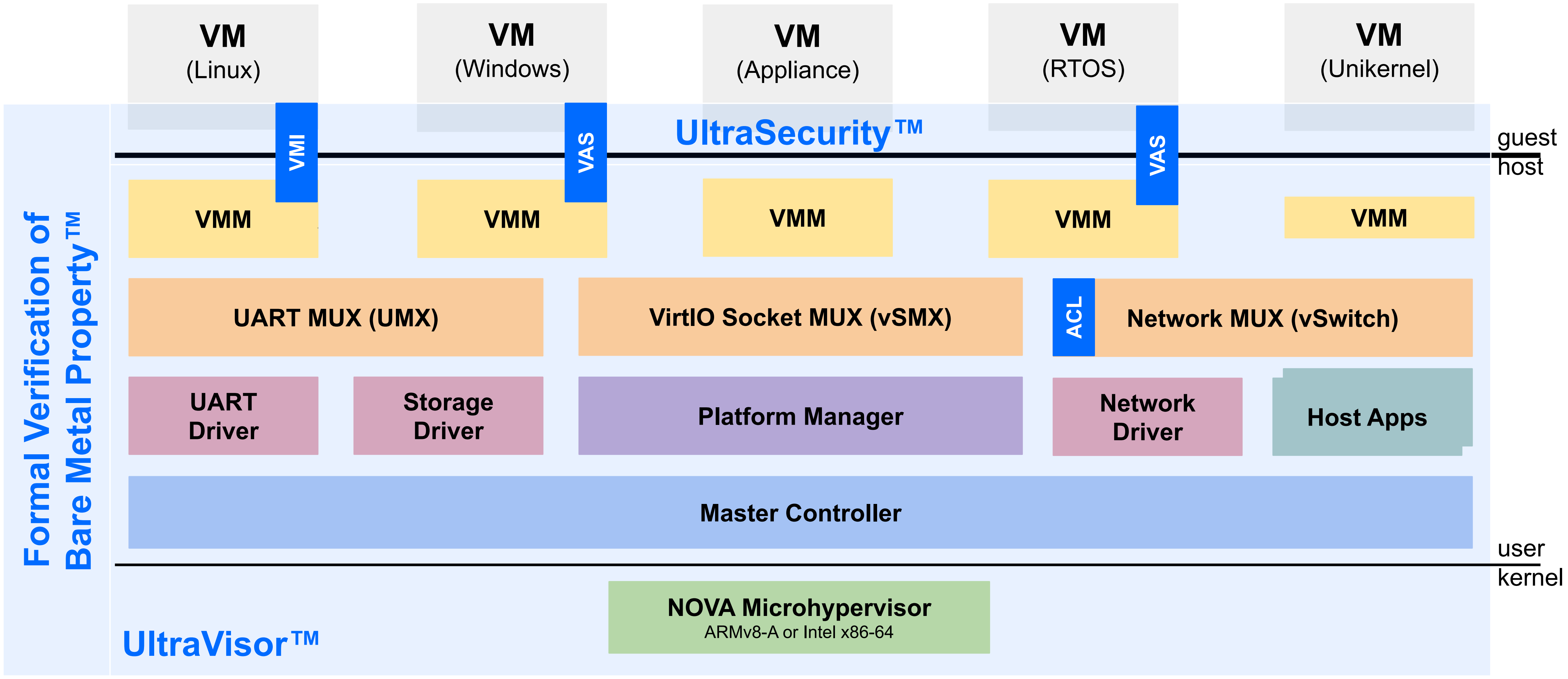
BedRock
Systems Inc

NOVA Microhypervisor Measured Launch

Udo Steinberg

BedRock Ultravisor Architecture



BedRock Systems Inc

Problem Statement

- Once you have a formally verified Ultravisor
  - … and a compiler that produced a correct set of binaries for the target architecture
- How do you ensure that some computer is running **those binaries**
  - … and not some other (malicious) software instead
  - … before you entrust that computer with your precious data
- In other words, how can you **establish trust** in the platform?
  - Either restrict the software that a computer will launch
  - Or determine what software has been launched on a computer

BedRock Systems Inc

Quick Intro: Trusted Computing

Terminology

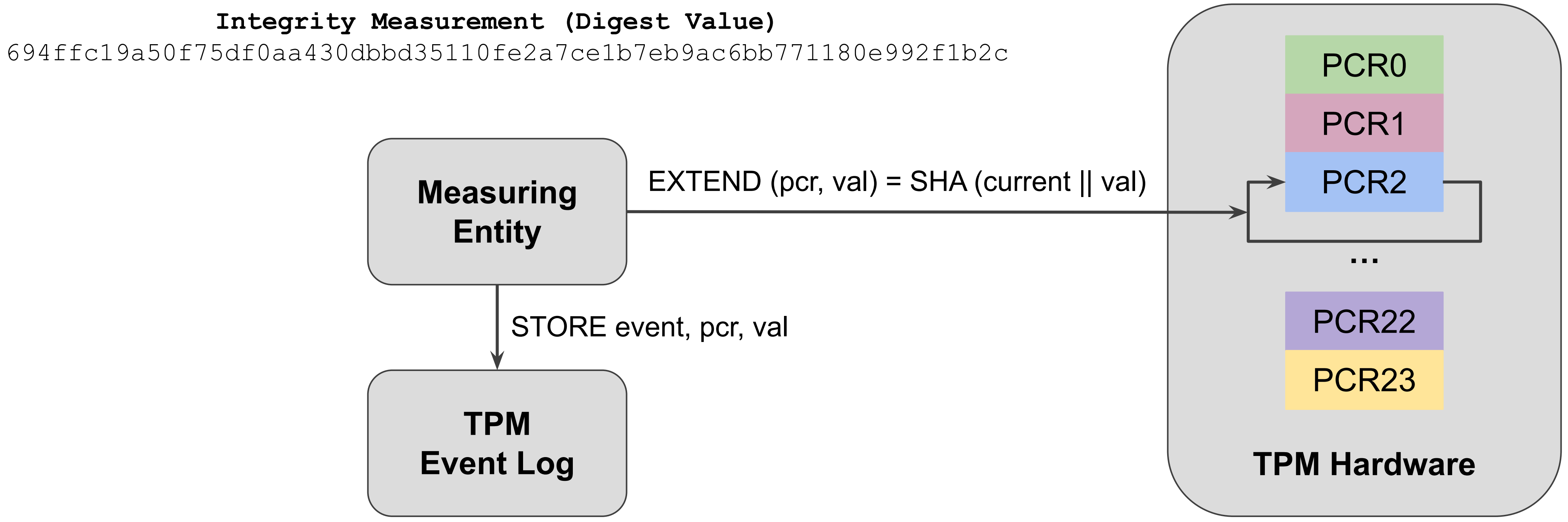
- **Trust**
  - Trusted Computing Group Definition: An entity can be trusted if it always behaves in the expected manner for the intended purpose.
- **Root of Trust**
  - A component that performs one or more security-specific functions, such as measurement, storage, reporting, verification and/or update. It is trusted to behave in the expected manner, because its misbehavior cannot be detected under normal operating conditions.

