OCP LOCK Integration

Zach Halvorsen, Carl Lundin, Jeff Andersen

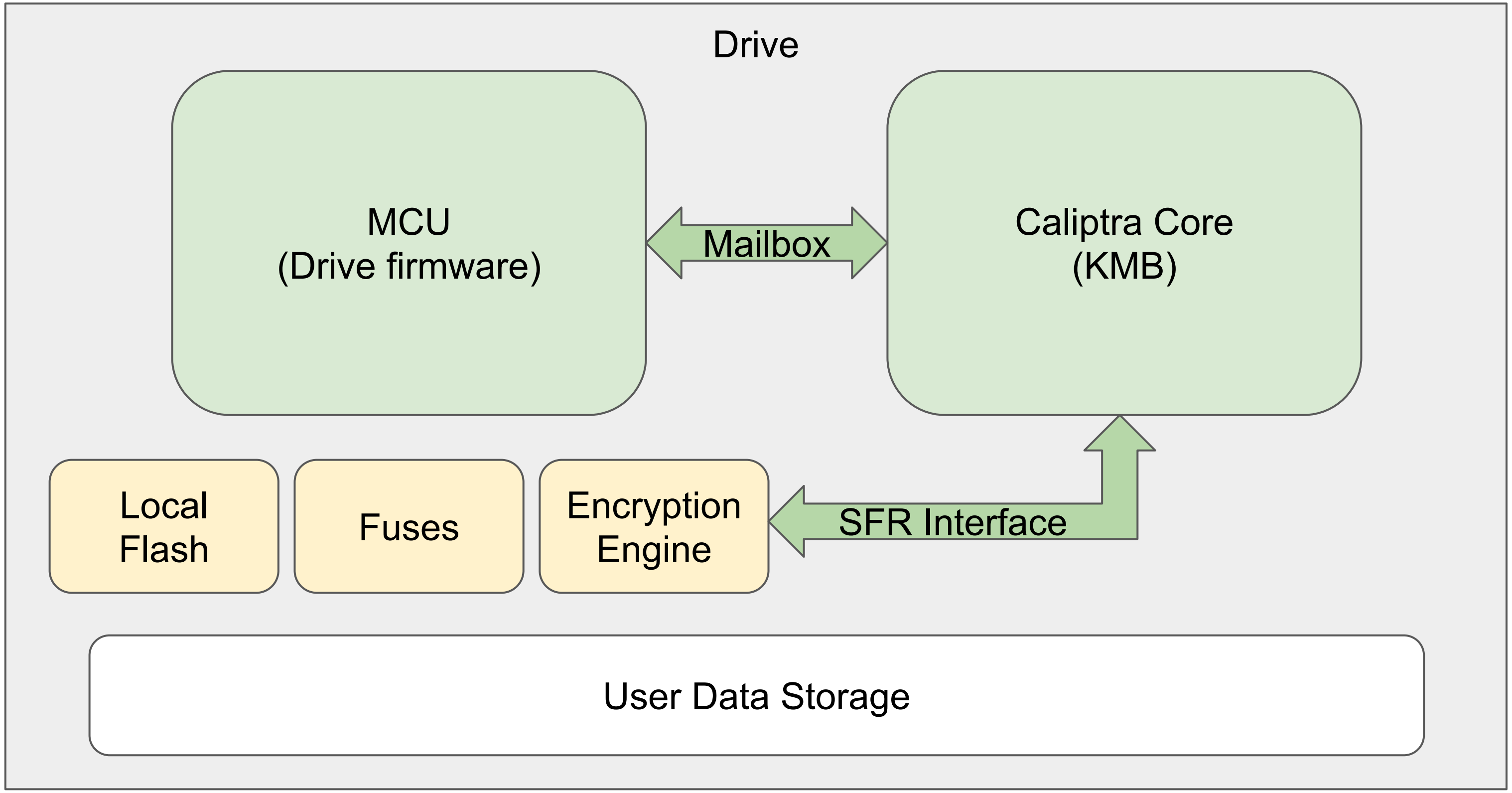
What is OCP LOCK?

