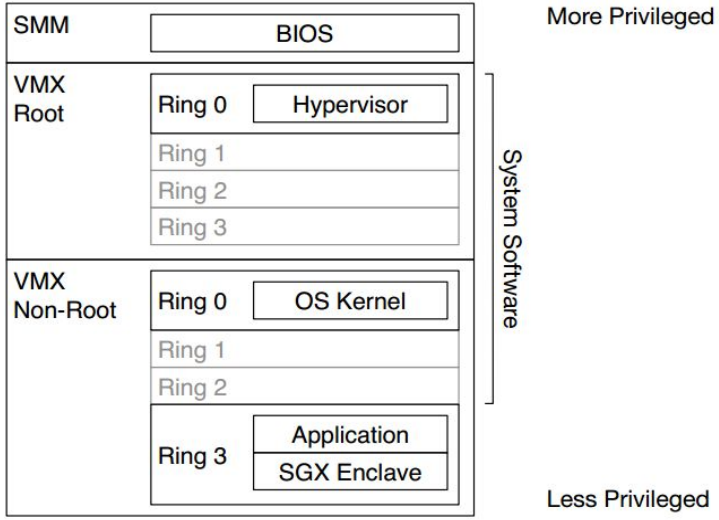
	Product Name	Vendor Name	Pro
1	Android	Google	OS
2	Debian Linux	Debian	OS
3	Ubuntu Linux	Canonical	OS
4	Flash Player	Adobe	Ap
5	Leap	Novell	OS
6	Opensuse	Novell	OS
7	Acrobat Reader Dc	Adobe	Ap
8	Acrobat Dc	Adobe	Ap
9	Acrobat	Adobe	Ap
10	Linux Kernel	Linux	OS

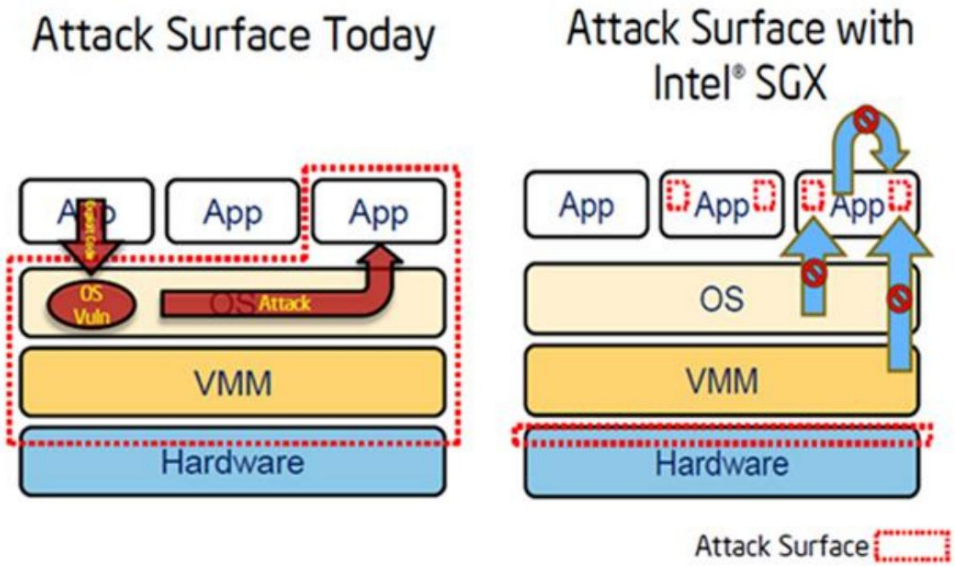


Attack Surface

What SGX Can Offer

- Enclave memory cannot be accessed from outside the enclave(OS, hypervisor, BIOS, drivers)
- **sgx_ecall** is the only way which performs several protection checks before calling a trusted enclave function
- Data will be encrypted with replay protection(nonce) before being moved to outside of enclave
- Attestation mechanism to ensure (remote) application runs on a genuine Intel SGX processor





SGX Limitations

- Timing side-channel attack(cache)[7, 8]
- Other sophisticated attacks(speculative execution, branch shadowing, rowhammer)[9, 10]
- Small enclave memory(128MB for now)
- Performance overhead
  - Ecall & Ocall
  - Data encryption & decryption inside enclave
  - Enclave creation, initialization and deletion
  - SGX attestation
- No CPUID and RDTSC inside enclave

