Software Guard Extension

Introduction

- New instruction set introduced by Intel in Skylake CPUs in 2015 as a standard addition but recently only on specific Intel Core CPUs
- Creates a secure environment that protects sensitive information and code from local/remote attacks and higher privilege processes
- A hardware-enforced security mechanism which requires:
  - Trusted Computing Base (TCB)
  - Hardware Secrets
  - Remote Attestation
  - Sealed Storage
  - Memory Encryption

Fundamentals of SGX

- Trusted Computing Base refers to the entirety of a computer system's hardware, firmware, and software components that work together to provide a system with a secure environment
- Hardware Secrets would consist of the two 128-bit keys specific to the CPU created at production
  - Root Provisioning Key (Known to Intel)
  - Root Seal Key (Unknown to Intel)

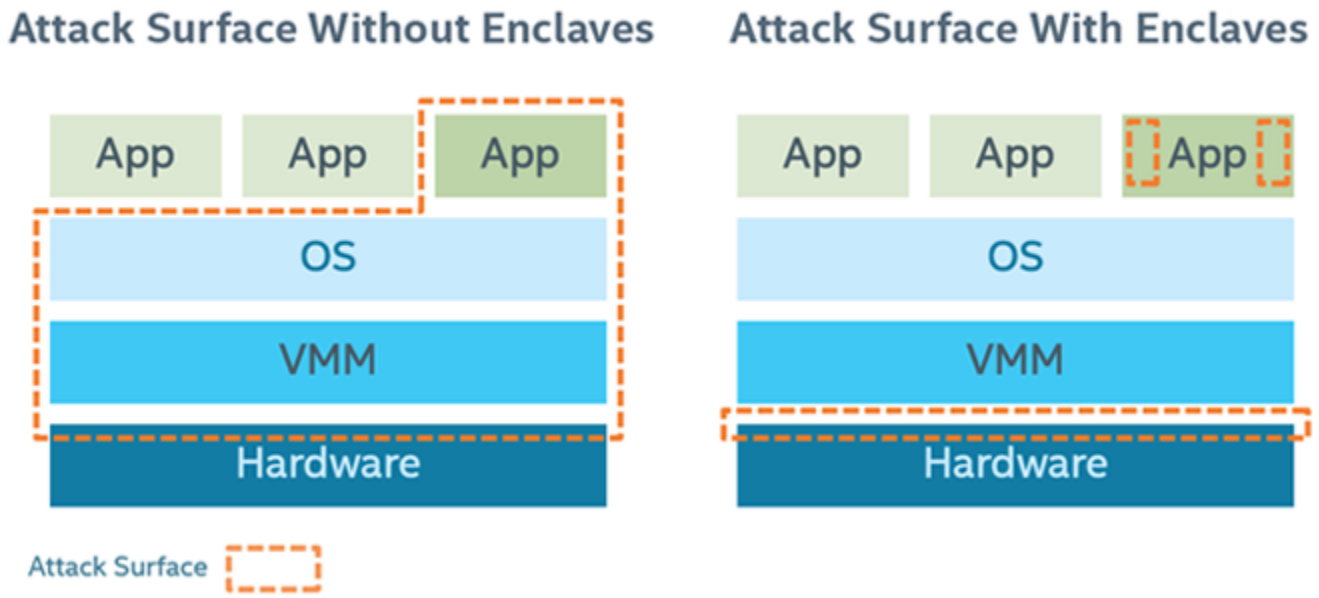
Fundamentals of SGX

- Remote Attestation is the process of how a client can prove to the service provider that an enclave is running a given software, inside a given CPU, with a given security level, for a given Software Vender (ISV)
- Sealed Storage is the ability to save secret data in untrusted media storage after encryption and retrieve later
- Memory Encryption is the process of how SGX hardware and software protects information such as through the memory encryption engine

Benefits of SGX

- Protections offered from SGX from hardware/software attacks:
  - Enclave memory is private and cannot be read/written from any entity outside of enclave regardless of privilege level
  - Enclaves cannot be debugged by software or hardware debuggers
  - Enclave environment cannot be entered through function call, jumps, register manipulation, or stack manipulation and can only be accessed through enclave function calls (ECall or OCall)
  - Enclave memory is encrypted and attempts to snoop or access memory will return encrypted data
  - Memory encryption key is randomly changed every power cycle and stored in CPU's inaccessible region
  - Data is isolated within each enclave (silos) and can only be accessed by code that shares the enclave

Benefits of SGX



Current SGX Weak Points

- Security Limitations of SGX:
  - Cache timing attacks
  - Physical attacks
  - Microcode malicious patching
  - Untrusted I/O
  - Human element
  - CPU bugs or bugs in dependencies

Terminology

- Untrusted Run-Time System (uRTS) – code that executes outside of the Intel SGX enclave environment and performs functions such as:
  - Loading and managing an enclave
  - Making calls to an enclave and receiving calls from within an enclave
- Trusted Run-Time System (tRTS) – code that executes within an Intel SGX enclave environment and performs functions such as:
  - Receiving calls into the enclave and making calls outside of an enclave
  - Managing the enclave itself
  - Standard C/C++ libraries and run-time environment

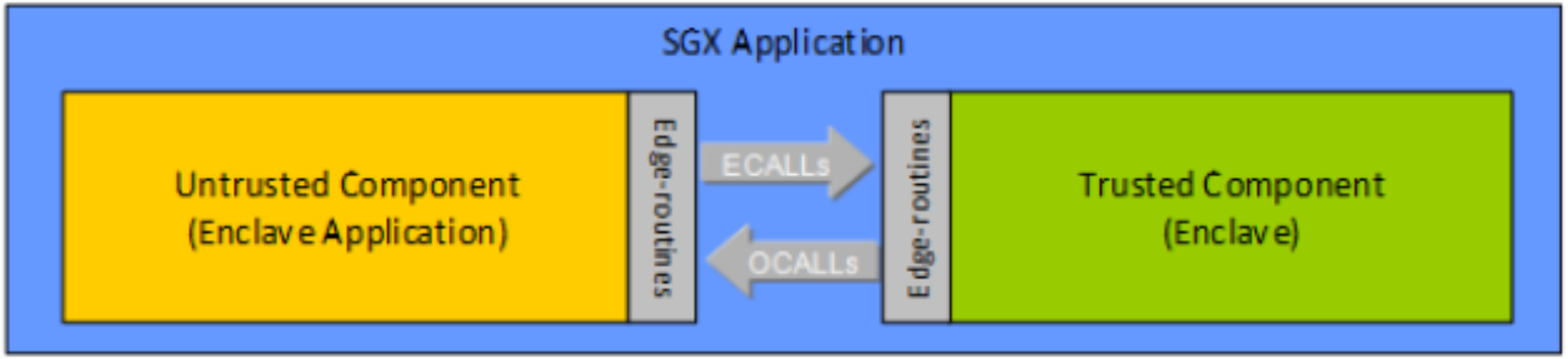
Terminology

- Edge Routines – functions that may run outside the enclave (untrusted edge routines) or inside the enclave (trusted edge routines) and serve to bind a call from the application within a function inside the clave or a call from the enclave with a function in the application
- 3rd Party Libraries – any library tailored to work inside the Intel SGX enclave not by Intel
- ECall – “Enclave Call” made into an interface within the enclave
- OCall – “Out Call” made from within the enclave to the outside application
- Trusted – Code or construct that runs inside an enclave

Terminology

- Trusted Thread Context – context for a thread running inside the enclave and composed of:
  - Thread Control Structure (TCS)
  - Thread Data/Thread Local Storage – data within the enclave specific to the thread
- Untrusted – code or construct that runs in the application and is outside of the enclave

Enclave/Enclave Application Interaction



SGX Data Structure Introduction

- There are 13 different Data Structures in Intel's SGX instruction set
  - 8 data structures are enclave specific
  - 3 data structures are associated with certain memory page(s)
  - 2 data structures are associated with overall resource management

SGX Data Structure Introduction

SGX Enclave Control Structure (SECS)	Thread Control Structure (TCS)
Page Information (PAGEINFO)	Security Information (SECINFO)
Paging Crypto MetaData (PCMD)	Version Array (VA)
Enclave Page Cache Map (EPCM)	Enclave Signature Structure (SIGSTRUCT)
EINT Token Structure (EINITTOKEN)	Report (REPORT)
Report Target Info (TARGETINFO)	Key Request (KEYREQUEST)
State Save Area (SSA)	

SGX Data Structure Introduction

- Resource Management:
  - Enclave Page Cache Map (EPCM)
- Memory Management:
  - Page Information (PAGEINFO)
  - Security Information (SECINFO)
  - Page Crypto MetaData (PCMD)
  - Version Array (VA)

SGX Data Structure Introduction

- Memory Content:
  - SGX Enclave Control Structure (SECS)
  - Thread Control Structure (TCS)
  - Save State Area (SSA)
- Attestation Structures:
  - Key Request (KEYREQUEST)
  - Report Target Info (TARGETINFO)
  - Report (REPORT)
- Enclave Structures:
  - EINIT Token Structure (EINITOOKEN)
  - Enclave Signature Structure (SIGSTRUCT)

SGX Data Structure Introduction

- Enclave Page Cache Map (EPCM)
  - Secure structure used by processor to track contents of EPC
  - Holds one entry per page currently loaded into PEC
  - Not assessable by software and layout of the EPCM fields are implementation specific (Elaborated later)
  - Contains information such as RWX permissions, type, linear address, state, and more

SGX Data Structure Introduction

- Page Information (PAGEINFO)
  - Architectural data structure used as a parameter to EPC management instructions
    - Linear Address
    - Effective address of page (virtual address)
    - SECINFO (Security Information)
    - SECS (SGX Enclave Control Structure)

SGX Data Structure Introduction

- Security Information (SECINFO)
  - Holds meta-data about enclave page information such as:
    - Read/Write/Execute (R/W/X) permissions
    - Page type (SECS, TCS, REG, or VA)

SGX Data Structure Introduction

- Paging Crypto MetaData (PCMD)
  - Tracks crypto meta-data associated with a paged-out page
  - Combines information provided by PAGEINFO to verify, decrypt, and reload a paged-out EPC page
  - EWB instruction that invalidates an EPC page and writes out to main memory will also write out the message authentication code (MAC)
  - ELDB/ELDU will read the MAC of a previously invalidated page prior to loading back in and decide whether to block or unblock the page depending on the MAC, version number check with version array, and the page encrypted contents
  - Contains enclave ID, SECINFO, and MAC

SGX Data Structure Introduction

- Version Array (VA)
  - Used mainly to record versions of evicted EPC pages currently in main memory
  - A special EPC page type
    - Contains 512 slots that has an 8-byte version number for each page evicted from EPC
  - When page is evicted, software will assign an empty slot in a VA page to the page and the slot receives the unique number of the page being evicted
  - When page is reloaded, a VA slot must hold the version of the page and this slot is cleared if successfully reloaded
  - VA pages can also be evicted in the same process as a normal EPC page and is stored in another VA

SGX Data Structure Introduction

- SGX Enclave Control Structure (SECS)
  - A special page type that represents each enclave
  - Contains meta-data such as hash, ID, size
  - Not accessible to any secure or non-secure code and only the processor can access
  - Immutable once instantiated

SGX Data Structure Introduction

- Thread Control Structure (TCS)
  - Each executing thread in the enclave is associated with a TCS
  - Contains information such as entry point of thread and pointer to State Save Area (SSA)
  - Only accessible by processor and is immutable

SGX Data Structure Introduction

- State Save Area (SSA)
  - Associated with TCS
  - Used to save a processor’s state during exceptions or interrupt handling
  - Written when exiting and read when resuming

SGX Data Structure Introduction

- Key Request (KEYREQUEST)
  - Structure used as the input parameter for the EGETKEY (create cryptographic key) instruction
  - Allows to choose which key to get
  - Additional parameters might also be needed for key derivation

SGX Data Structure Introduction

- Report Target Info (TARGETINFO)
  - Structure used as the input parameter to the EREPORT (create a cryptographic report)
  - Used to identify which enclave (hash and attributes) will be able to verify the REPORT generated by the CPU (from EREPORT)
    - Contains hash and attributes of target enclave

SGX Data Structure Introduction

- Report (REPORT)
  - Structure that is the output of the EREPORT instruction
    - Attributes of the enclave
    - Hash of the enclave
    - Signer of the enclave
    - Set of data used for communication between enclave and target enclave
    - A cryptographic message passing authentication on the report using the report key

SGX Data Structure Introduction

- EINIT Token Structure (EINITTOKEN)
  - EINIT token used by EINIT to verify that the enclave is permitted to launch during enclave setup
  - Contains attributes, hash, and signer of enclave, and etc
  - Authenticated with CMAC on EINITTOKEN using launch key

SGX Data Structure Introduction

- Enclave Signature Structure (SIGSTRUCT)
  - Each enclave is associated with a SIGSTRUCT
    - Signed by the author
  - Contains information such as:
    - Enclave measure
    - Signer public key
    - Version number (ISV, reflecting the security level)
    - Product identifier (ISVPRODID, to differentiate between enclaves by the same author)
    - ENCLAVEHASH as SHA256
    - 3072-bit integers (MODULUS, SIGNATURE, Q1, Q2)
  - Allows enclave that has not been modified to be re-signed with a different key

SGX Application Design

- Enclave:
  - Application is divided into a trusted and untrusted component based on identifying what assets to be protected
    - Application can have more than one trusted component/enclave
    - The actual partition of the application needs to be contemplated to ensure smallest TCB while balancing overhead
- Attestation:
  - Process of demonstrating that a specific enclave was established on an approved platform and has the required security measures
    - Local Attestation (Intra-machine)
    - Remote Attestation (Inter-machine)

SGX Application Design

- Sealing:
  - Process of encrypting data when writing to untrusted memory or storage to ensure contents are not revealed
  - Data can be read by enclave again later and decrypted
  - Encryption keys for sealing are derived internally on demand and are not exposed to the enclave
    - Two sealing methods:
      - Enclave Identity: Key protocol that produces key unique to exact enclave sealing
      - Sealing Identity: Key protocol that produces key based on identity of enclave's sealing authority and multiple enclaves from the same sealing identity can derive the same key

