arm

Runtime Security Engine (RSE)

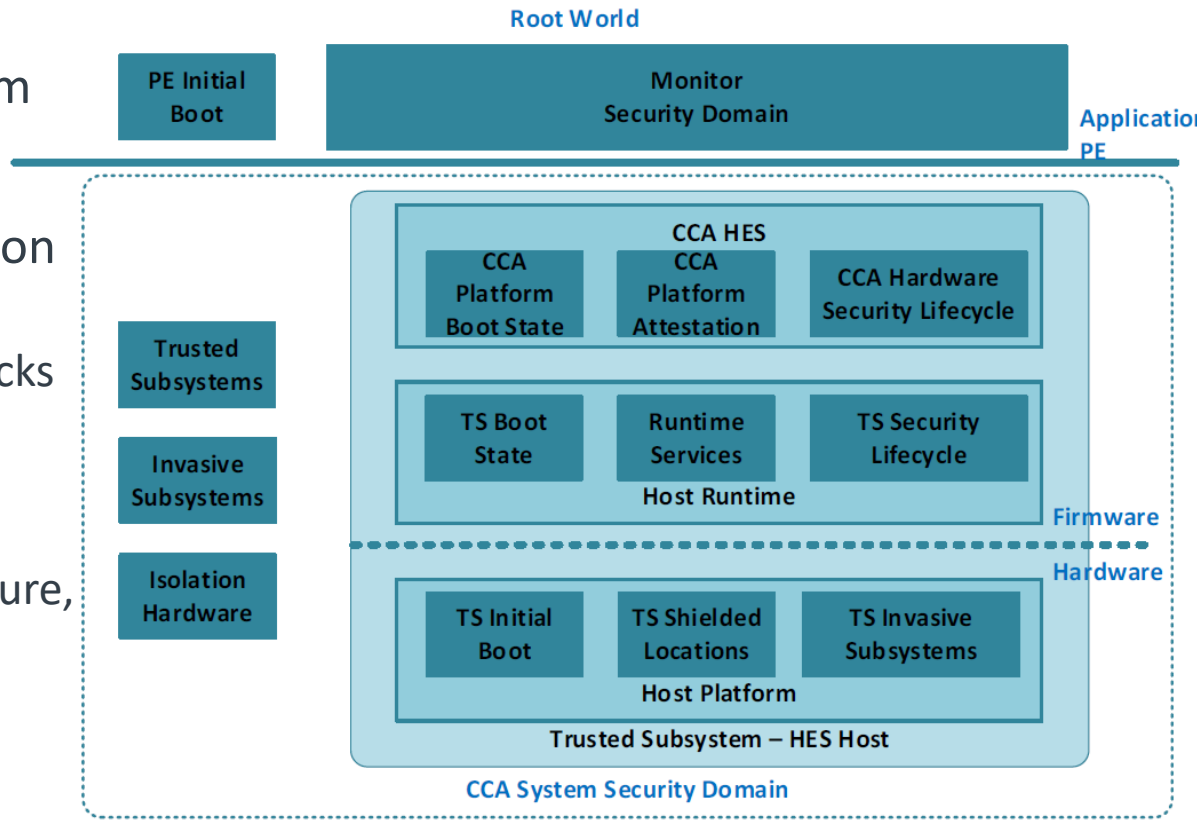
Jamie Fox

November 2023

© 2023 Arm

Runtime Security Engine (RSE) - Background

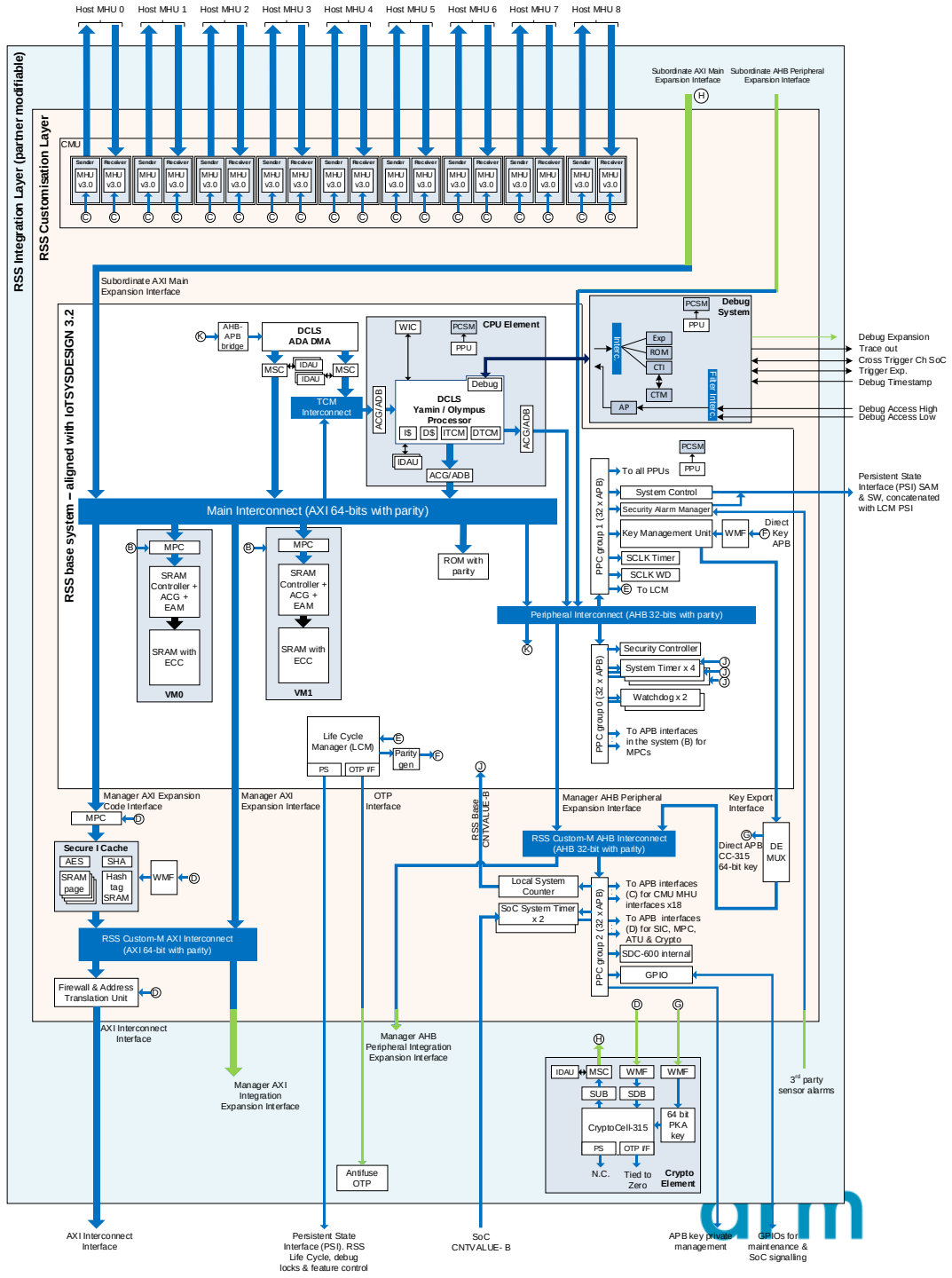
- Arm CCA Security Model strongly recommends ‘Hardware Enforced Security’ (CCA HES)
- CCA HES is required to be a tenant in a Trusted Subsystem
- Security domain services moved away from the application PE into a hardware isolated execution environment
  - With mitigations against fault injection & side-channel attacks
- RSE provides an Arm implementation of the CCA HES
  - M-profile TrustZone to separate HES services from Non-secure, Secure and Realm world services
- Trusted Firmware-M project implements RSE firmware providing CCA HES services.