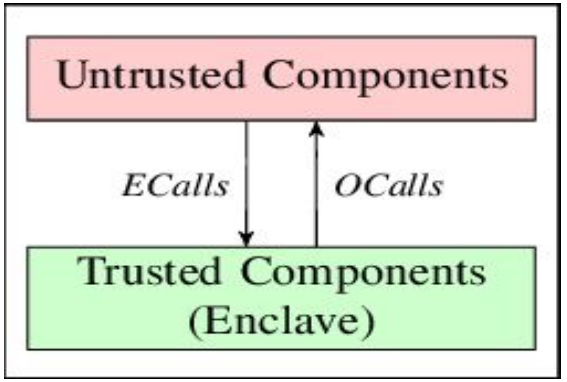
Instruction Level Overhead

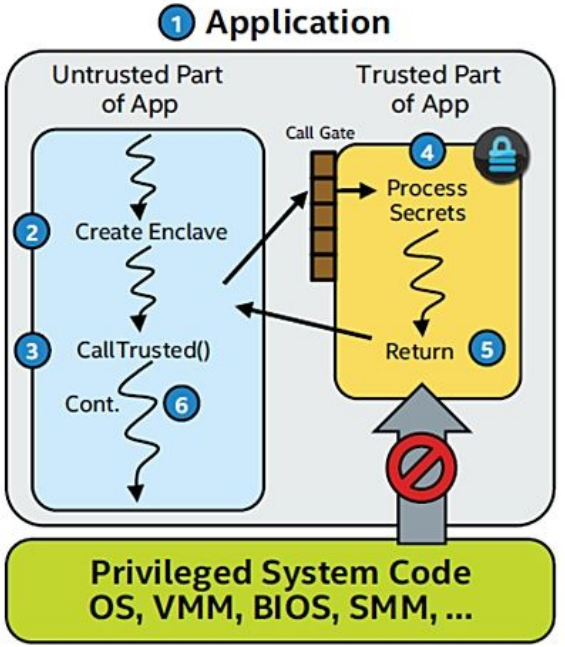
Micro-benchmark	Description	Latency (cycles)
`ecall` (warm cache)	Calling an enclave w/o parameters, and immediately return	8,640
`ecall` (cold cache)	Same as above, but the entire cache is flushed	14,170
`ecall` (buffer)	Calling an enclave func, passing 2K buffer, to the enclave	9,816
	Calling an enclave func, passing 2K buffer, from the enclave	11,172
	Calling an enclave func, passing 2K buffer, to and from enclave	10,827
`ocall` (warm cache)	Calling untrusted w/o parameters, and immediately return	8,314
`ocall` (cold cache)	Same as above, but the entire cache is flushed	14,160
`ocall` (buffer)	Calling untrusted func, passing 2K buffer, to the untrusted	9,254
	Calling untrusted func, passing 2K buffer, from the untrusted	11,418
	Calling untrusted func, passing 2K buffer, to and from untrusted	9,801

Source: "Regaining Lost Cycles with HotCalls" [WBA17] http://www.ofirweisse.com/ISCA17_Ofir_Weisse.pdf

SGX App Flow

- App is partitioned into trusted and untrusted parts
- App creates enclave
- Trusted function is called through ***sgx_ecall***
- ***sgx_ocall*** is used to interact with untrusted part when executing code within enclave
- code running inside enclave can decrypt and process data





Enclave Architecture Overview

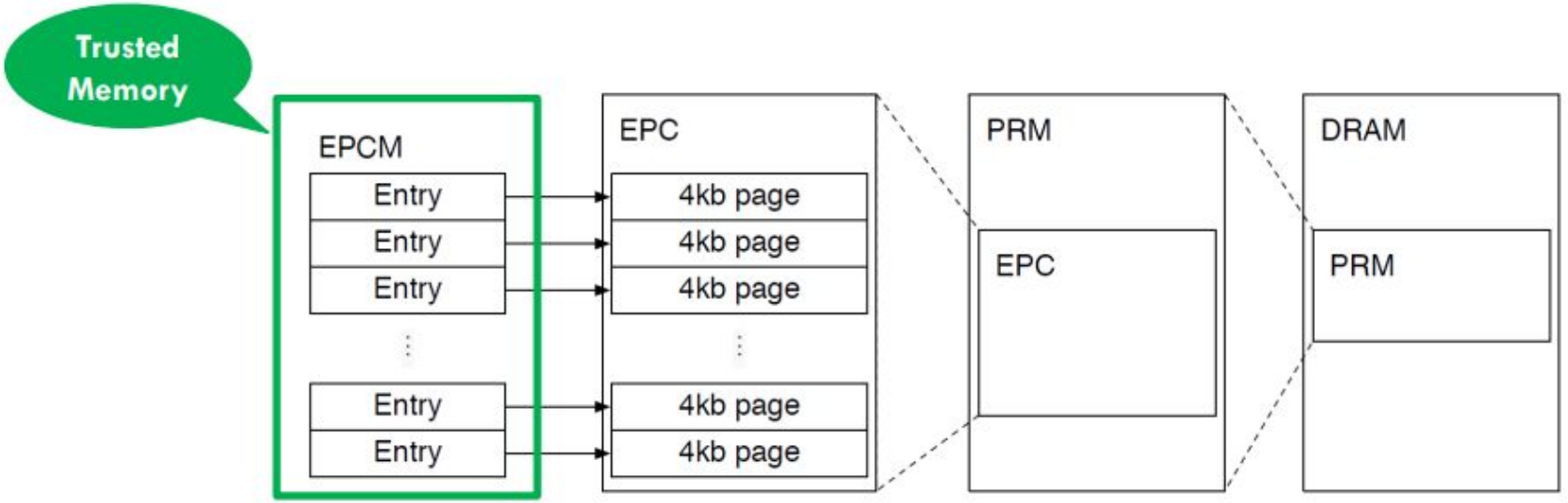
- Enclave memory organization
- Enclave control structure
- Thread control structure
- Address translation in SGX enclave
- Enclave measurement
- Enclave lifecycle
- Enclave definition language

Enclave Memory Organiz

- The Processor's Reserved Memory (PRM) is a res
- The Enclave Page Cache (EPC) contains enclave
- The Enclave Page Cache Map (EPCM) contains a

