

Trusted Platform Modules and Their Applicability to Hardware and Software Security Mitigations

Chandni Bhowmik & Topher Timzen

Intel, Security Center of Excellence

Legal Disclaimers

- The comments and statements are those of the authors and not necessarily Intel's
- Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at [intel.com](https://www.intel.com), or from the OEM or retailer.
- No computer system can be absolutely secure.

Outline

Introduction

Trusted Computing

TPM

TEEs

How TEE and TPM are used in HW/FW/SW

Comparative Analysis

Demos

Conclusion

Questions

#whoami

Chandni Bhowmik

- Does stuff
- Hacks things
- Annoys Topher

Topher Timzen

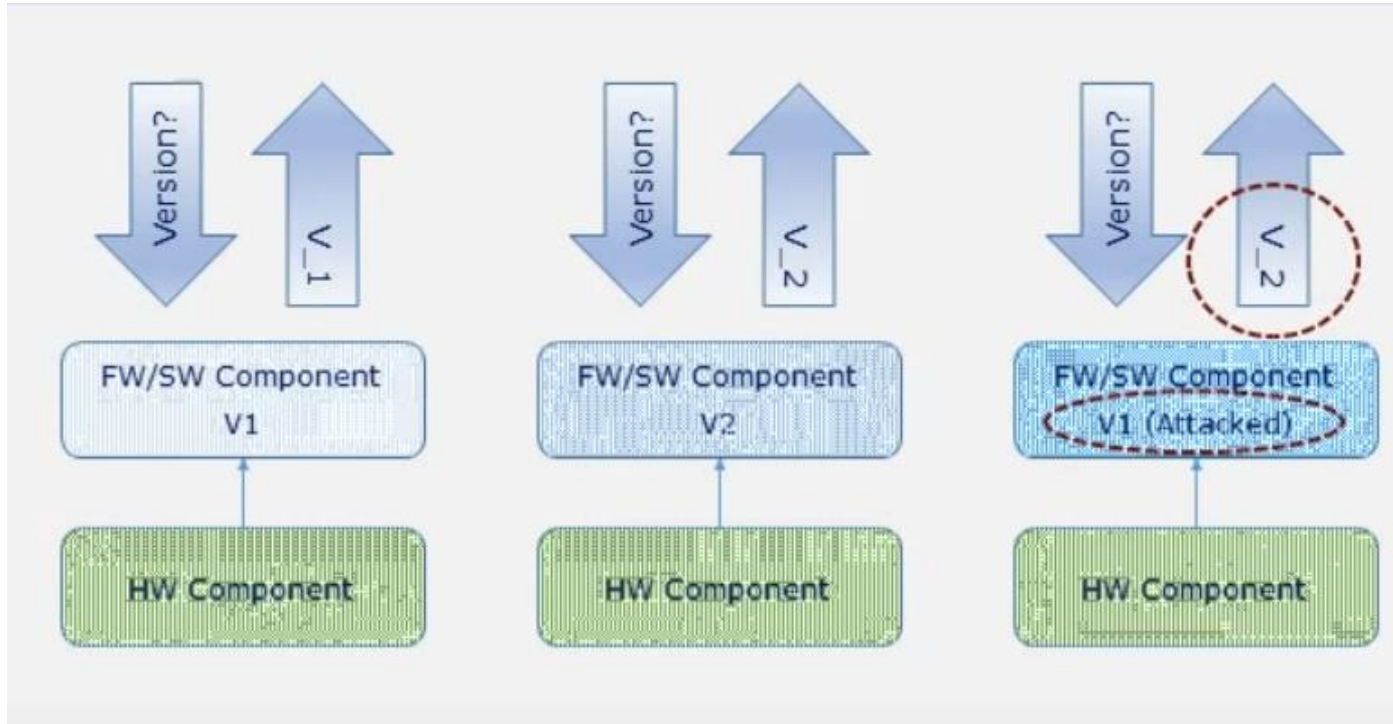
- Does stuff
- Hacks things
- Annoys Chandni

Trusted Computing

Problem Statement

Why are computers not “trustworthy”?

Software alone cannot solve the problem.



Modern security threats to PCs:
Vulnerable programs (coding bugs, buffer overflows, parsing errors), Malicious programs (spyware, Trojans), Misconfigured programs (security features not turned on), Social engineering (phishing/pharming), Physical theft (laptops), Electronic eavesdropping (capturing email)

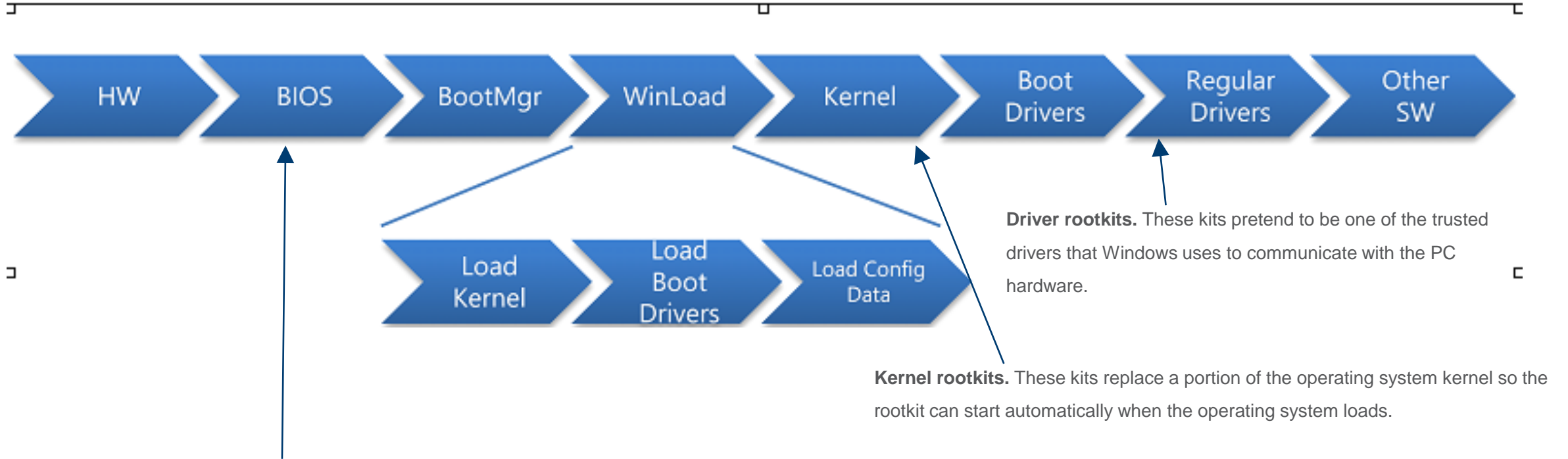
SW/FW (Mutable)

HW (Immutable)

[1]

[1] <https://trustedcomputinggroup.org/using-tpm-solve-todays-urgent-cybersecurity-problems/>

Modern PC state trustworthiness

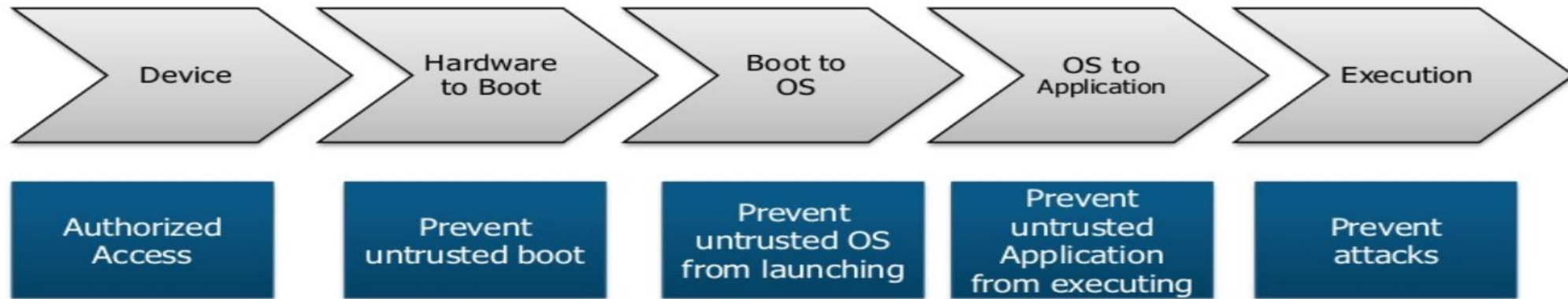


Bootkits - These kits replace the operating system's bootloader (the small piece of software that starts the operating system) so that the PC loads the bootkit before the operating system.

<https://technet.microsoft.com/en-us/windows/dn168167>

Trusted Computing Goal

Goal of trusted computing



Establishing SW / HW Trust

1. Hardware to BootROM
2. BootROM to Operating System
3. Operating System to Application

<https://www.slideshare.net/mentoresd/security-for-io-t-apr-29th-mentor-embedded-hangout>

“Trust” in trusted computing

An entity can be trusted if it always behaves in the expected manner for the intended purpose. (TCG 2004)

To be trusted,

- Bind SW trust to HW.
- Public key authentication -
 - proof of authentication of SW Identity.
- Attestation -
 - proof of authentication of system identity on which the SW ran.
- Integrity Measurements -
 - proof of integrity that cannot be broken by compromising SW alone.
 - at least be able to show an audit trail of what ran on the system without making assumptions on present system state

Modern threats: to PCs:
Vulnerable programs (coding bugs, buffer overflows, parsing errors), Malicious programs (spyware, Trojans), Misconfigured programs (security features not turned on), Social engineering (phishing/pharming), Physical theft (laptops), Electronic eavesdropping (capturing email)

SW/FW (Mutable)

HW (Immutable)

https://www.trustedcomputinggroup.org/wp-content/uploads/TCG_Glossary_Board-Approved_12.13.2012.pdf

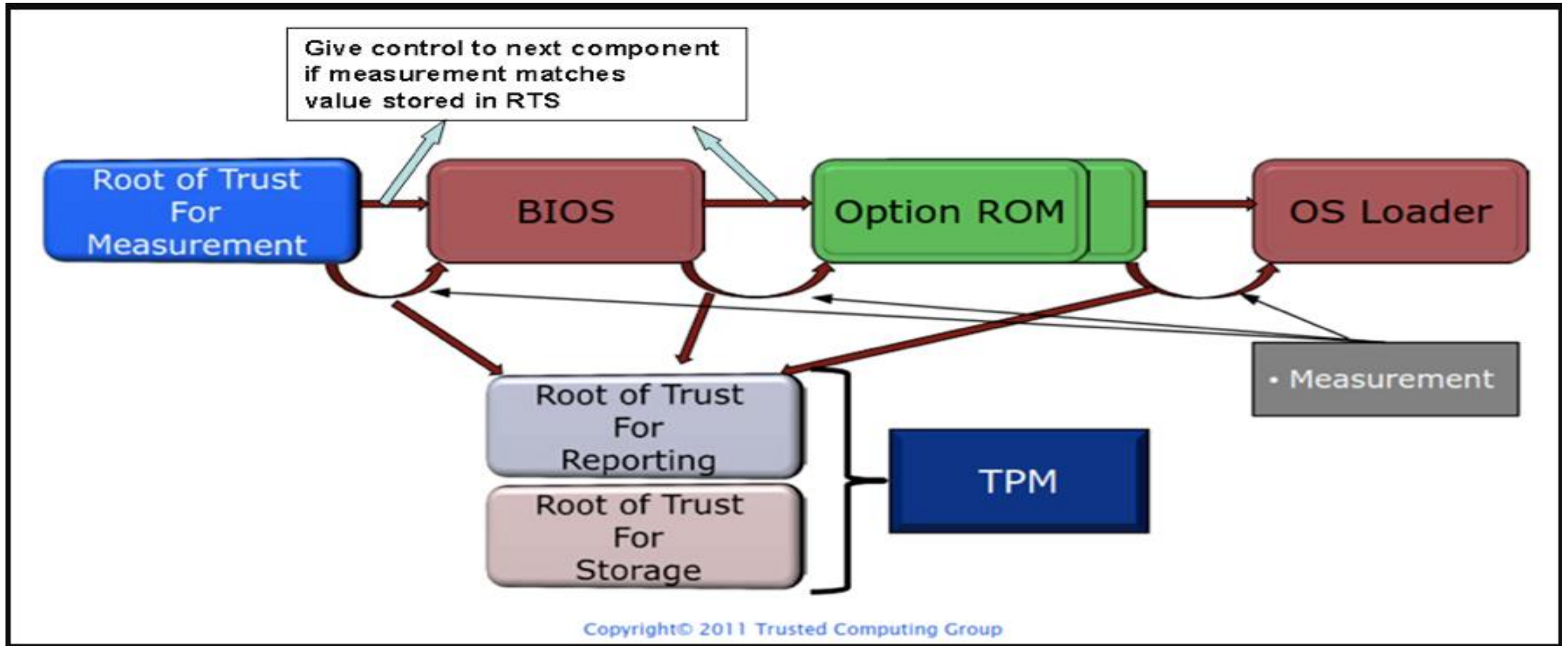
Root of Trust & Chain of Trust

Root of Trust concept from TCG

Root of Trust (RoT) A component that **must always behave in the expected manner**, because its misbehavior cannot be detected. The complete set of Roots of Trust has at least the **minimum set of functions** to enable a description of the platform characteristics **that affect the trustworthiness of the platform.**

https://www.trustedcomputinggroup.org/wp-content/uploads/TCG_Glossary_Board-Approved_12.13.2012.pdf

Chain of Trust



<https://www.ibm.com/blogs/cloud-computing/2014/09/intel-software-delivering-trusted-cloud-platforms/>

TCG Design Goals

TCG design goals

A mechanism to audit/measure what software is/was running (Integrity Measurement)

- Start with HW anchor.
- Monitor system boot.
- Store audit result in shielded location that cannot be corrupted.

Secure storage

- Allow access to sensitive data only if system is in trusted state.

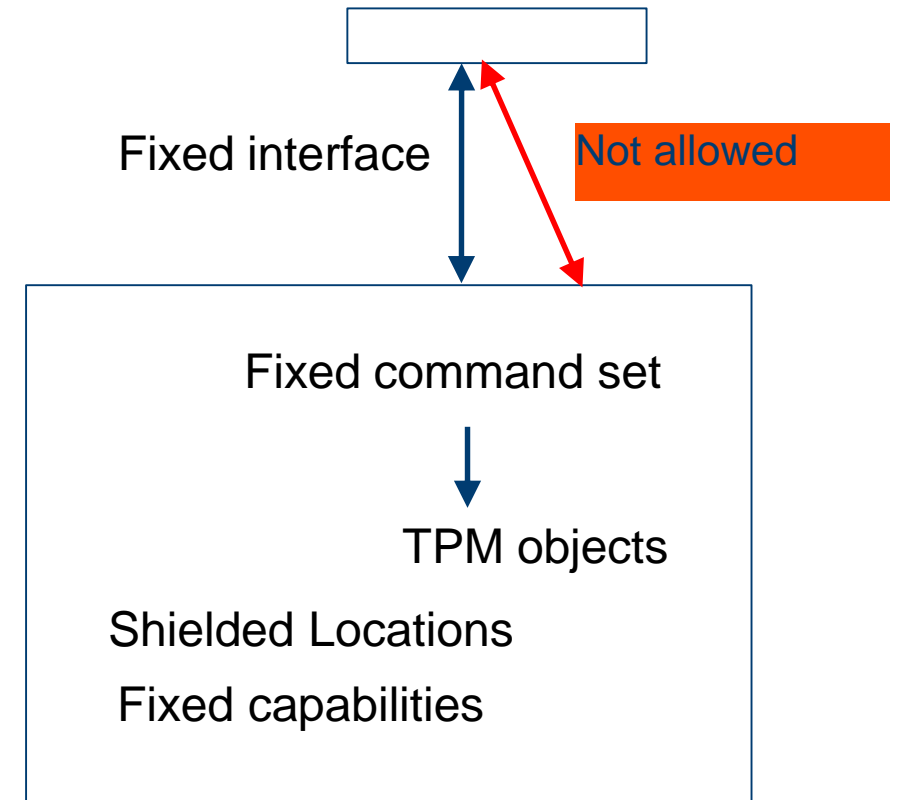
Mechanism to report system state

- Securely report measurement to trusting authority.

TPM design goals

Isolation and Tamper resistance

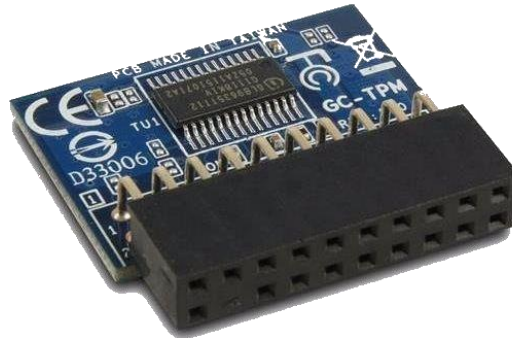
- Shielded locations
 - Storage area that is not exposed to external software for modification.
 - Only defined set of functions / protected capabilities can act on the shielded locations.
- Protected capabilities
 - A function expected to behave correctly every time so that TPM can be trusted.
- Controlled interface (platform responsibility)



TPM

What is a TPM?

- TPM – Trusted Platform Module
- Passive IO device either on LPC bus or SPI bus
- Has several Platform Control Registers (PCR[0...23])
- Operations Include
 - Crypto Stuff
 - Seal/Unseal
 - Quote



Why use a TPM?