⇒ RoT is typically some immutable component, e.g. hardware or ROM

Trusted Platform Module (TPM)

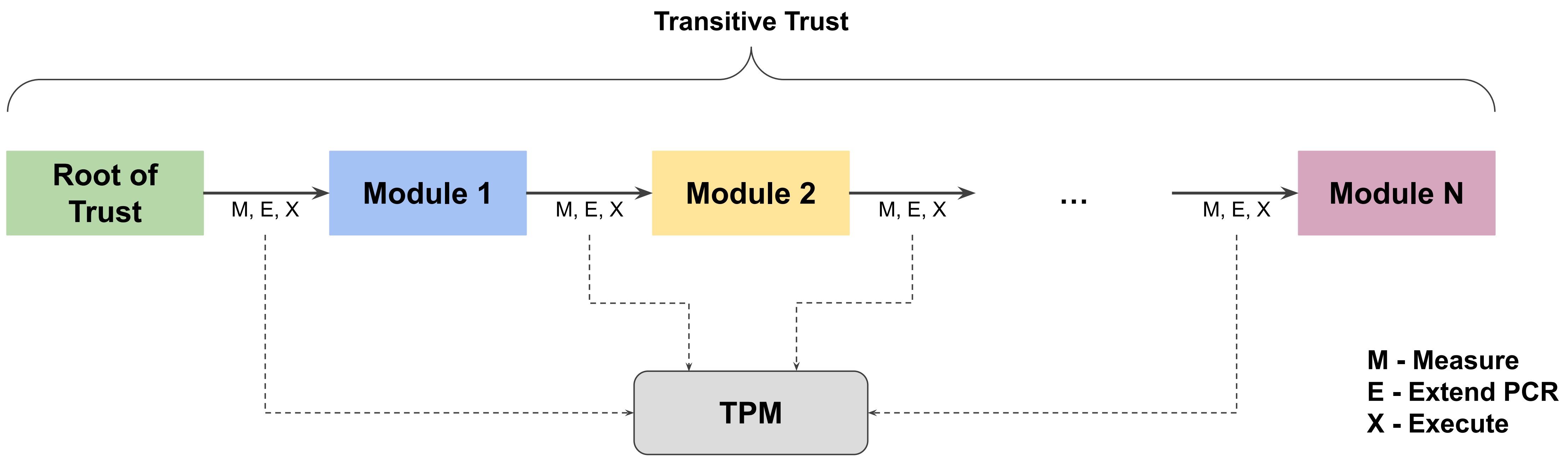
- Tamper-Resistant Opt-In Attestation Device for the Platform
  - Discrete TPM (dTPM)
  - Firmware TPM (fTPM), e.g. Intel PTT
- Building Blocks
  - Cryptographic Functions
  - Random Number Generator
  - Platform Configuration Registers (PCRs)
  - Non-Volatile Secure Storage (limited capacity)
  - Sealed Storage

TPM Platform Configuration Registers (PCRs)


Integrity Measurement (Digest Value)
694ffc19a50f75df0aa430dbbd35110fe2a7ce1b7eb9ac6bb771180e992f1b2c
Measuring Entity
EXTEND (pcr, val) = SHA (current || val)
STORE event, pcr, val
TPM Event Log
PCR0
PCR1
PCR2
...
PCR22
PCR23
TPM Hardware

Event Log can be used to recompute (validate) the current value of each PCR

Chain of Trust



❖ Integrity measurement is a cryptographic hash ⇒ unique + indicative to changes in the module

❖ Chain of Trust uses integrity measurements to determine trustworthiness of software stack