Enclave Introduction

- SGX allows the creation of private regions of memory called enclaves that protect data/code within the region from higher privileged processes such as:
  - BIOS, Hypervisor, Operating System
- Developer can partition an application with more parts of the application in the enclave but must consider the balance between stability and security
  - More data/code in enclave will increase size of TCB and make attack surface larger
  - Smaller code/terse code lower chances of defects in software

Physical Memory Organization

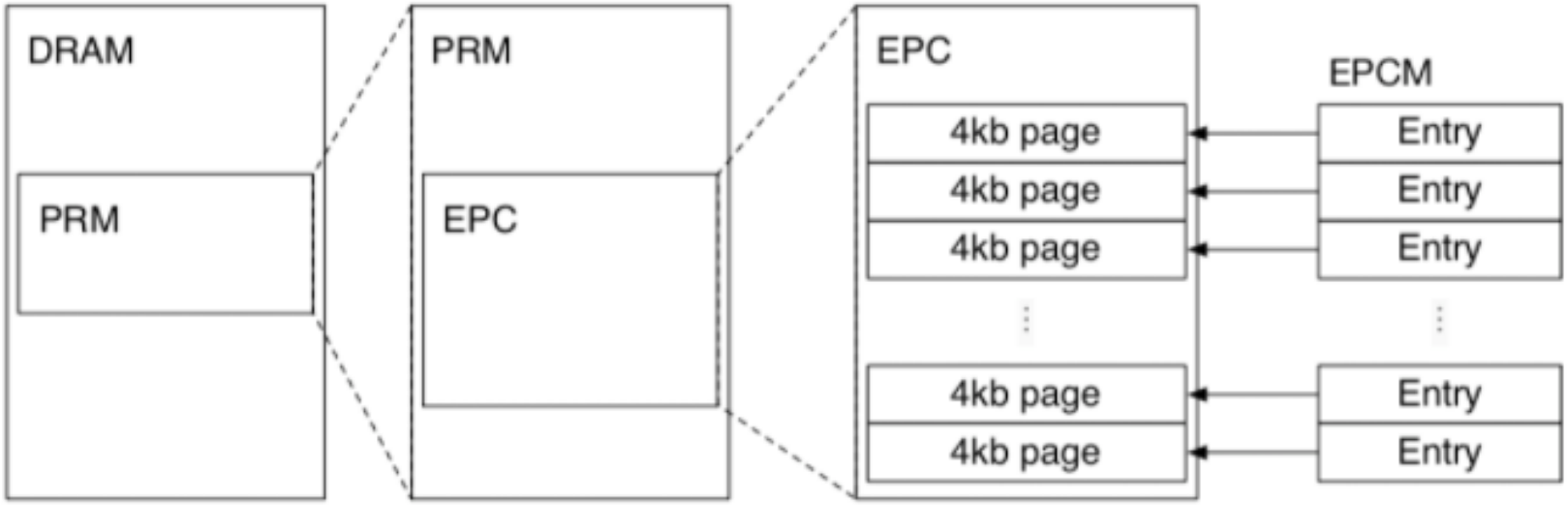
- Enclave data and code is stored in Processor Reserved Memory (PRM) and cannot be accessed by even higher privileged rings
  - PRM is a subset of DRAM
- The contents of enclaves and associated data structures are stored in the Enclave Page Cache (EPC)
  - EPC is a subset of PRM
  - 1 EPC with max size of 128 MB (93MB used by application and rest by metadata)
- EPC is split into 4 KB size pages that can be later assigned to different enclaves to support multiple enclaves for different processes
  - Page allocation is still managed by the same system software that manages memory for the rest of the computer

Physical Memory Organization

- The Enclave Page Cache Map (EPCM) is a secure structure (array-like) used by the processor to track contents of the EPC
  - Holds one entry for each page that is currently loaded into the EPC
  - Not accessible by software
  - Layout of EPCM fields are implementation specific

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

Physical Memory Organization



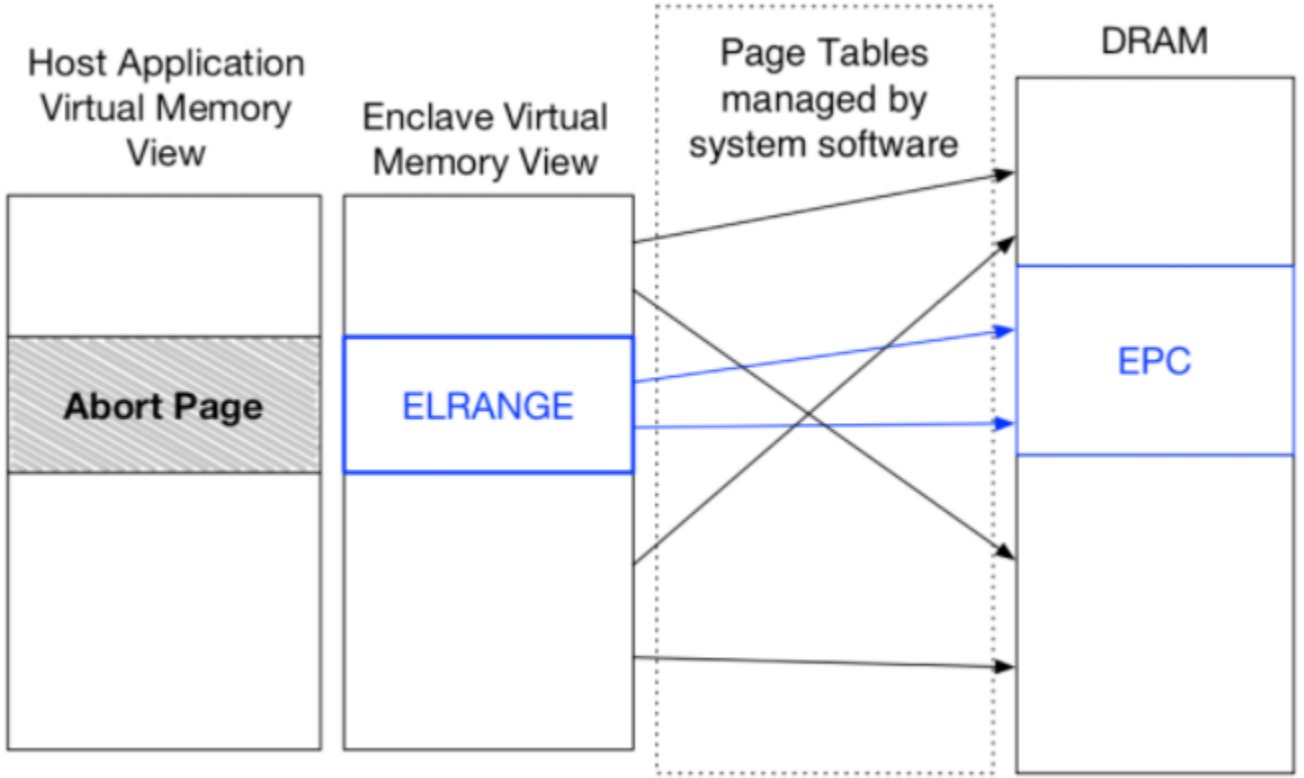
Physical Memory Organization

- SGX Enclave Control Structures (SECS) represents one enclave each and is stored in a dedicated EPC page with page type PT_SECS
  - Contains meta-data such as hash and size
  - Not mapped to enclave's address space as should not have direct access to the control structure (not accessible to secure or non-secure code)
    - Initialized and manipulated by hardware (processor) only
  - Synonymous with the identity of an enclave as the SECS is the first page allocated when initializing an enclave
  - EPCM entry field which identifies which enclave owns which EPC page points to the SECS for reference
  - System software uses the virtual address of SECS to identify enclaves when running SGX instructions and creates page tables that point to the SECS of enclaves but cannot access the SECS as in PRM

Memory Layout of Enclave (Virtual Memory)

- Enclave Linear Address Range (ELRANGE) was designed to ensure that for a given enclave, it would have a virtual address range that will map the code and sensitive data stored in the EPC pages
  - Virtual address space outside of ELRANGE is used to map to access non-EPC memory but is not guaranteed as system can be compromised
    - Only ELRANGE is setup to always obey virtual memory abstraction and access EPC
  - ELRANGE is specified in the SECS (size and base address)
    - Size in power of 2 as must conform with page size and must be within PRM range

Memory Layout of Enclave (Virtual Memory)



SGX Enclave Attributes

- An enclave's attributes are recorded in the attributes field of an enclave's SECS
  - DEBUG – enabling read/write of an enclave's memory in debug mode
  - XFRM – defines extended features request mask to specify architectural extensions
  - MODE64BIT – set true for enclaves that use 64-bit Intel architecture

Field	Bits	Description
DEBUG	1	Opts into enclave debugging features.
XFRM	64	The value of XCR0 (§ 2.6) while this enclave's code is executed.
MODE64BIT	1	Set for 64-bit enclaves.

Address Translation for SGX Enclaves

- System software such as the operating system or hypervisor is still in control of page tables, extended page tables, and each enclave's code uses the same address translation process as its application host counterpart
  - Since SGX is based on trusting the CPU as the only hardware component, since translation is still managed by untrusted system software, it opens up avenues of attack (address translation attacks)
- SGX's active memory mapping attack defense mechanisms ensure that each EPC page can only be mapped at a specific virtual address
  - Recorded in EPCM entry for the page in the address field

Address Translation for SGX Enclaves

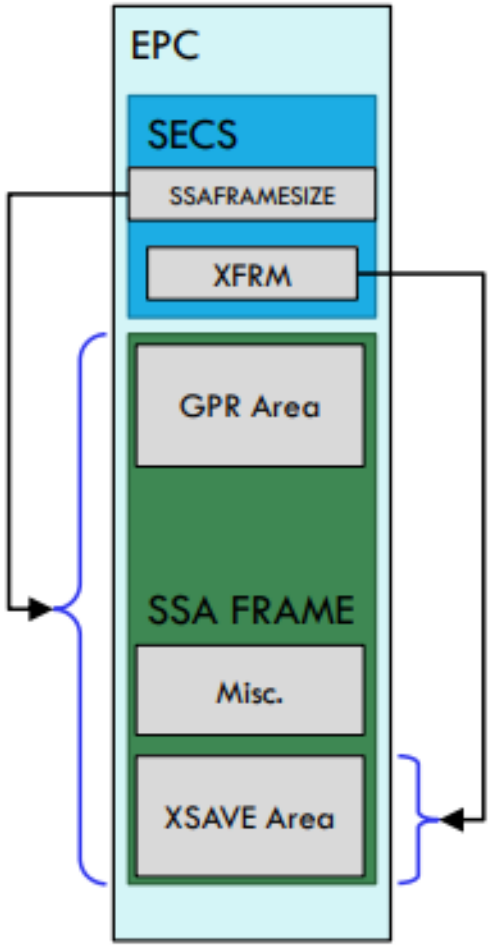
- Address translation will be checked again when physical memory access is requested
  - CPU checks that the given virtual address matches the expected virtual address recorded in the EPCM of the page
- Passive memory mapping attack and fault injection attacks are protected against by ensuring that access permissions are setup when a page is allocated (R/W/X)
  - Developer would include the memory layout alongside the enclave to let system software know the expected virtual memory address and access permissions to guarantee consistency
- ELRANGE ensures that virtual memory for enclave maps to EPC and prevents translation attacks

Thread Control Structure

- Each thread process that uses SGX has its own Thread Control Structure (TCS) that contains assets such as the entry point of thread or pointer to the State Save Area (SSA)
- TCS is stored in a dedicated EPC page similarly to SECS and has the entry of PT_TCS on the EPCM
- TCS cannot be accessed directly (similar restrictions as with SECS)
- Architectural fields in the TCS belong to mainly the architectural part of the structure and is assumed the same throughout all SGX supported processors while the rest are undocumented
  - The fields also lay out the context switches performed by the logical processor when transitioning between enclave and non-enclave code

State Save Area (SSA)

- Enclave specific executing code that was interrupted (requiring privilege level switch) will be temporarily saved in the State Save Area (SSA) in the PRM region of DRAM
  - To ensure that no information in execution can be revealed to the untrusted system software
- SSA start at the beginning of an EPC page and will use up the number of EPC pages specified in the SSAFRAMESIZE field in the SECS
- General Purpose Registers (GPRs) and special feature registers are stored in separate regions in SSA



SGX Instructions

- There are a total of 18 different instructions in the SGX instruction set:
  - 13 Supervisor Instructions
  - 5 User Instructions

SGX Instructions - Supervisor

Supervisor Instruction	Description
ENCLS[EADD]	Add a page
ENCLS[EBLOCK]	Block an EPC page
ENCLS[ECREATE]	Create an enclave
ENCLS[EDBGRD]	Read data by debugger
ENCLS[EDBGWR]	Write data by debugger
ENCLS[EEXTEND]	Extend EPC page measurement
ENCLS[EINIT]	Initialize an enclave
ENCLS[ELDB]	Load an EPC page as blocked
ENCLS[ELDU]	Load an EPC page as unblocked
ENCLS[EPA]	Add version array
ENCLS[EREMOVE]	Remove a page from EPC
ENCLS[ETRACK]	Activate EBLOCK checks
ENCLS[EWB]	Write back/invalidate an EPC page

SGX Instructions - User

User Instruction	Description
ENCLU[EENTER]	Enter an Enclave
ENCLU[EEXIT]	Exit an Enclave
ENCLU[EGETKEY]	Create a cryptographic key
ENCLU[EREPORT]	Create a cryptographic report
ENCLU[ERESUME]	Re-enter an Enclave

Enclave Lifecycle

- ECREATE – Initializes enclave by turning an EPC page into the SECS for the enclave
  - ECREATE will copy the SECS structure from outside the EPC into a SECS page inside the EPC
  - Software will set the source structure fields:
    - BASEADDR
    - SIZE
    - ATTRIBUTES
  - ECREATE will also validate if information is valid and will result in page fault or general protection fault if information is not valid
  - Other causes of page fault:
    - SECS target page in use already, outside EPC, addresses not aligned, unused PAGEINFO (structure) are not zero

Enclave Lifecycle

- EADD – instructions to load initial code and data into the enclave from application
  - Used to create both TCS pages and regular pages
  - Function will copy source page from non-enclave memory into EPC and associate the EPC page with a SECS page in the EPC
    - Linear address (virtual address) and security attributes will be stored in the EPCM
  - Input data read from Page Info (PAGEINFO) structure