A TPM has several advantages to the security of a platform:

Supports Cryptographic protocols - encryption, decryption, signing/verification, key generation

- Enable Root of Trust for Measurements (RTM)
 - Platform Configuration Register (PCR) for storing PC audit trail / measurements
- Root of Trust for Storage (RTS)
 - Trusted entity for storing measurements that is not exposed to external alteration
 - PCRs cannot be written. Only read/extended.
 - Secure Authorization - A hash of $f(x)$ is stored in a PCR after initial measurement and the same hash is needed to unseal the secret
- Root of Trust for Reporting (RTR)
 - Trusted entity responsible for reporting measurements securely and correctly
 - Attest to PCR measurements by signing with RSA ID keys.
 - Proof of platform identity

TPM functional units

Component class	Function provided/data stored
Functional units	Random numbers Cryptographic hashes Message authentication codes (HMAC) RSA key-pair generation RSA encryption and decryption
Nonvolatile memory	Endorsement key Storage root key Owner authorization secret key
Volatile memory	RSA key pairs Platform configuration registers Key handles Authorization session handles

Salvador Abreu, Francisco Rocha, Miguel Correia, "The Final Frontier: Confidentiality and Privacy in the Cloud", *Computer*, vol. 44, no. , pp. 44-50, Sept. 2011, doi:10.1109/MC.2011.223

TPM PCRs and Root of Trust for Measurements

Goals:

- Create audit trail of PC boot
- Use the audit trail to detect change in FW integrity
- Cannot alter measurement

PCR extension:

- Platform Configuration Registers (single or multiple banks of 20 byte registers)
- Cannot write to PCRs, Only extend
 - $\text{New_PCR} = \text{Hash}(\text{existing_PCR} \mid \text{Hash of FW integrity})$
- SHA-1, SHA-256 etc.
- SHA-1 chain: Output 160 bit (20 byte) value
 - Accumulate hashes or audit trail of system FW components
 - Hash is ordered ($\text{Hash}(X \mid Y) \neq \text{Hash}(Y \mid X)$)

PCR in PC Client spec

There are 24 in TPM 1.2 Specification

- 0 - CRTM, BIOS and Platform Extensions
- 1 - Platform Configuration
- 2 - Option ROM Code
- 3 - Option ROM Configuration and Data
- 4 - IPL Code (MBR Information and Bootloader Stage 1)
- 5 - IPL Code and Configuration Data (for use by IPL Code)
- 6 - State Transition and Wake Events
- 7 - Reserved for future usage. Do not use.
- 8 - Bootloader Stage 2 Part 1
- 9 - Bootloader Stage 2 Part 2
- 10 - Not in Use.
- 11 - Not in Use.
- 12 - Bootloader Command line Arguments
- 13 - Files checked via check-file routine
- 14 - Files which are actually loaded (e.g. Linux kernel, initrd, modules..)
- 15 - Not in Use.
- 16 - Not in Use.
- 17 - DRTM
- 18 ... 23 - Not in Use.

0-6 = Preboot

10 = Integrity Measurement Architecture

17, 18,19 = records the DRTM chain

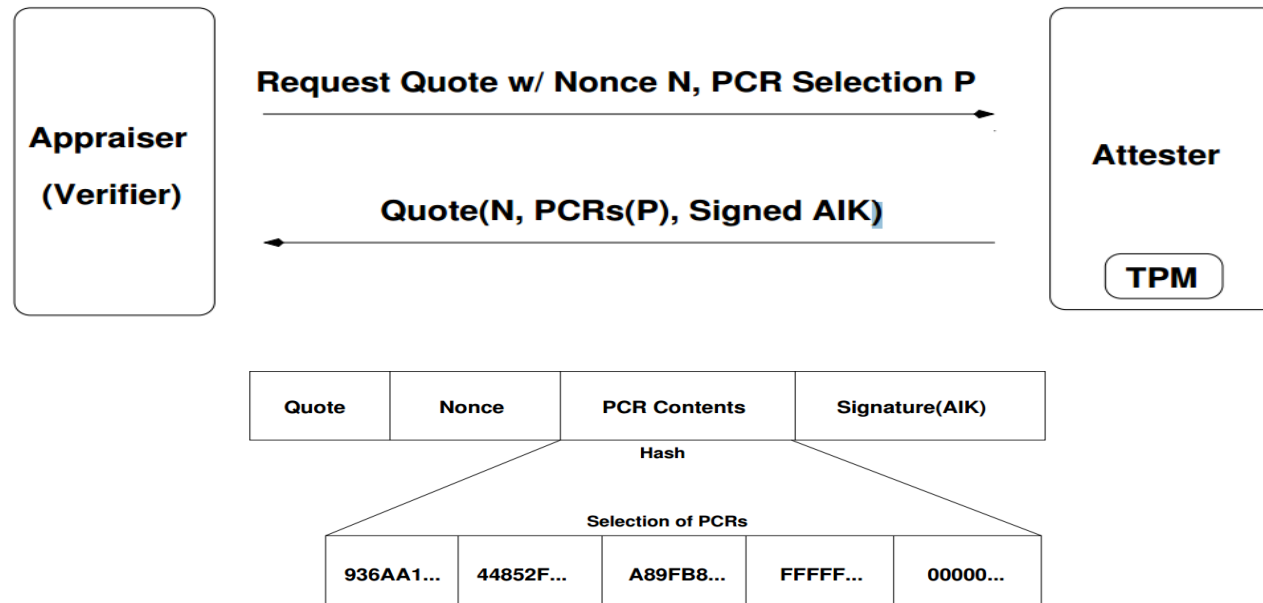
0-15 are for SRTM and cannot be reset (except at boot) but can be extended.

17-22 are for DRTM

Root of Trust for Reporting (RTR)

Digitally sign platform integrity report (PCR log) with TPM attestation key and send digital signature to appraiser. Prove to appraiser:

- Platform integrity report is genuine and originated from a platform appraiser trusts
 - appraiser trusts/verifies TPM by its Signature (AIK)



http://opensecuritytraining.info/IntroToTrustedComputing_files/Day2-1-auth-and-att.pdf

Secure Identity using TPM (RTR)

To prove to a 2nd party that TPM ID key belongs to genuine TPM the party trusts.

- Unique Endorsement Key (EK) per TPM at manufacturing time.
 - 2048 bits RSA key pair generated inside TPM and the TPM ships with it
 - Unique to each TPM
 - EK Private is non-migratable outside TPM
 - EKPub can be certified by manufacturer EKCert
 - Not used for encryption or signing data.
 - Signs other TPM identity keys.

Attestation ID key (AIK) - Signed by EK.

- Sign PCRs (Attestation)

Root of Trust for Storage (RTS)

Storage keys - for encrypting keys and data. Encrypt with public key (cannot sign), stores other storage or signing keys.

- Storage Root Key (SRK)
 - Storage key generated at TPM activation and reset
 - 2048 bit RSA Key pair, non-migratable, root of TPM key chains.
 - Wraps/encrypts platform migratable key, wraps user storage key
- Bind data securely to TPM. Bind and Unbind (no PCR usage)
- Seal secret to PCR. PCR Seal and PCR Unseal

Root of Trust for Storage (RTS)

TPM Bind: TPM can generate an asymmetric encryption key pair on request for use in secure storage of symmetric keys.

- TCB encrypts secret with the public part of TPM key and this can be done outside TPM.
- However, decryption can only happen within TPM using TPM Private key which is non-migratable. (TPM Unbinding)
- Binding secrets to TPM such that without TPM data cannot be retrieved.

Root of Trust for Storage (RTS)

Sealing options

- Use a storage key for encryption, Optional authorization for storage key
- Record current PCR state
- Seal against recorded PCR state
 - Decryption will require PCR state to match

Unsealing

- Decrypt with storage key, May need authorization to use storage key
- Verify against PCR to decrypt
 - If PCR mismatch – don't decrypt sealed data

TPM keys and Usages

Keys	Description	Usages	Comments
Storage keys	Encrypts with public key (not for signing), stores other storage or signing keys.	Protect private keys from being stolen.	Generate key pair in TPM. Private key can be migrated off TPM but encrypted. Key Decryption can only be done in TPM. Add integrity measurement constraint so malicious SW identity cannot get the key if requests TPM.
Binding keys	Asymmetric non-migratable key pair for encrypting symmetric keys and storing in TPM.	Prevents Encryption keys from being stolen.	Encrypt data and seal key against PCRs.
Identity Keys	Signing keys for PCRs and signing other ID keys.	TPM Identity, User Identity, SRK is root.	Attest to platform identity by signing PCR extend log.

Localities

PCR	Alias (description)	Extendable in localities
0-15	Static RTM	4,3,2,1,0
16	Debug	4,3,2,1,0
17	Locality 4	4,3,2
18	Locality 3	4,3,2
19	Locality 2	3,2
20	Locality 1	3,2,1
21	Dynamic OS controlled	2
22	Dynamic OS controlled	2
23	Application Specific	4,3,2,1,0

Locality 4: Trusted hardware component. This is used by the D-CRTM to establish the Dynamic RTM.

Locality 3: Auxiliary components. Use of this is optional and, if used, it is implementation dependent.

Locality 2: Dynamically Launched OS (Dynamic OS) “runtime” environment.

Locality 1: An environment for use by the Dynamic OS.

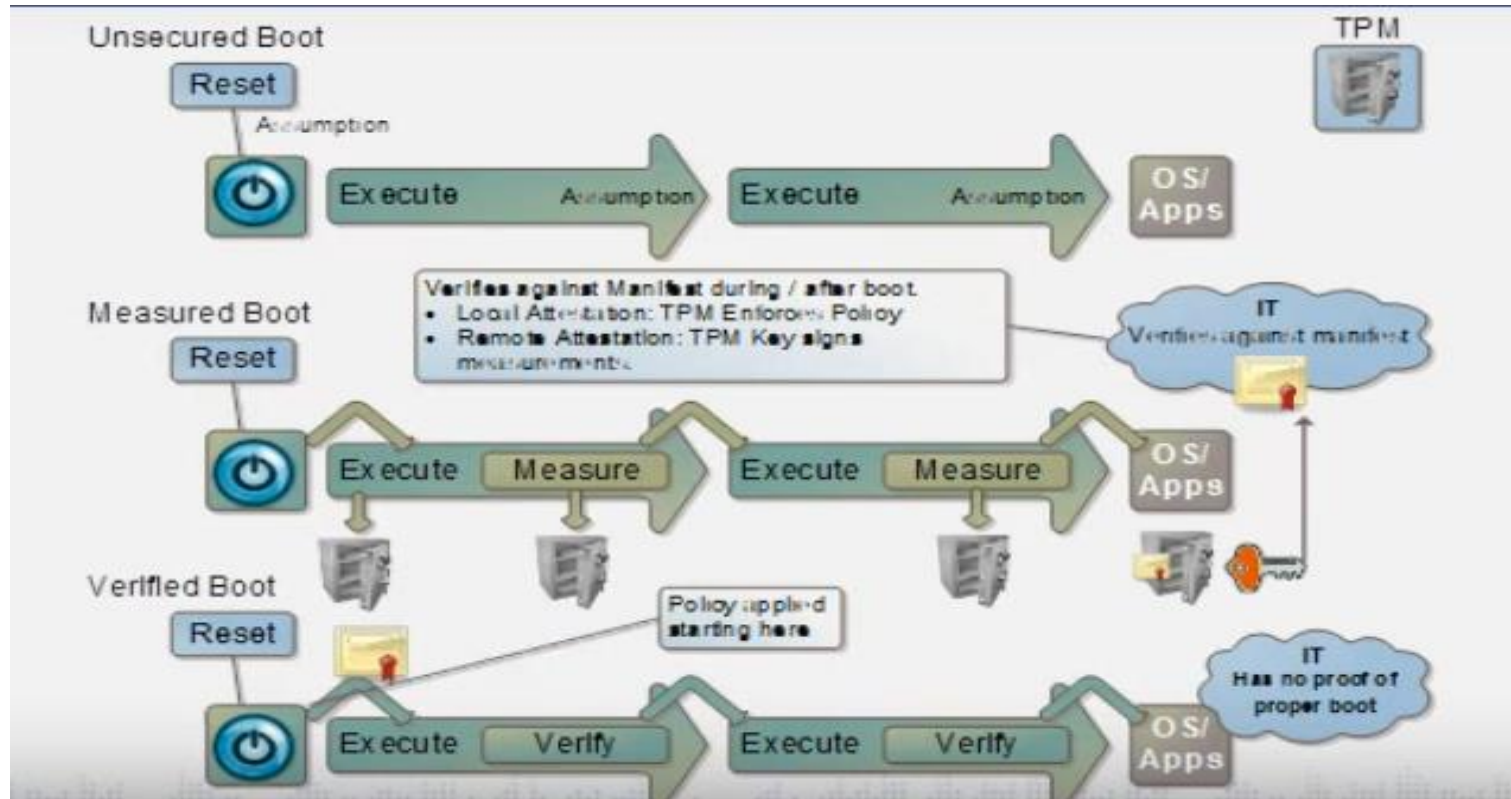
Locality 0: The Static RTM, its chain of trust and its environment.

Localities

There are 4 localities for changing the TPM (Primitive caller-based access control for TPM)

- 1) General purpose for MLE
- 2) Runtime of the MLE and can reset 20-22 and extend all PCR
- 3) SINIT ACM for DRTM (ACPI/DMAR tables for DMA) and measure the MLE in PCR 18
- 4) Hardware/microsoft as root of DRTM/TXT. Used for PCR 17-20 and send measurement to PCR 17

Insecure vs measured vs verified vs secure boot



SRTM & DRTM

TPM Measurements - Static Root of Trust Measurements (SRTM)

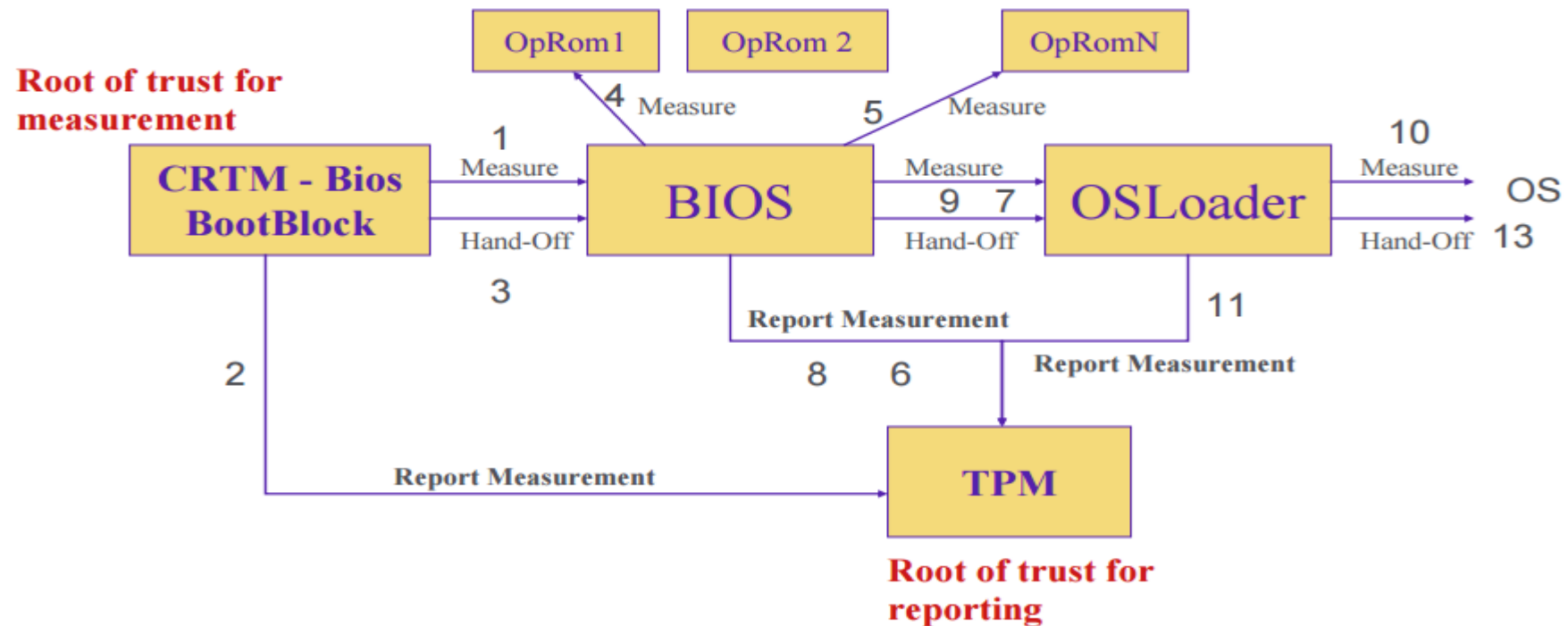
SRTM

- Measures code at launch time (no protection from run time corruption)
- Core Root Of Trust for Measurement (CRTM) runs at boot or platform reset
 - Immutable trusted static code blob (typically BIOS Initial Boot Block)

Problems with SRTM?

- Need to measure every possible piece of code that was executed since system boot
- doesn't scale
- Too long of a chain-of-trust

TPM Measurements - Static Root of Trust Measurements (SRTM)



<http://www.ieee802.org/1/files/public/docs2004/af-congdon-tcg-overview-1104.pdf>

TPM Measurements - Dynamic Root of Trust Measurement (DRTM)

Launch a trusted environment from an untrusted state (late launch)

Root of trust can be initialized at any point in runtime unlike SRTM which relies on a “static trust”

Secure Virtual Machine (SVM) is AMD’s approach

Trusted Execution Technology (TXT) is Intel’s implementation

- SENTER instruction to apply VT-d protections around a hypervisor image, measure it and only load it if trusted

Problems with DRTM?