© 2023 Arm

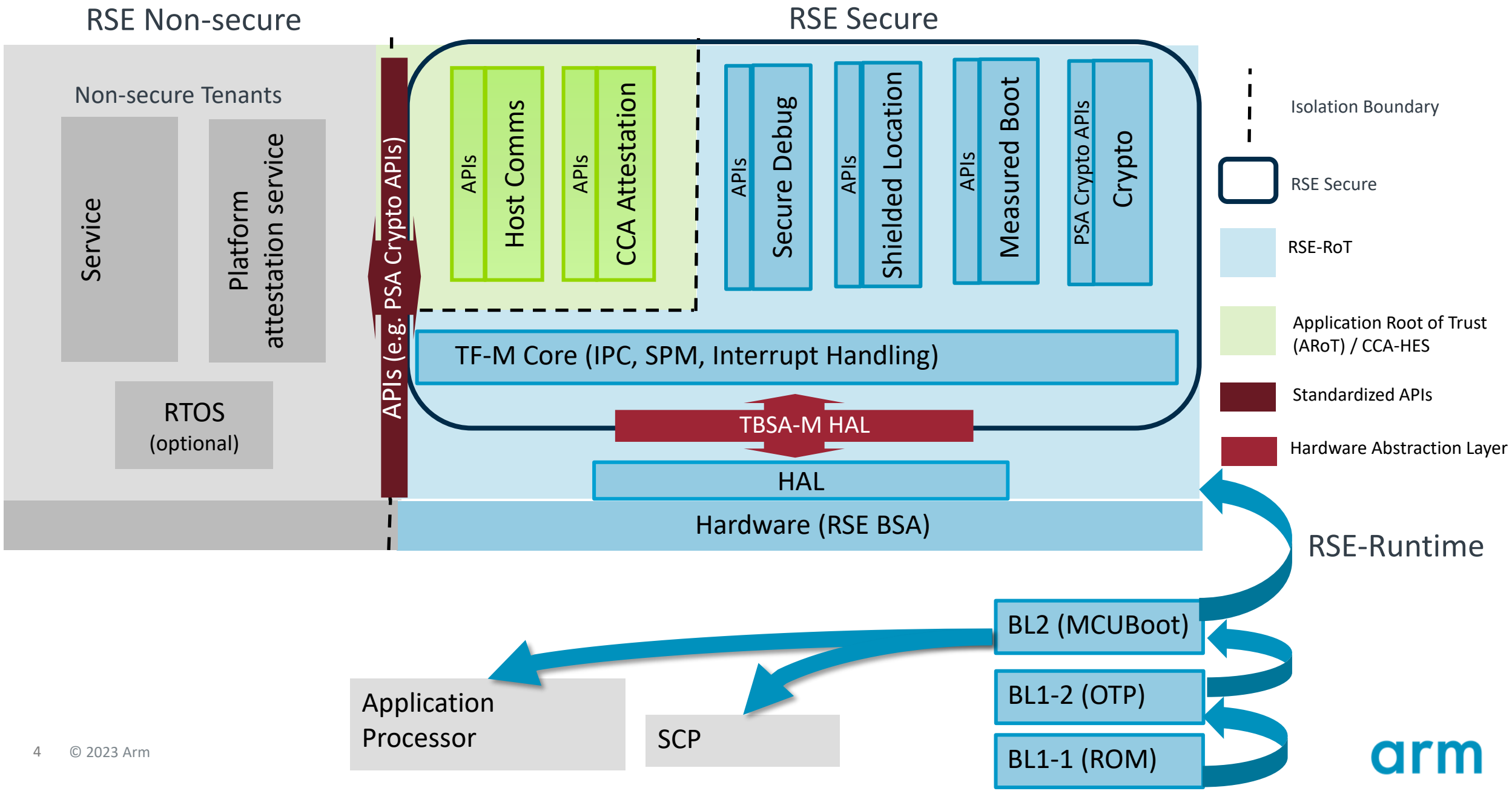
RSE HW overview

- Single M55 (or M85) with TrustZone & Helium (MVE)
- Lifecycle & OTP manager (LCM)
  - SoC debug access signals
    - Aligned with PSA ADAC (Authenticated Debug Access Control)
- Key management unit (KMU)
- CryptoCell-315 (updated CC-312 from CryptoIsland)
  - In integration layer, replaceable
- MHUv3 for host communication
  - Point-to-point link bypassing the “untrusted” system bus
  - Reduces the possibility of rogue bus manager accessing secrets
- DMA-350 (Ada DMA) for bulk data transfers
- Secure Instruction Cache
  - Read-only access for code and constants from flash or DRAM
  - Reduces SRAM requirements and therefore area (cost)
- Address Translation Unit (ATU)
  - Maps host physical addresses into region in M55’s memory map
- Security fault injection protection
  - CPU DCLS, bus parity, SRAM ECC, & HW life cycle management