Verified vs. Measured Boot

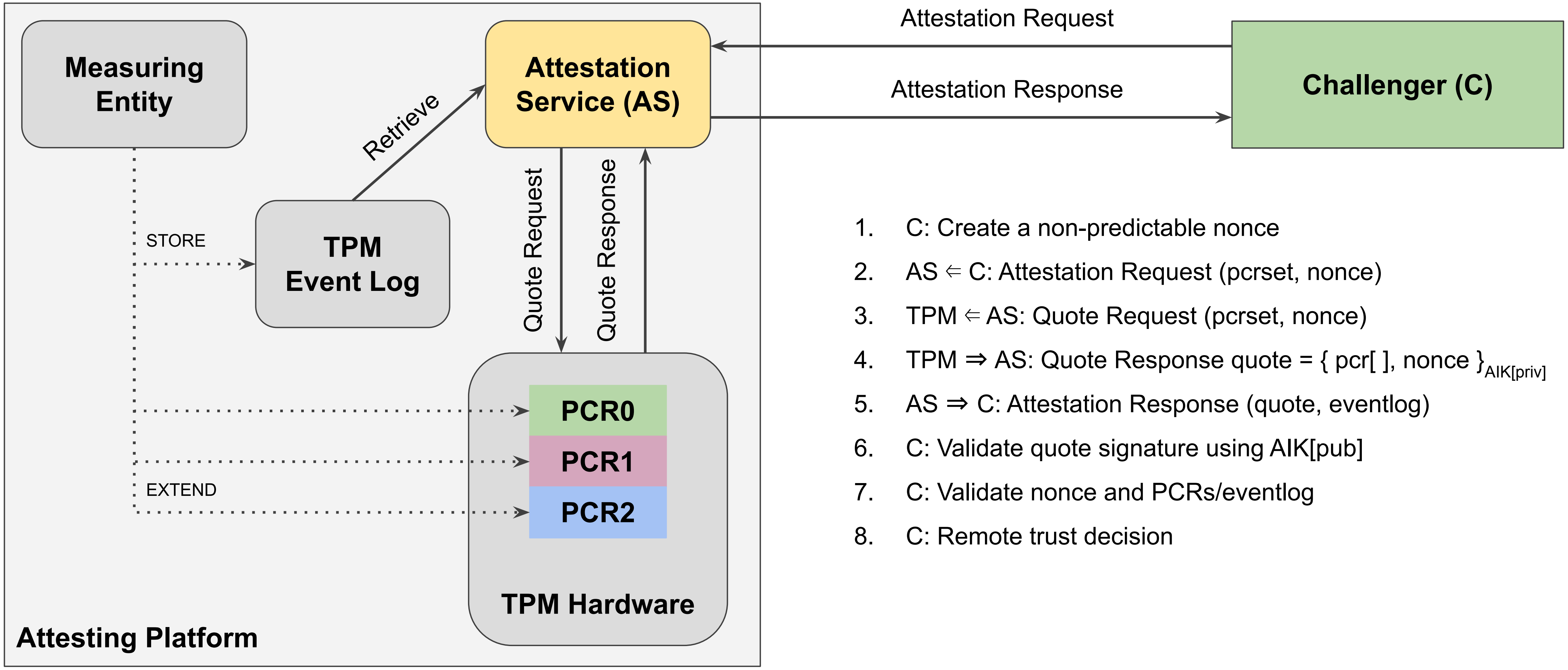
- **Verified Boot**
  - Boot policies are enforced during the boot process.
  - Starting with the Core Root of Trust for Verification (CRTV), the currently executing module verifies the next module against a policy (i.e. manifest).
- **Measured Boot**
  - Integrity measurements are stored in the TPM during the boot process.
  - Starting with the Core Root of Trust for Measurement (CRTM), the currently executing module extends the integrity measurement for the next module into the TPM.

⇒ Secure Boot could mean either of those mechanisms, or a combination of both

Attestation

- Prove platform properties to a challenger
  - Currently operating Hardware / Software / Configuration
- Attestation Device Requirements
  - Accurate Measurements
  - Storage of the Measurements
  - Verifiable Report of the Measurements
  - Tamper-Resistant Storage and Reporting Mechanisms
- Trusted Platform Module (TPM) fulfills these requirements

Remote Attestation (Example)

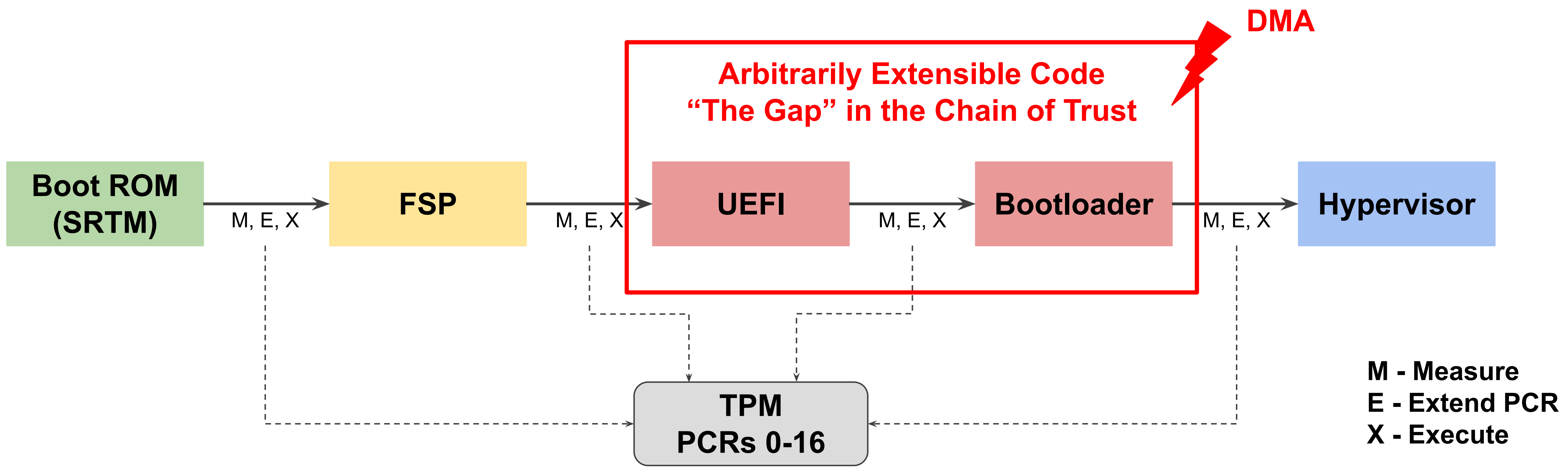


1. C: Create a non-predictable nonce
2. AS ⇐ C: Attestation Request (pcrset, nonce)
3. TPM ⇐ AS: Quote Request (pcrset, nonce)
4. TPM ⇒ AS: Quote Response quote = { pcr[ ], nonce }$_{\text{AIK[priv]}}$
5. AS ⇒ C: Attestation Response (quote, eventlog)
6. C: Validate quote signature using AIK[pub]
7. C: Validate nonce and PCRs/eventlog
8. C: Remote trust decision