- BIOS and SMM

BIOS and TPM

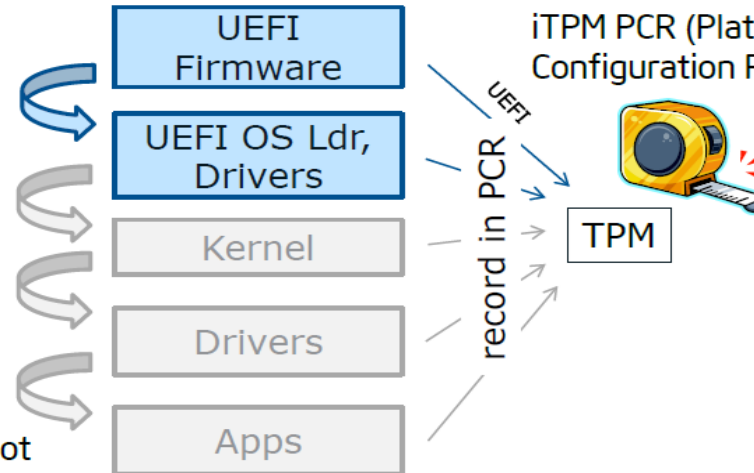
What

UEFI Secure Boot VS TCG Trusted Boot

UEFI authenticate OS loader (pub key and policy)

Check signature of before loading

- UEFI Secure boot will stop platform boot if signature not valid (OEM to provide remediation capability)
- UEFI will require remediation mechanisms if boot fails



UEFI PI will measure OS loader & UEFI drivers into iTPM PCR (Platform Configuration Register)

- TCG Trusted boot will never fail
- Incumbent upon other SW to make security decision using attestation

[Secure Boot Ecosystem Challenges](#)

FDE

FDE and the TPM

Full disk encryption passwords need to be stored somewhere

TPM can be utilized to seal the secret decryption key

In Human Terms “I want to use BitLocker”

Protects against Evil Maid Attacks

- A TPM with STRM would detect malicious pre-boot environment

FDE – Bitlocker case study

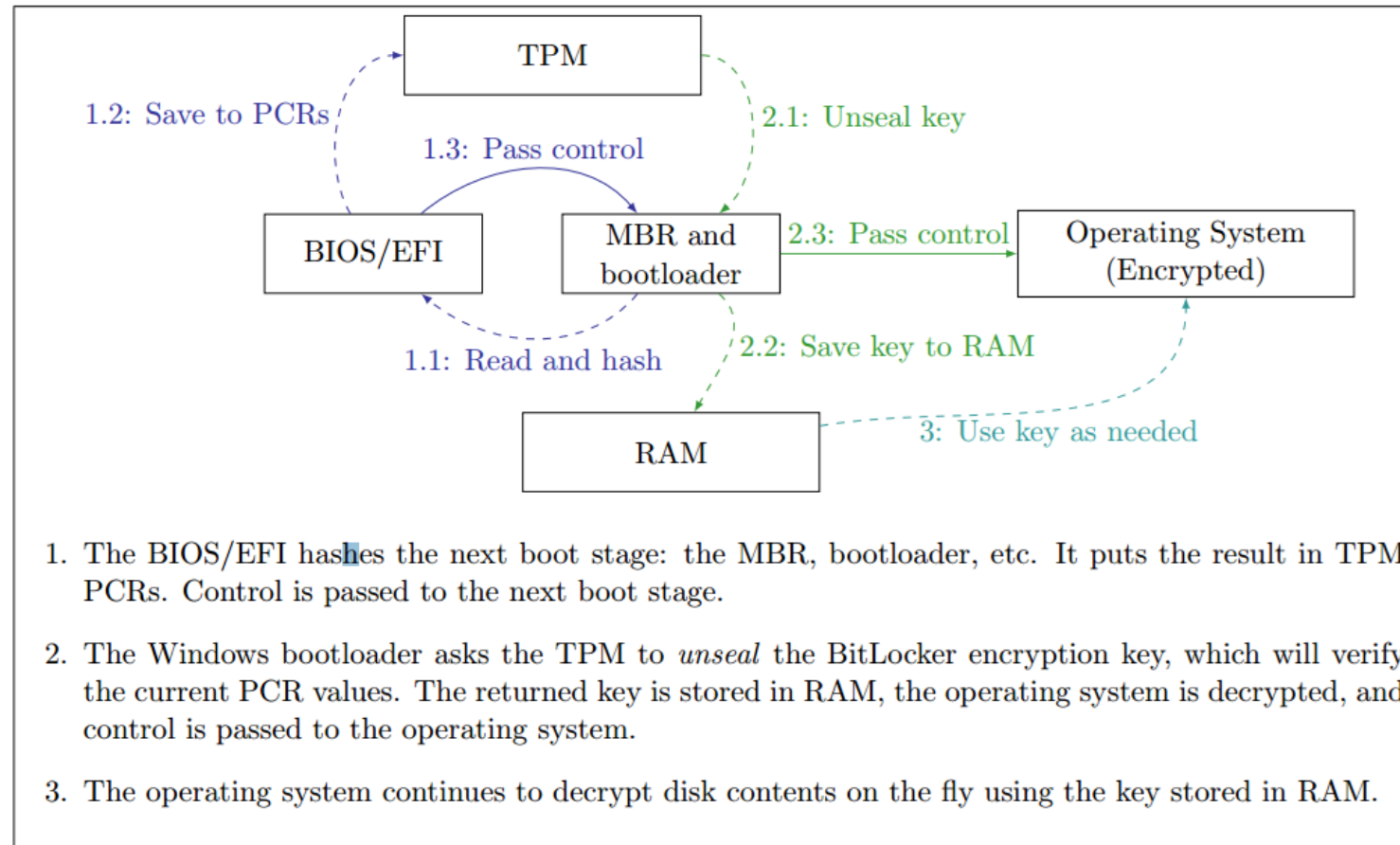
Removable Flash Drive for FDE is not a good option and is easier to obtain by malicious actors

TPM allows for transparent secure key storage allowing Bitlocker to decrypt the operating system volume on boot without pre-boot authentication

The TPM will unseal the key to the operating system which is used to decrypt the drive

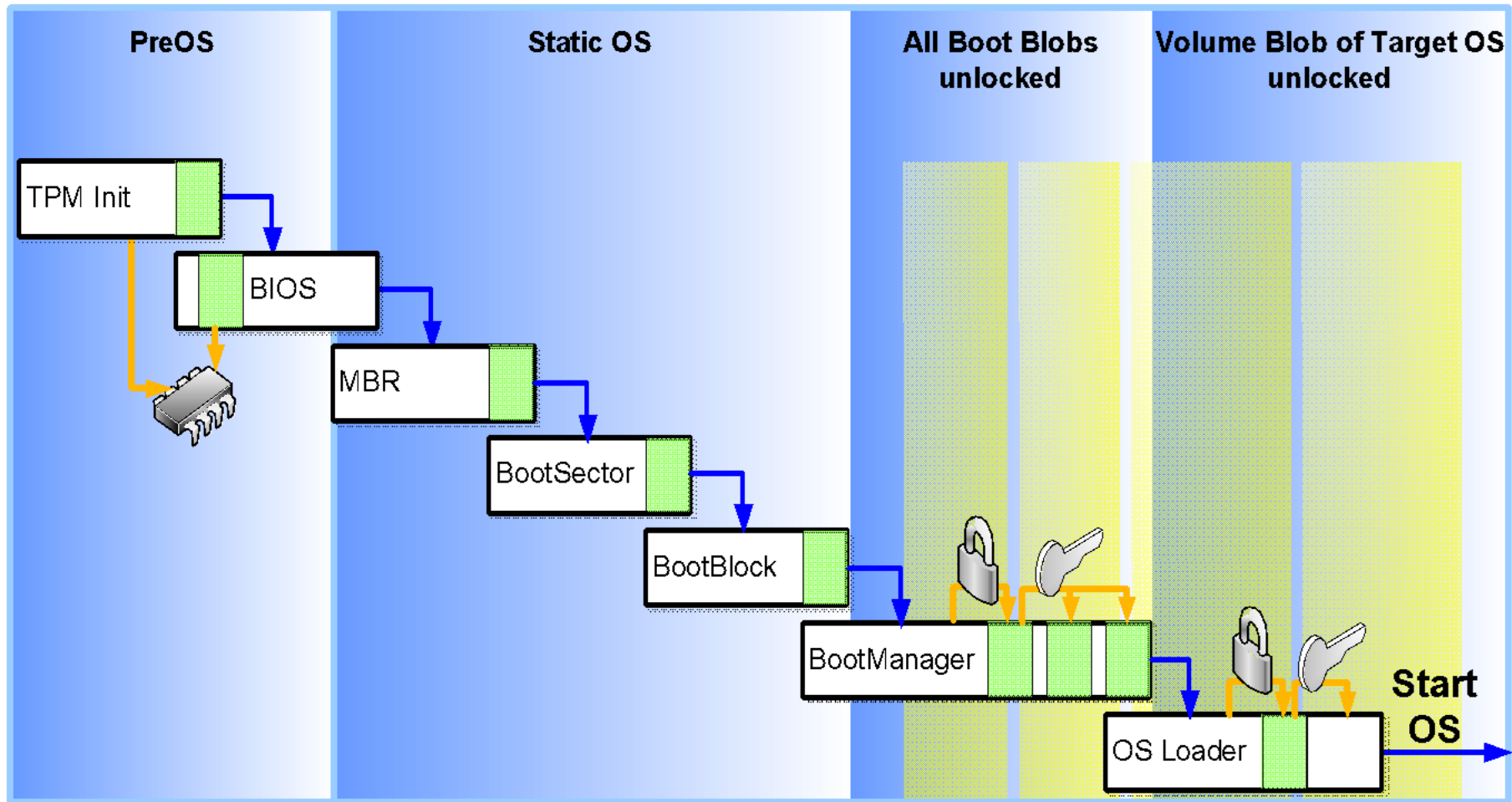
Static Root of Trust Measurement (SRTM) is used with bitlocker.

FDE – Bitlocker case study



<https://www.blackhat.com/docs/eu-15/materials/eu-15-Haken-Bypassing-Local-Windows-Authentication-To-Defeat-Full-Disk-Encryption-wp.pdf>

FDE – Bitlocker case study



http://download.microsoft.com/download/5/b/9/5b97017b-e28a-4bae-ba48-174cf47d23cd/cpa064_wh06.ppt

TEE

TPM vs TEE

TEE:

- Trusted area on an SoC
- Protected/Secure virtualized environment
- Utilizes secure boot to ensure all system components are trusted

TPM:

- Hardware that is physically isolated from the core processor that provides
 - Attestation
 - Binding
 - Sealing

What is a TEE

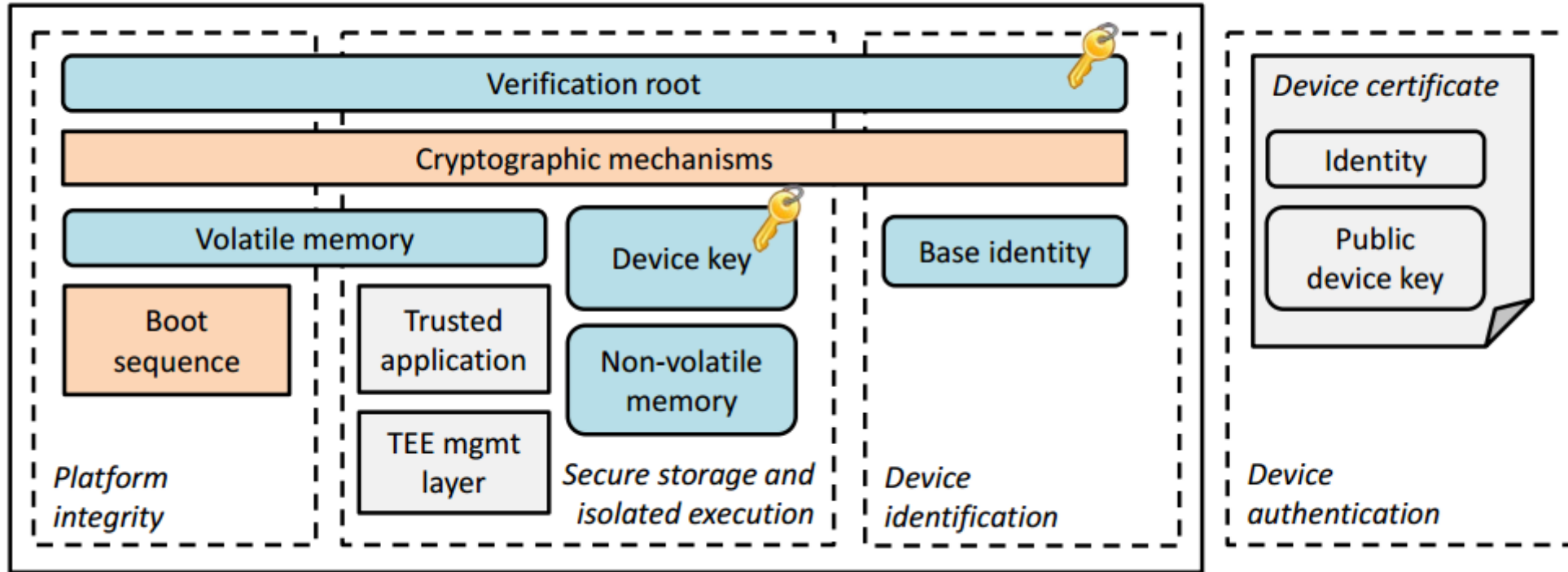
Isolated execution area on an SoC

- Processing, memory and storage capabilities

A Trusted Execution Environment provides

- Platform integrity
- Isolated execution
- Device Identification
- Device Authentication
- Secure storage