TCS: thread controll structure

Enclave Lifecycle

- EEXTEND – updates the enclaves measurement used in the software attestation process
  - Updates MRENCLAVE measurement register of a SECS with the measurement
- EINIT – concludes the MRENCLAVE measurement and enclave is ready to start user code execution using the EENTER instruction
  - INIT attribute will be set to true and allows the enclave to start executing in ring 3
  - Measurement compared with ISV's signed SIGSTRUCT data structure
  - Sealing identity is recorded in SECS of the enclave (sealing authority, product ID, SVN)
  - After INIT is set true, EADD can no longer be invoked on that specific enclave anymore
    - System software must load all pages that make up the enclave's initial state before executing the EINIT instruction

ISV: Independent Software Vendor

Enclave Lifecycle

- EENTER (Synchronous Enclave Entry)
  - Thread Control Structure (TCS) is checked to ensure not busy, and contents of the Translation Lookaside Buffer (TLB) is flushed
  - Control is transferred from outside the enclave to a pre-determined location inside the enclave
  - Operation mode changed to enclave mode
  - RSP/RBP saved for later restore of operation upon enclave asynchronous exit
  - Save XCR0 and replace it with enclave XFRM value

Enclave Lifecycle

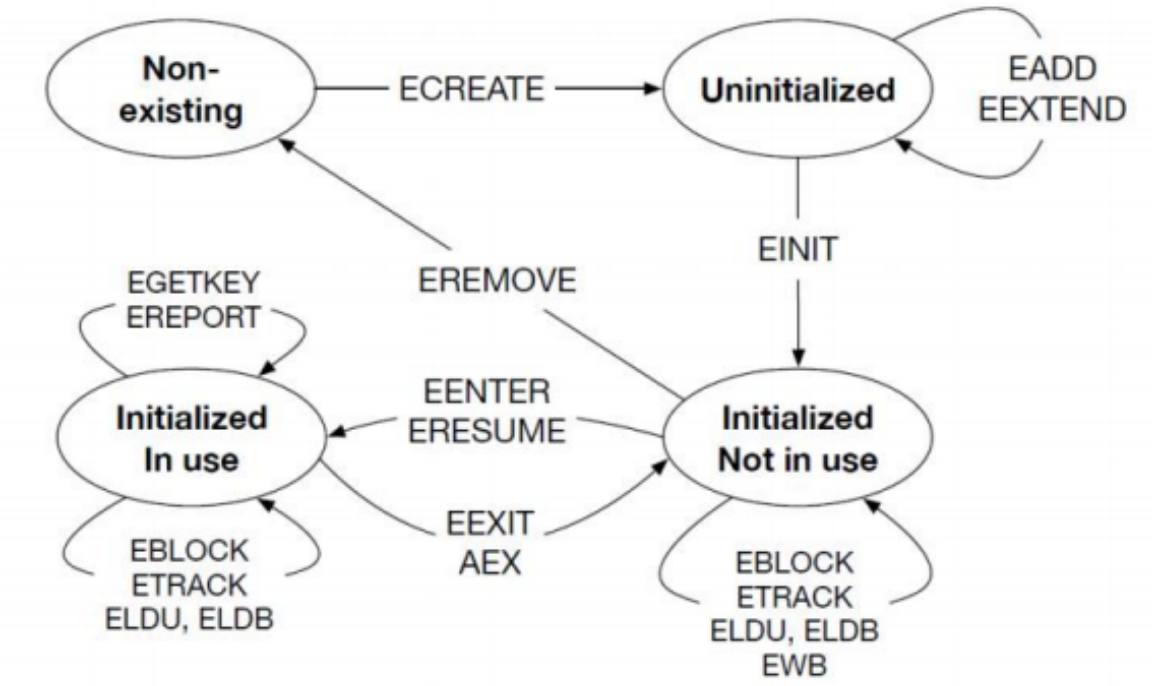
- EEXIT (Synchronous Enclave Exit)
  - Enclave mode and TLB entries for enclave addresses are cleared
  - Control is transferred from inside the enclave to another location outside of the enclave
    - TCS is marked as not busy
  - The task of clearing the register state is the responsibility of the enclave designer and not a built-in functionality of Intel SGX

Enclave Lifecycle

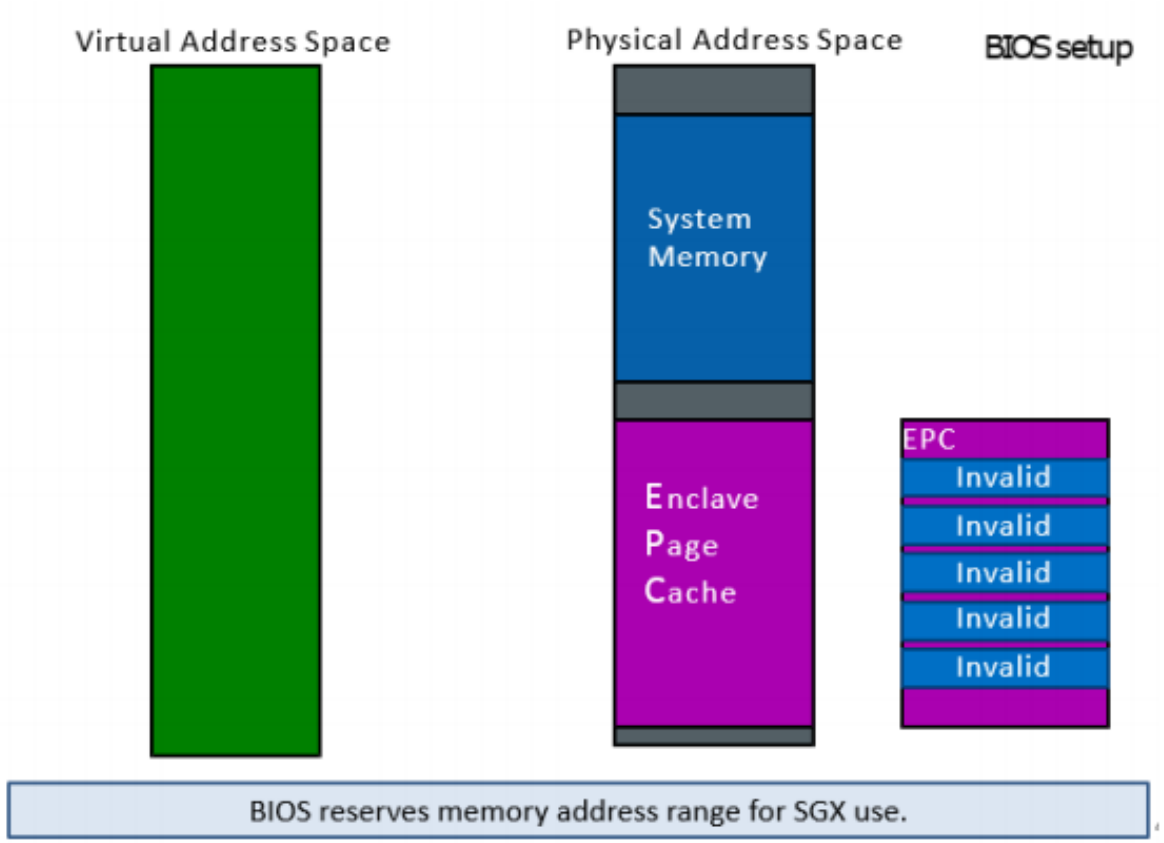
- EREMOVE (Teardown)
  - Deallocates/removes a 4 KB page permanently from the EPC
    - Page cannot be removed until no thread is executing code within the enclave
    - SECS page cannot be removed until all regular pages of the enclave are removed
    - SECS page is removed at the very end, and this effectively destroys the enclave

Enclave Lifecycle

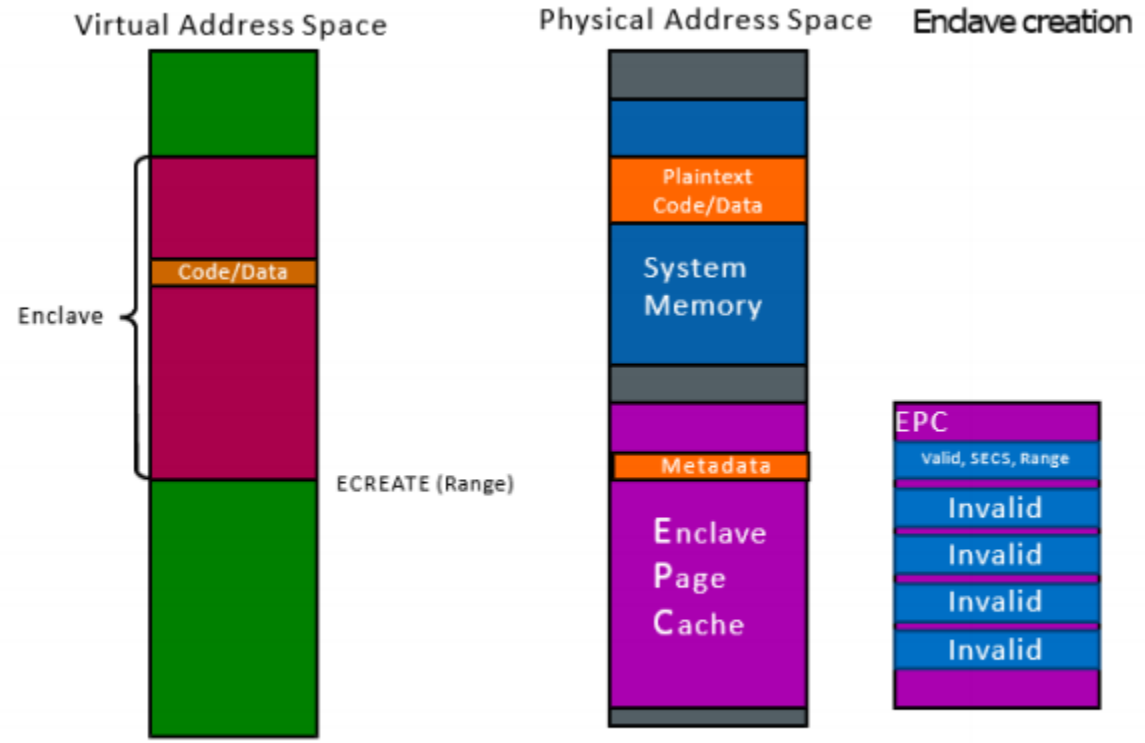
1. Creation (ECREATE)
2. Loading (EADD, EEXTEND)
3. Initialization (EINIT)
4. Enter/Exit the Enclave (EENTER/EEXIT)
5. Teardown (EREMOVE)



Enclave Lifecycle – BIOS Setup

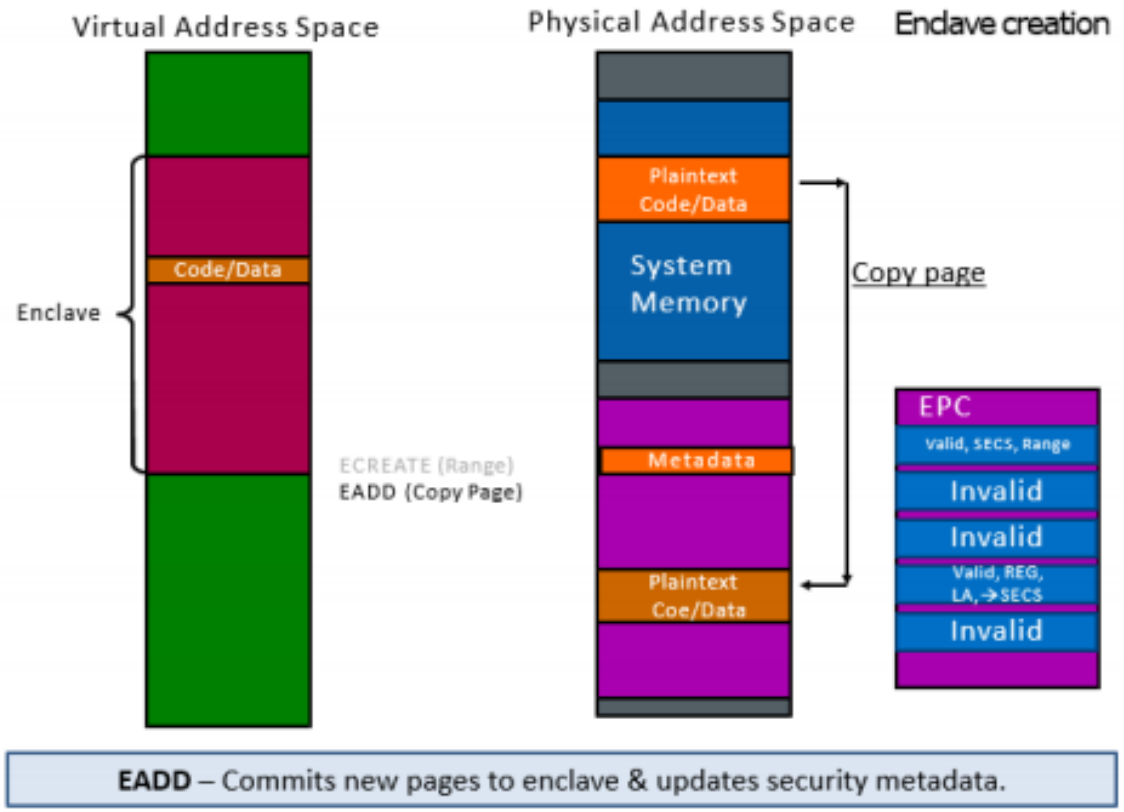


Enclave Lifecycle – ECREATE



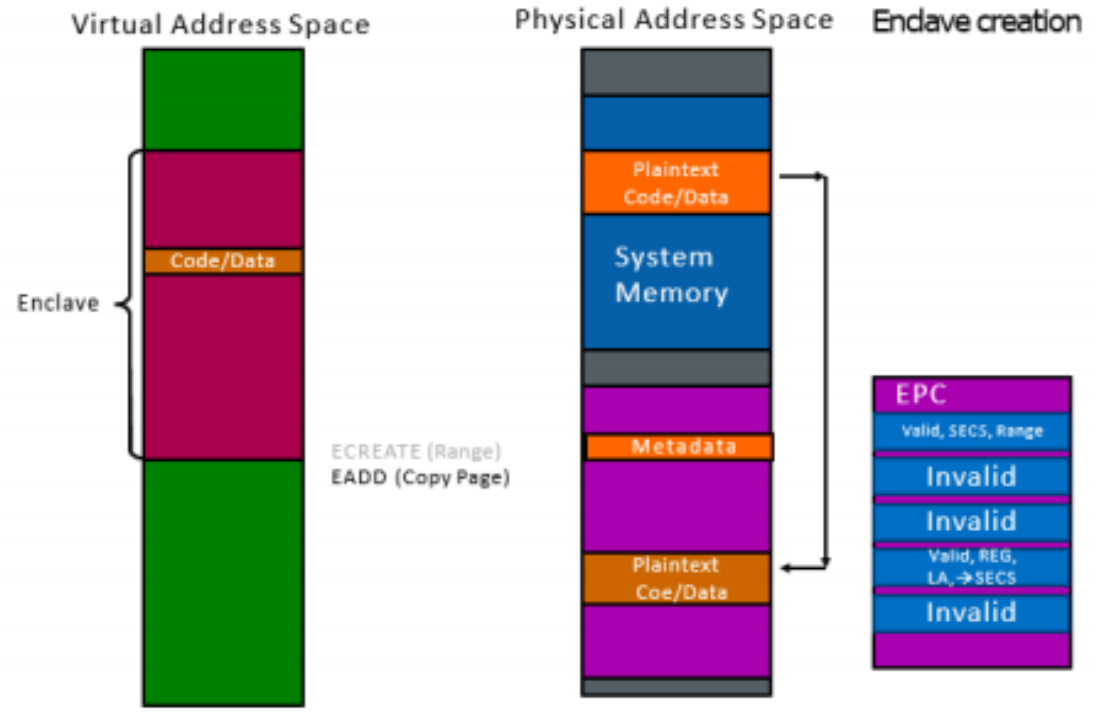
ECREATE - Stores enclave attributes (mode of operation, debug, etc.) in metadata.