BedRock
Systems Inc

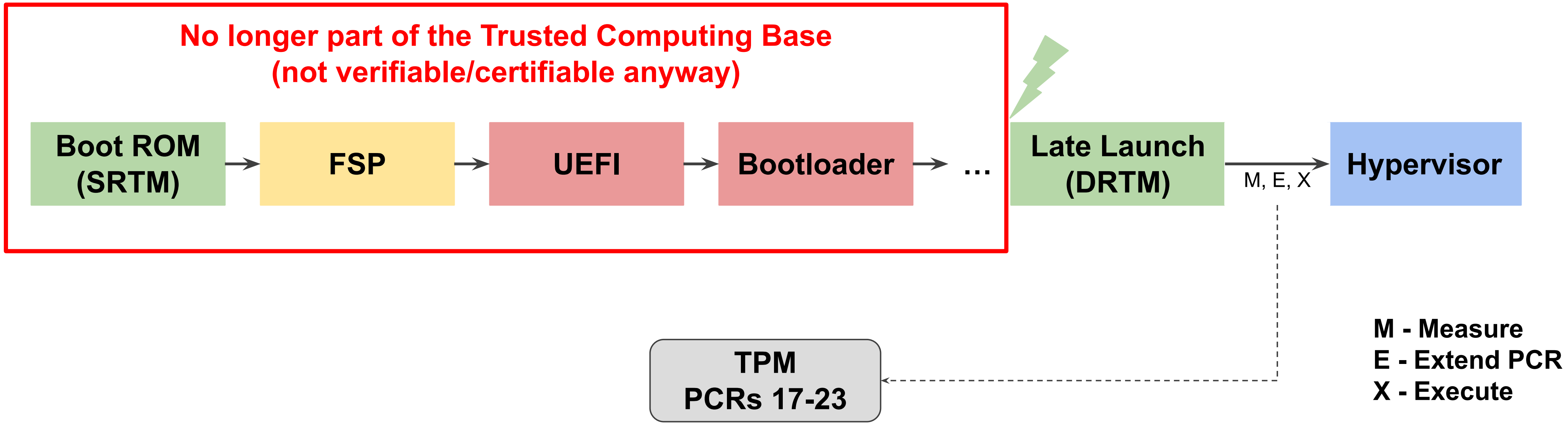
Measured Boot

Measured Boot: Static Root of Trust (SRTM)



- SRTM Flow is inherently brittle
  - Forcing vendors to maintain extensive “allow” or “block” lists
  - One minor change can invalidate the entire chain of trust

Measured Boot: Dynamic Root of Trust (DRTM)

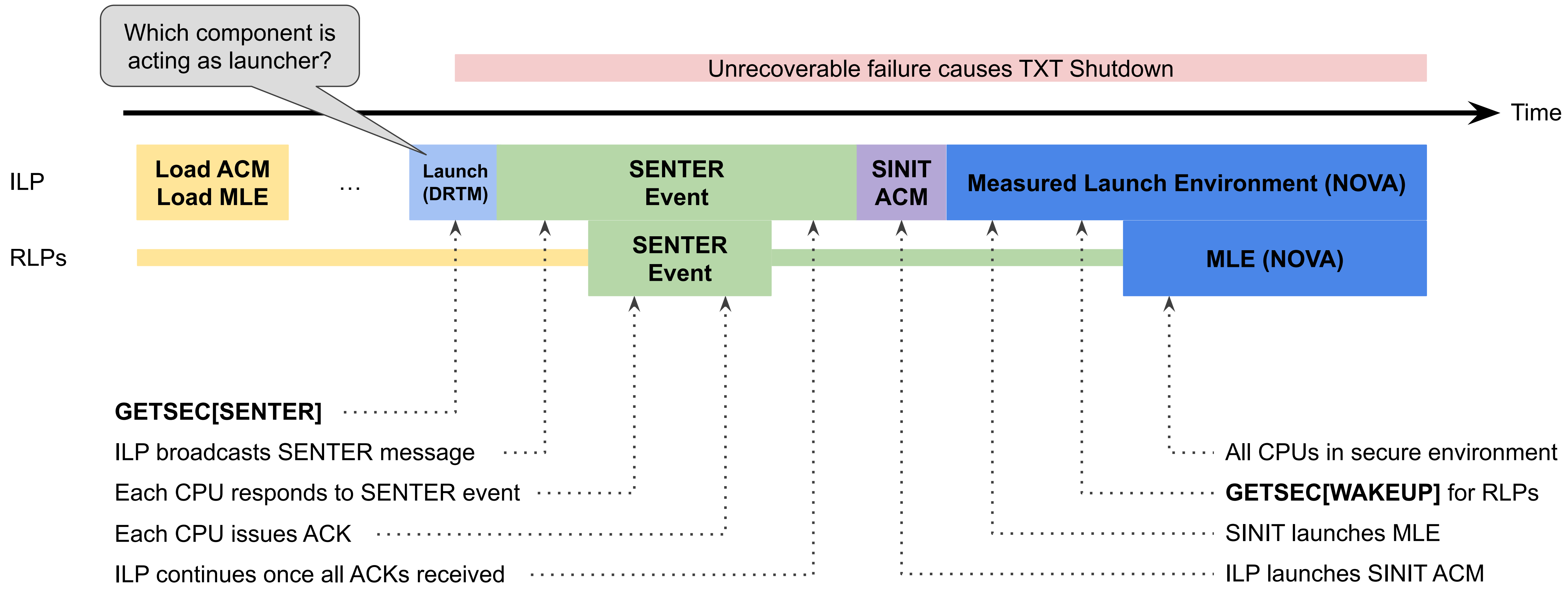


- DRTM Flow lets system boot into an untrustworthy state (initially)
  - Measured Launch later “resets” system into a trustworthy safe state
  - Takes control of all CPUs and forces them down a <u>protected</u> and <u>measured</u> code path

Intel Trusted Execution Technology (TXT)

- Provides a Dynamic Root of Trust (DRTM)
- Prerequisites
  - CPU support (SMX features)
  - TXT-capable chipset (DMA protection)
  - TPM 2.0 (preferably) or 1.2
  - SINIT Authenticated Code Module (ACM)
- Use Cases
  - Remote Attestation (via TPM Quote)
  - Local Attestation (via Launch Control Policy)