- OCP L.O.C.K. (Layered Open-source Cryptographic Key management) provides secure key management for Data-At-Rest protection in self-encrypting storage devices.
  - Official OCP LOCK specification at https://www.opencompute.org/documents/ocp-lock-specification-v1-0-rc2-pdf.
  - Developed on GitHub: https://github.com/chipsalliance/Caliptra
- OCP LOCK talks later this week:
  - “OCP LOCK/Caliptra 2.1 and MEK-MPA Use Cases”
    - 9:00 AM Thurs SJCC – Concourse Level – 220C
  - “OCP LOCK with Caliptra 2.1”
    - 11:15 AM Thurs SJCC – Concourse Level – 211

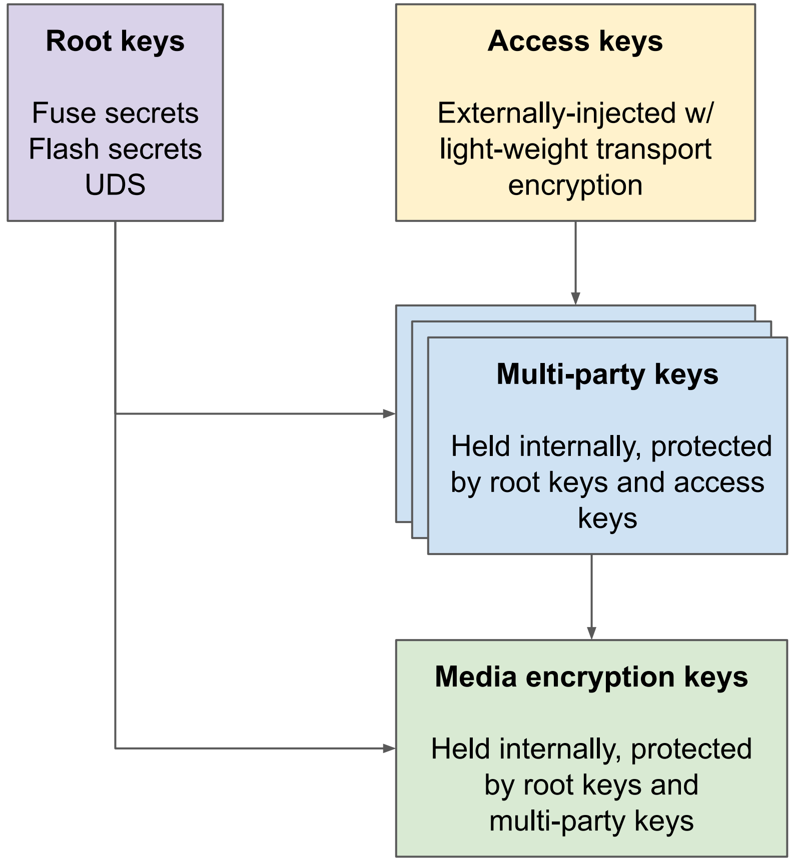
FAQ

- Will Caliptra 2.1 fully support OCP LOCK 1.0 as it is currently available?
- Will Caliptra 2.1 need to be updated to fully support LOCK 1.0?
- Are there any changes required to firmware, etc., to support both OCP LOCK 1.0 and Caliptra 2.1?
- What are the differences between OCP LOCK 0.85 and OCP LOCK 1.0?
- Are there any immediate changes expected in OCP LOCK 1.0 (e.g., bug fixes) that I should wait for?
- When is the next release of OCP LOCK expected, and what are the expected improvements?
- Should we expect a Caliptra update to support future OCP LOCK updates?
- When is the next version of Caliptra expected, and will it support OCP LOCK 1.0?

Overview Diagram



Key Hierarchy


Root keys
Fuse secrets
Flash secrets
UDS
Access keys
Externally-injected w/ light-weight transport encryption
Multi-party keys
Held internally, protected by root keys and access keys
Media encryption keys
Held internally, protected by root keys and multi-party keys

Integration Surfaces

Fuses / Straps

Flash

Encryption Engine

Integration Surfaces - Fuses / Straps

- OCP LOCK Enable
  - OCP L.O.C.K. shall be enabled by setting the input strap ss_ocp_lock_en in Caliptra Core or cptra_ss_strap_ocp_lock_en_i in Caliptra Subsystem.
- MEK Release Address
  - DMA address for MEK KV release. KMB will release the MEK to this address.
- MEK Size
  - Informs KMB of the MEK size. All implementations should currently set this to 64 bytes

Integration Surfaces - Fuses / Straps (cont.)