Enclave Lifecycle – EADD



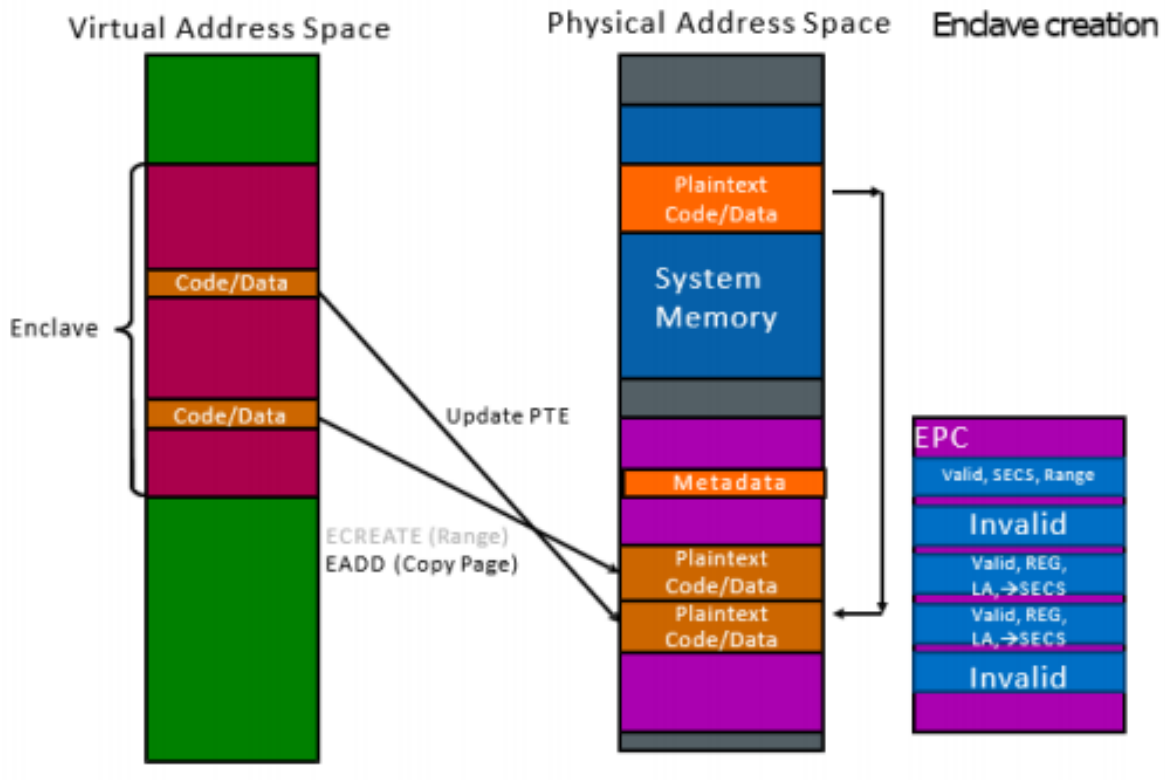
EADD – Commits new pages to enclave & updates security metadata.

Enclave Lifecycle – EADD

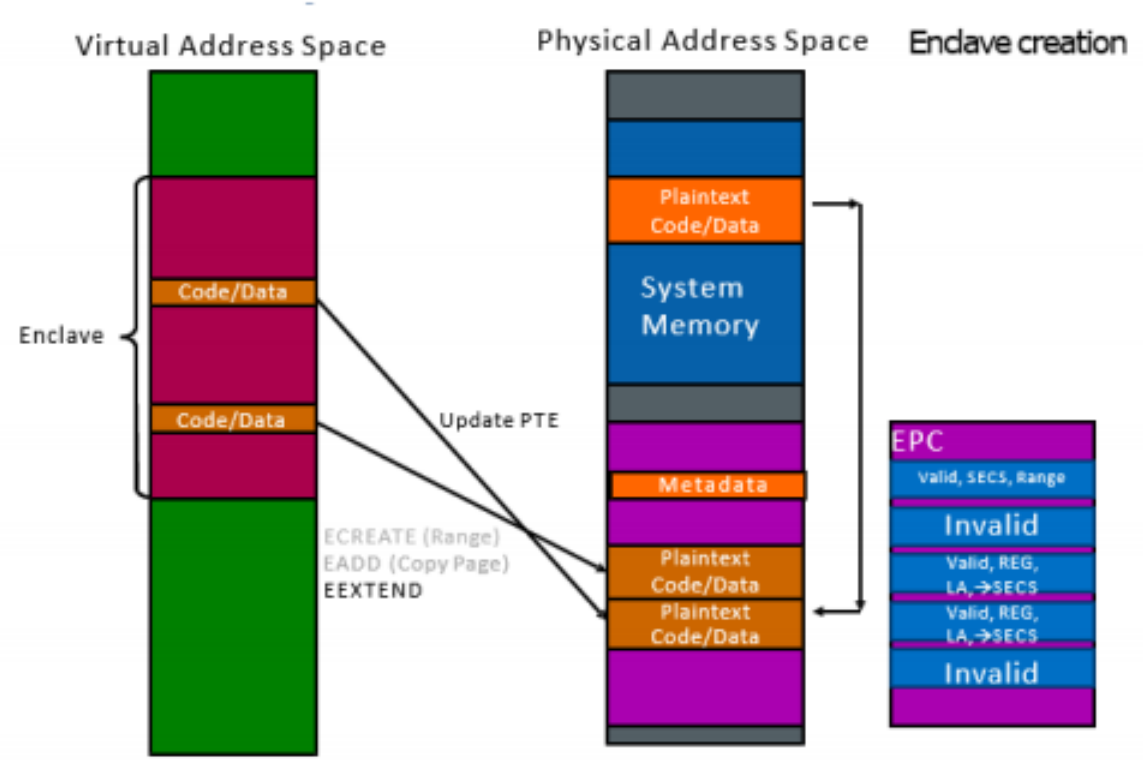


EADD – Commits code, data or SGX control structure page types.