GETSEC[SENTER] Late Launch Sequence

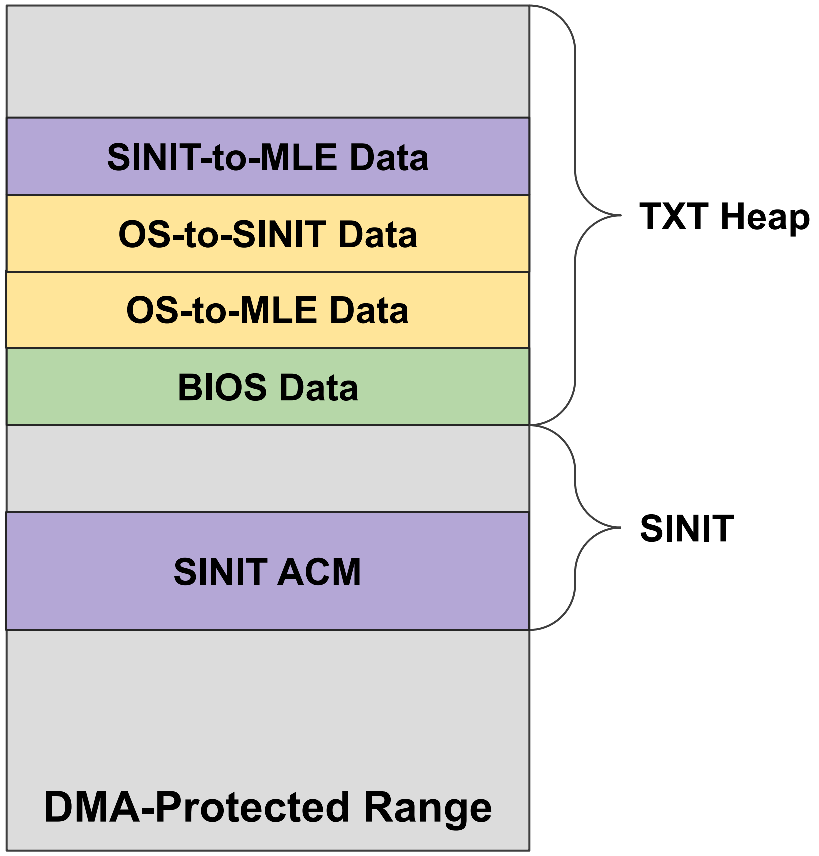

Which component is acting as launcher?
Unrecoverable failure causes TXT Shutdown
Time
ILP
RLPs
Load ACM
Load MLE
...
Launch (DRTM)
SENTER Event
SINIT ACM
Measured Launch Environment (NOVA)
SENTER Event
MLE (NOVA)
GETSEC[SENTER]
ILP broadcasts SENTER message
Each CPU responds to SENTER event
Each CPU issues ACK
ILP continues once all ACKs received
All CPUs in secure environment
GETSEC[WAKEUP] for RLPs
SINIT launches MLE
ILP launches SINIT ACM

Who should execute GETSEC[SENTER]?

- TXT specification defines layout of the TXT heap
  - Defines launcher (OS) and hypervisor (MLE) as separate components that exchange information via the TXT heap
  - Introduces a dependency between launcher and MLE
- Design Decision: NOVA is late-launching itself
  - Lots of similar code exists in launcher and MLE
  - Early boot code shared between launcher and MLE

⇒ Measured boot in NOVA is independent of any bootloader*

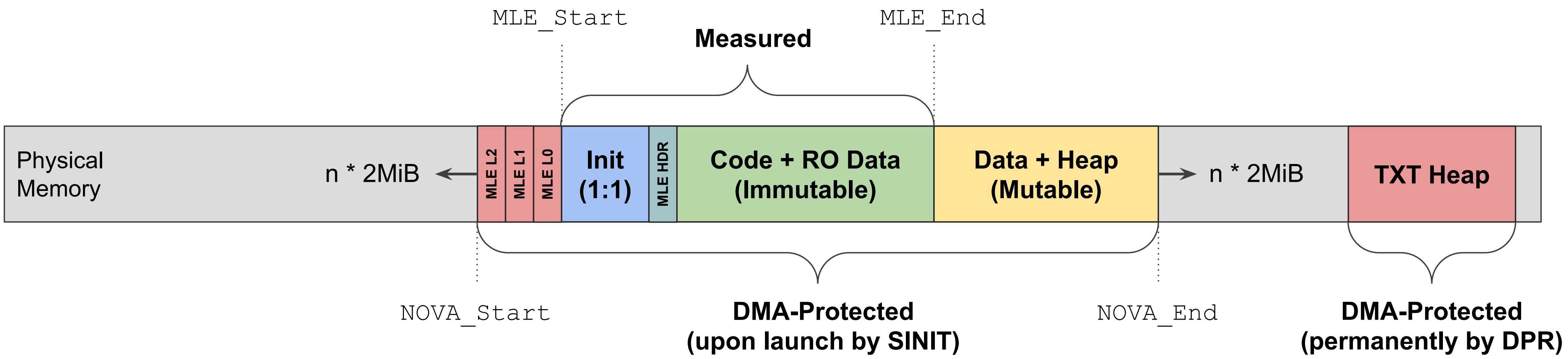
* as long as someone puts the SINIT ACM in memory



What should (not) be measured?

- Integrity measurement is **sensitive** to
  - Any modifications to the NOVA binary (immutable portions) before/during launch
  - Command line parameters passed to NOVA
  - Launch capability downgrades to workaround ACM issues
- Integrity measurement is **insensitive** to
  - The Intel platform on which NOVA runs (hardware, firmware, bootloader, etc.)
  - The physical memory range into which NOVA has been loaded
  - NOVA patching its code to adapt to available platform features
  - UEFI and multiboot parameters (memory map, multiboot modules, etc.)