© 2023 Arm

RSE SW overview

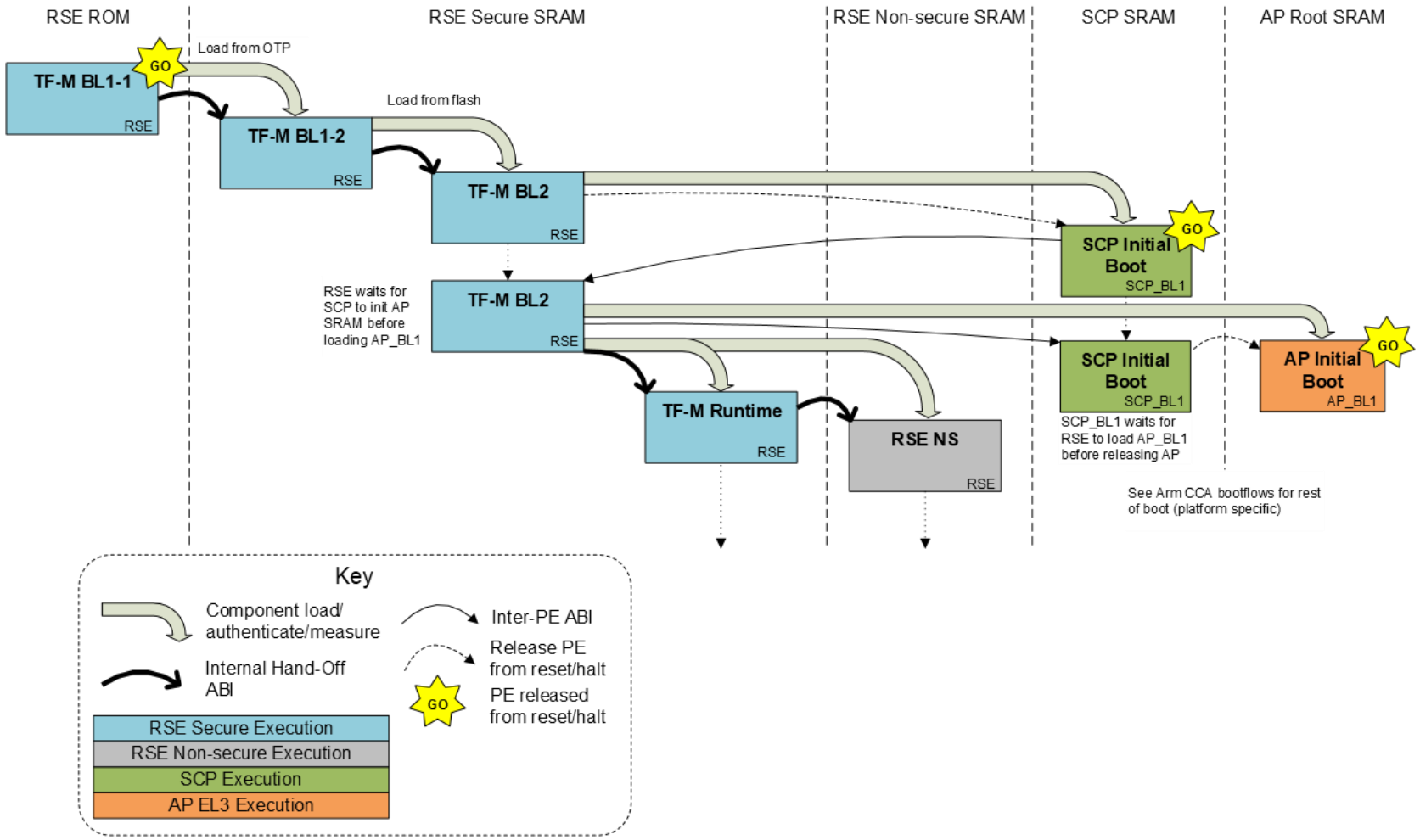


© 2023 Arm

RSE boot flow

TF-M BL1-1

- XIP from ROM upon reset de-assertion
- Checks LCM life cycle state: continues if it is Secured, otherwise provisions
- Loads TF-M BL1-2 from OTP into RSE internal SRAM
- Computes SHA-256 hash of loaded BL1-2 image and compares to provisioned OTP value
  - A fault countermeasure
- Jumps to BL1-2…