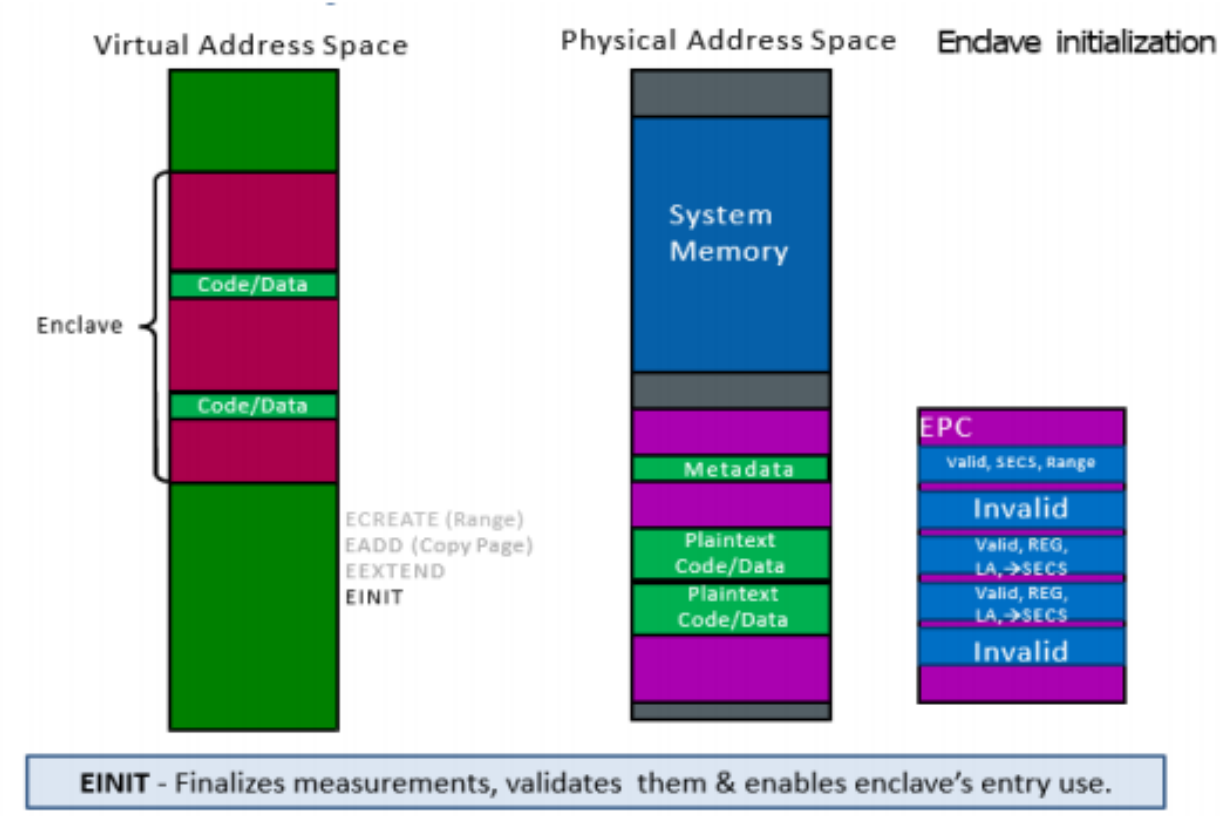
Enclave Lifecycle – EADD



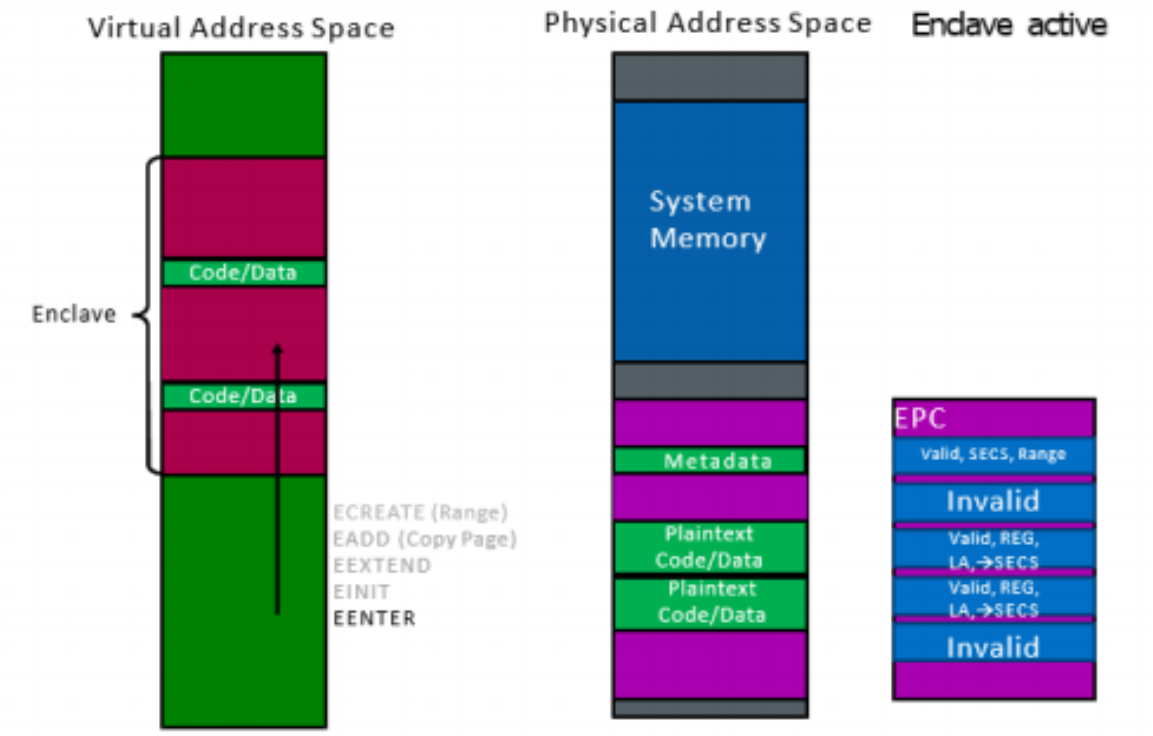
Enclave Lifecycle – EEXTEND



Enclave Lifecycle – EINIT

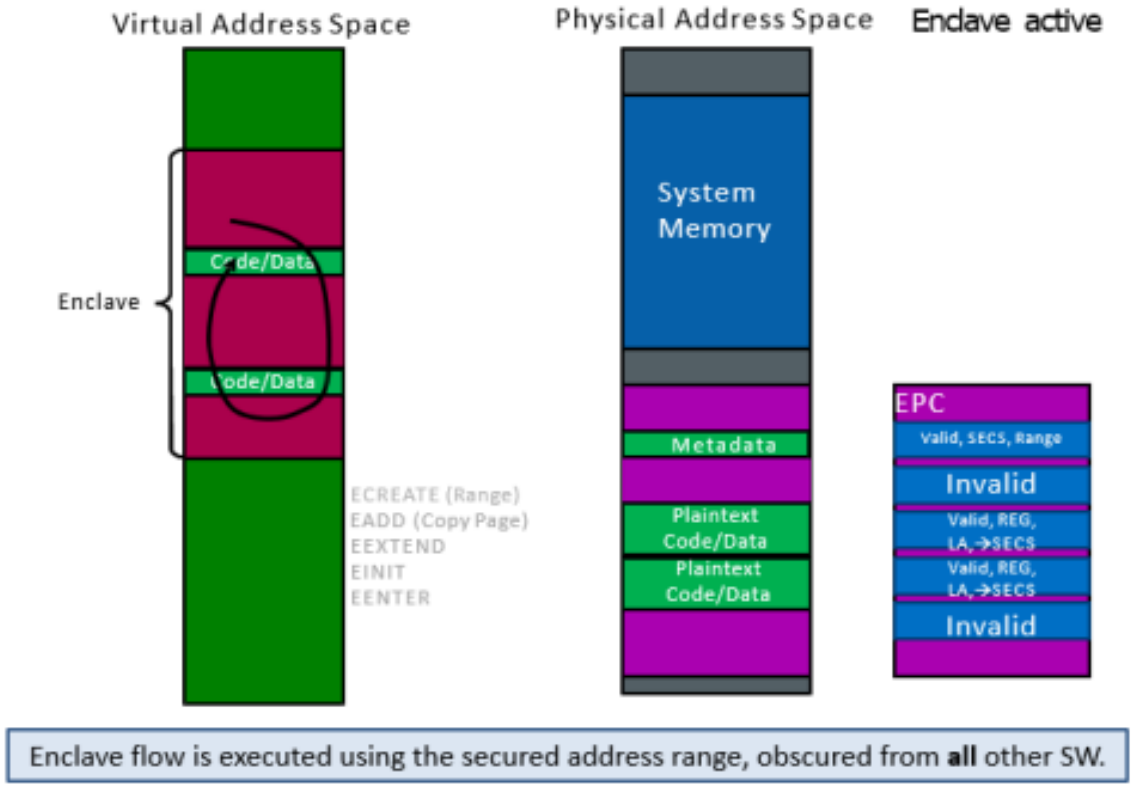


Enclave Lifecycle – EENTER



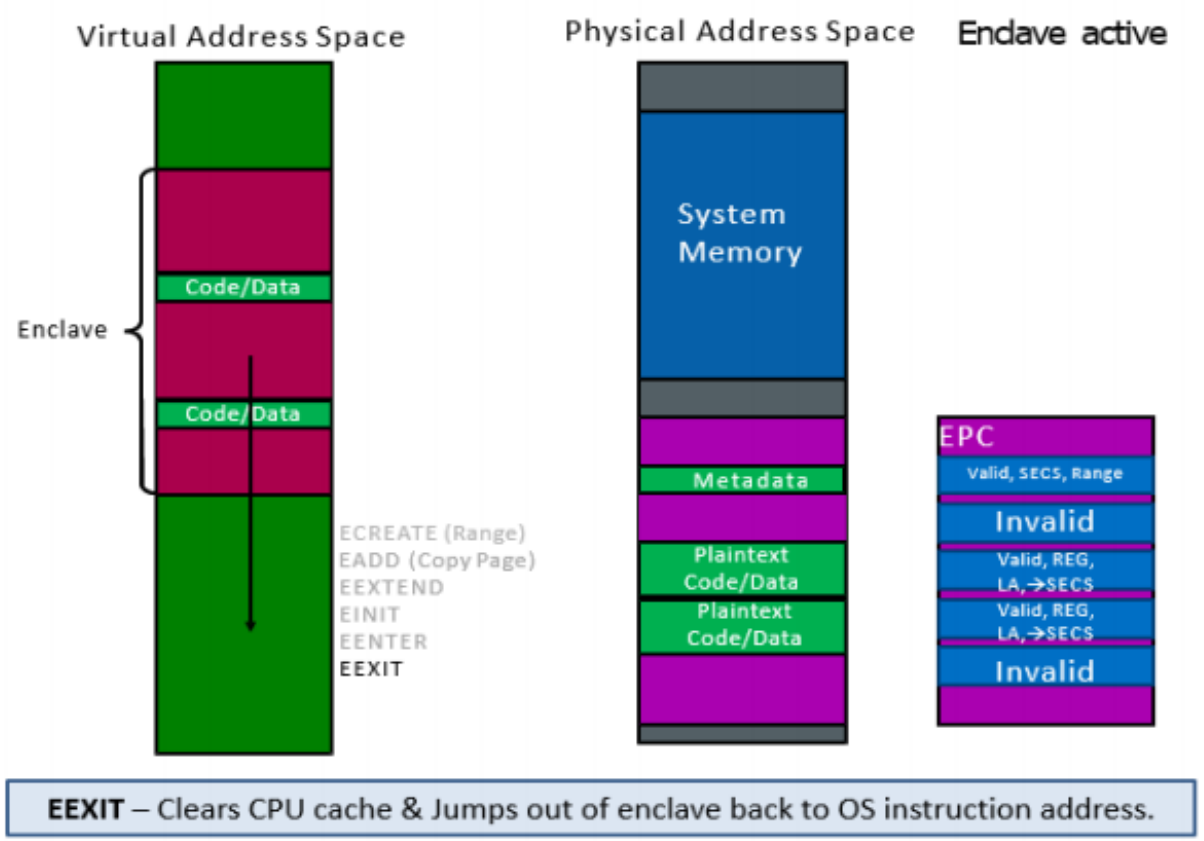
EENTER - Verifies enclave entry & sets CPU operation mode to “enclave mode”.

Enclave Lifecycle – EENTER



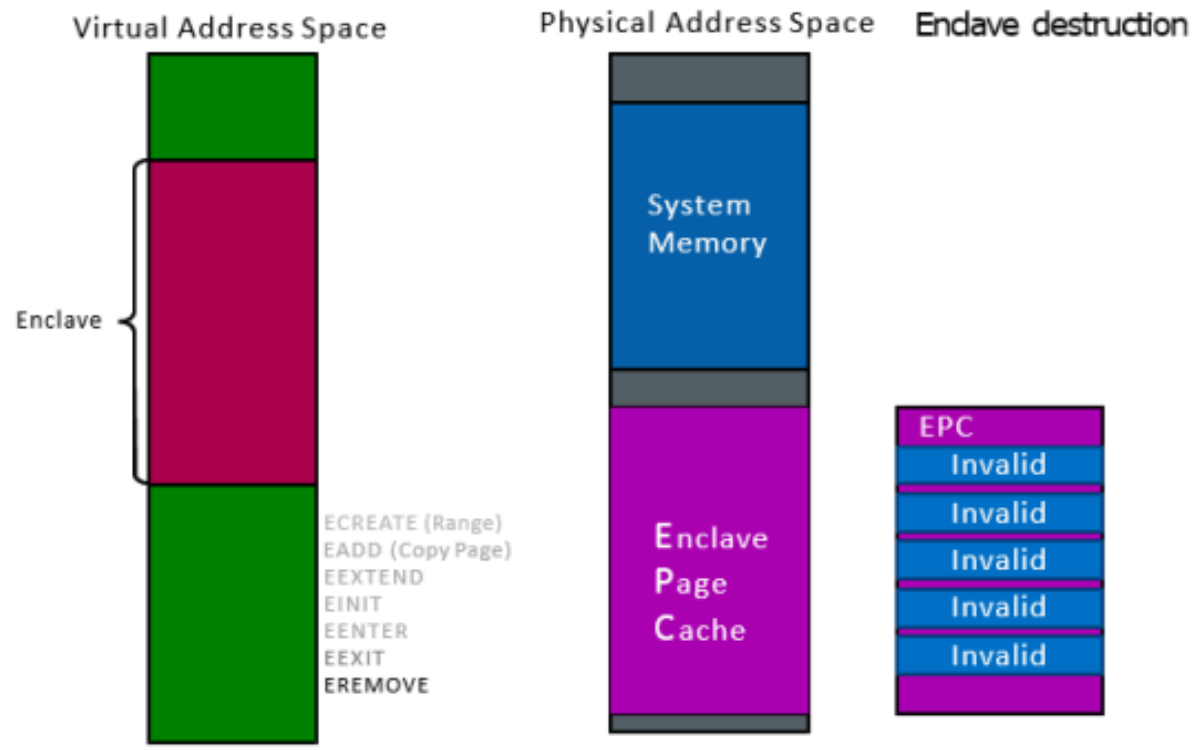
Enclave flow is executed using the secured address range, obscured from **all** other SW.

Enclave Lifecycle – EEXIT



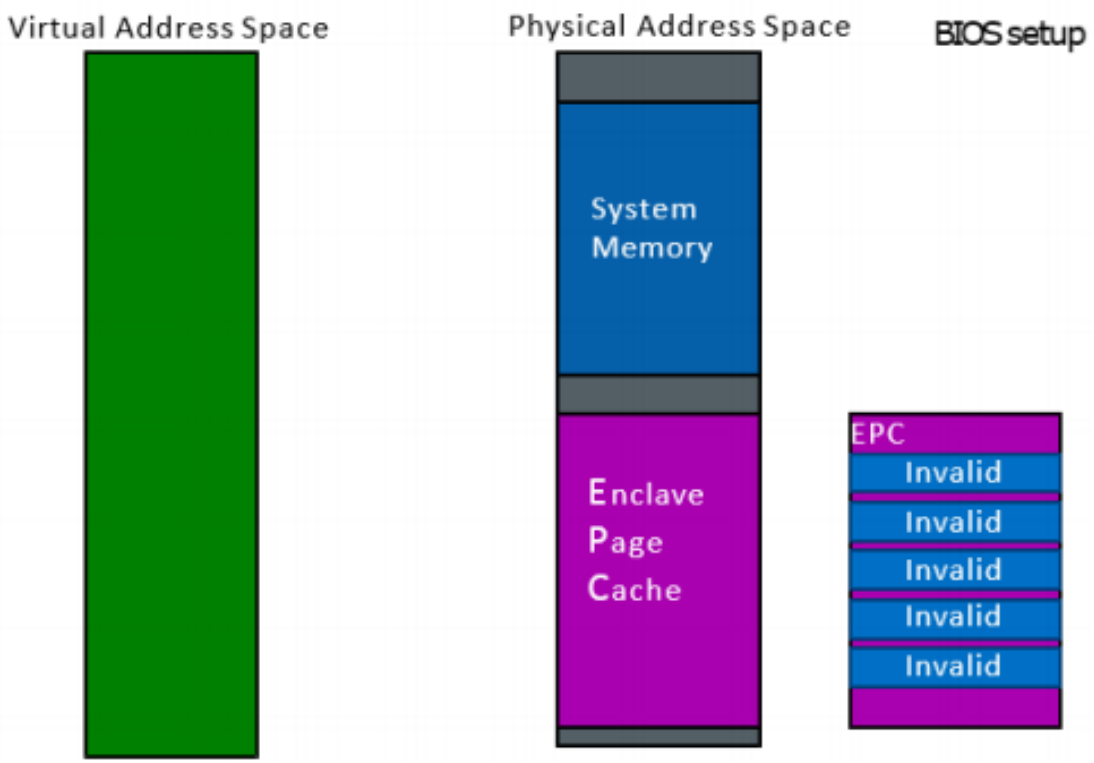
EEXIT – Clears CPU cache & Jumps out of enclave back to OS instruction address.

Enclave Lifecycle – EREMOVE



EREMOVE – Clears enclave's trusted virtual address range reserved by ECREATE.

Enclave Lifecycle – EREMOVE



Architectural Enclaves

- Launch Enclave (LE)
- Provisioning Enclave (PvE)
- Provisioning Certificate Enclave (PcE)
- Quoting Enclave (QE)
- Platform Service Enclave (PSE)

Architectural Enclaves

- Launch Enclave (LE)
  - Responsible for assigning EINITTOKEN to other enclaves wishing to launch on the same platform
  - Checks the signature and identity of the enclave to see if valid
  - Generates tokens using the Launch Key
    - Only the Launch Enclave can retrieve the Launch Key

Architectural Enclaves

- Provisioning Enclave (PvE)
  - Responsible for retrieving the Attestation Key by communicating with Intel Provisioning Service (IPS) servers
  - Proves the authenticity of the platform via the certificate provided by the Provisioning Certificate Enclave (PcE)

Architectural Enclaves

- Provisioning Certificate Enclave (PcE)
  - Responsible for signing the processor certificate destined to go to the Provisioning Enclave (PvE)
  - Uses the Provisioning Key to provision the certificate
  - Both PvE and PcE are implemented in SGX as a single enclave

Architectural Enclaves

- Quoting Enclave (QE)
  - Responsible for providing trust in both the identity of the enclave and the environment in which it executes in during the remote attestation process
  - Decrypts the Attestation Key from the PvE and uses this key to transform a REPORT structure (locally verifiable structure) into a QUOTE structure (remotely verifiable structure)

Architectural Enclaves

- Platform Service Enclaves (PSE)
  - Enclaves that offer other enclaves services such as
    - Monotonous counters
    - Trusted time
  - Uses the Management Engine (ME)

SGX Key Directory

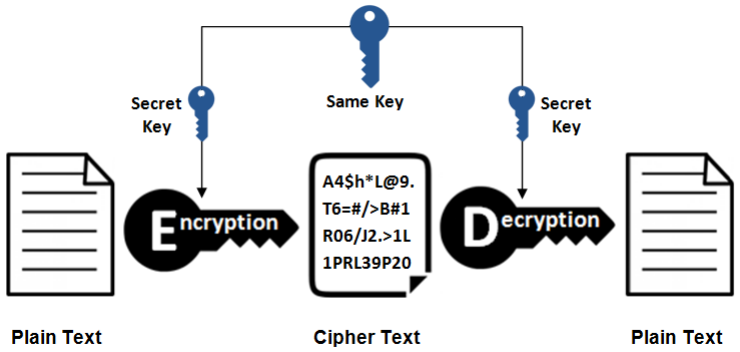
- Each Intel CPU that supports SGX contains two root keys that are stored on e-fuses:
  - Root Provisioning Key (RPK)
  - Root Seal Key (RSK)
- Root keys stored in the e-fuses are extractable, but Intel CPU design makes it highly difficult and costly to attempt
  - Only one encrypted version of the keys stored on the e-fuses
- Physical Unclonable Function (PUF) is used to store symmetric keys used to decipher the other keys during processor execution