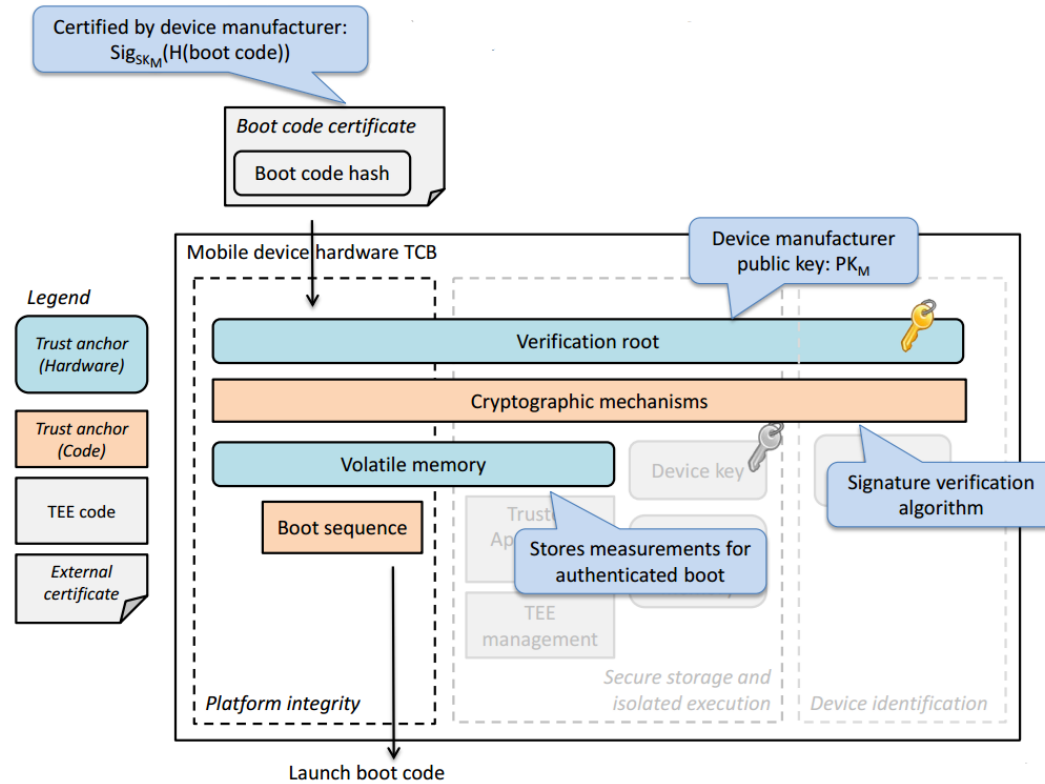
TEE



<http://asokan.org/asokan/Padova2014/tutorial-mobileplatsec.pdf>

Platform Integrity

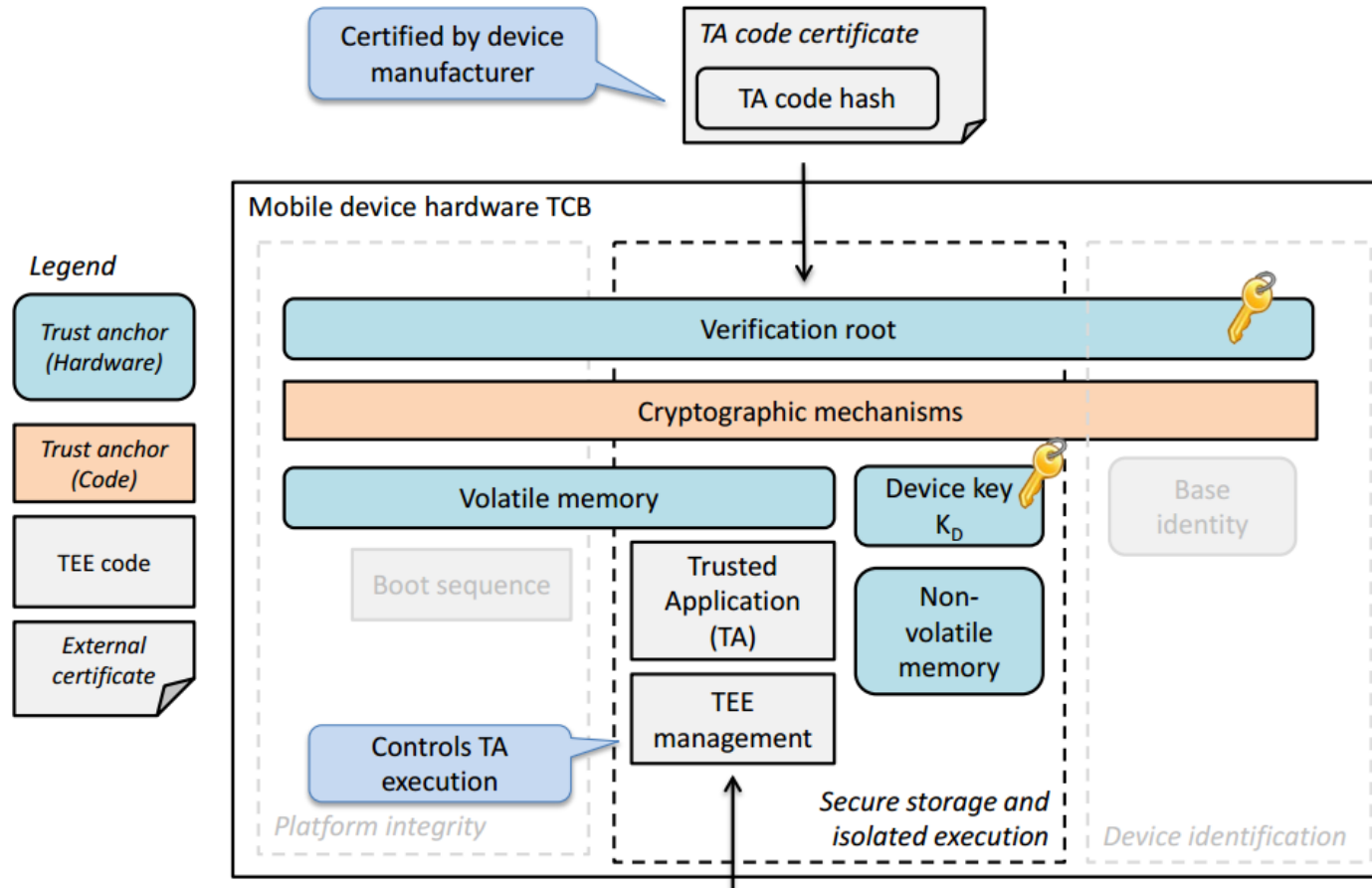
Secure / Authenticated boot



<http://asokan.org/asokan/Padova2014/tutorial-mobileplatsec.pdf>

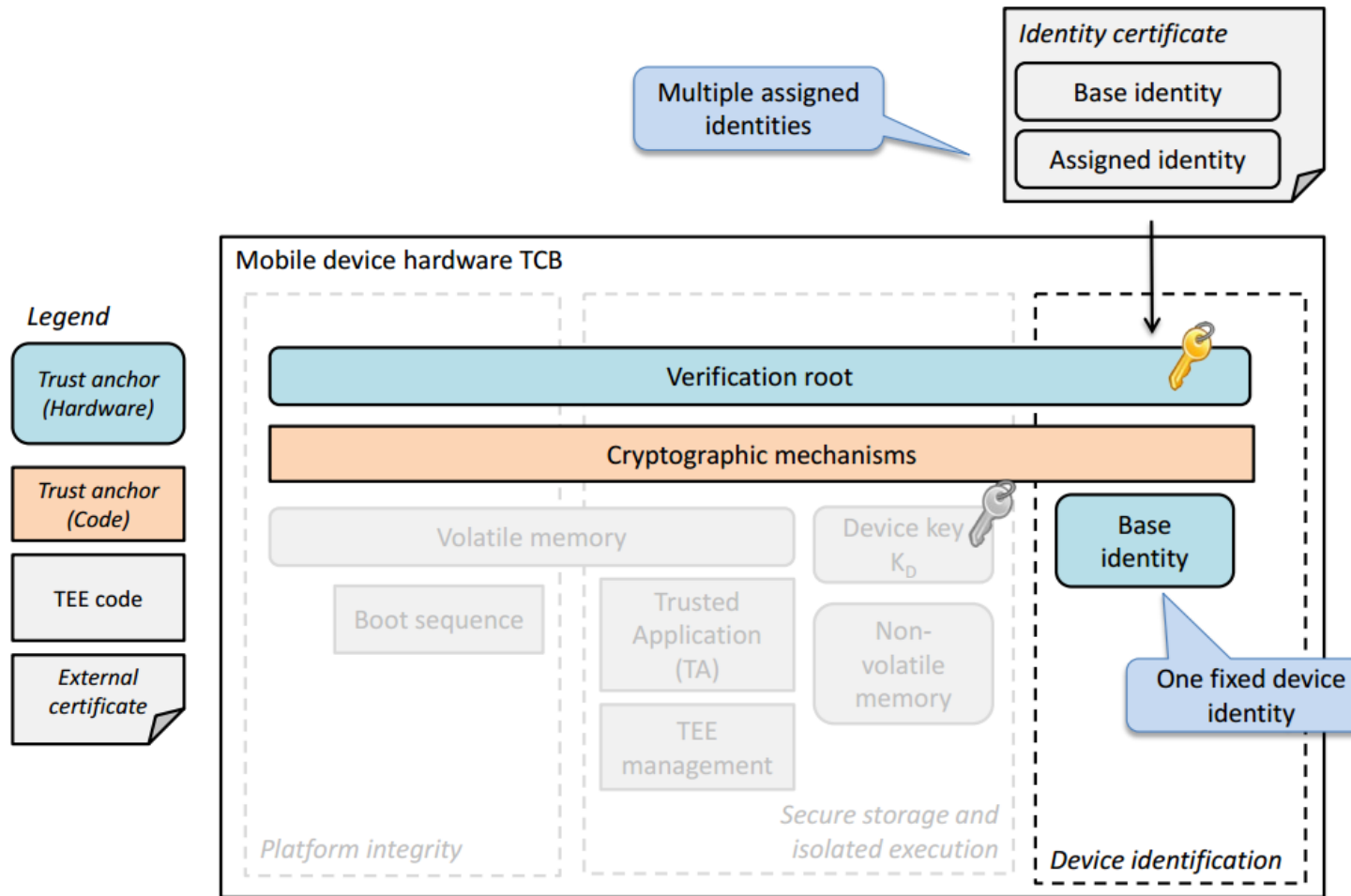
Isolated Execution

The core of the TEE



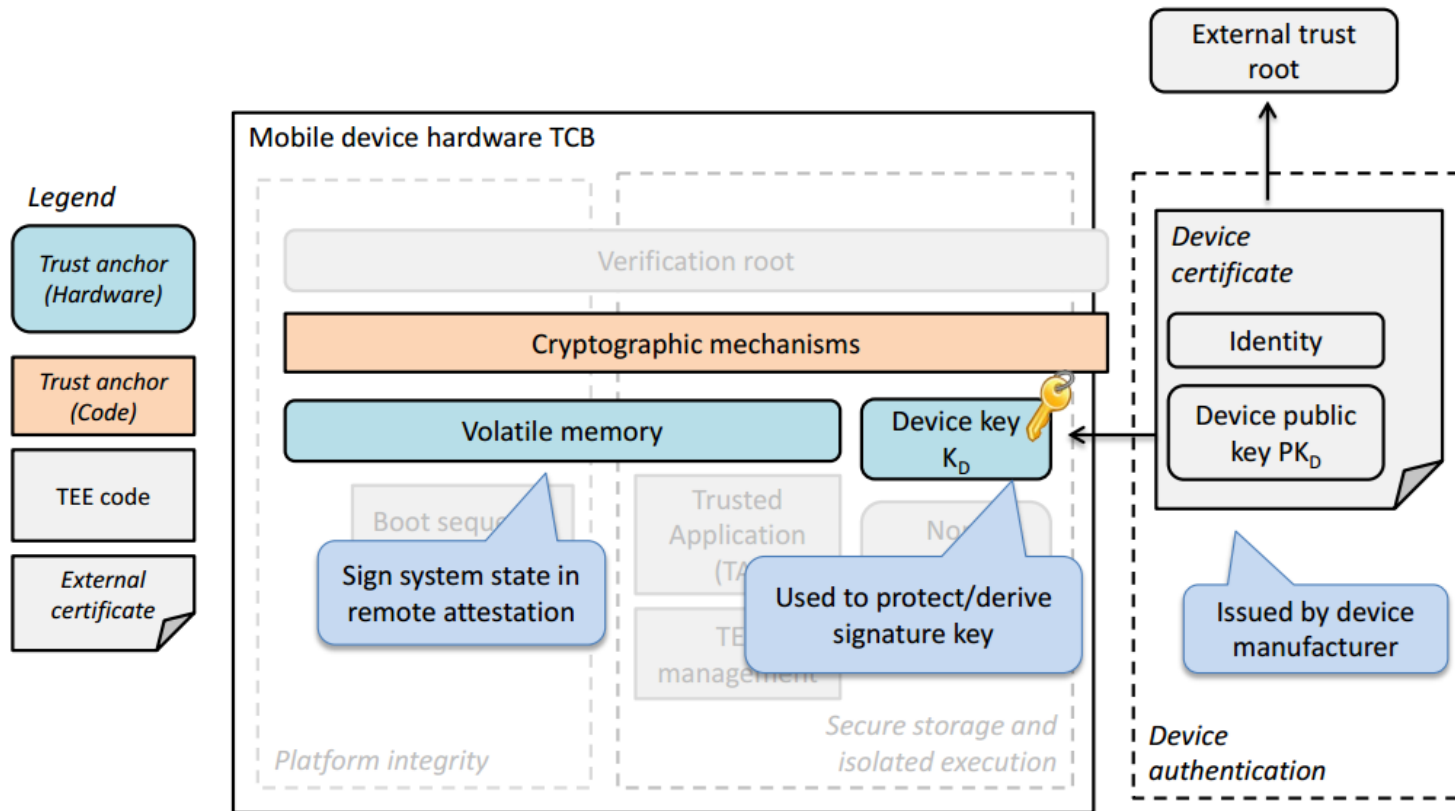
<http://asokan.org/asokan/Padova2014/tutorial-mobileplatsec.pdf>

Device Identification



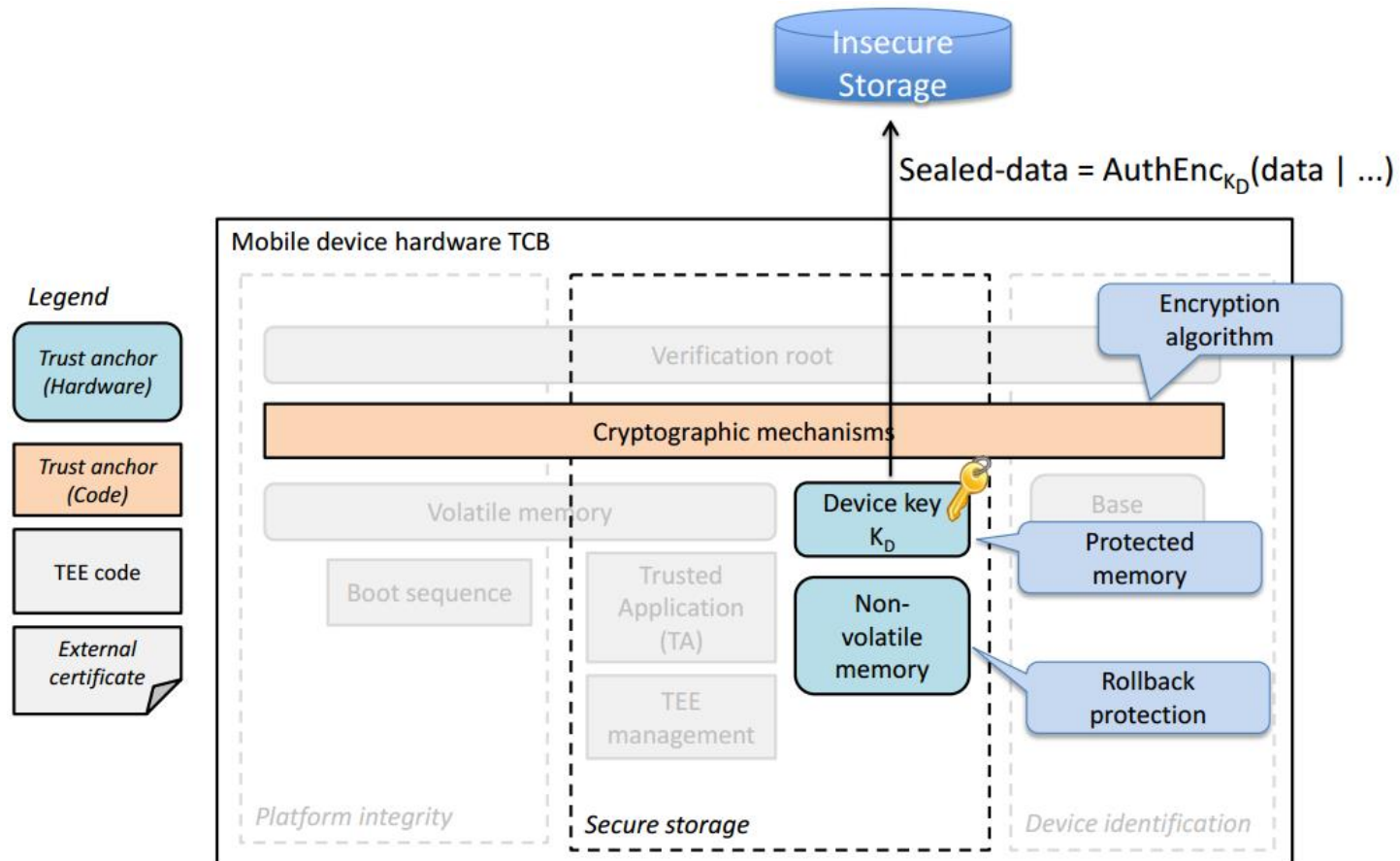
<http://asokan.org/asokan/Padova2014/tutorial-mobileplatsec.pdf>

Device Authentication



<http://asokan.org/asokan/Padova2014/tutorial-mobileplatsec.pdf>

Secure Storage



<http://asokan.org/asokan/Padova2014/tutorial-mobileplatsec.pdf>

Types of TEEs

Intel TXT

Intel SGX

ARM TrustZone

TXT

What is Trusted Execution Environment (TXT)?