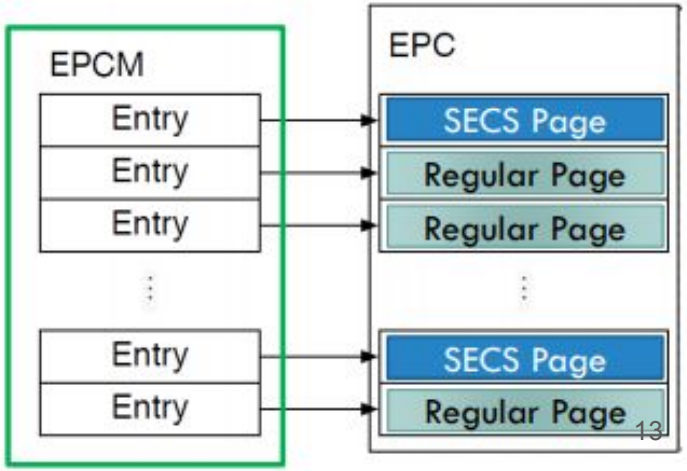
Field	Bits	Description
VALID	1	0 for un-allocated EPC pages
PT	8	page type
ENCLAVESECS		identifies the enclave owning the page

EPCM



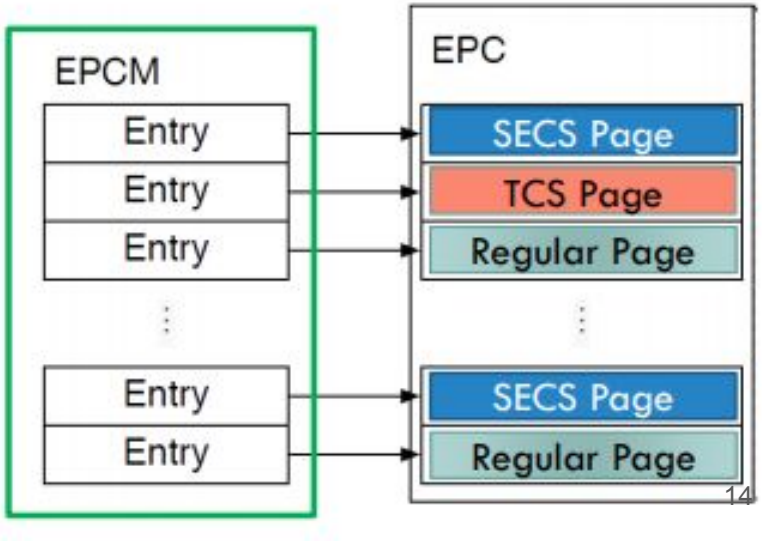
SGX Enclave Control Structure (SECS)

- The SGX Enclave Control Structure (SECS) stores critical metadata of each SGX enclave
  - Enclave's measurement for software attestation
  - Enclave Attributes
- Each SECS is stored in a dedicated EPC page
- SECS Pages cannot be accessed by:
  - System Software (OS/Hypervisor etc.)
  - Even the enclave's code itself



Thread Control Structure (TCS)

- Concurrently execute the same enclave's code at the same time, via different threads
- Each TCS is stored in a dedicated EPC Page
- The contents of an EPC page that holds a TCS cannot be directly accessed, even by the code of the enclave that owns the TCS



Address Translation for SGX Enclave

- The OS and hypervisor are in full control of the page tables and Extended Page Table.
- When an EPC page is allocated, its intended virtual address is recorded in the ADDRESS field.
- R/W/X attributes from EPCM entry override the permissions specified in page tables.

Field	Bits	Description
ADDRESS	48	the virtual address used to access this page
R	1	allow reads by enclave code
W	1	allow writes by enclave code
X	1	allow execution of code inside the page, inside enclave

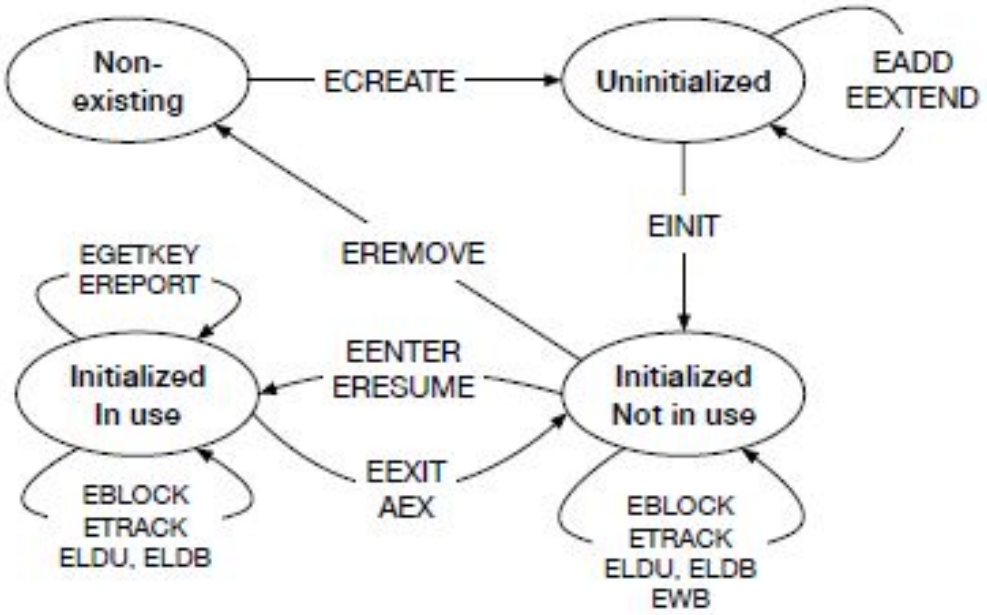
SGX Enclave Measurement(MRENCLAVE)