Sidenote – Symmetric vs Asymmetric Encryption

- Cryptography is the method of using mathematical principles both storing and transmitting data in a specific form in which only those intended to read/process can access it in raw form
- Key concepts in cryptography:
  - Encryption
    - The process of locking up information using cryptography so that to read the information it needs to be decrypted
  - Decryption
    - Process of unlocking encrypted information using cryptographic techniques
  - Key
    - The secret password used to encrypt and decrypt information

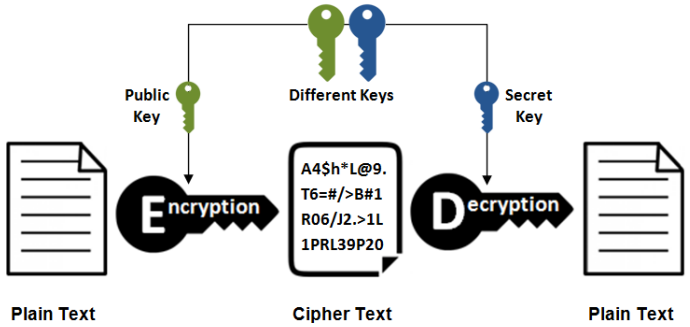
Sidenote – Symmetric vs Asymmetric Encryption

- Symmetric Encryption
  - The same secret key is used for both encryption and decryption of information
    - Can be either a number, work, or random string of letters
  - The key must be held by both the sender and recipient to share information



Sidenote – Symmetric vs Asymmetric Encryption

- Asymmetric Encryption
  - There are two sets of keys
    - Public Key: Made available to anyone and can be shared (anyone can send a message to the receiver)
    - Private Key: Only the specific receiver (original sender) holds this key
  - Information encrypted by the private key can be decrypted by the specific public key while information encrypted by the public key can only be decrypted by the private key



Sidenote – Symmetric vs Asymmetric Encryption

- Asymmetric Encryption in Digital Certificates
  - A method of discovering public keys such is through using digital certificates in a client-server model of communication
    - A certificate is a package of information that identifies a user and a server and contains key information such as the organization's name, the issuer of the certificate, user's email, country, and the user's public key
  - When secure encrypted communication is required, both parties can send a query over the network to the other party, which in turn, will send back a copy of the certificate
    - The other party's public key can be extracted from the certificate and can also be used to uniquely identify the holder

Root Keys

- Root Provisioning Key (RPK)
  - The first key created by Intel during the manufacturing process and is generated randomly on a dedicated Hardware Security Module (HSM) located inside a facility called the Intel Key Generation Facility (iKGF)
  - Intel maintains a database containing all keys produced by the HSM
  - The generated RPK are later sent to multiple production facilities to be embedded inside the processor’s e-fuse

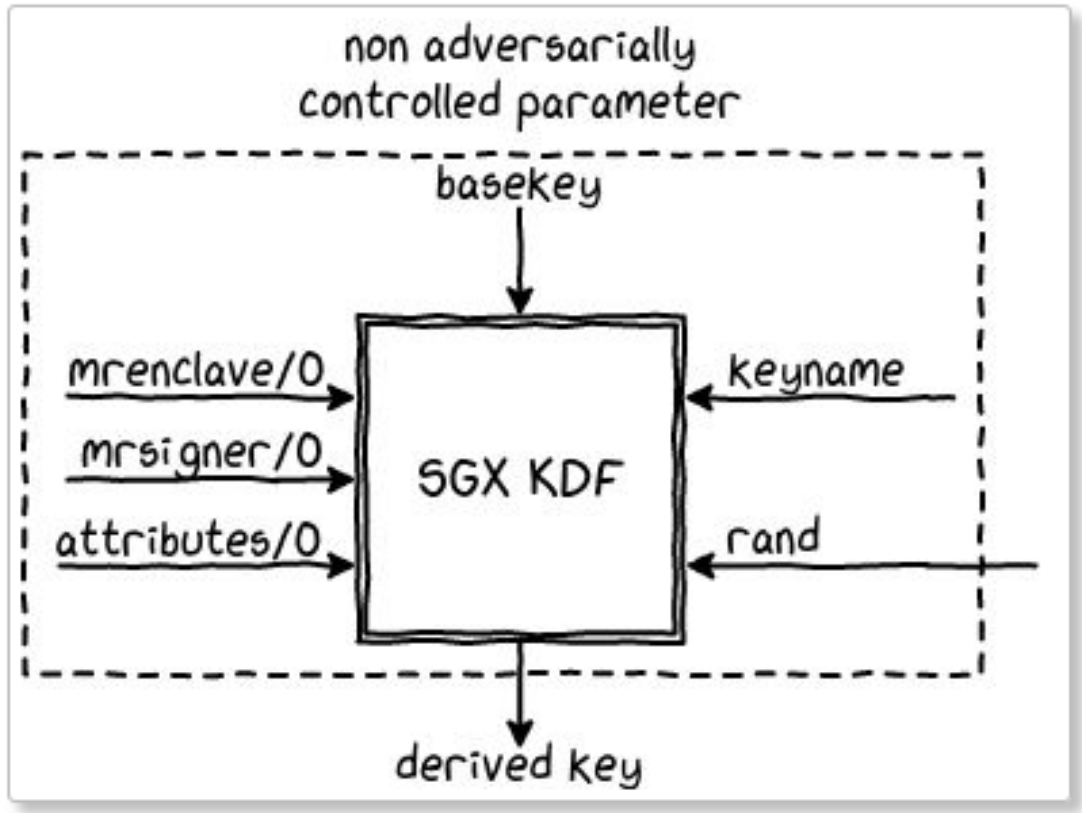
Root Keys

- Root Sealing Key (RSK)
  - The second key located inside the e-fuse on the processor
  - Similarly, to the RPK, it is also generated to be different between each unit produced
  - Intel erases all traces of the RSK from their production chain to ensure that only the platform knows and has a unique key

Key Derivation

- Enclaves do not have access to root keys, but they can access keys that are derived from root keys
- Through the key derivation function (KDF), the enclave author can specify the key derivation policy
  - Can include the following parameters:
    - MRENCLAVE
    - MRSIGNER
    - Attributes of the enclave
  - To add entropy, that is user specific, a value called the Owner Epoch is used as a parameter during the derivation
    - Configured at boot-time by a derivation of a password and saved during each power cycle in non-volatile memory
    - Must stay the same for an enclave to be able to retrieve the same key and chances when owner changes to prevent access to previous personal information

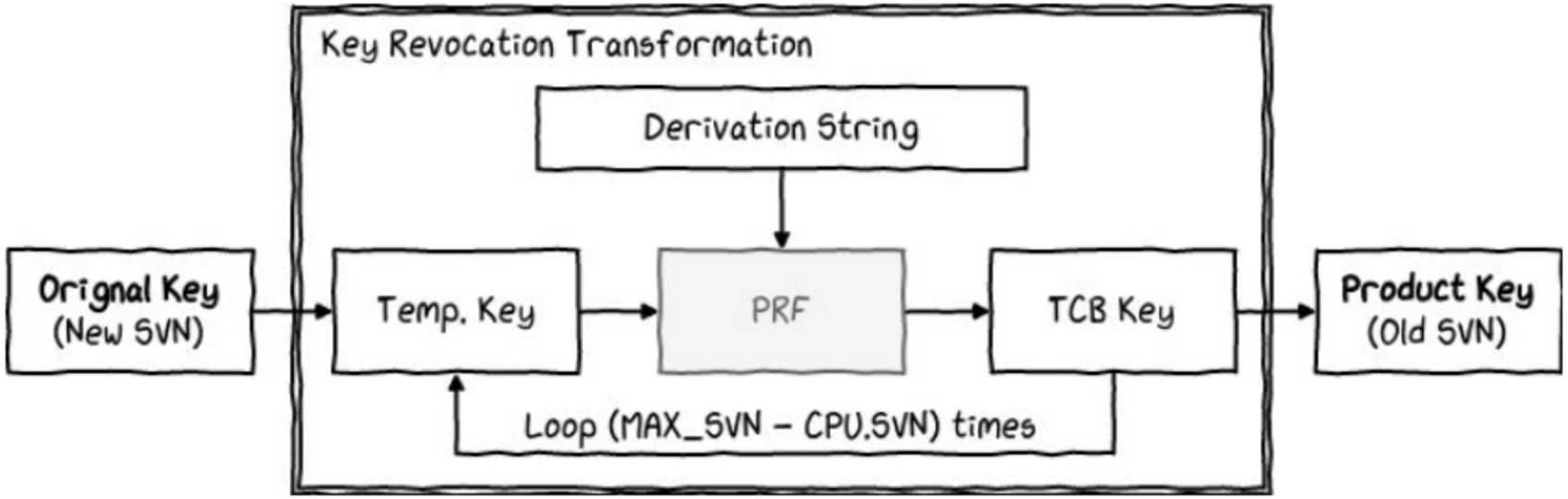
Key Derivation



Key Derivation

- SGX infrastructure supports TCB updates of hardware and software components
  - Each component has a Security Version Number (SVN) which is incremented after each security update
  - Each new SVN leads to a new Sealing Key
- Newer TCB can access the Sealing Key of older TCB for data migration while older TCB cannot access the keys of newer TCB

Key Derivation



Derived Keys

- There are five main keys derived from the two root keys:
  - Provisioning Key
  - Provisioning Seal Key
  - Launch Key
  - Seal Key
  - Report Key

Derived Keys

- Provisioning Key
  - Derived from the RPK and used as the root of trust between Intel Provisioning Service and the processor
  - Only enclaves signed by Intel (MRSIGNER) will be given access to this key by the Launch Enclave to prevent non-SGX processors access to a group of legitimate SGX processors
- Provisioning Seal Key
  - Derived from both the RPK and RSK
  - During enrolment of a processor in the group, the private key of each platform (EPID) is encrypted with this key and sent to Intel Attestation Services

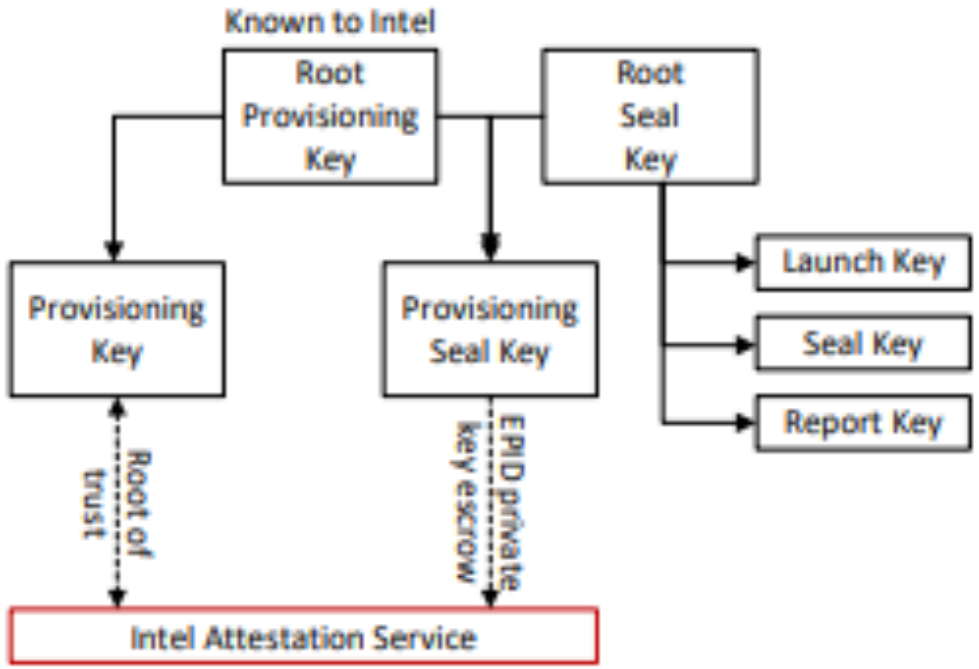
Derived Keys

- Launch Key
  - Derived from the RSK and used by the Launch Enclave to generate an EINITTOKEN
  - Each enclave not signed by Intel must obtain this token or else the processor will not instantiate it
  - Only a specific MRSIGNER (whose corresponding private keys are only known to Intel can access the Launch Key)

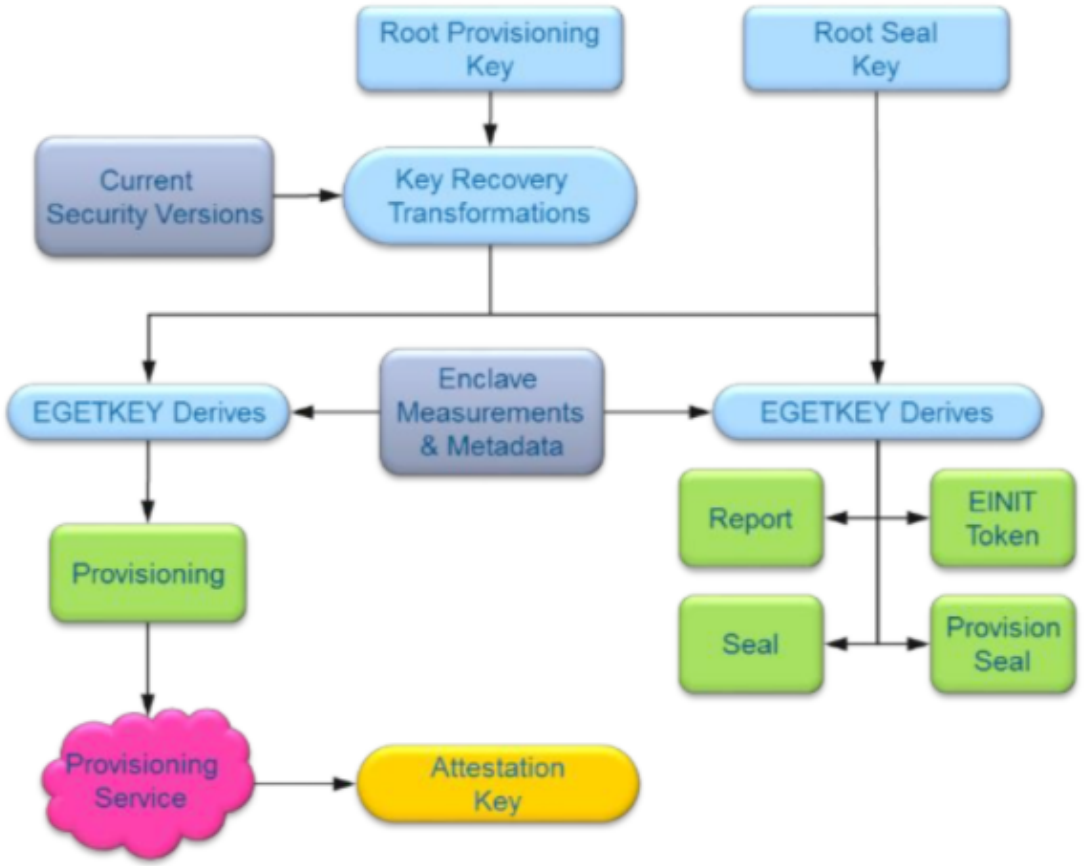
Derived Keys

- Seal Key
  - Derived from the RSK and used to encrypt data related to the current platform
- Report Key
  - Derived from the RSK and used for the local attestation process