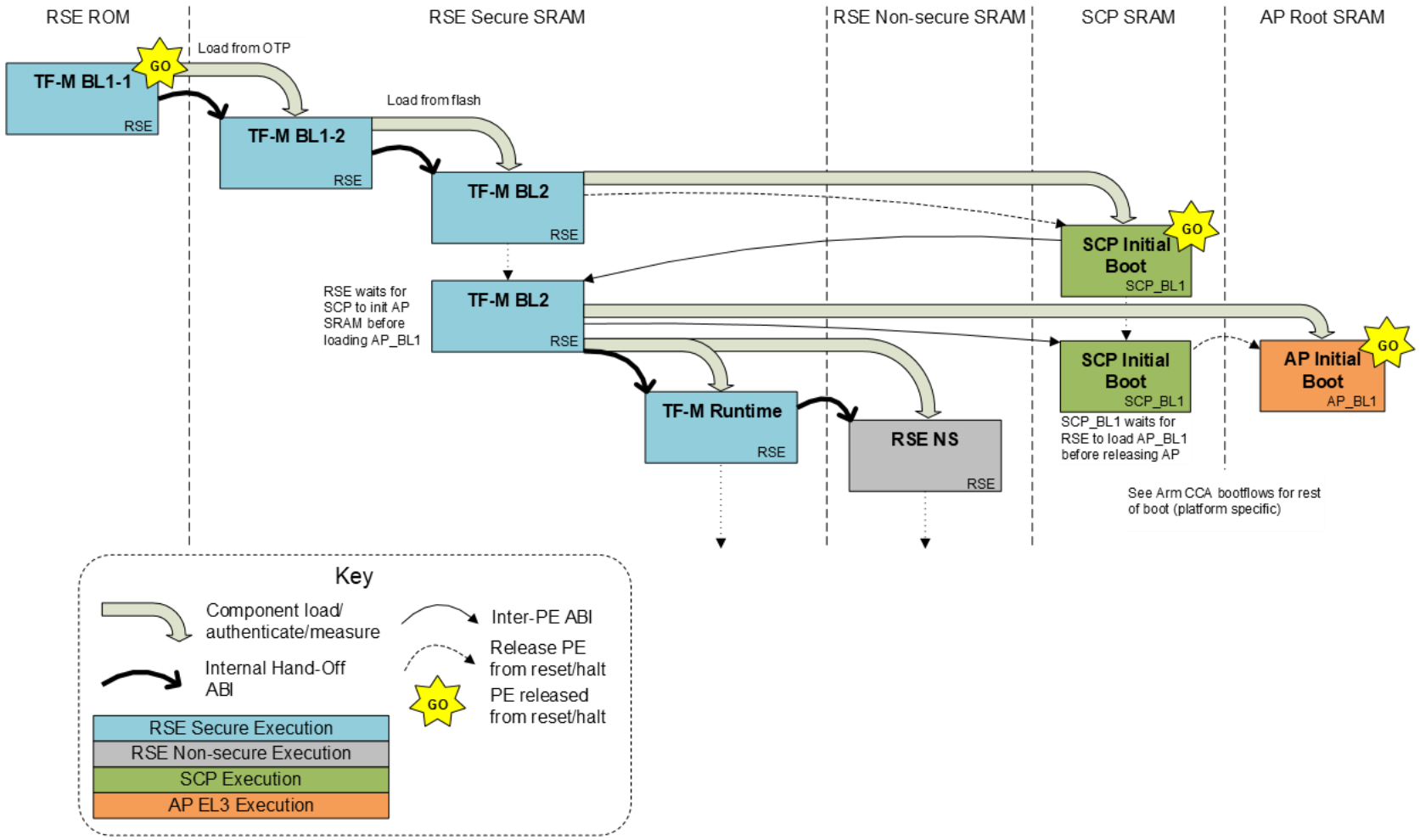


© 2023 Arm

RSE boot flow

TF-M BL1-2

- Loads TF-M BL2 from external flash into RSE SRAM
- Computes BL2 hash for measured boot
- Authenticates BL2 using LMS post-quantum signature scheme
  - BL1-2 is the last immutable stage: using quantum-safe crypto means rest of firmware could be updated to post-quantum crypto
  - Trade-off: limited to 1024 updates of BL2
- Checks BL2 freshness using anti-rollback counter in OTP
- Jumps to BL2…