When building an enclave, SGX generates a cryptographic log of all the build activities

- Content: Code, Data
- Location of each page within the enclave
- Security flags being used

Enclave Lifecycle

- Creation (**ECREATE**)
- Loading (**EADD**, **EEXTEND**)
- Initialization (**EINIT**)
- Enter/Exit the Enclave (**EENTER**/**EEXIT**)
- Teardown (**EREMOVE**)



Enclave Definition Language(.EDL)

```
enclave {
    // Include files
    // Import other edl files
    // Data structure declarations to be used as parameters of the function prototypes in edl
    trusted {
        // Trusted function prototypes (ECALLs)
    };
    untrusted {
        // Untrusted function prototypes (OCALLs)
    };
};
```

SGX Attestation Overview

- Preliminary instructions and secret keys
- Trust chain in software attestation
- Local attestation
- Remote attestation

SGX Attestation Preliminaries

- **EGETKEY** gets derivatives of device keys. EGETKEY produces symmetric keys for different purposes
- **EREPORT** generates a cryptographic structure, called REPORT, for caller enclave's measurement and attributes
- **SIGSTRUCT** holds enclave's MRENCLAVE together with other enclave attributes, and it is a mandatory supplement for launching any enclave

EGETKEY

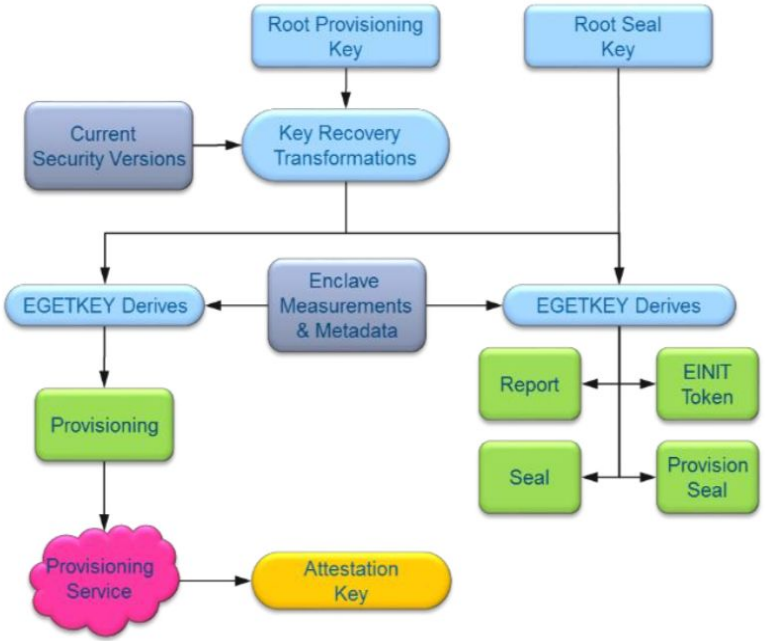
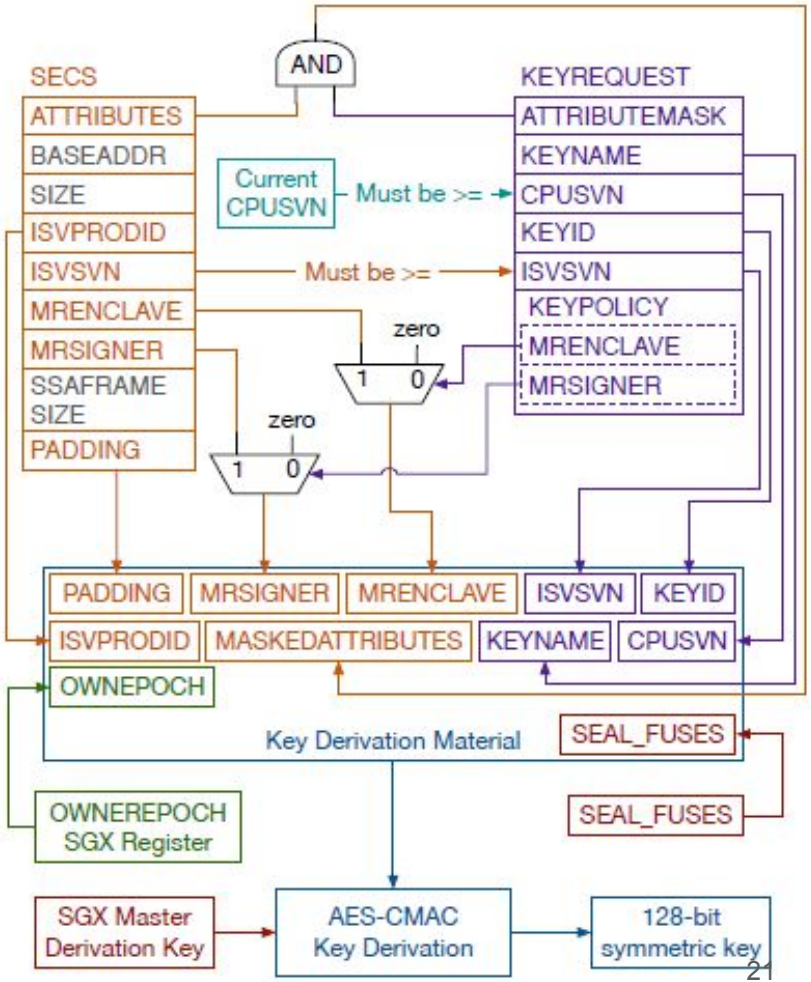