Derived Keys



Derived Keys



SGX Features

- There are two main features of Intel SGX:
  - Sealing
  - Attestation
    - Local
    - Remote

SGX Features - Sealing

- Sealing
  - The process of encrypting enclave secrets for persistent storage in untrusted memory on the disk
  - This allows the ability to have the contents retrieved if the enclave is torn down
    - Enclave secrets are stored in volatile memory and will be wiped if any interruption to power or if the user orders the destruction of said enclave
  - The key derivation process of obtaining the Sealing Key to seal enclave secrets is derived from RSK and the chosen key policy by the author

Sidenote - Measurement

- Each enclave is provided with two measurement registers:
  - MRENCLAVE
    - The “Enclave Identity” consists of a SHA-256 digest consisting of all internal logs that record the activities done while the enclave is built such as contents of pages, positions of pages, and any security flags associated with the pages stored in the MRENCLAVE register
      - MRENCLAVE value is finalized after EINIT instruction therefore any changes to the enclave will result in a different MRENCLAVE value
  - MRSIGNER
    - The “Sealing Identity” consists of a SHA-256 hash of the public key used to sign the enclave’s SIGSTRUCT
      - Value created when the enclave builder has hardware check the expected MRENCLAVE value vs the actual MRENCLAVE value that has been signed and the public key of the Sealing Authority
        - If both passes, the hash of the public key of the Sealing Authority is stored in the MRSIGNER register
      - Multiple enclaves can be signed by the same Sealing Authority and therefore can be shared

Sidenote – Data Structures and Instructions Related with Sealing

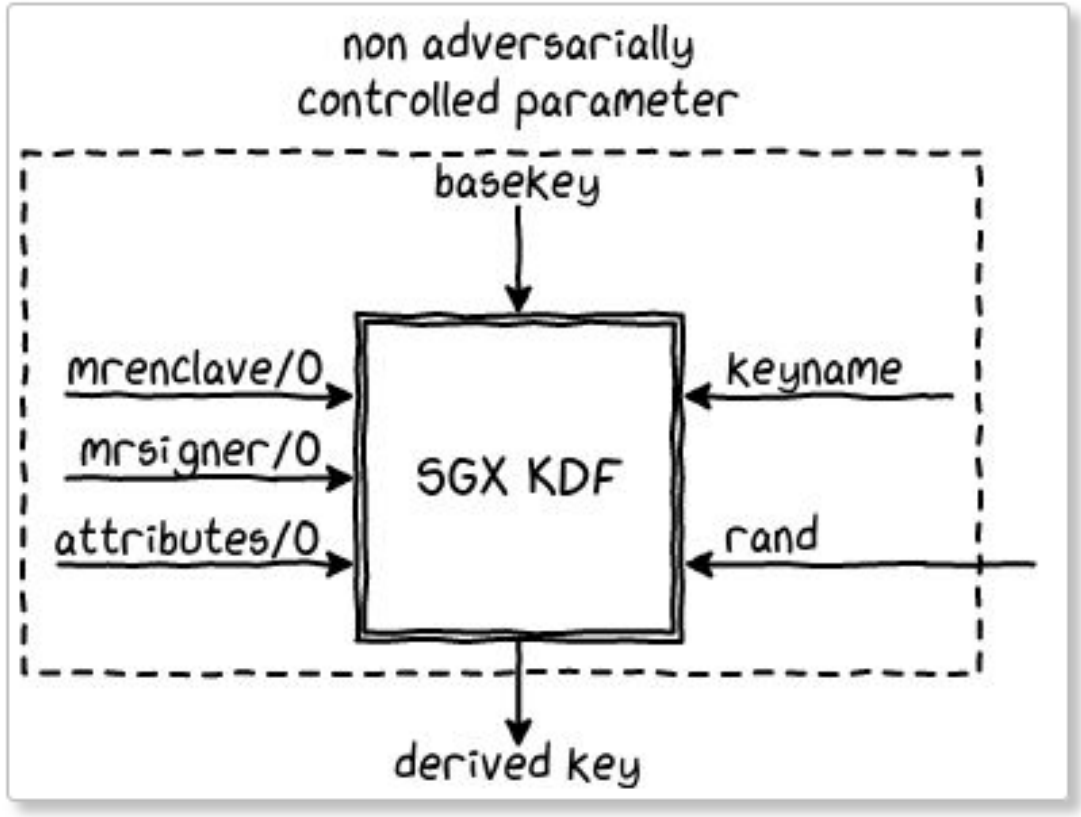
- Structures
  - Key Request (KEYREQUEST)
    - Used as the input parameter for the EGETKEY instruction (generate key) and allows it to choose which key to get and supply additional parameters which may be necessary for the derivation process
- Instructions
  - EGETKEY
    - Generates a cryptographic key

SGX Features - Sealing

- Sealing
  - Key policy
    - MRENCLAVE will result in sealed secrets that can only be unsealed by the exact enclave in which it originated from (provided that the enclave has not been altered)
    - MRSIGNER will result in sealed secrets that can be unsealed by enclaves by the same author (same product ID) with equal or newer SVN
  - Key is derived through the EGETKEY instruction in which the author will then use the appropriate key policy to derive the sealing key

SGX Features - Sealing

- Parameters are verified
- KEYREQUEST instruction (parameter for EGETKEY) started
- EREPORT called to obtain ISV and TCB SVN used information
- Key required (Seal Key) identified
- Key policy selected (MRENCLAVE or MRSIGNER)
- Key ID through RDRAND to obtain a random number for key wear-out protection
- Attribute Mask: Bitmask that indicates which attributes the seal key should be bound to
- EGETKEY is called with KEYREQUEST to obtain the seal key
- Enclave secrets are sealed with the seal key
- Seal key is deleted to prevent accidental leaks



SGX Features - Unsealing

- Unsealing
  - Parameters are verified
  - KEYREQUEST is retrieved alongside the sealed data structure
  - EGETKEY is called with the KEYREQUEST to obtain the seal key
  - Decryption is done with the seal key
  - The seal key is deleted from memory
  - Confirm that the hash generated by the decryption algorithm matches the one generated during encryption to ensure integrity

SGX Features - Attestation

- Attestation is the process of demonstrating that a specific enclave was established on an approved platform and has met the required security measures
  - Split between Local Attestation and Remote Attestation
    - Local Attestation (Intra-machine)
    - Remote Attestation (Inter-machine)
- Information conveyed from one party to another includes:
  - The identities of the software environment being attested
  - Details of any non-measurable state (mode the software environment may be running in
  - Data which the software environment wishes to associate with itself
  - Cryptographic binding to the platform TCB making the assertion

Sidenote – Data Structures and Instructions Related with Attestation

- Structures
  - Key Request (KEYREQUEST)
    - Used as the input parameter for the EGETKEY instruction (generate key) and allows it to choose which key to get and supply additional parameters which may be necessary for the derivation process
  - Report Target Info (TARGETINFO)
    - Used as the input parameter for the EREPORT instruction (generate report) and is used to identify which enclave (hash and attributes) will be able to verify the REPORT generated y the CPU
  - Report (REPORT)
    - The output from the EREPORT instruction and contains the enclave's attributes, measure, signer identity, and other user data to share between the source and destination enclave
      - Processor performs a MAC over this structure with the Report Key

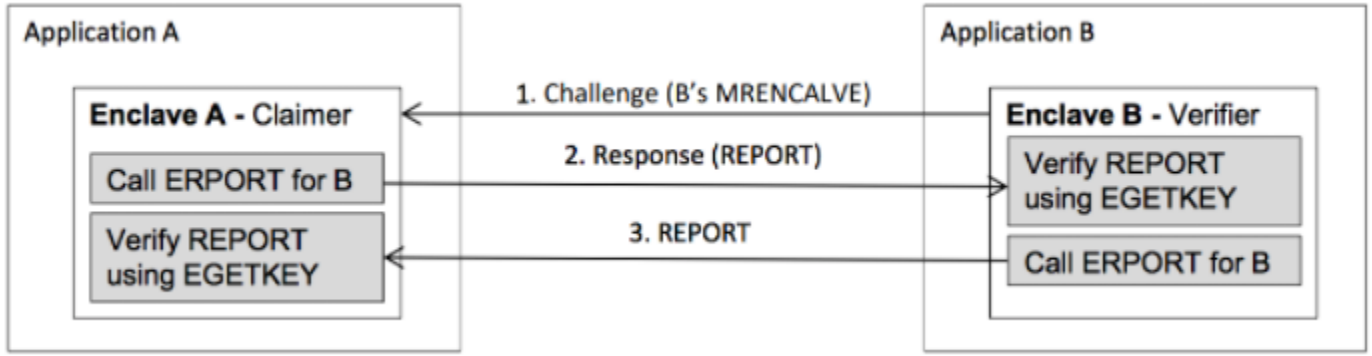
Sidenote – Data Structures and Instructions Related with Attestation

- Instructions
  - EGETKEY
    - Generates a cryptographic key
  - EREPORT
    - Generates a cryptographic report that contains information about the enclave and will be authenticated using the Report Key of the destination enclave

SGX Features – Attestation (Local)

- Local Attestation is the process done between two enclaves on the same platform
- Two enclaves will exchange data structures and verify each other prior to engaging in secured communication/sharing
- One enclave will authenticate the other locally using Intel SGX Report mechanism to verify that the counterpart is truly running on the same TCB by applying the REPORT based Diffie-Hellman Key Exchange

SGX Features – Attestation (Local)

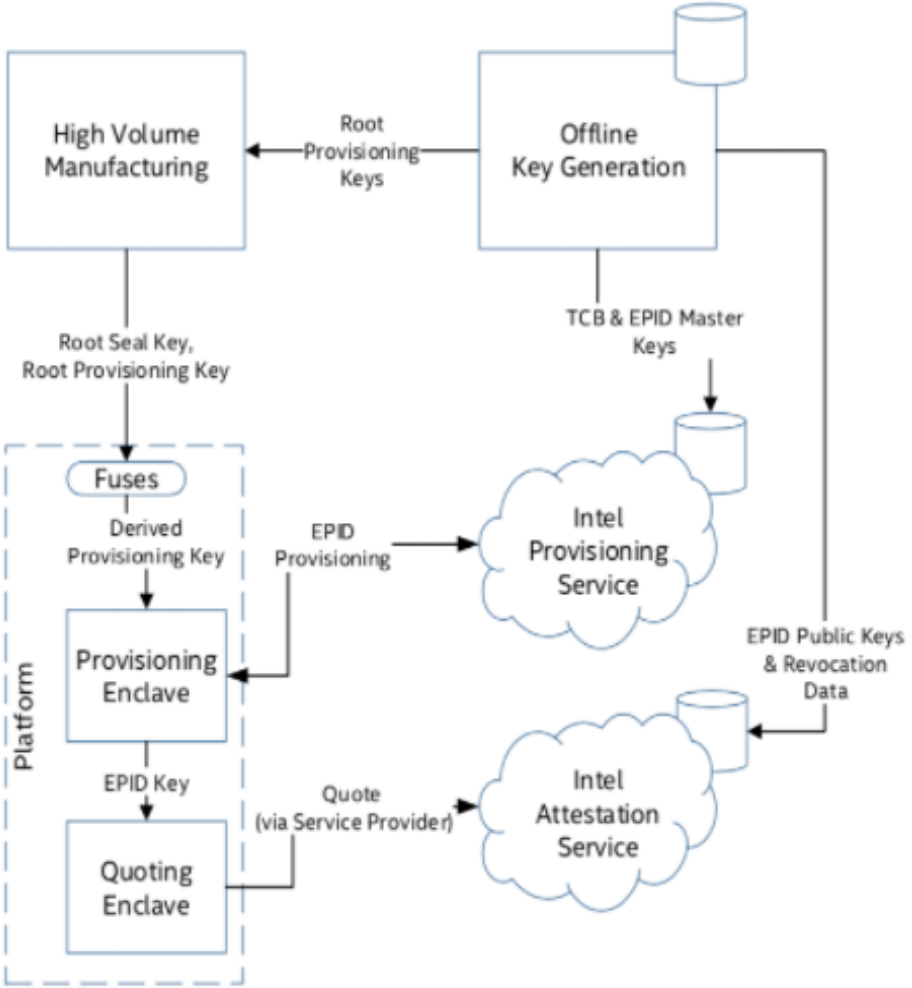


- B retrieve its MRENCLAVE measurement and sends it to A via the untrusted channel
- A uses EREPORT to produce a report data structure for B using B's MRENCLAVE measurement and sends it back to B
  - A can also include Diffie-Hellman Key Exchange data in the report to establish a trusted communication channel in the future
- B verifies the report data structure by calling EGETKEY to obtain the key to verify the report via the report key
  - If the report can be verified, then B can be sure that A is on the same platform as the report key is platform specific
- B uses the MRENCLAVE measurement from A's report to create another report via EREPORT and sends back to A
- A verifies the report with the report key obtained via EGETKEY and finally both can confirm they are on the same platform running the same TCB requirements
- Diffie-Hellman Key Exchange can be used to establish a secure channel from the data exchanged earlier in the report