- HEK Seeds
  - Randomized seed for HEK
  - Zeroize to wipe the HEK and keys derived from HEK
  - Must support 4 <= n <= 16 HEK seed slots
  - HEK seed states
    - Empty
    - Zeroized
    - Corrupted
    - Randomized
    - Permanent
  - Hek availability
    - Seed state
    - Caliptra Lifecycle state
    - HEK seed zeroized

Integration Surfaces - Flash

- SEK (Soft Epoch Key)
  - Drive firmware is responsible for generating, storing, and zeroizing SEK
- Access Keys
  - Held external to drive
  - Unlocks a MPK (Multi-party Protection Key)
- HPKE (IETF RFC 9180)
  - Encrypts Access Keys in transit
  - Stored by KMB in volatile memory
  - Endorsed by Runtime Alias Key
- Locked MPKs
  - MPK Encrypted by HEK, SEK and Access Key
  - Stored by drive firmware

Integration Surfaces - Encryption Engine

- Vendor Specific block for encrypting / decrypting user data using MEK
- MEK released from KMB to EE using DMA
- Drive Firmware manages which MEK to load to EE
- KMB uses Special Function Registers (SFRs) to communicate with EE
  - MEK register
    - Holds MEK
  - Metadata Register
    - Vendor defined MEK metadata
  - Aux Register
    - Vendor defined data associated with the MEK
  - Control Register
    - EE Status
    - KMB manage EE/MEK lifecycle