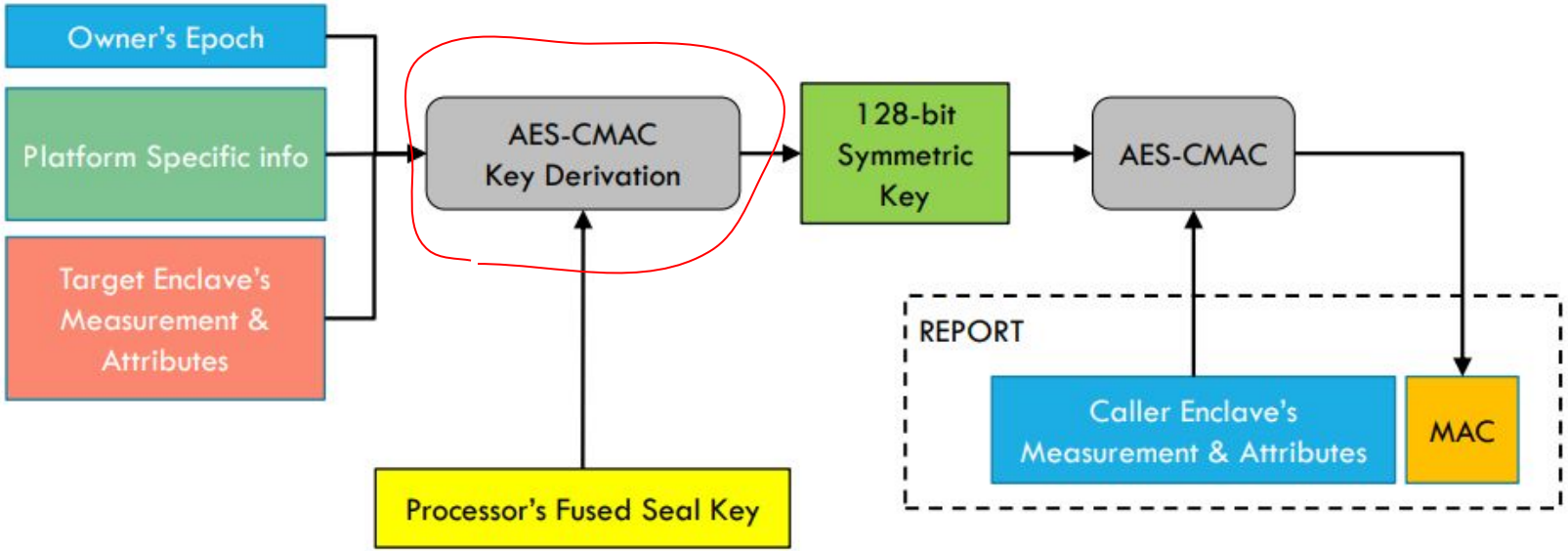
Figure 5: SGX Key Hierarchy



SGX Attestation Preliminaries

- **EGETKEY** gets derivatives of device keys. EGETKEY produces symmetric keys for different purposes
- **EREPORT** generates a cryptographic structure, called REPORT, for caller enclave's measurement and attributes
- **SIGSTRUCT** holds enclave's MRENCLAVE together with other enclave attributes, and it is a mandatory supplement for launching any enclave

REPORT Generation(EREPORT)



SGX Attestation Preliminaries

- **EGETKEY** gets derivatives of device keys. EGETKEY produces symmetric keys for different purposes
- **EREPORT** generates a cryptographic structure, called REPORT, for caller enclave's measurement and attributes
- **SIGSTRUCT** holds enclave's MRENCLAVE together with other enclave attributes, and it is a mandatory supplement for launching any enclave

SGXSTRUCT

Field	Bytes	Description
ENCLAVEHASH	32	Must equal the enclave's measurement (§ 5.6).
ISVPRODID	32	Differentiates modules signed by the same private key.
ISVSVN	32	Differentiates versions of the same module.
VENDOR	4	Differentiates Intel enclaves.
ATTRIBUTES	16	Constrains the enclave's attributes.
ATTRIBUTEMASK	16	Constrains the enclave's attributes.

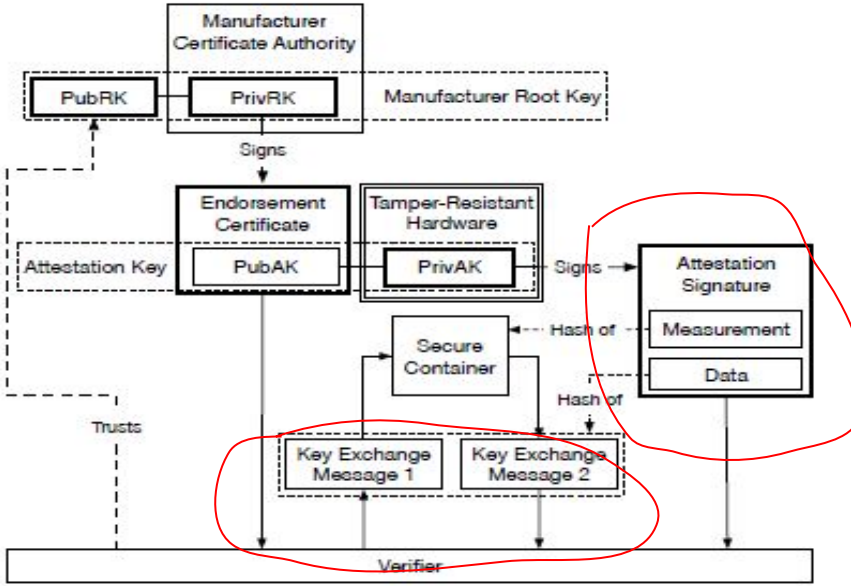
Table 19: A subset of the metadata fields in a SIGSTRUCT enclave certificate