SGX Features – Attestation (Remote)

- Remote Attestation is like Local Attestation where it is trying to prove to another enclave that is running on a proper SGX system but is done between platforms instead of on same platform
- This process requires additional architectural enclaves mentioned earlier called the Quoting Enclave and Provisioning Enclave/ Provisioning Certificate Enclave alongside verification services
- The main process is to have the REPORT (locally verifiable structure) be transformed a QUOTE (remotely verifiable structure) and have it signed by the Provisioning Key attest

SGX Features – Attestation (Remote)



Sidenote – Additional Information

- Intel Enhance Privacy ID (EPID)
  - Extension of existing Direct Anonymous Attestation (DDA) scheme
  - Enables the signing of objects without leaving a trace that can be backtracked to the signer which makes the signing process anonymous
    - Process is done by dividing signers to groups (EPID groups) based on processor type
      - Each can create their own secret keys, but signatures can be verified only with the public key of the group in which they belong to and therefore can only be traced back to the group but not to a specific signer
- Revocation List
  - An addition to DDA scheme of EPID which checks lists which hold revoked groups, keys or signatures of a group (Group-RL, Priv-RL, Sig-RL)

Sidenote – Provisioning Scenarios

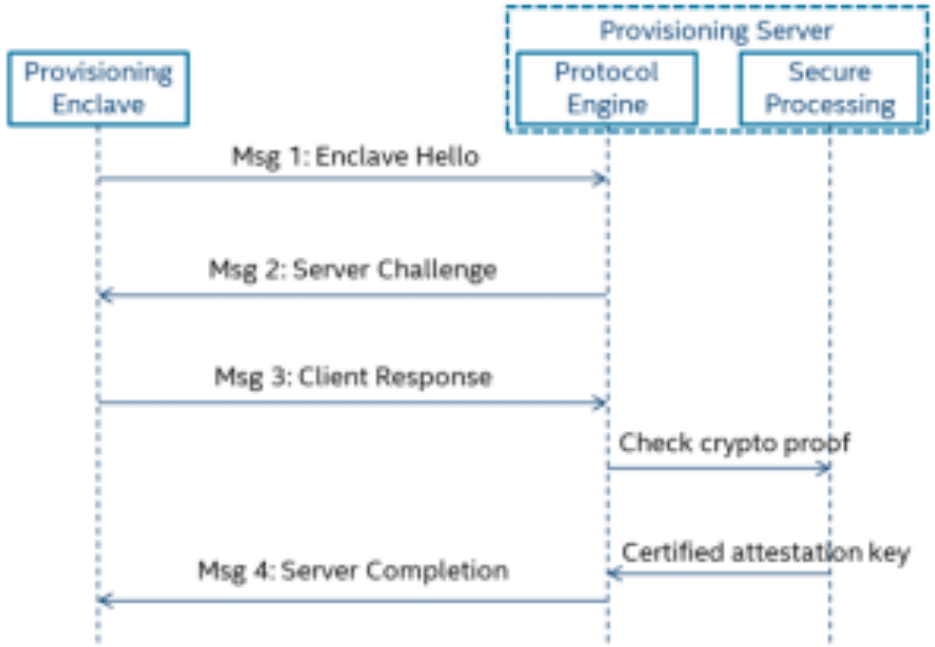
- Three scenarios:
  - First key generation
  - New key generation after TCB upgrade
  - Retrieving previous key
- First Key Generation:
  - Attestation key needs to be established when first deployed by proving that it is running on genuine Intel hardware at the TCB it claims to be in
    - Joins the EPID group selected by the EPID Provisioning Server
  - Platform encrypts a copy of its private EPID key for backup using the PSK

Sidenote – Provisioning Scenarios

- New Key Generation after TCB Upgrade:
  - Request for new EPID key from EPID Provisioning Server after upgrade but needs to prove that the platform's previous key(s) have not been revoked
    - Platform will send previous copies of EPID key to IPS via the server challenge
- Retrieving a Previous Key:
  - Similar process to requesting new key
    - Checks are done to ensure previous keys have not been revoked

Sidenote – Provisioning Protocol

- Consists of the following steps/messages:
  - Enclave Hello
  - Server Challenge
  - Client Response
  - Server Completion



Sidenote – Provisioning Protocol

- Enclave Hello
  - First message sent that contains the Platform Provisioning ID and claimed TCB version
    - PPID is derived from first provisioning key available to the provisioning enclave (PvE)
  - The message is encrypted with the public key of the IPS
  - Only authorized enclaves can access their derived provisioning key
    - Only the provisioning certificate enclave (PcE) can retrieve the provisioning key
      - PvE and PcE are implemented as a single architectural enclave

Sidenote – Provisioning Protocol

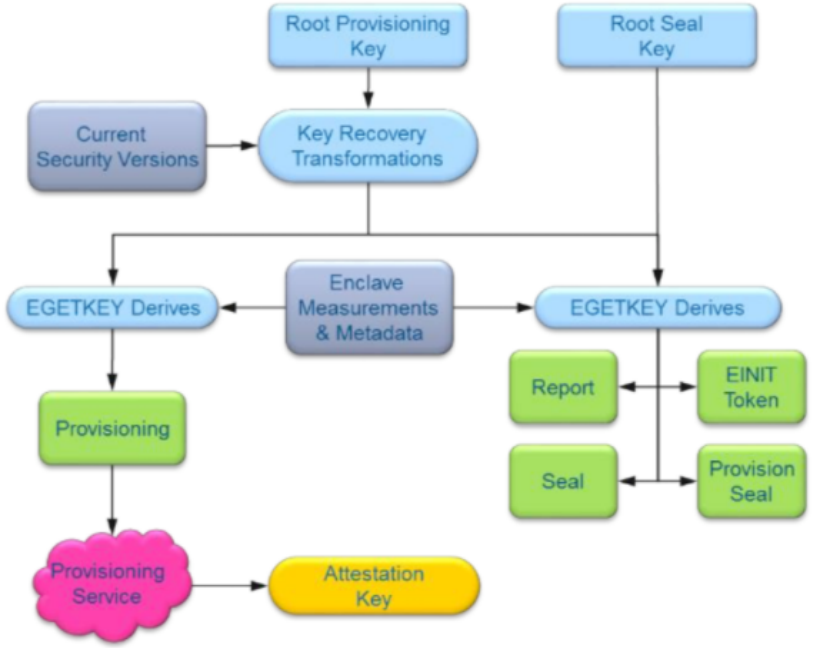
- Server Challenge
  - IPS uses PPID to determine if the platform has already been provisioned
  - If it has not been provisioned prior, the IPS will select an EPID group for the enclave and return the public key
- Client Response
  - Provisioning enclave will use provisioning key to decrypt the TCB challenge received from IPS and generates a TCB proof using the TCB challenge as a key to the CMAC received from IPS
  - PvE generates a hidden EPID and hides it
  - Client provides the non-hidden encrypted (PSK) of EPID

Sidenote – Provisioning Protocol

- Server Completion
  - IPS validates TCB proof received with the value from iKGF
    - Process continues if all is successful
  - Hidden key is processed to create a unique certificate signed with the EPID group issuer key and stored alongside the encrypted key for future provisioning events
  - Final message from IPS will be sent with a signed certificate
  - Platform’s membership key alongside the signed matching certificate creates the unique EPID private key (attestation key)

Sidenote – Provisioning Protocol

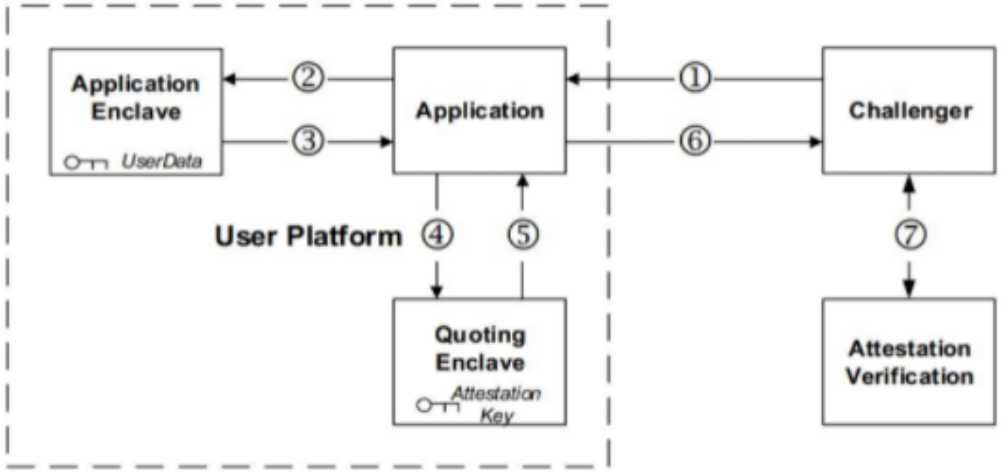
- Final
  - PvE will encrypt the attestation key with the PSK and store on platform for future use



SGX Features – Attestation (Remote)

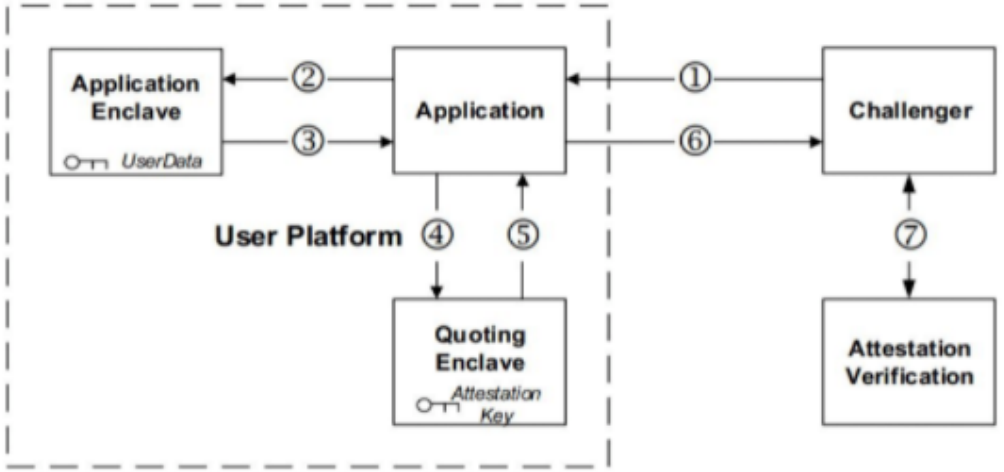
- There are three to four platforms that are involved in Remote Attestation
  - The service provider (challenger)
  - The application with its enclave and QE
  - Intel Attestation Service (IAS) that verifies the enclave
  - Intel Provisioning Service (IPS) that verifies the membership of the enclave and provides the Attestation Key (EPID)
- Remote Attestation uses both symmetric and asymmetric key systems
  - Local Attestation portion uses symmetric
  - Remote Attestation portion uses asymmetric

SGX Features – Attestation (Remote)



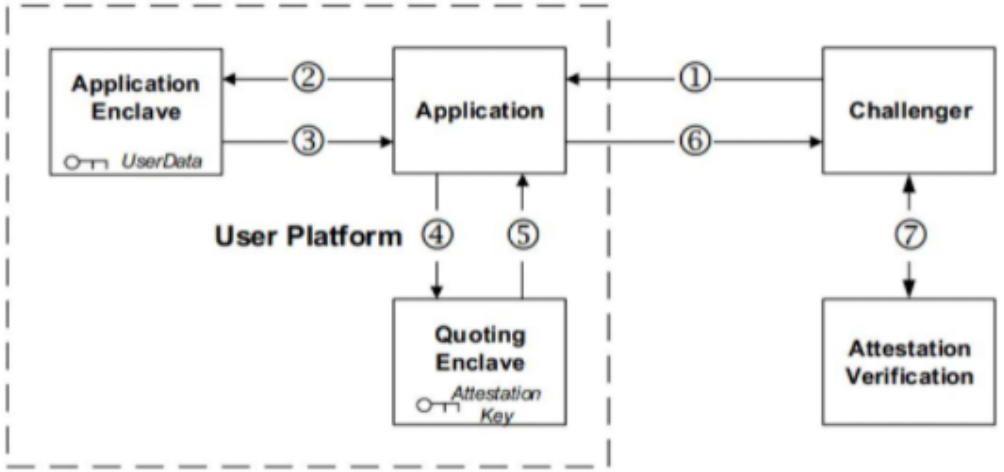
1. ISV enclave sends out an initial request to the remote service provider which includes the EPID group that the platform claims to be a member of
2. If the service provider wishes to serve the members of the claimed group, it may proceed by requesting an updated SigRL from IAS
3. If SigRL passes, the service provider constructs a challenge message consisting of its Service Provider ID (SPID registered with IAS and binded to the TLS certificate), the updated SigRL and optional base name parameter
4. Application passes the Quoting Enclave Identity alongside the challenge to the application enclave
5. Enclave will respond by generating an ephemeral public key to be used by the challenger for future asymmetric encryption communication and a REPORT from a hash digest of the manifest of user data

SGX Features – Attestation (Remote)



6. The REPORT is passed back to the application and the application passes it to the QE
7. QE calls EGETKEY to obtain the REPORT KEY and verifies the report
8. If STEP 7 is successful, QE will call EGETKEY again to receive the platform's Provisioning Seal Key to decrypt the platform's remote attestation key (EPID private key)
9. Attestation key is used to compute the two signatures of knowledge over the MRENCLAVE measurement to that the identity signature was signed with a key certified by Intel and that the signature is not on the list of the SigRL
10. The QUOTE structure (holds identity of the attesting enclave, execution mode details such as SVN, and additional data) is forwarded to the service provider for verification

SGX Features – Attestation (Remote)



11. Service provider (challenger) forwards the QUOTE to IAS for verification
12. IAS examines the QUOTE by validating the EPID proofs against the identity signature and then checks that the signature is not on the group Priv-RL
13. IAS will create a new attestation called the verification report to the service provider
14. A positive attestation verification report confirms that the enclave is running on a specific piece of code on a genuine Intel SGX processor