Supported TCG Specs

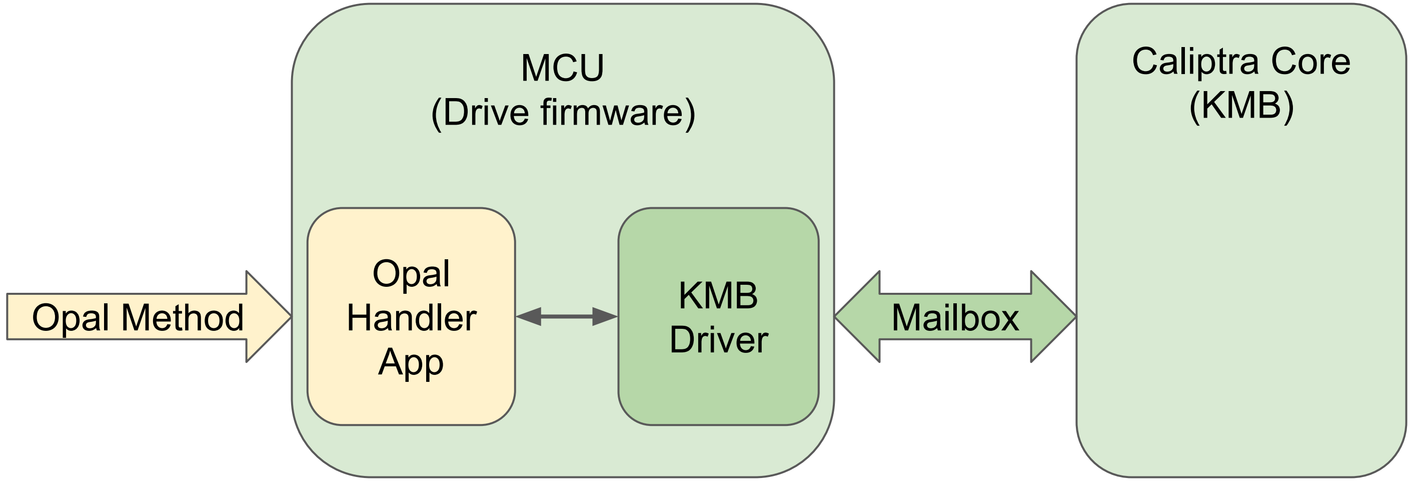
Opal

Opal w/ MEK-MPA

Key Per I/O

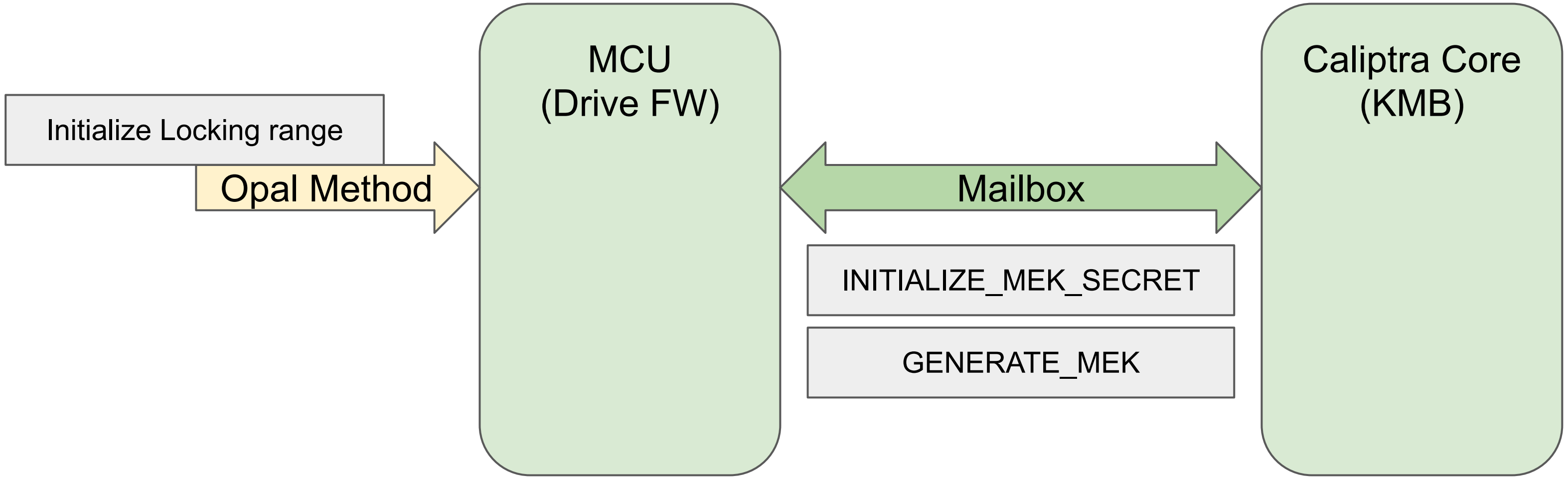
TCG Specs - Opal

- KMB is designed to support all TCG Opal functionality out of the box
    - KMB driver will be included in upstream MCU firmware