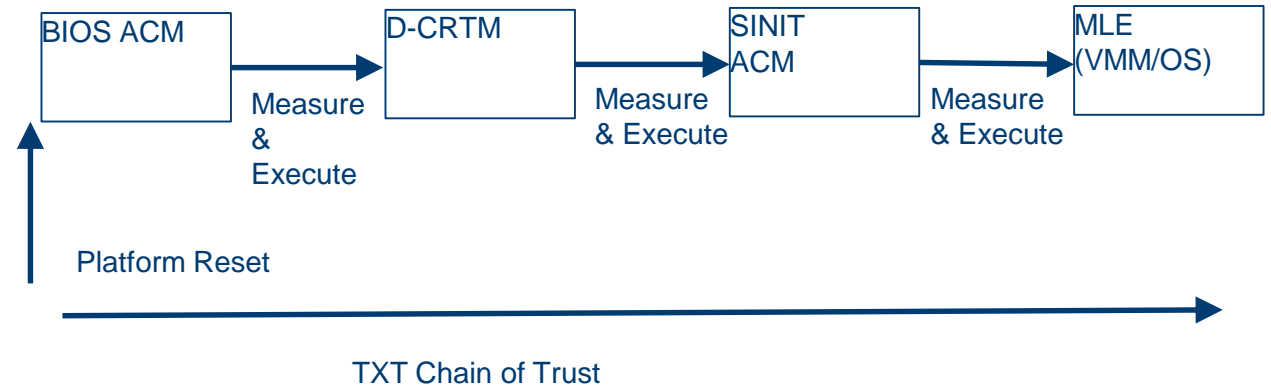
Intel based TEE, formerly Intel LaGrande technology

Hardware based support to provide Dynamic Root of Trust for Measurement (DRTM) and/or verification

Enhanced platform extensions (SMX) to provide launch and control interfaces

- Late launch of Measured Launch Environment (MLE)

Designed to launch a VMM without platform reset (unlike SRTM)



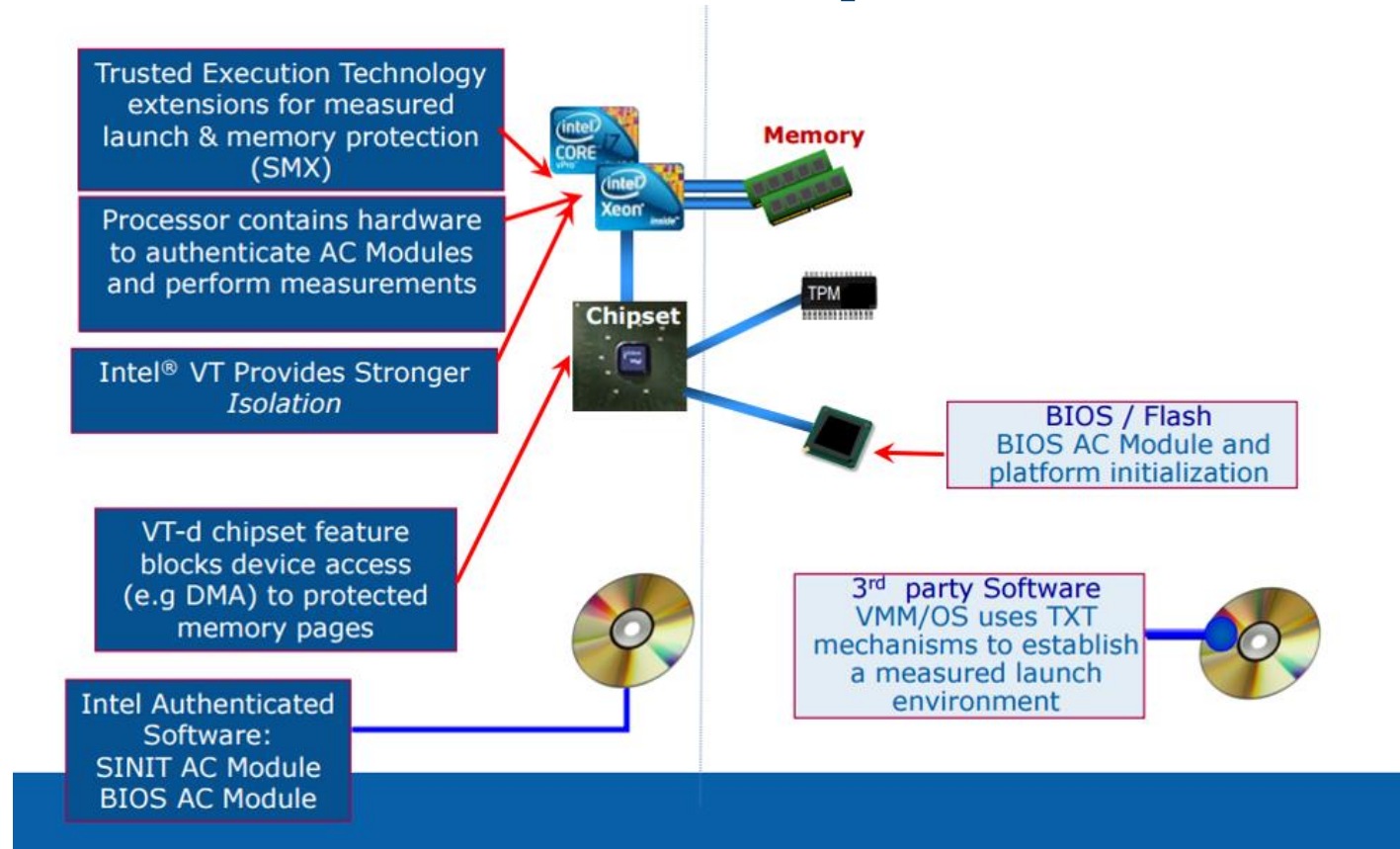
Extends SRTM

- Measures code at launch time (no protection from run time corruption)
- CRTM runs at boot or platform reset

Problems with SRTM?

- Need to measure every possible piece of code that was executed since system boot
- doesn't scale
- Too long of a chain-of-trust

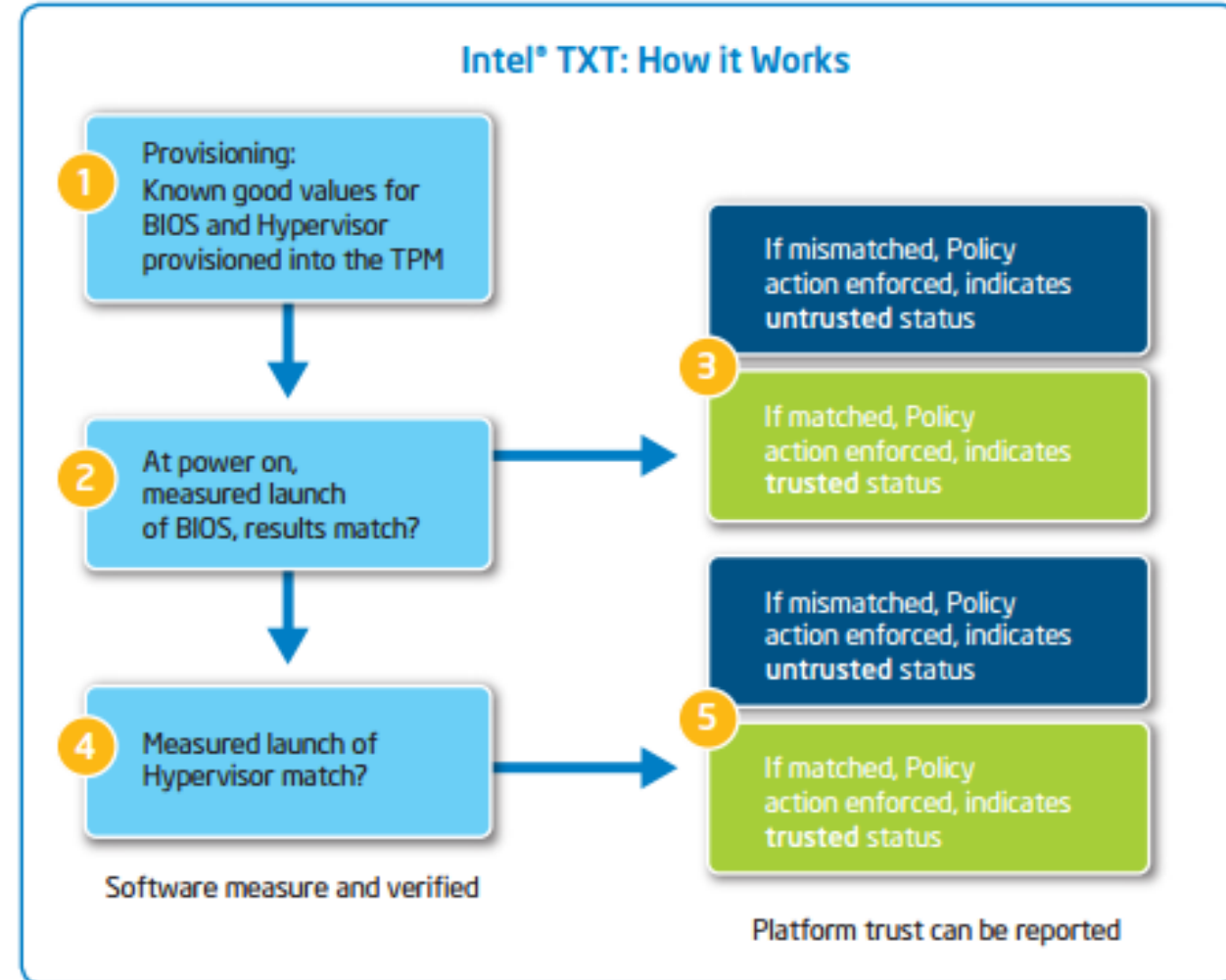
TXT Components



http://tce.webee.eedev.technion.ac.il/wp-content/uploads/sites/8/2015/09/GG_Cyber_conference_TXT_9_15.pdf

Scope of TXT

- Launch “trusted” (measured and/or verified) 3rd party SW later in system runtime without invoking platform reset (vs S-CRTM)
- Memory configuration protection to avoid SW corruption through memory configuration attacks
- SCLEAN – Clean resume (avoid reset based attacks against TPM/TXT)



Safer Mode Extensions (SMX)

Safer Mode Extensions (SMX) provide a programming interface for system software to establish a measured environment within the platform to support trust decisions by end users. The measured environment includes:

- Measured launch of a environment, referred to as a Measured Launched Environment (MLE). Example: a measured VMM is referred to as MVMM2.
- Mechanisms to ensure the above measurement is protected and stored in a secure location in the platform. (TPM usage)
- Protection mechanisms that allow the VMM to control attempts to modify the VMM.

<http://www.intel.com/content/dam/www/public/us/en/documents/manuals/64-ia-32-architectures-software-developer-vol-2d-manual.pdf>

SMX GETSEC instructions and Leaf functions

SMX features in 64bit Intel CPU provide TXT API Via GETSEC instruction via leaf functions.

- GETSEC[CAPABILITIES] - Returns a bit map of available GETSEC leaves /functions
- GETSEC[SENDER]
EBX = SINIT ACM_BASE , ECX = SINIT ACM_SIZE,
EDX = 0
- Authenticates SINIT ACM *
- Measures ACM and extends in TPM PCR#17
- GETSEC[ENTERACCS] leaf – Loads authenticated ACM of [ECX] bytes at [EBX] address

Table 6-2. GETSEC Leaf Functions

Index (EAX)	Leaf function	Description
0	CAPABILITIES	Returns the available leaf functions of the GETSEC instruction.
1	Undefined	Reserved
2	ENTERACCS	Enter
3	EXITAC	Exit
4	SENDER	Launch an MLE.
5	SEXIT	Exit the MLE.
6	PARAMETERS	Return SMX related parameter information.
7	SMCTRL	SMX mode control.
8	WAKEUP	Wake up sleeping processors in safer mode.
9 - (4G-1)	Undefined	Reserved

<http://www.intel.com/content/dam/www/public/us/en/documents/manuals/64-ia-32-architectures-software-developer-vol-2d-manual.pdf>

TXT Modules

Two code modules are loaded in memory during TXT operations

- SINIT ACM, Intel signed code
 - Used only for measurement and verification
 - SINIT ACM Measures itself and MLE (example: MVMM)
- Measured Launch Environment (MLE)
 - The software launched using the SMX instructions is known as the Measured Launched Environment (MLE). Example: MVMM

<https://www.intel.com/content/dam/www/public/us/en/documents/guides/intel-txt-software-development-guide.pdf>

Authenticated Code Modules (ACM)

1) GETSEC[SENDER] - Loads SINIT ACM to ACEA in CPU internal memory and authenticates.

2) SINIT ACM header contains RSA public key and signature. CPU within ACEA -

2.1) Decrypts $_{RSA_PUB_KEY}$ (RSA signature of SINIT header and user code area) = hash of SINIT ACM

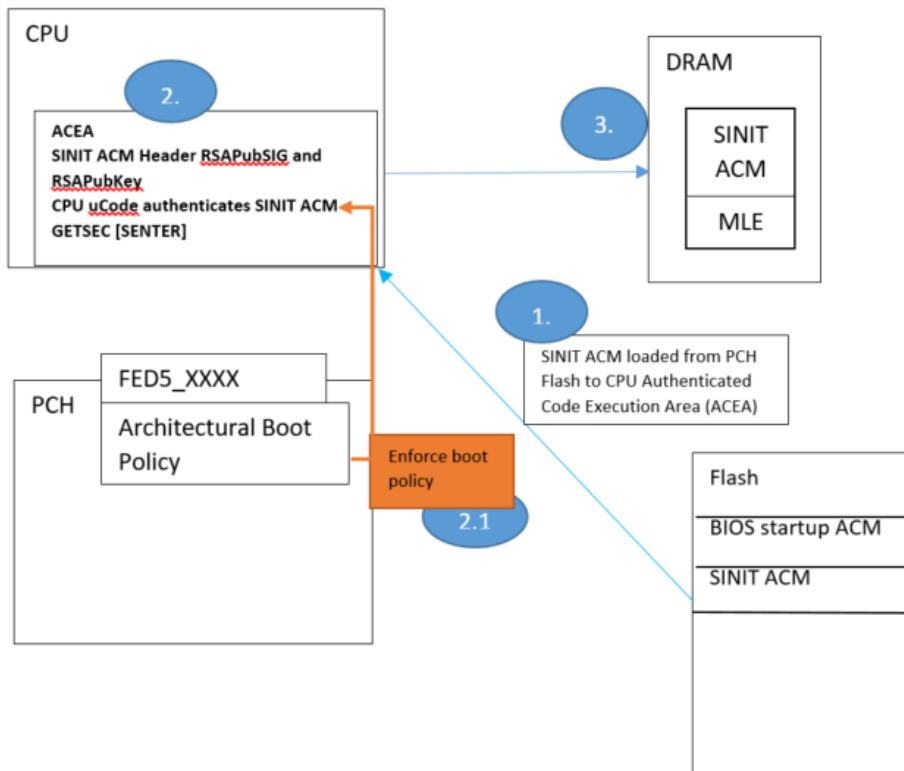
- CPU computes Hash(Loaded SINIT ACM) within ACEA

2.2) Compare hashes. Match?

-Yes - continue boot

-Error - TXT Shutdown (TXT.ERRORCODE)

3) Processor loads SINIT ACM to system memory/DRAM Processor begins ACM execution with EIP = AC Module EntryPoint field + module base address (EBX)



TXT TPM Usage

Root of Storage (RTS): Stored in PCR3

Root of Reporting (RTR)

Legacy and Details/Authorities PCR mappings

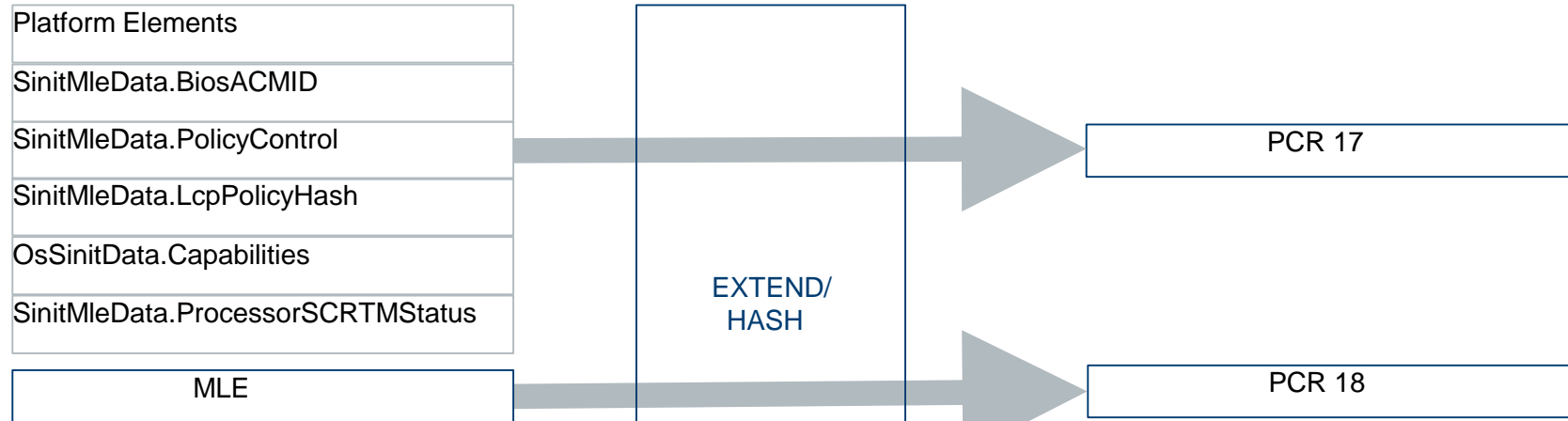
NV Storage access locality protected ensuring BIOS ACM measurement protection (within MLE TCB)

Launch Control Policy storage

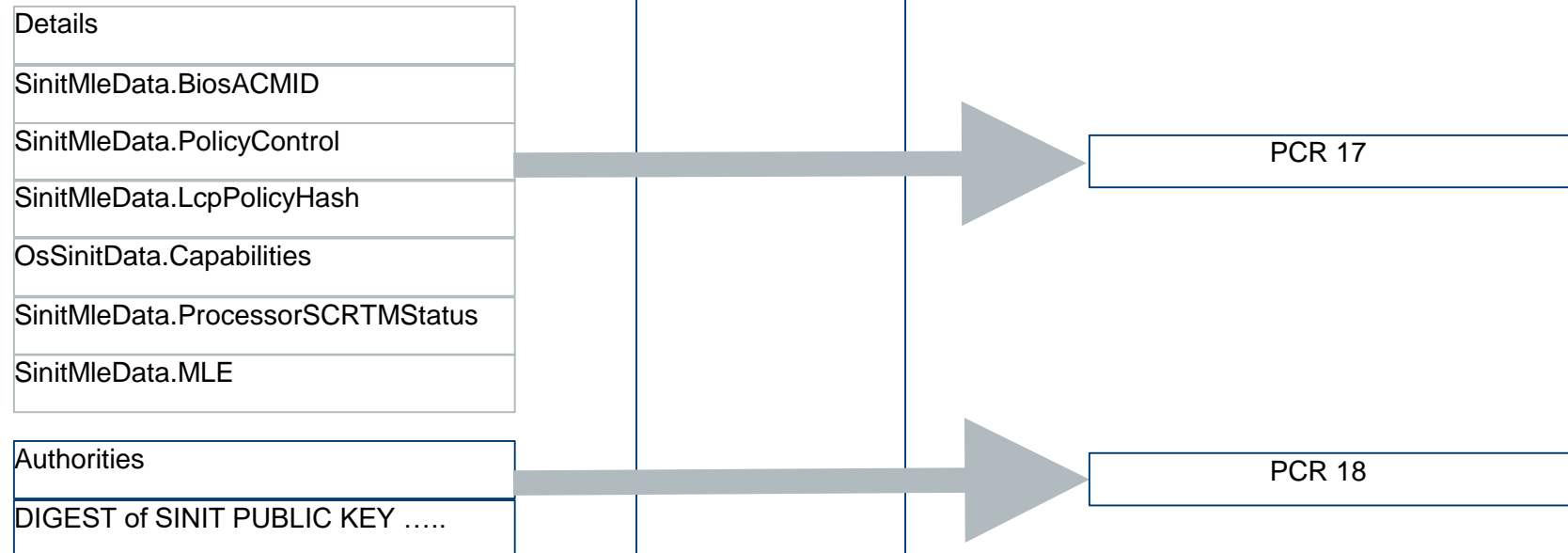
- Platform supplier LCP
- Platform Owner LCP