SGX Attestation Preliminaries

- **Root Provisioning Key (RPK).** Intel maintains a database of all RPKs ever produced. Intel stores all RPKs for SGX processors to demonstrate their genuineness
- **Root Sealing Key (RSK)**. Intel declares that it attempts to erase all RSKs so that each SGX processor should assume that its RSK is both **unique and known only to itself**

Trust Chain in Attestation

- Manufacturer is the root of trust
- Secure processor signs the data and measurement inside each container for attestation
- Verifier ensures:
  - The attestation signature belongs to the target software
  - The software runs in an isolated secure container the verifier trusts



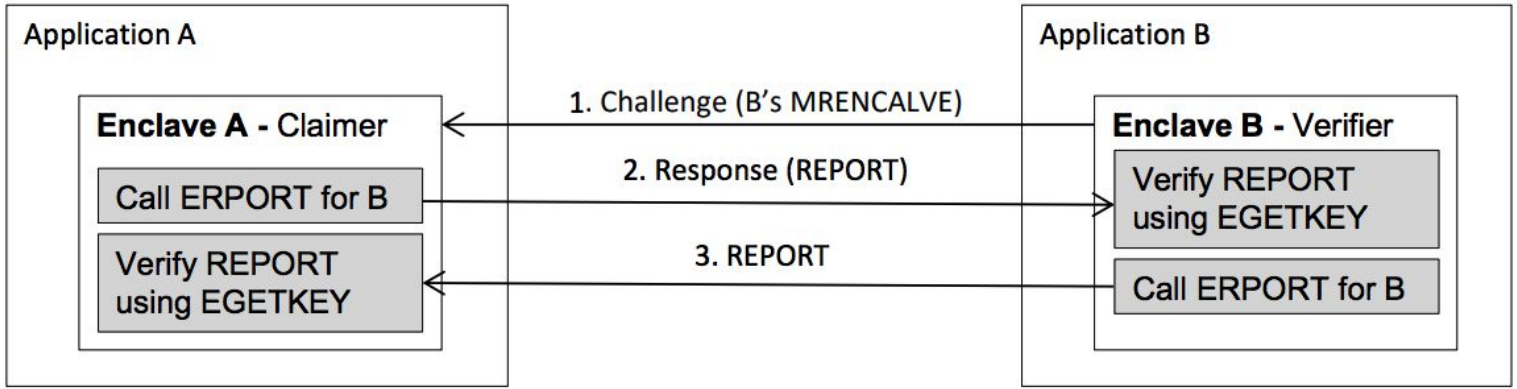
SGX Attestation

Enclave proves its trustworthiness to verifier

- ***Local attestation*** allows one enclave to attest its Thread Control Block (TCB) to another enclave on the *same* SGX processor
- ***Remote attestation*** allows one enclave to attest its TCB to another entity outside of the SGX processor