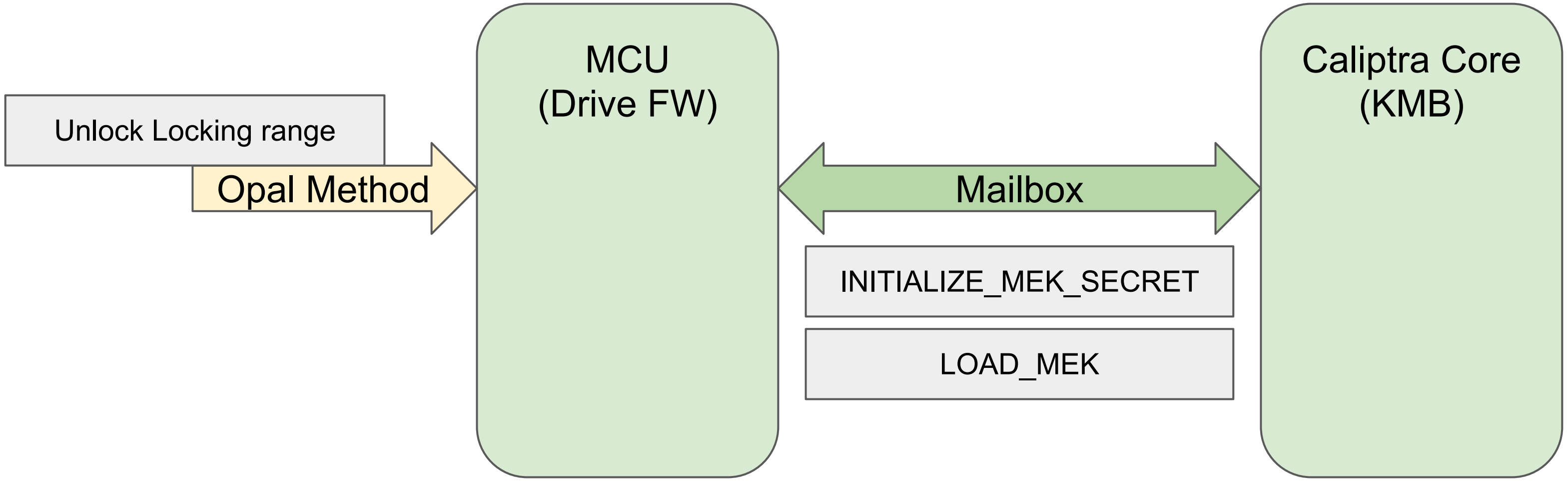
TCG Specs - Opal

Example of LOCK commands to initialize a locking range



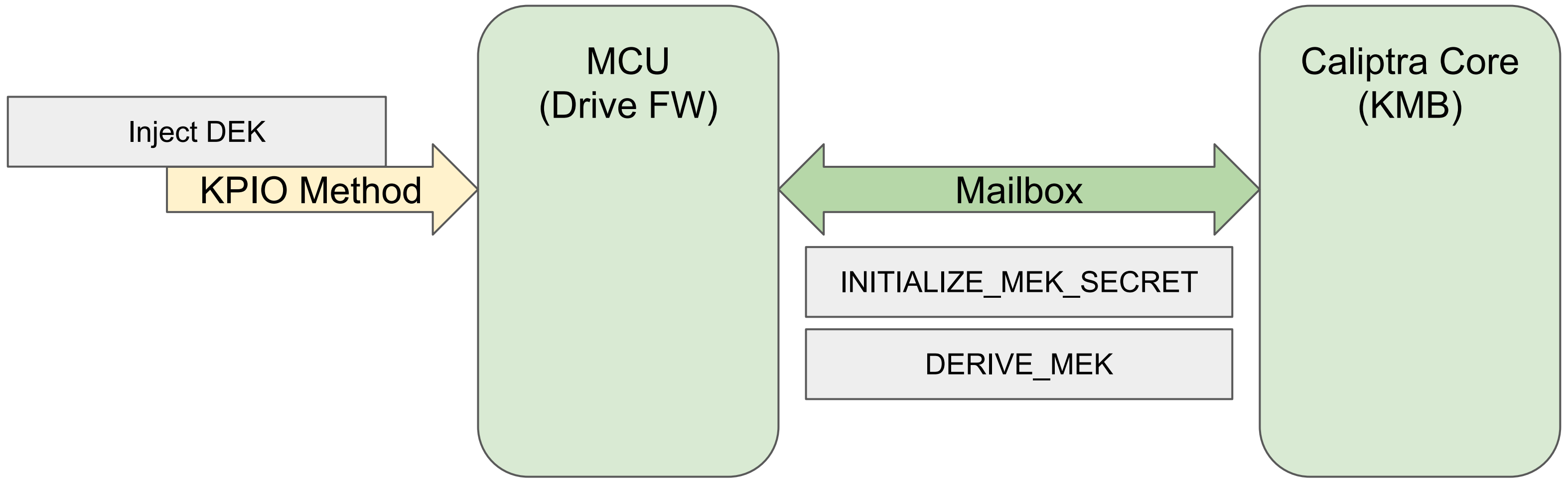
TCG Specs - Opal

Example of LOCK commands to unlock a locking range



TCG Specs - Key Per I/O

Example of LOCK commands to program DEK into the encryption engine

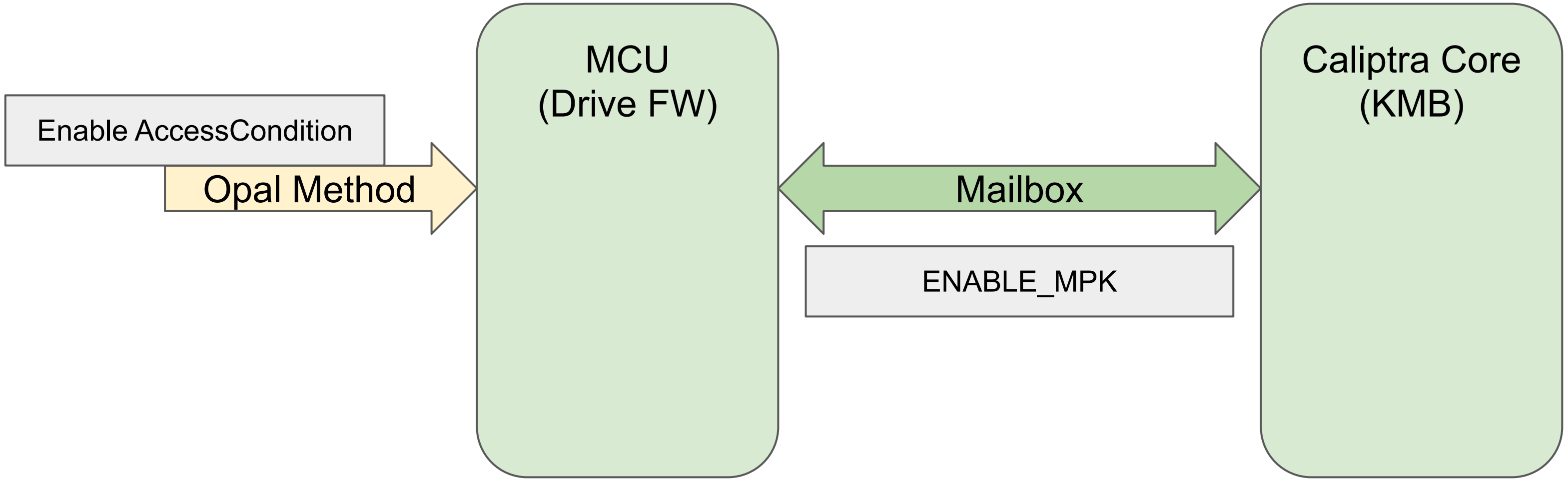


TCG Specs - Opal MEK-MPA

- Opal MEK-MPA was borne out of OCP LOCK