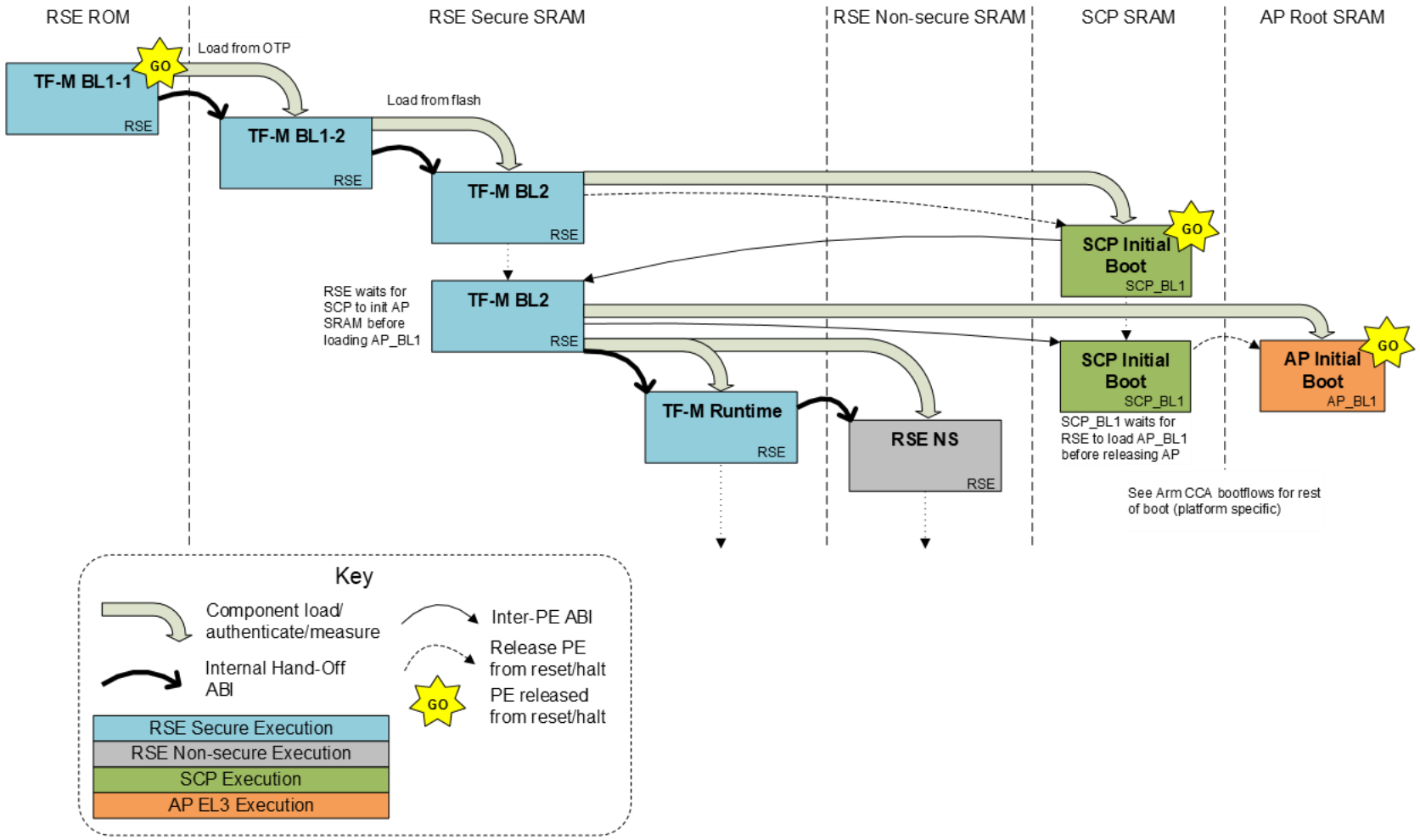


© 2023 Arm

RSE boot flow

TF-M BL2

- Using existing TF-M MCUBoot bootloader
- Loads images from flash into PEs local SRAM, measures and (RSA or ECC) authenticates, for:
  - Other PEs BL1s
  - TF-M runtime
  - RSE NS
- Inter-PE synchronization required
  - Release SCP from halt
  - Wait for SCP to init AP SRAM before loading AP BL1
  - Notify SCP after loading AP BL1
  - RSE<->SCP notifications are via MHU doorbells
- Jump to TF-M Secure runtime
  - TF-M runtime is responsible for any jump to NS (as usual on Cortex-M)