TXT TPM PCR Mapping

Legacy



Details/authorities



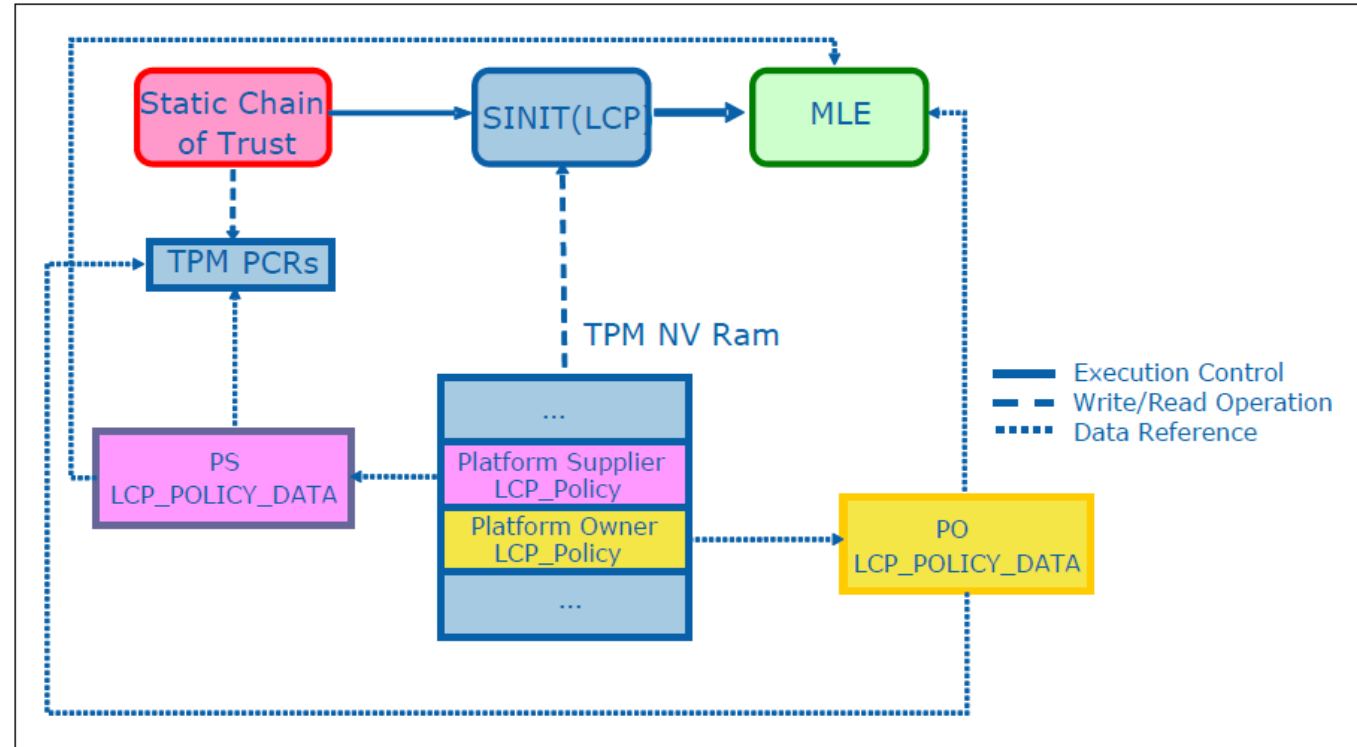
TXT Launch Control Policy (LCP)

Facilitates “Verified” TXT MLE launch

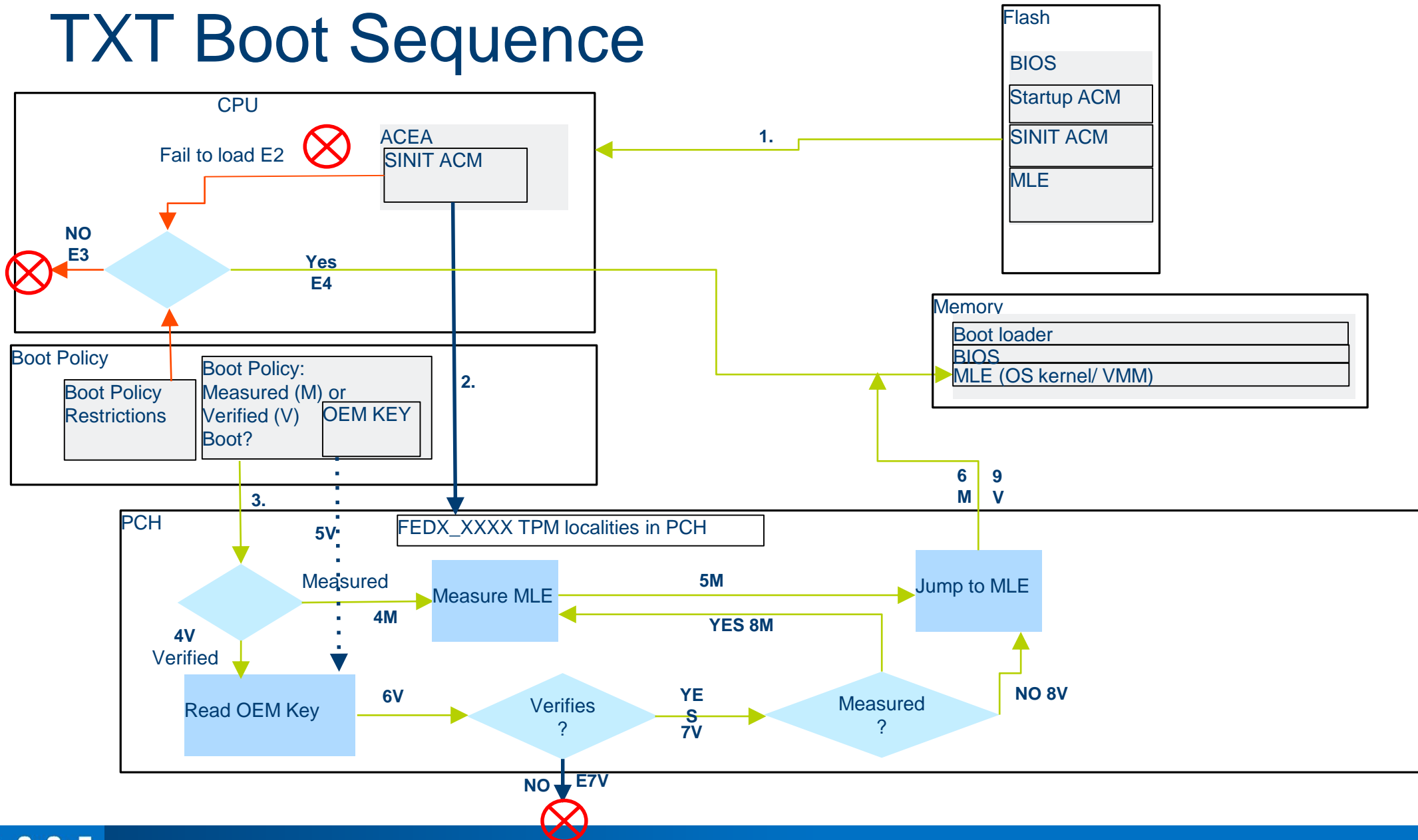
Does the platform configuration and/or MLE meet a specified criteria?:

YES:NO: Take some action!

- LCP Policy Engine (part of SINIT ACM and enforces the policy)
- LCP Policy (stored in TPM, SINIT ACM will enforce these)
- LCP PolicyData objects (list of valid policy elements – measurements of MLE or platform config)



TXT Boot Sequence



TXT vs TPM

TPM is not TXT alternative Or vice-versa.

TPM – Static Root of Trust for Storage (RTS) and Reporting (RTR) of measurements.

Essential for “Trusted Boot” (SRTM based boot).

TXT serves DRTM and heavily relies on TPM for ensuring SRTM, RTS and RTR.

TXT Attack Modelling

TXT is used to launch OS kernel/VMM/MLE as such in a “measured and/or verified” environment without making any assumptions on the current state of the system (can be infected with bootkit/rootkit).

- DMA protection is optional but can work without it. - no isolation of memory from non-measured/verified untrusted device drivers in memory
- TXT does not guarantee runtime protections.
- SMM is the highest execution privilege mode for system software. SMI interrupts enter CPU in SMM Mode. SMM memory is highly access controlled but has access to entire system memory.
- SMI handler code – platform vendor software (Platform BIOS/OS/device driver); Not measured / verified as part of TXT.
- Huge attack surface if SMI handler code is vulnerable and exploitable.

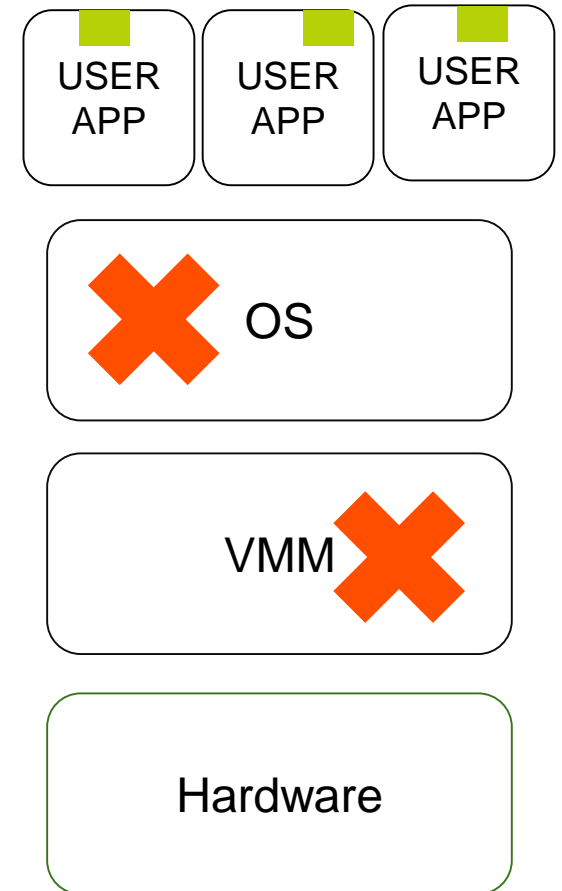
SGX

SGX scope

Protected Ring 3 application address space - Enclave.

With SGX: Reduce attack surface by guarding user space application (Ring 3) address space from OS and VMM

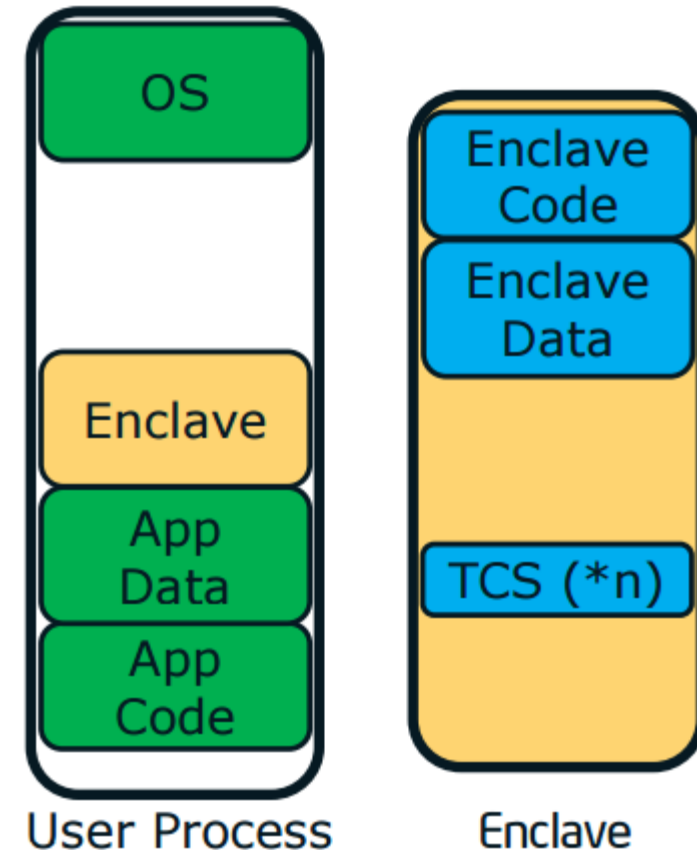
- separation in application address space protected by hardware.
- OS and VMM can manage paging but cannot affect integrity of an enclave pages.
- Integrity, confidentiality and replay protection



SGX and trusted execution environment (TEE)

SGX provides TEE service for user process by providing

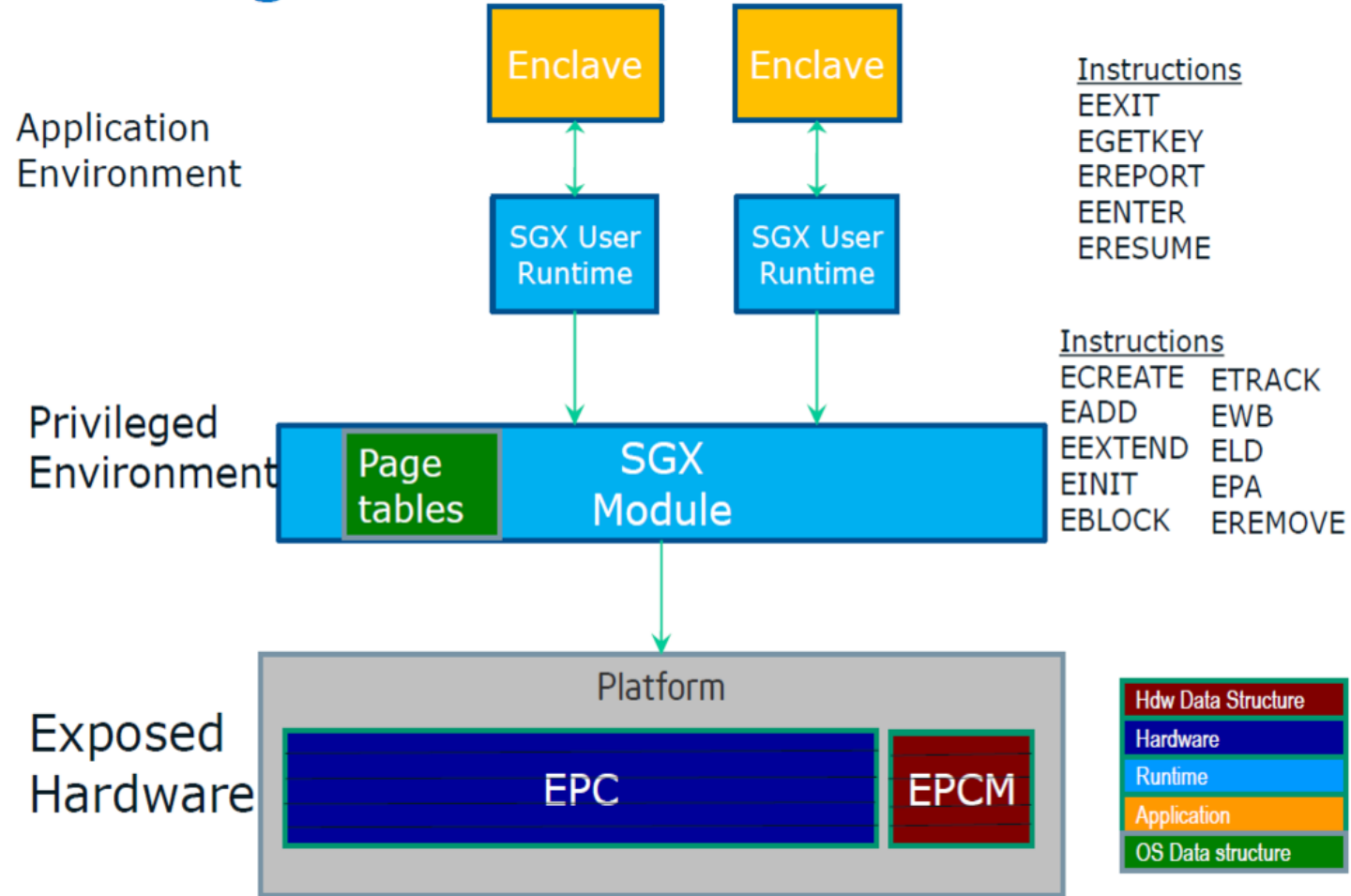
- Protection of enclave code and data in processor
- Protection of enclave code and data in memory
- CPU SGX instruction set
- Memory Encryption Engine
- Hardware anchor for ring 3 code trustworthiness



<https://web.stanford.edu/class/ee380/Abstracts/150415-slides.pdf>

SGX architecture

SGX High-level HW/SW Picture



<https://web.stanford.edu/class/ee380/Abstracts/150415-slides.pdf>

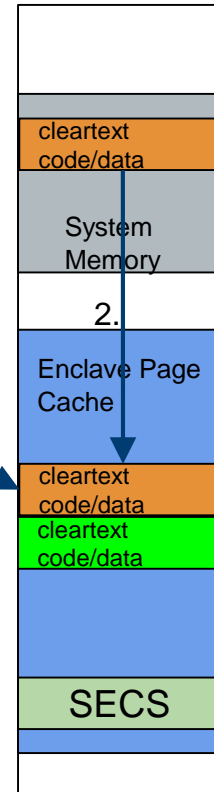
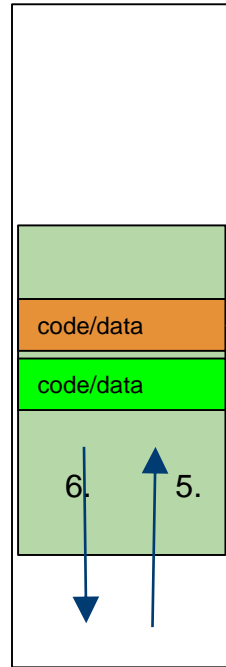
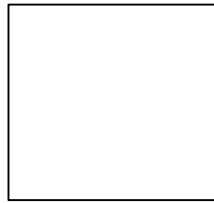
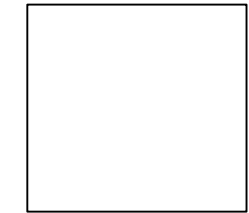
SGX architecture

Virtual Address Space

Physical Address Space

Virtual Address Space

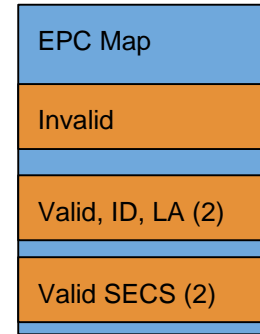
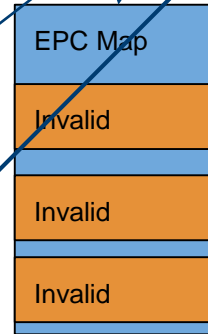
Physical Address Space



memory claimed by HW at reset or stage 0




3.