- The feature set is supported natively
- As a rule of thumb
  - MPK in LOCK == "AccessCondition" in MEK-MPA
  - Binding an MEK to zero MPKs allows for legacy Opal, Enterprise, or Key Per I/O support

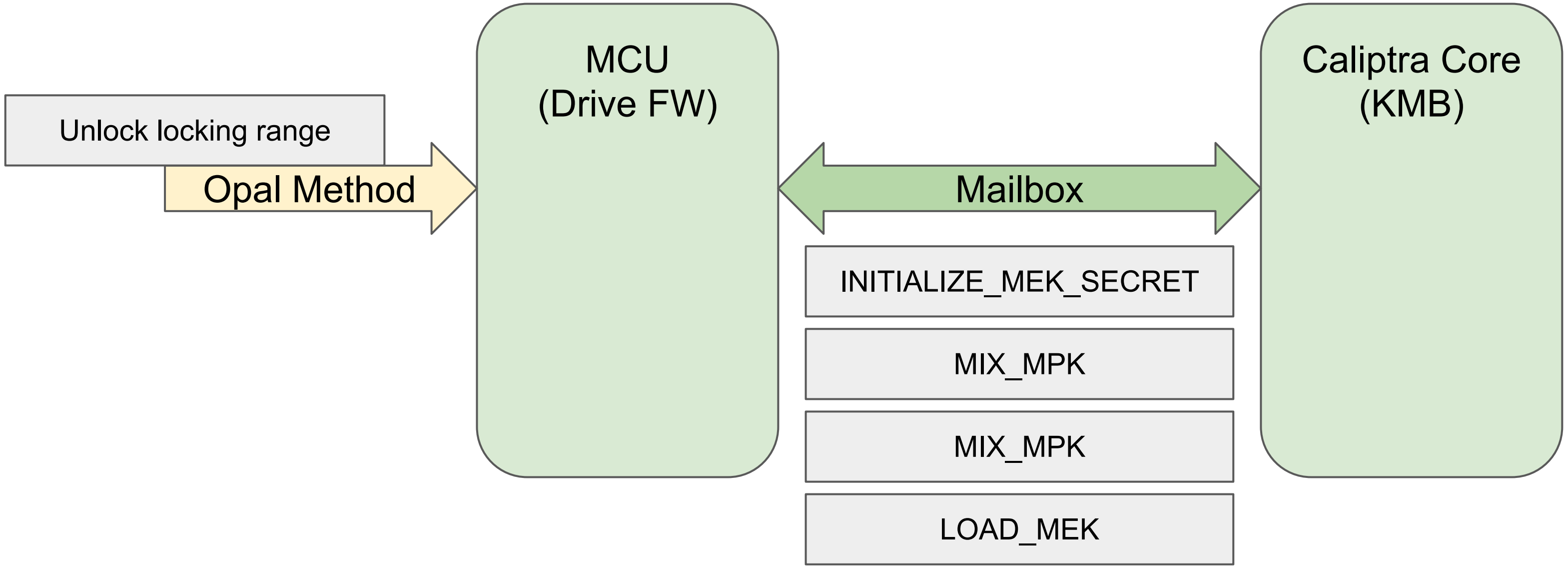
TCG Specs - MEK-MPA

Example of LOCK commands to enable an AccessCondition object



TCG Specs - MEK-MPA

Example of LOCK commands to unlock a locking range protected by two AccessCondition objects