© 2023 Arm

Measured boot

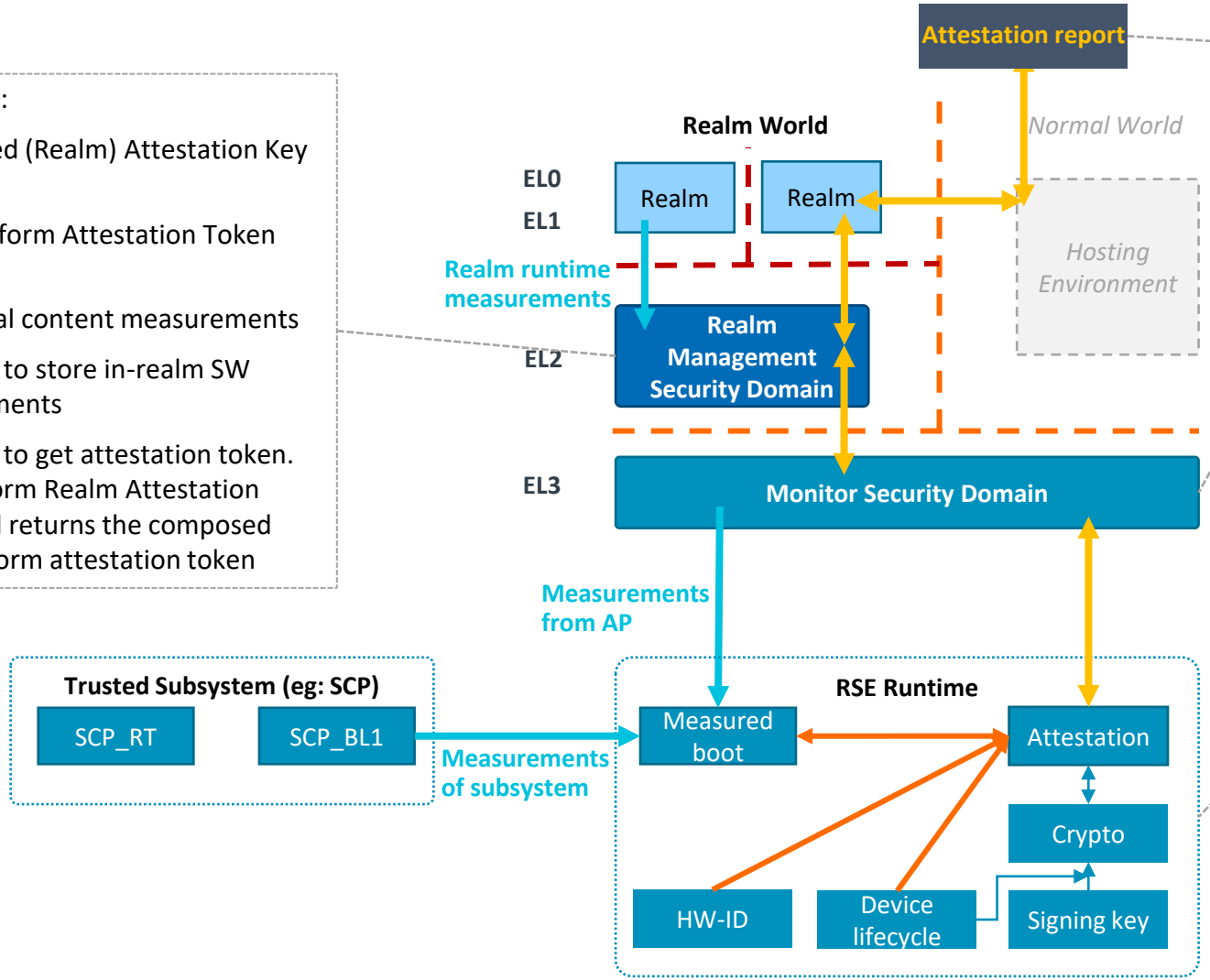
- Stores measurements of boot components after each is loaded
  - Measurements stored in RSE internal SRAM and rebuilt each boot
  - Complementary to secure boot
- Measurement =
  - SHA-256 hash of boot image, calculated over loaded image in-place
  - Associated metadata
    - Signer ID (hash of public key)
    - Version
    - Measurement algorithm (e.g. SHA-256)
    - SW type (e.g. “RSE_BL2”)
- RSE directly measures the components it loads
  - Receives further measurements over MHU for components loaded by other elements
- Extend semantics
  - measurement_slot_value = hash(new_measurement || old_measurement_slot_value)
  - But… since RSE puts measurements in RAM, we are not space constrained and recommendation is one slot per measurement
- API
  - tfm_measured_boot_extend_measurement(…)
  - tfm_measured_boot_read_measurement(…)