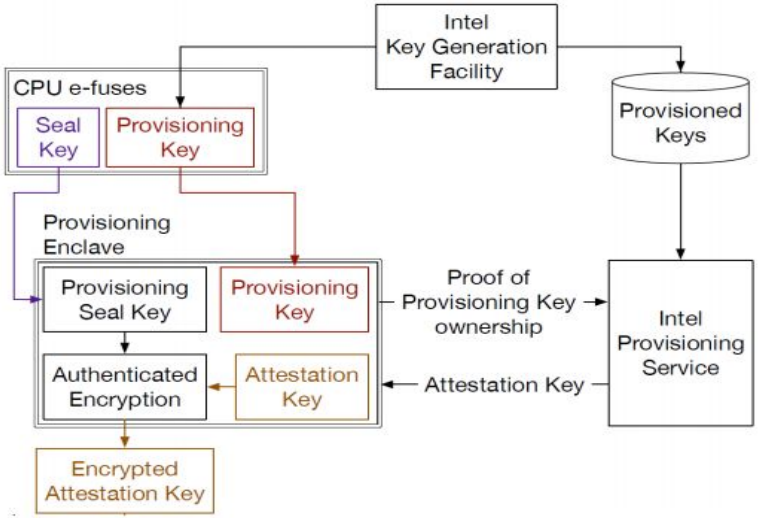
Local Attestation in SGX

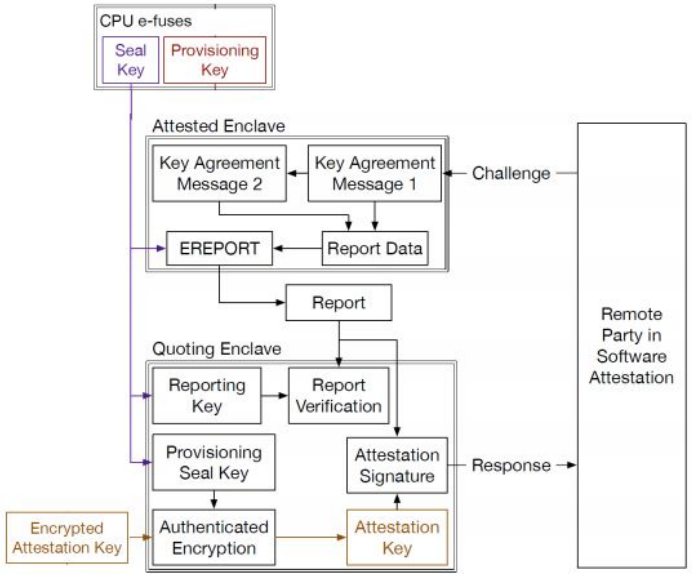
- Verifier enclave sends its MRENCLAVE to Claimer enclave
- Claimer enclave calls EREPORT instruction to generate REPORT structure for the verifier enclave
- Verifier enclave call EGETKEY to retrieve REPORT KEY that is specific to the processor and verifies the REPORT
- Reciprocate the above steps to make claimer ensure that verifier runs on the same processor with it



Remote Attestation in SGX

- Intel Provisioning Service issues an Attestation Key
- Quoting Enclave performs Local Attestation of application's Enclave
- Quoting Enclave decrypts the Attestation Key and signs the REPORT
- The Intel Attestation Service verifies the signature
- The IAS creates a new attestation verification report





SGX Application

- Develop SGX software from scratch thro
  - Partition software into **trusted** and **untrusted**
  - Use SGX SDK to create enclave, generate stu
- Porting existing software on LibOS(Have
  - Application requires a manifest to specify whic
  - Most of the time, application runs in enclave

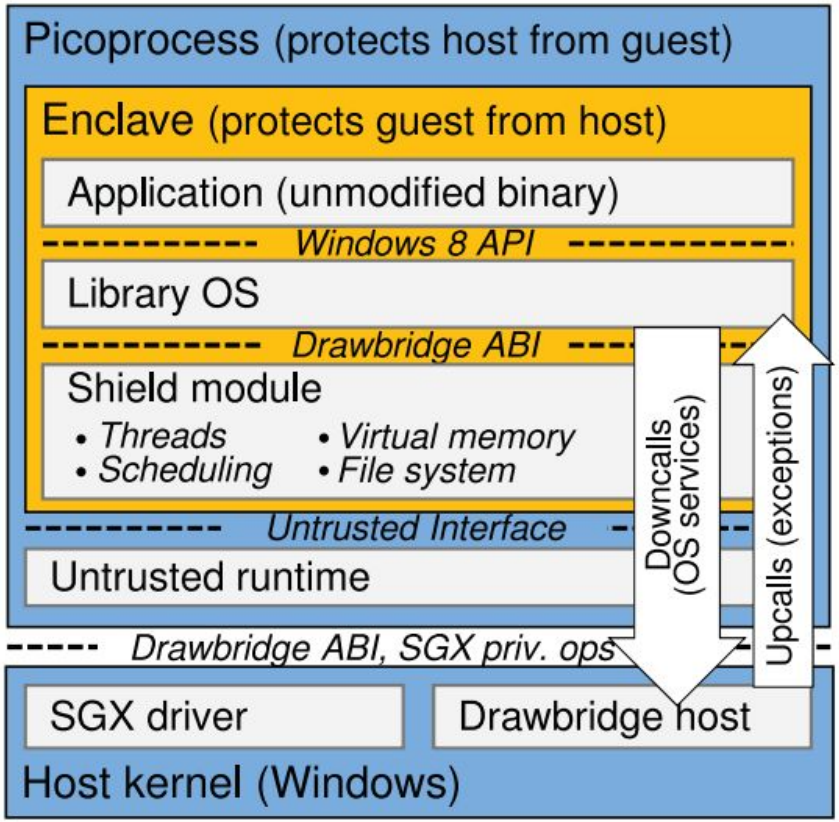
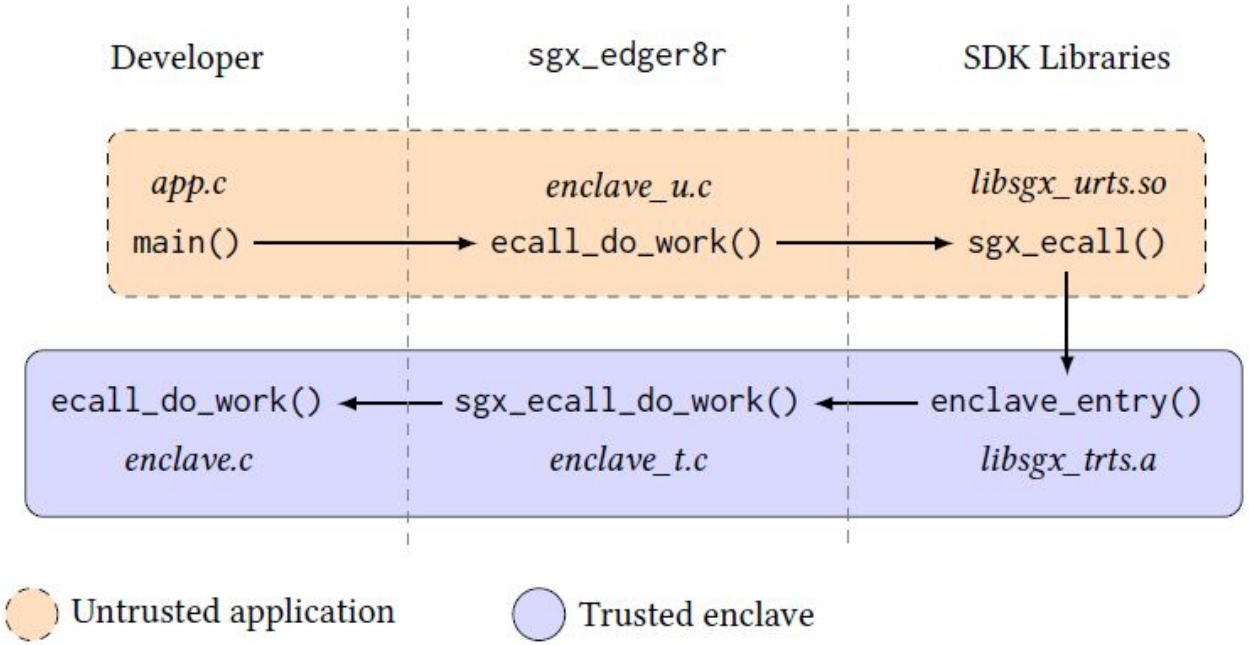