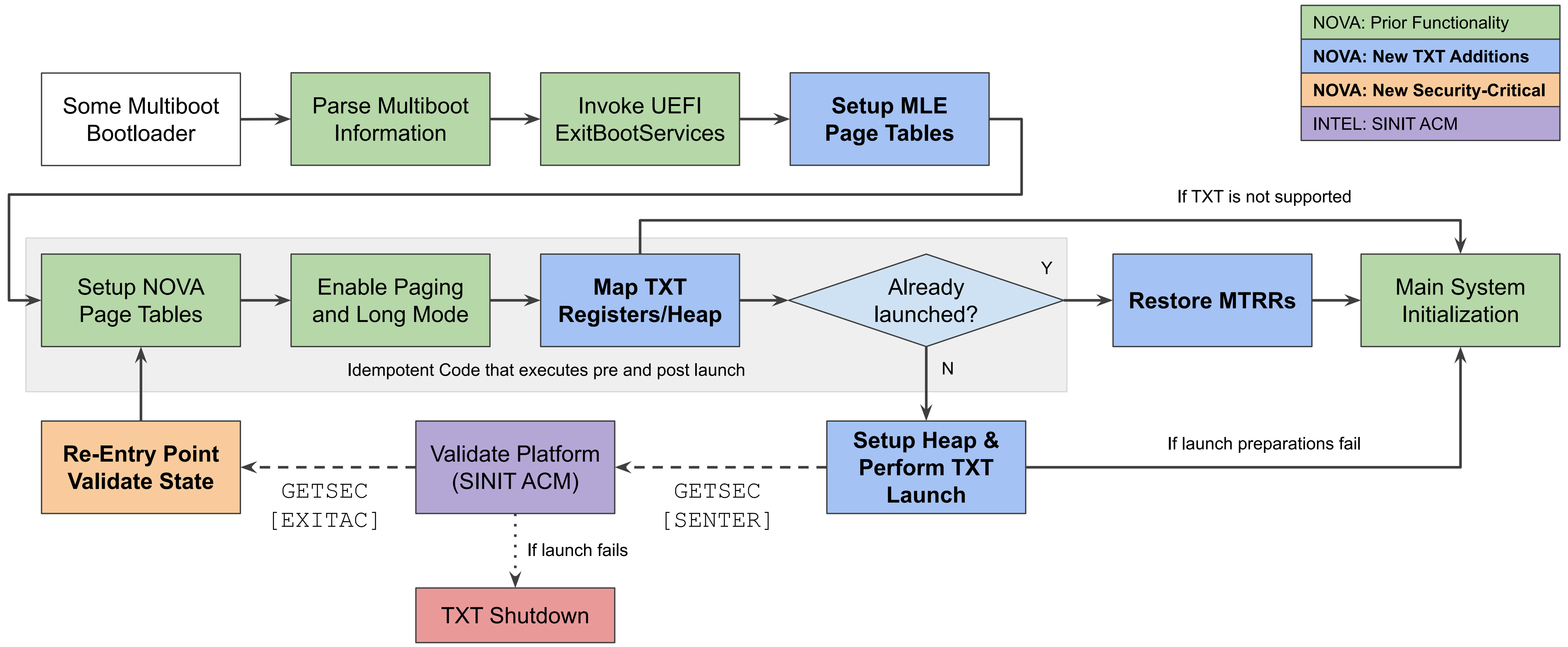
Memory Layout



- MLE Page Tables enumerate the to-be-measured memory pages
  - ➢ Must precede the MLE in physical memory
  - ➢ Must list MLE pages with ascending physical addresses

⇒ MLE page tables not measured, because NOVA is relocatable in physical memory

NOVA Late Launch Process (BSP)



NOVA: Prior Functionality
NOVA: New TXT Additions
NOVA: New Security-Critical
INTEL: SINIT ACM
Some Multiboot Bootloader
Parse Multiboot Information
Invoke UEFI ExitBootServices
Setup MLE Page Tables
If TXT is not supported
Setup NOVA Page Tables
Enable Paging and Long Mode
Map TXT Registers/Heap
Already launched?
Y
N
Restore MTRRs
Main System Initialization
Idempotent Code that executes pre and post launch
Re-Entry Point Validate State
Validate Platform (SINIT ACM)
Setup Heap & Perform TXT Launch
If launch preparations fail
GETSEC [EXITAC]
GETSEC [SENTER]
If launch fails
TXT Shutdown

BedRock Systems Inc

TXT Shutdown

- Can occur between GETSEC[SENTER] and GETSEC[SEXIT]
  - When recovery from an error condition is not considered reliable (abort of MLE)
  - CPU or ACM writes error code to TXT.ERRORCODE register
  - Platform reset via TXT.CMD.RESET register and shutdown state (until RESET takes effect)
- TXT.ERRORCODE chipset register survives soft reset
  - Software can diagnose shutdown condition during next boot
- NOVA will not attempt another measured launch until error code is cleared
  - Power cycle will clear shutdown error

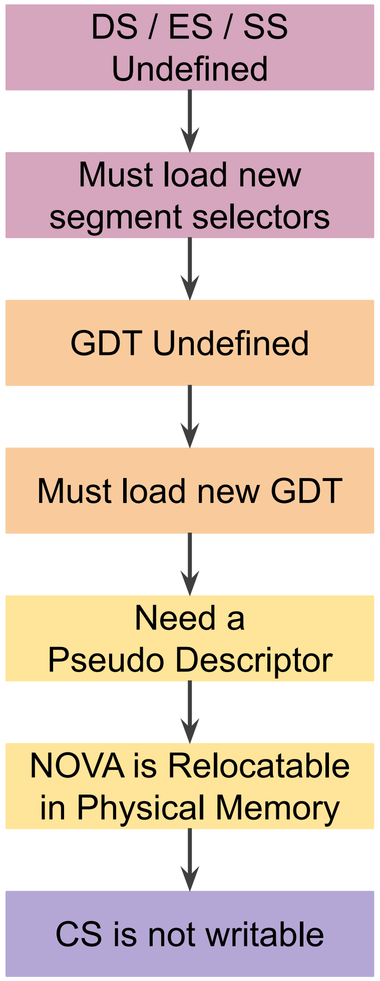
TXT Protections

- GETSEC[SENTER] is a disruptive (reset-like) event for the platform
  - Except that SINIT ACM starts executing instead of reset vector
- Protecting against rogue CPUs
  - Chipset detects CPUs not participating in rendezvous or additional CPUs showing up later
  - INIT converts to TXT shutdown ⇒ INIT-SIPI-SIPI startup sequence not possible
- Protecting against surprise RESET
  - Measured environment must exit gracefully using GETSEC[SEXIT]
  - While TXT.SECRETS is set, memory controller blocks access until SCLEAN has run