2. Update PTE

1. ECREATE (Range)
2. EADD (Copy Range)
3. EEXTEND (Measure)
4. EINIT
5. EENTER
6. EEXIT
7. EREMOVE

 Measured code/data pages

 EPC

 Code/Data pages

<https://web.stanford.edu/class/ee380/Abstracts/150415-slides.pdf>

SGX Scope

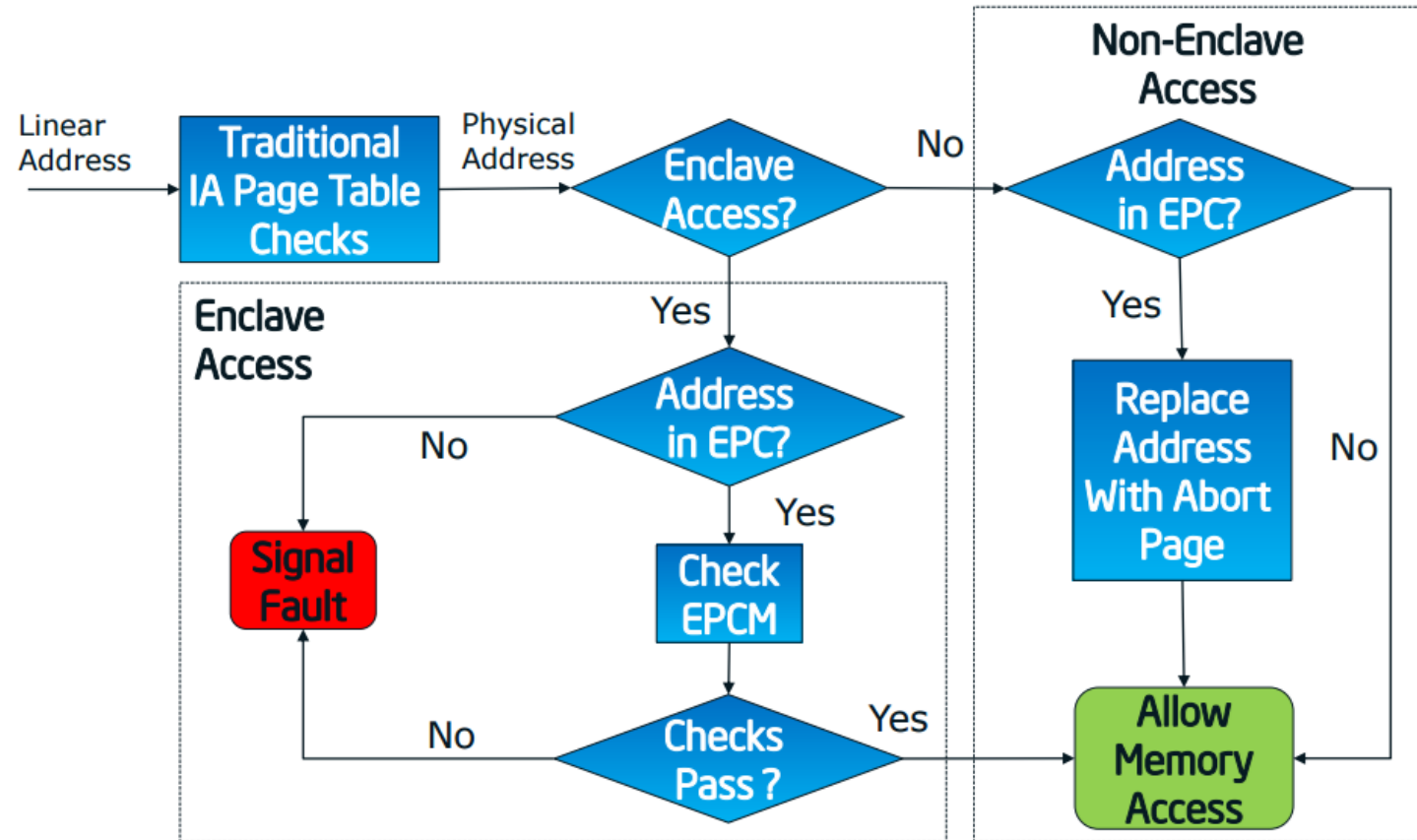
Data and code are unencrypted inside CPU package.

Outside CPU package, data and code are encrypted and/or integrity protected by special hardware MEE.

- Protected against memory snooping. MRENCLAVE is 256 bit digest of enclave measurement log and serves the SW TCB.
 - Crypto to hash the enclave and verify hash. Don't have right measurement, won't get secret.
 - Integrity alteration will be detected.

Enclave paging – Enclave page encrypted out to memory. Bring it back. Metadata in EPC encrypted also.

Attestation services with Intel Quoting enclaves.



<https://web.stanford.edu/class/ee380/Abstracts/150415-slides.pdf>

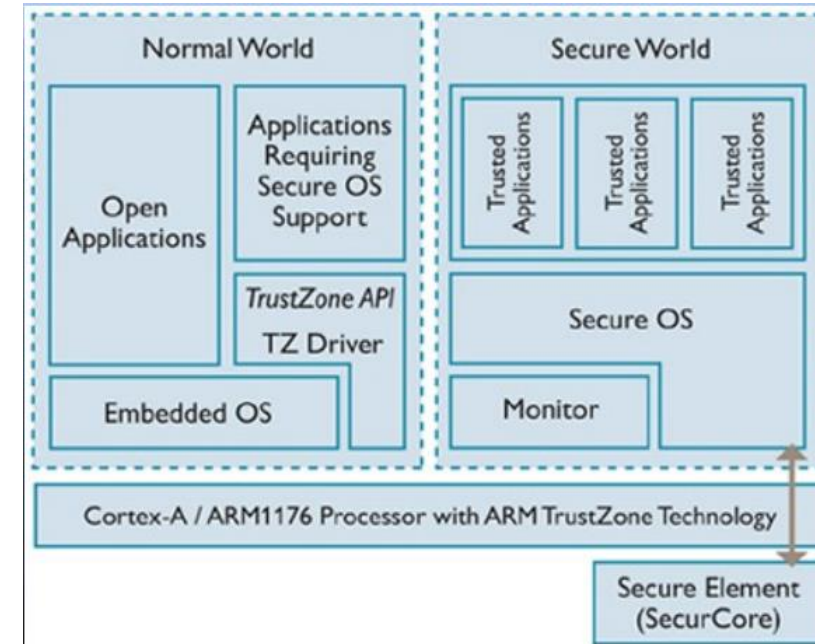
TrustZone

TrustZone

ARM TrustZone is a SoC TEE

Hardware-based security

- Processor based isolation
- Secure / Non-secure world
- Secure Monitor is used to switch execution modes
- Encompasses processor, memory, software, bus transactions, interrupts, debug and peripherals



<https://www.arm.com/products/security-on-arm/trustzone>

So?

Embedded devices are handling sensitive data

Open platforms with a lot of 3rd party code being ran

TrustZone allows a dedicated processing block

Partitioned SoC hardware and software resources in the two worlds

ARM Processor cores contain TrustZone extensions so a single processing core can time-slice secure and normal world

- Accomplished with the AMBA3 AXI system bus and control signals for read and write channels (NS Bits)

Architecture Explained

L1 Memory System partitions the TLB and MMU with NS bits to distinguish worlds.

L2 memory mapping is in secure / non-secure regions

Executing in Secure World

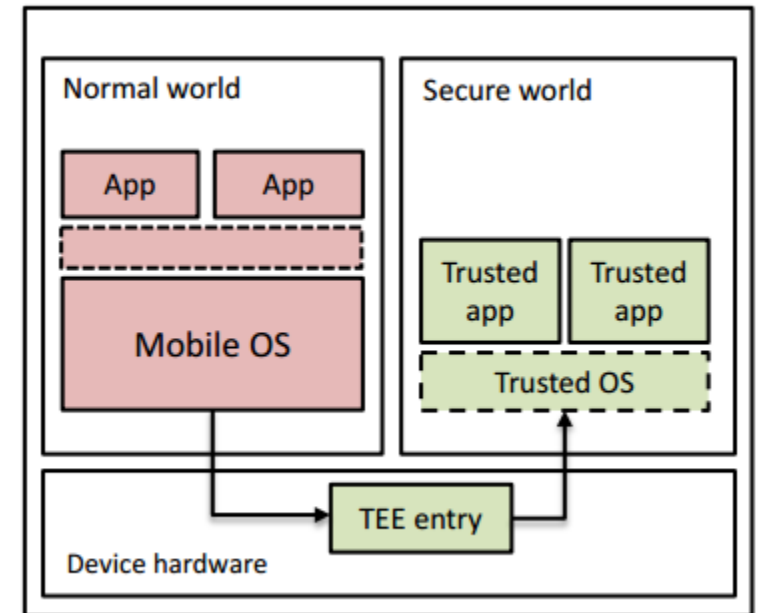
- Secure Monitor Call (SCM)
 - Entry to the secure monitor to determine whether or not to switch processing world
- Interrupts
- External Aborts

Architecture Explained . . .

User and Kernel mode is defined by the Current Program Status Register (CPSR)

The "Worlds" are defined by the Secure Configuration Register

- Used by the Secure Monitor (Lives in Secure World)
- Defines secure or non-secure worlds
- The world that is executing
- $SCR[0] = NS$
 - 0 = Secure; 1 = Non-secure



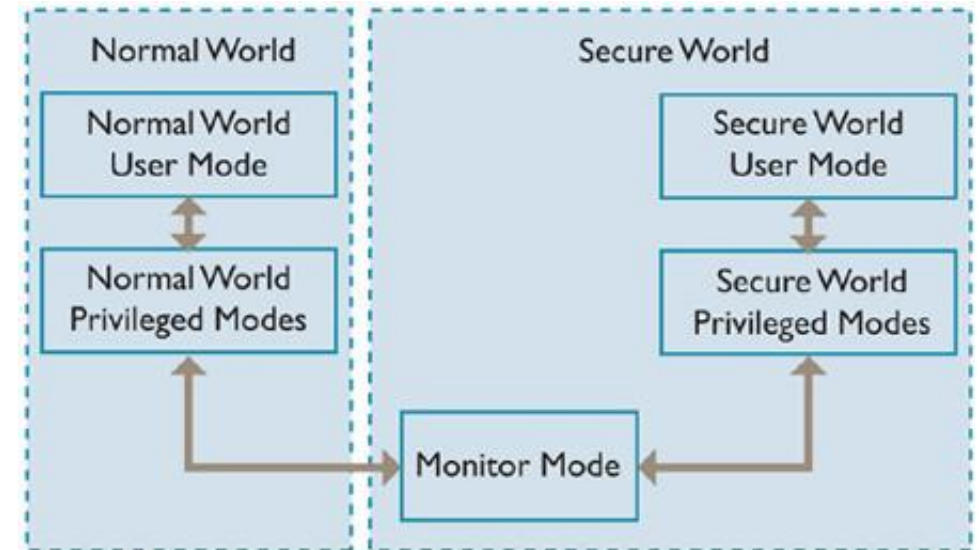
TrustZone Monitor

Secure Mode regardless of NS bit

Entry is mostly gained by the Secure Monitor Call (SMC) to perform a service call.

Determines if the change from non-secure to secure mode is valid and controls the NS bit on the processor (recommended).

Saves processor context on switch



<http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0333h/Chdfjdgi.html>

QSEE

Qualcomm Secure Execution Environment (QSEE)

QSEE is the most wide spread TEE built on top of ARM TrustZone

- Mobile, Mobile, Mobile
- Android, Android, Android

Provides to Android

- Verifying kernel integrity (TIMA)
- Using the Hardware Credential Storage (used by "keystore", "dm-verity")
- Secure Element Emulation for Mobile Payments
- Implementing and managing Secure Boot
- DRM (e.g. PlayReady)
- Accessing platform hardware features (e.g. hardware entropy)

QSEE TrustZone internals

From Qualcomm

"...a system-wide approach to security for a wide array of client and server computing platforms, including handsets, tablets, wearable devices and enterprise systems. Applications enabled by the technology are extremely varied but include payment protection technology, digital rights management, BYOD, and a host of secured enterprise solutions."

Qseecom driver accessible by the following processors

- surfaceflinger (running with "system" user-ID)
- drmserver (running with "drm" user-ID)
- mediaserver (running with "media" user-ID)
- keystore (running with "keystore" user-ID)

Trustlets

The executable binaries in secure mode are “Trustlets”

The Normal World holds the binaries and a Trustlet is loaded by "QSEECOM_start_app":

TrustZone loads the trustlet segments into secapp-region and assigns an ID to it

- Afterward the entry function is loaded

QSEOS_CLIENT_SEND_DATA_COMMAND with the Trustlet ID, request and response buffers is used to interact with a trustlet

TrustZone Attack Modeling

SCM (Secure Channel Manager) is used to interact with “Secure” mode from “Normal” mode

SMC (Secure Monitor Calls) used to transfer execution from normal to secure mode are a large attack surface

- The SMM of TrustZone for you IA nerds

Shared Memory

Hardware Exceptions / Interrupt Handlers

eMMC Flash (Secure boot in theory protects this asset)

- Regular and Atomic

So what?

Attackers have been able to exploit vulnerabilities in trustlets to obtain secure mode code execution

Secure Mode can write anything to Normal mode

- `qsee_on_ns_memory()` or `qsee_range_not_in_region()` can be used incorrectly

Having a secure processing mechanism requires attention to implementation details

- Just because you have a secure thing doesn't mean anything if used incorrectly

Operating Systems

Windows 10

Several windows features require TPM 2.0 and TPM firmware or discrete TPM must be enabled by default for Windows 10 Anniversary

- Bitlocker
- Virtual Smart Card
- Credential Guard
- Measured Boot
- Device Health Attestation

Windows 10 ...