Attestation

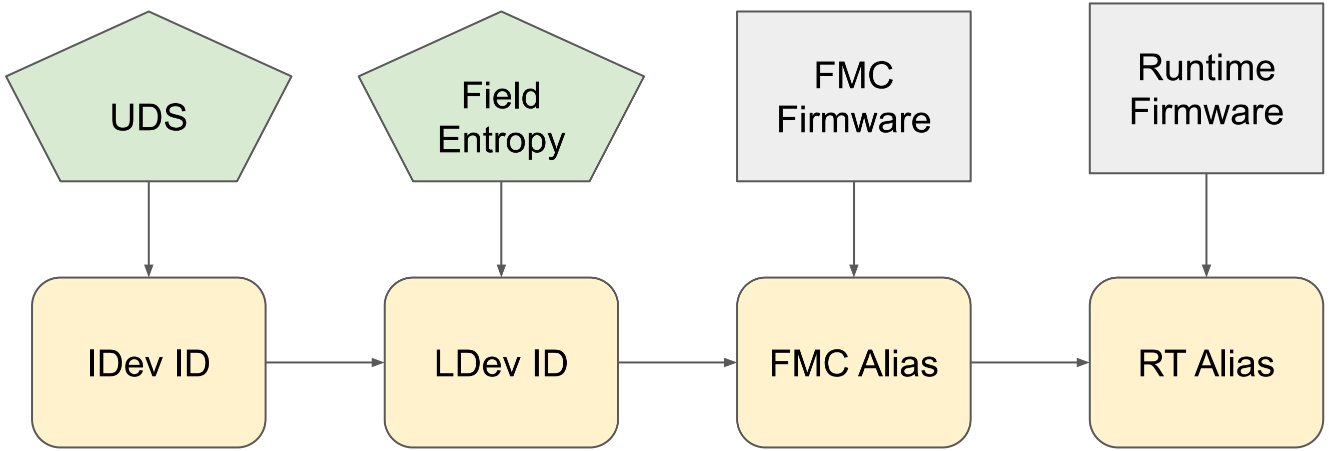
Device Identity

Why do drive owners care?

Provisioning stages

Attestation - Device Identity

- Caliptra contains layers of identity
    - UDS (Unique Device Secret) - 384 fuse bits
    - Field Entropy - 256 fuse bits
    - Alias identities derived from measurements
- Cryptographically identifies drive
- Allows drive to attest to information



Attestation - Why do drive owners care?

- Identify a given drive as trustworthy

- Attested cryptographic erasure
    - KMB will attest to the state of the root keys using CBOR encoded Entity Attestation Tokens (EAT)

    - Attestation signed by the drive’s DICE keychain

    - By attesting to the state of the root keys (and preventing user data from being written if the keys are not programmed), the drive is able to cryptographically prove if it contains user data

Attestation - LOCK provisioning stages

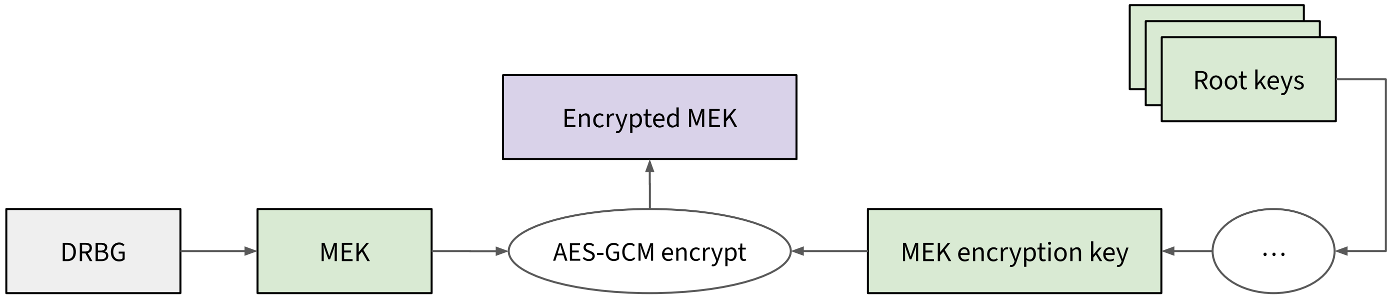
	Fuses		Flash
Manufacturing	UDS	HEK_0	SEK_0
Owner Ingestion	Field Entropy		
Owner Lifecycle		$HEK_{1..N}$	$SEK_{1..N}$