SINIT Authenticated Code Module

- Runs in isolated ACRAM after CPU has successfully validated it
- Configures and checks various hardware settings for secure operation
  - Locks Memory Configuration
  - Checks for Physical Memory Aliasing
  - Checks Power and Frequency Settings (Undervolting, Overclocking, etc.)
  - Validates platform configuration and certain ACPI tables for correctness
  - Most of the checks and settings are chipset-specific (but software ⇒ extensible)
- Measures and launches the MLE (NOVA)
- Enforces the Launch Control Policy (LCP)

Post-Launch Re-Entry Point



All data reads require CS segment override

But that requires a GDT

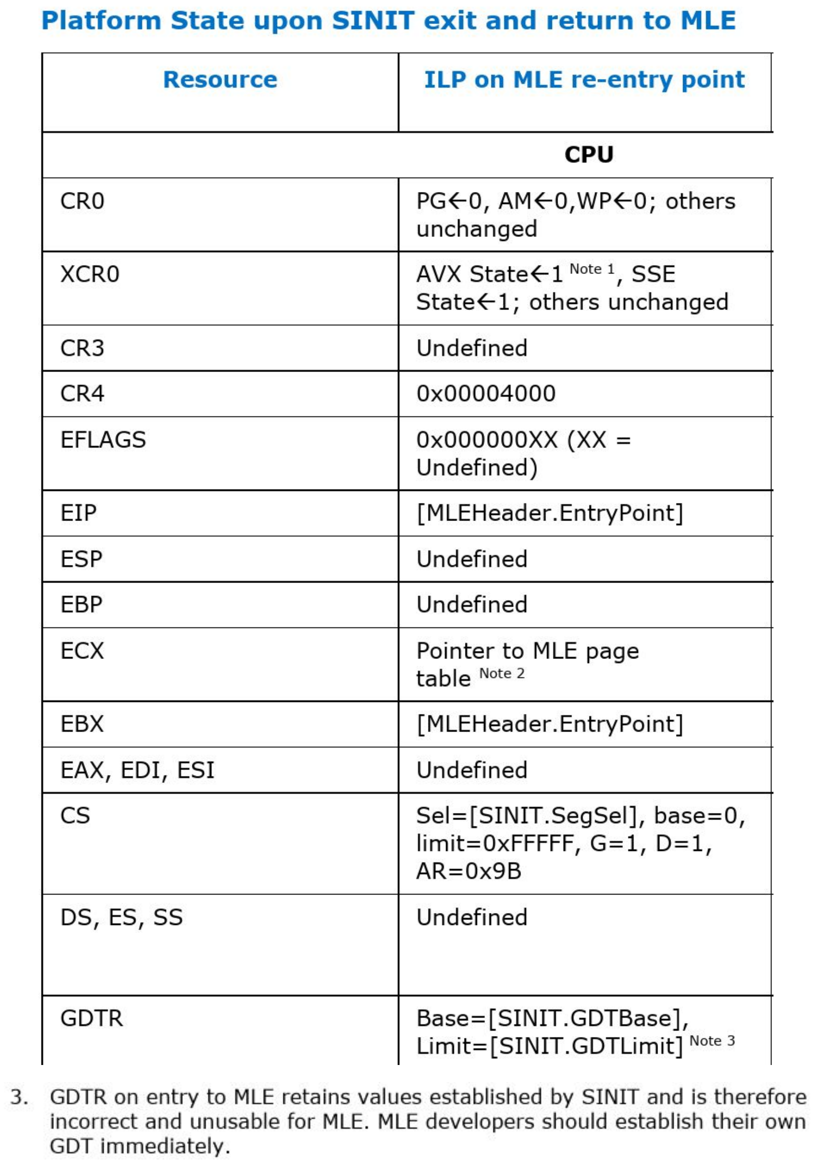
The SINIT GDT is not usable

Just need to point the CPU at NOVA's GDT

Defines GDT physical address and size

Physical address in descriptor must be initialized

Cannot initialize the pseudo descriptor



Platform State upon SINIT exit and return to MLE

Resource	ILP on MLE re-entry point
	CPU
CR0	PG←0, AM←0,WP←0; others unchanged
XCR0	AVX State←1 Note 1, SSE State←1; others unchanged
CR3	Undefined
CR4	0x00004000
EFLAGS	0x000000XX (XX = Undefined)
EIP	[MLEHeader.EntryPoint]
ESP	Undefined
EBP	Undefined
ECX	Pointer to MLE page table Note 2
EBX	[MLEHeader.EntryPoint]
EAX, EDI, ESI	Undefined
CS	Sel=[SINIT.SegSel], base=0, limit=0xFFFFF, G=1, D=1, AR=0x9B
DS, ES, SS	Undefined
GDTR	Base=[SINIT.GDTBase], Limit=[SINIT.GDTLimit] Note 3

3. GDTR on entry to MLE retains values established by SINIT and is therefore incorrect and unusable for MLE. MLE developers should establish their own GDT immediately.

Securing the Measured Launch Process

- Intel TXT does not protect against an attacker initiating the launch process
  - But, any attack that produces a different or no integrity measurement will be exposed
- Attacker Goal
  - Subvert the measured launch in a way that the integrity measurement does not change
- Re-Entry point can make only few assumptions about pre-SENTER state
  - Must either validate or reinitialize any state the attacker can control before launch
  - Validation requires knowledge of the correct state
  - Reinitialization is often easier, but not always possible