Win 10 Feature	TPM 1.2	TPM 2.0	Details
UEFI Secure Boot			<ul style="list-style-type: none"> No TPM requirement
Conditional Access			<ul style="list-style-type: none"> No TPM requirement
Enterprise Data Protection			<ul style="list-style-type: none"> No TPM requirement
Windows Defender - Advanced Threat Detection			<ul style="list-style-type: none"> No TPM requirement
Device Guard / Configurable Code Integrity			<ul style="list-style-type: none"> No TPM requirement
Windows Hello			<ul style="list-style-type: none"> No TPM requirement
Credential Guard	Yes	Yes	<ul style="list-style-type: none"> More secure with TPM 2.0
Measured Boot	Yes	Yes	<ul style="list-style-type: none"> More secure with TPM 2.0
Device Health Attestation	Yes	Yes	<ul style="list-style-type: none"> Requires TPM
Virtual Smart Card	Yes	Yes	<ul style="list-style-type: none"> Requires TPM
Passport: Domain AADJ Join	Yes	Yes	<ul style="list-style-type: none"> Supports both versions, but requires TPM with HMAC and EK certificate for key attestation support.
Passport: MSA / Local Account	Yes	Yes	<ul style="list-style-type: none"> Requires TPM 2.0 for HMAC and EK certificate for key attestation support
BitLocker	Yes	Yes	<ul style="list-style-type: none"> TPM 1.2 or later required or a removable USB memory device such as a flash drive
Device Encryption		Yes	<ul style="list-style-type: none"> For Modern Standby devices, all require TPM 2.0

https://sec.ch9.ms/slides/winHEC/03_WindowsSecurity.pdf

TPM and GPO

Windows 8.1+ has GPO settings for TPM configuration

- Computer Configuration\Administrative Templates\System\Trusted Platform Module Services\
 - Mostly sets TPM backup to AD Services, blocked commands...
 - Turn on bitlocker
- Useful to deploy things like Credential Guard

TPM and Powershell

```
PS C:\windows\system32> Get-Tpm
```

```
TpmPresent      : True
TpmReady       : False
ManufacturerId  : 1229346816
ManufacturerVersion : 4.40
ManagedAuthLevel : Delegated
OwnerAuth      :
OwnerClearDisabled : True
AutoProvisioning : Enabled
LockedOut      : False
SelfTest       : {128, 0, 1, 255}
```

Cmdlet	Description
Clear-Tpm	Resets a TPM to its default state.
ConvertTo-TpmOwnerAuth	Creates a TPM owner authorization value.
Disable-TpmAutoProvisioning	Disables TPM auto-provisioning.
Enable-TpmAutoProvisioning	Enables TPM provisioning for the next restart.
Get-Tpm	Gets an object that contains information about a TPM.
Get-TpmEndorsementKeyInfo	Gets information about the endorsement key and certificates of the TPM.
Get-TpmSupportedFeature	Verifies whether a TPM supports specified features.
Import-TpmOwnerAuth	Imports a TPM owner authorization value to the registry.
Initialize-Tpm	Performs part of the provisioning process for a TPM.
Set-TpmOwnerAuth	Changes the TPM owner authorization value.
Unblock-Tpm	Ends a TPM lockout.

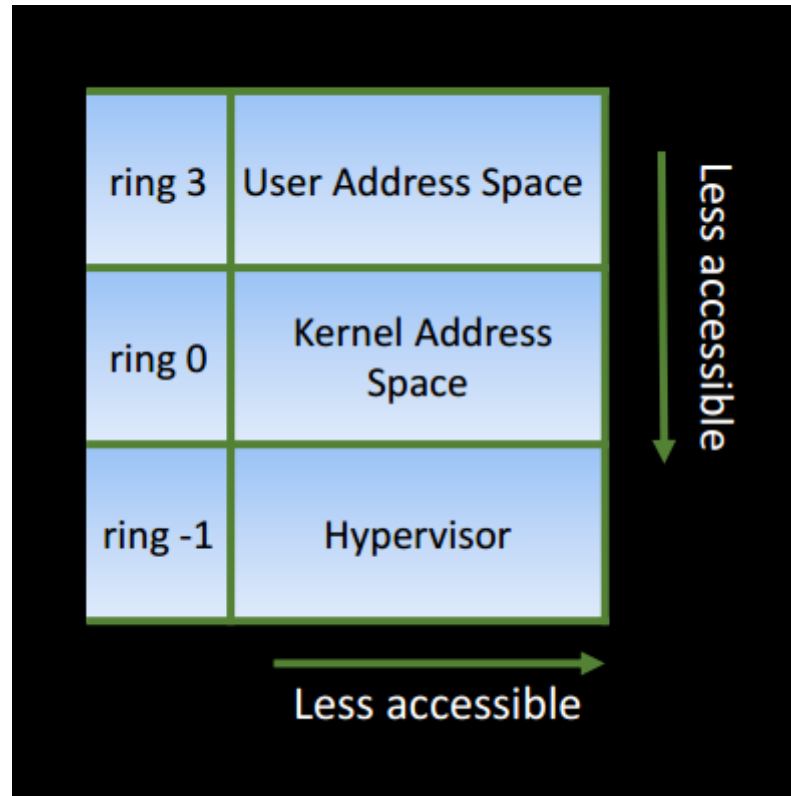
Windows Virtualization Based Security

VSM platform requirements

- Virtualization Extensions (Intel VT-x)
- Second Level Address Translation, SLAT (Intel Extended Page Tables, EPT)
- IOMMU (Intel VT-d)
- UEFI 2.3.1
- TPM 2.0
- Optional Performance Enhancement - Mode Based Execution Control (MBEC)
- Optimal performance for CI enforcement
- Fall-back to S/W based implementation

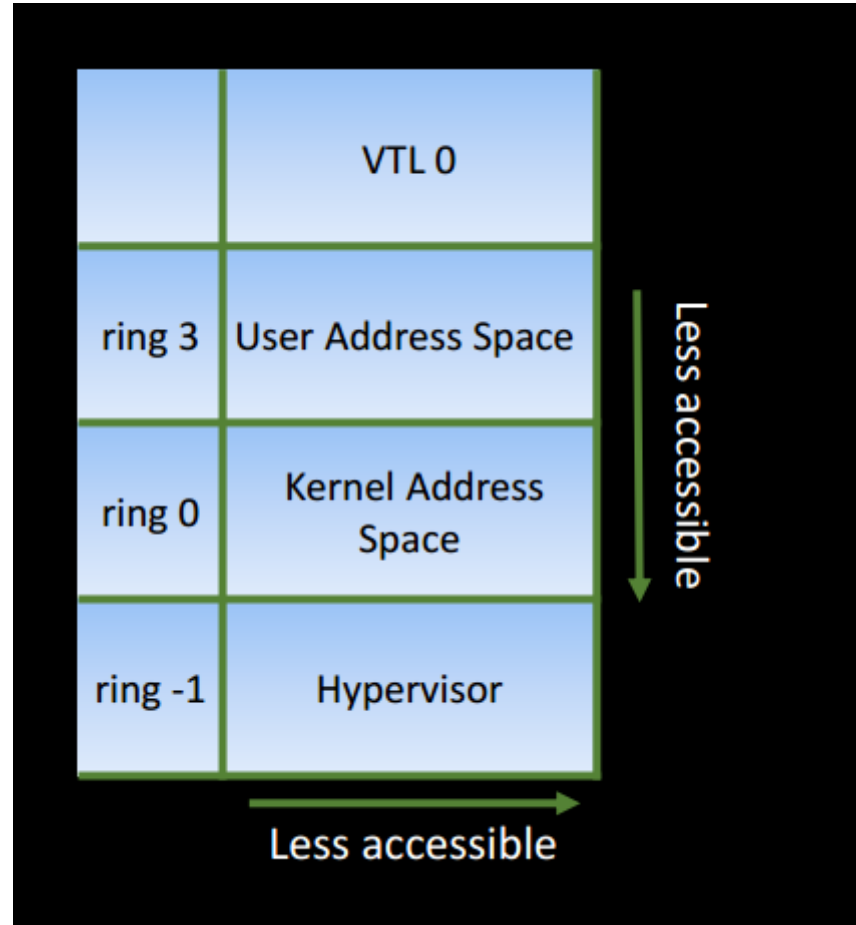
<https://www.blackhat.com/docs/us-15/materials/us-15-Moore-Defeating%20Pass-the-Hash-Separation-Of-Powers.pdf>

Windows Virtualization Based Security



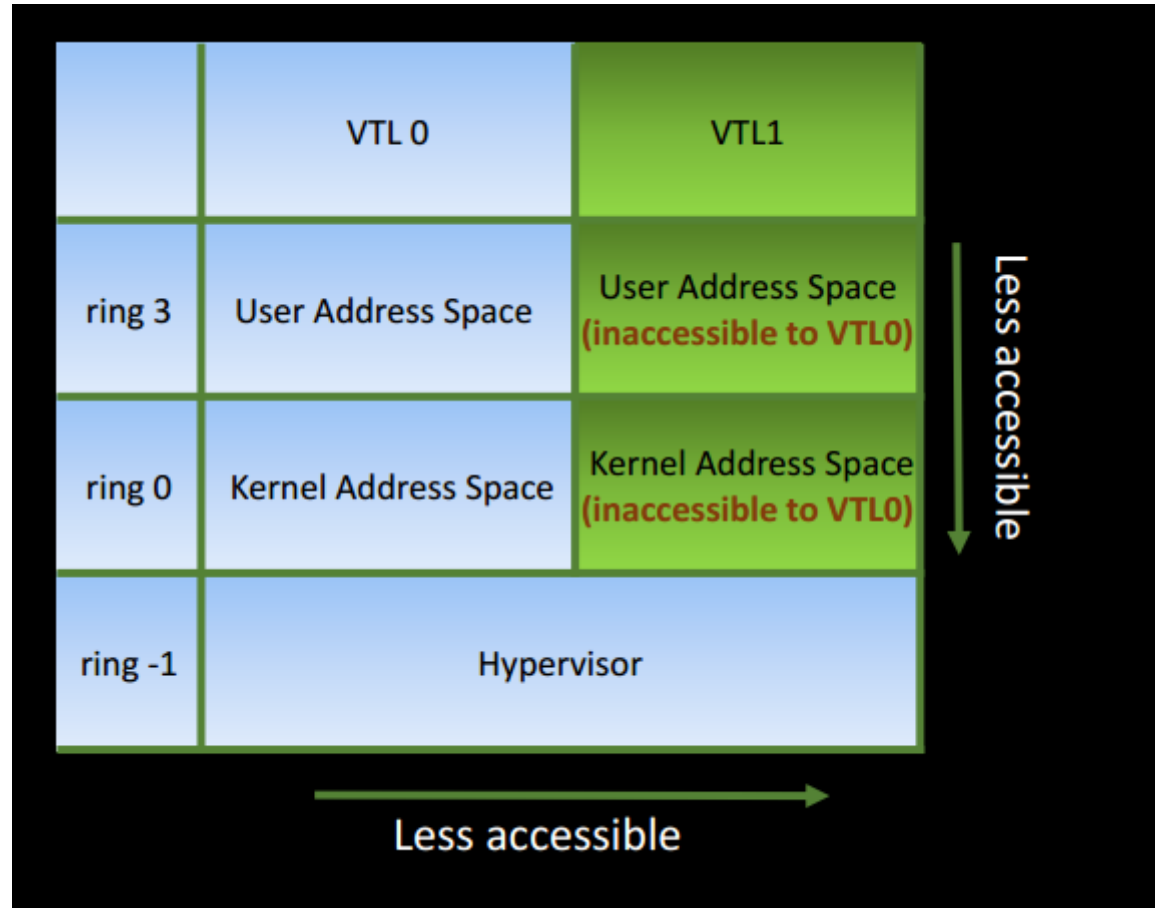
<https://www.blackhat.com/docs/us-15/materials/us-15-Moore-Defeating%20Pass-the-Hash-Separation-Of-Powers.pdf>

Windows Virtualization Based Security



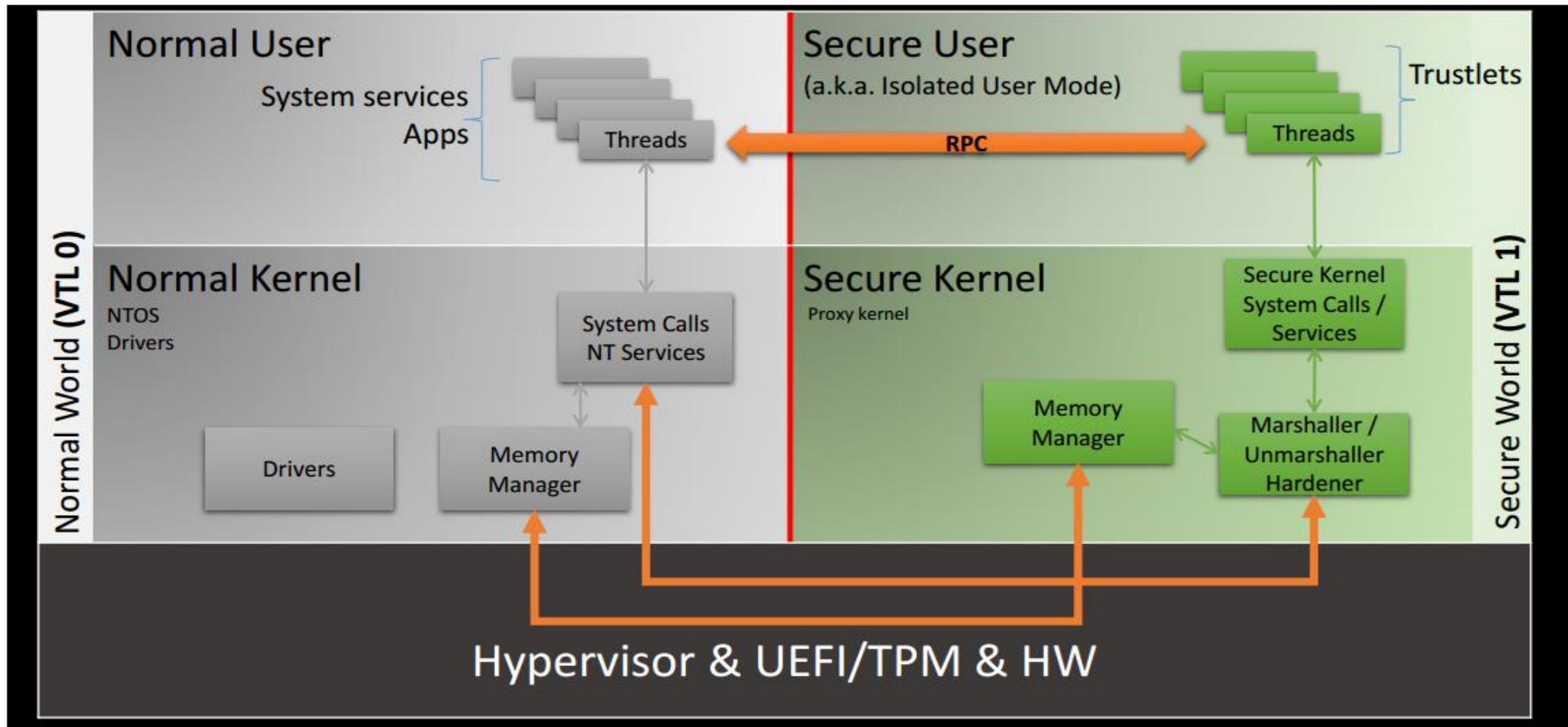
<https://www.blackhat.com/docs/us-15/materials/us-15-Moore-Defeating%20Pass-the-Hash-Separation-Of-Powers.pdf>

Windows Virtualization Based Security



<https://www.blackhat.com/docs/us-15/materials/us-15-Moore-Defeating%20Pass-the-Hash-Separation-Of-Powers.pdf>

Windows Virtualization Based Security



<https://www.blackhat.com/docs/us-15/materials/us-15-Moore-Defeating%20Pass-the-Hash-Separation-Of-Powers.pdf>

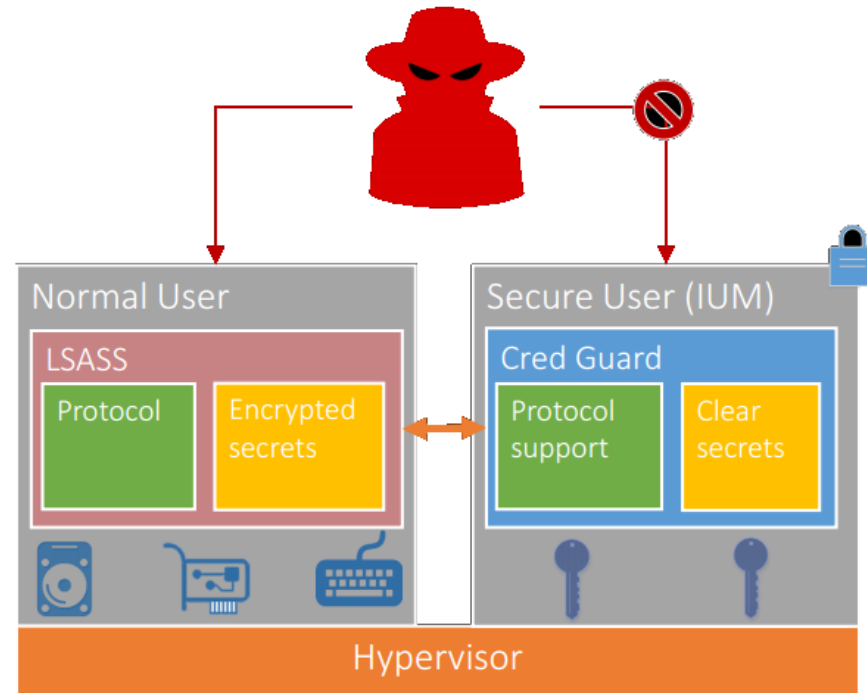
Credential Guard

Utilizes VBS to isolate secrets (NTLM password hashes and Kerberos credentials) that have lead to Pass-The-Hash attacks (see Mimikatz)

TPM 1.2 and 2.0 provides protection for encryption keys that are stored in the firmware. TPMs, either discrete or firmware will suffice, but this is a must have requirement for Credential Guard.

TPM provides protection for the VBS encryption keys.

Credential Guard



<https://www.blackhat.com/docs/us-15/materials/us-15-Moore-Defeating%20Pass-the-Hash-Separation-Of-Powers.pdf>

S4 Sleep

Suspend-to-disk sleep (Hibernation)

All devices are powered off and a hibernation file, `hiberfile.sys`, is written to disk containing an image of memory.

With VBS, the hiberfile is encrypted when entering S4 and the secret is sealed into the TPM

No TPM? Seal the key in UEFI variable

- Don't do that

Not only does S4 shut down the machine, but if Bitlocker is enabled it also will prompt for authentication.

Conclusion

Questions?