Fig. 2. Haven components and interfaces.

SGX App Demo

- Read file row by row
- Encrypt each row inside enclave
- Move encrypted row
- Decrypt row inside e



Related TEE Platform

- ARM TrustZone
- The IBM 4765 Secure Coprocessor
- The Trusted Platform Module (TPM)
- The Aegis Secure Processor

SGX Related Paper

- SCONE: Secure Linux Containers with Intel SGX
- Ryoan: a distributed sandbox for untrusted computation on secret data
- SecureKeeper: Confidential ZooKeeper using Intel SGX
- SGX-Log: Securing System Logs With SGX
- Glamdring: Automatic Application Partitioning for Intel SGX
- Enhancing Security and Privacy of Tor's Ecosystem by Using Trusted Execution Environments
- Cache Attacks on Intel SGX
- Graphene-SGX: A Practical Library OS for Unmodified Applications on SGX
- EnclaveDB: A Secure Database using SGX
- OBLIVIATE: A Data Oblivious Filesystem for Intel SGX
- ShieldStore: Shielded In-memory Key-value Storage with SGX
- Towards Memory Safe Enclave Programming with Rust-SGX
- OPERA: Open Remote Attestation for Intel’s Secure Enclaves
- Telling Your Secrets Without Page Faults: Stealthy Page Table-Based Attacks on Enclaved Execution\
- CrypTFlow : Secure TensorFlow Inference
- …….

SGX Installation Guide

- Test if your machine supports SGX using SGX Hardware
- Download SGX SDK & driver from Intel and install them
- Copy libraries to /usr/lib
- Play with demo codes inside Linux-SGX

Details Not Covered

- How SGX deals with hardware exception
- Enclave linear memory address space mapping
- EPC page eviction and TLB update
- Refer to Intel SGX manual for other instructions

Summary

- A hardware based trusted execution environment that minimize application attack surface with acceptable overhead
- Two ways of developing SGX application
  - From scratch; partition into untrusted and trusted parts
  - Porting existing application software on LibOS

References

1. Friedrich Beck. Integrated Circuit Failure Analysis: a Guide to Preparation Techniques. John Wiley & Sons, 1998.
2. Seaborn M, Dullien T. Exploiting the DRAM rowhammer bug to gain kernel privileges. Black Hat. 2015 Mar;15:71.
3. Van Bulck J, Weichbrodt N, Kapitza R, Piessens F, Strackx R. Telling your secrets without page faults: Stealthy page table-based attacks on enclaved execution. In26th {USENIX} Security Symposium ({USENIX} Security 17) 2017 (pp. 1041-1056).
4. Bernstein DJ. Cache-timing attacks on AES.
5. Bonneau J, Mironov I. Cache-collision timing attacks against AES. International Workshop on Cryptographic Hardware and Embedded Systems 2006 Oct 10 (pp. 201-215). Springer, Berlin, Heidelberg.
6. Kim T, Lin Z, Tsai CC. CCS'17 Tutorial Abstract/SGX Security and Privacy. InProceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security 2017 Oct 30 (pp. 2613-2614).
7. Götzfried J, Eckert M, Schinzel S, Müller T. Cache attacks on Intel SGX. InProceedings of the 10th European Workshop on Systems Security 2017 Apr 23 (pp. 1-6).
8. Moghimi A, Irazoqui G, Eisenbarth T. Cachezoom: How SGX amplifies the power of cache attacks. International Conference on Cryptographic Hardware and Embedded Systems 2017 Sep 25 (pp. 69-90). Springer, Cham.
9. Jang Y, Lee J, Lee S, Kim T. SGX-Bomb: Locking down the processor via Rowhammer attack. InProceedings of the 2nd Workshop on System Software for Trusted Execution 2017 Oct 28 (pp. 1-6).
10. Chen G, Chen S, Xiao Y, Zhang Y, Lin Z, Lai TH. SgxPectre: Stealing Intel secrets from SGX enclaves via speculative execution. In2019 IEEE European Symposium on Security and Privacy (EuroS&P) 2019 Jun 17 (pp. 142-157). IEEE.
11. Baumann A, Peinado M, Hunt G. Shielding applications from an untrusted cloud with haven. ACM Transactions on Computer Systems (TOCS). 2015 Aug 31;33(3):1-26.
12. Tsai CC, Porter DE, Vij M. Graphene-sgx: A practical library {OS} for unmodified applications on {SGX}. In2017 {USENIX} Annual Technical Conference ({USENIX}{ATC} 17) 2017 (pp. 645-658).

Thanks!