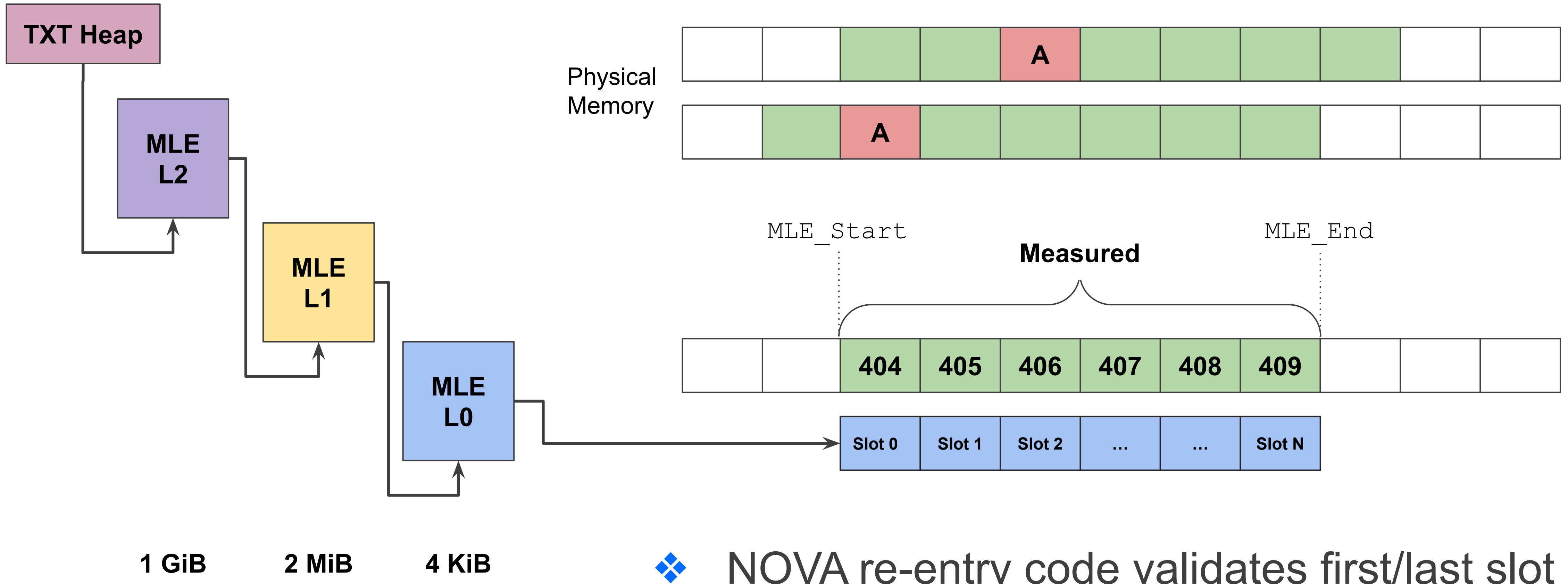
Simple Attacks & Countermeasures

- Changing NOVA’s code or measured launch code
  - Changes the integrity measurement
- Invoking NOVA at the wrong re-entry point (defined in MLE header)
  - Changes the integrity measurement
- Using attacker’s own launch code
  - To launch something else ⇒ changes the integrity measurement
  - To launch NOVA correctly ⇒ that’s ok
  - To launch NOVA incorrectly ⇒ let’s see how

What could an attacker potentially control?

- Pre-SENTER launch code (if they bring their own)
  - NOVA's own launch code is measured ⇒ attacker cannot change that
- MLE page tables and TXT heap (if attacker initiates the launch process)
  - Both are not included in the integrity measurement
  - Both influence the measured launch process, but are retained across GETSEC[SENTER]
- **If the correct integrity measurement is stored in the TPM, then unmodified NOVA code has gained execution at the correct NOVA re-entry point**
  - And can subsequently inspect the TXT heap and the MLE page tables

Subverting the MLE Page Tables



- NOVA re-entry code validates first/last slot

BedRock Systems Inc

Subverting the TXT Heap

- Changing MLE Size or MLE header pointer
  - SINIT validates that MLE header is within [MLE_Start, MLE_End] region
  - SINIT validates that MLE Size == MLE_End - MLE_Start (from MLE header)
  - SINIT measures contents of [MLE_Start … MLE_End] region
- Changing MLE page-table pointer
  - NOVA re-entry code validates that TXT Heap ⇒ MLE_L2 ⇒ MLE_L1 ⇒ MLE_L0
- Changing DMA-protected region via PMR
  - SINIT ensures that PMR covers at least [MLE_Start, MLE_End] region
  - NOVA re-entry code validates that PMR covers entire NOVA image range

Current Status

- TXT support is part of the upcoming NOVA 23.26 release
  - Will be publicly available end of this month
- Measured launch works on all TXT-capable machines in our lab
  - Client: BDW, SKL, KBL, WHL, CFL, CML, RKL, TGL, ADL
  - Server: HSX/BDX (Grantley), SKX/CLX (Purley), ICX (Whitley), ICX-D (Idaville)
- Some platforms had TXT problems
  - We worked with Intel to fix a few issues on newer platforms
  - Intel won’t fix EOL platforms ⇒ NOVA has a quirk list with workarounds

What is the correct integrity measurement?

- ❖ Local or Remote Attestation needs to know what value to expect
- ❖ Could just boot NOVA once and see what the result is
  - ➢ No guarantee that your machine has not already been subverted
- ❖ Better: Write a tool to compute the correct measurement offline
  - ➢ Takes a NOVA x86 binary as input
  - ➢ Outputs the measurable portion to feed through a hash program
  - ➢ Not a lot of code

⇒ Works with any hash algorithm

BedRock Systems Inc

Demo

Intel Security Technologies in NOVA

- **Confidentiality**
  - Total Memory Encryption with Multiple Keys (TME-MK) - since 22.52
- **Integrity**
  - Control-Flow Enforcement Technology (CET IBT+SS) - since 22.17
  - Trusted Execution Technology (TXT/CBnT) - since 23.26
- **Availability**
  - Cache Allocation Technology (CAT/CDP) - since 22.26
  - Memory Bandwidth Allocation (MBA) - since 22.26

Links

- NOVA Microhypervisor: https://github.com/udosteinberg/NOVA
- BedRock Systems: https://bedrocksystems.com/
- Trusted Computing: https://trustedcomputinggroup.org/resources/
- Intel Trusted Execution Technology: MLE Writer’s Guide
- Arm: DRTM Architecture for Arm
- FOSDEM: https://fosdem.org/
  - NOVA Microhypervisor Feature Update (2023)
  - NOVA Microhypervisor on ARMv8-A (2020)