AES-GCM

Special IV Handling

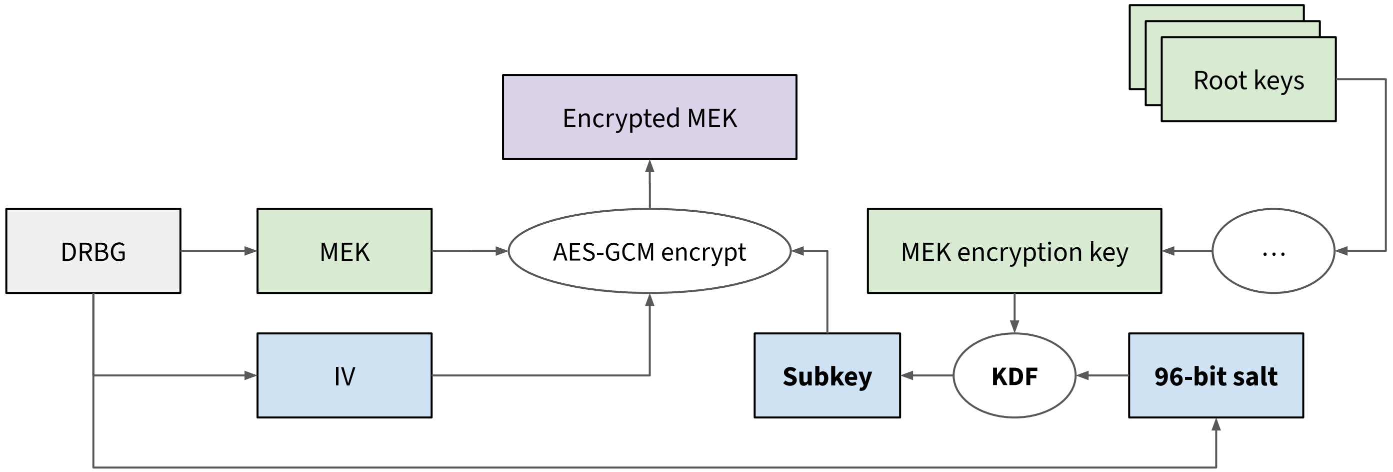
Protecting Media Encryption Keys

- We use root keys to compute an MEK encryption key, which encrypts a random MEK



Protecting Media Encryption Keys

- We derive a subkey based on a random 96-bit salt.
- This lets us use a given encryption key up to 280 times.
- At 1 op / µs, our safety margin lasts ~3x the age of the universe.



Integration Requirements

Integration Requirements

- 15 item checklist
    - “Compliance Requirements Table”
- No surprise requirements

Integration Requirements - A few examples

Item	Requirement
1	The device shall integrate Caliptra.
2	OCP L.O.C.K. shall be enabled by setting the input strap `ss_ocp_lock_en` in Caliptra Core.
4	The encryption engine shall remove all MEKs from the encryption engine on a power cycle or during zeroization of the storage device.
9	The drive shall only provide HEK seeds or SEKs that are cryptographically-strong random values.
12	KMB shall have access to the SFRs defined in Table 6 through the address defined by `OCP_LOCK_MEK_ADDRESS`. See Section 4.7.1.2.

OPEN Compute Project®

Thank you

Enumerate interfaces

- What do you need to know about integrating with KMB
  - Fuses
    - Attested erase
  - Flash
    - Attested erase
  - Opal (Mailbox commands)
    - Possibly List Opal and how that is actuated with KMB
    - Works out of box
    - Opal MEK-MPA also supported
  - Key Per I/O
    - Works out of box
  - Encryption engine
    - Will require DMA access from KMB to write the MEK