© 2023 Arm

CCA Attestation

RMM responsibilities:

- Requests Delegated (Realm) Attestation Key (DAK) from RSE
- Requests CCA Platform Attestation Token from RSE
- Makes Realm initial content measurements
- Provides interface to store in-realm SW runtime measurements
- Provides interface to get attestation token. Encodes data to form Realm Attestation Token, signs it and returns the composed Realm + CCA Platform attestation token



Attestation:

- Allows the user of a Realm to determine the trustworthiness of the Realm and the CCA platform it is running on
- Request received through Host OS & routed to corresponding realm through virtio

Monitor responsibilities:

- Pushes boot measurements (EL3, R-EL2) to RSE
- Mediates between RMM and RSE

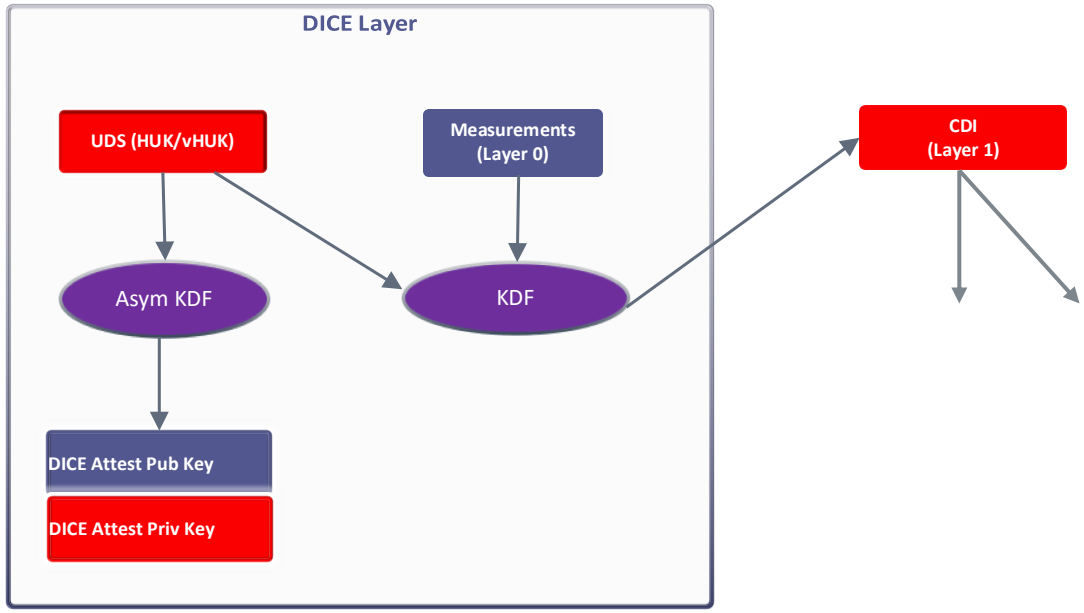
RSE ("HES Host") responsibilities:

- Owns initial attestation key (IAK), lifecycle state and HW-ID in shielded OTP
- Provides measured boot service to collect boot measurements
- Derives Delegated Attestation Key (DAK) from provisioned key, lifecycle state & boot measurements
- Provides interface to get CCA Platform Attestation Token, using hash of DAK public key as the challenge to create hash binding between Realm and CCA Platform tokens
- Encodes CCA Platform token, including all boot measurements, and signs with IAK

© 2023 Arm
arm

DICE with RSE

- RSE is well suited to implement further attestation usecases in addition to CCA
- Trusted Computing Group specifies the Device Identifier Composition Engine (DICE) concept:
  - A Compound Device Identifier (CDI) is derived from Unique Device Secret and measurement of first mutable code
  - Subsequent CDIs are derived from previous CDI and measurement of boot component
    - Forming a chain that corresponds to the boot chain
  - Attestation keys can be derived from each CDI and used to sign certificates
    - If any boot component changes, the whole certificate chain from that point changes
- DICE Protection Environment is an isolated environment used to manage DICE secrets, perform DICE derivations and sign attestation certificates
  - Protects CDIs from leakage
  - RSE implementation of DPE in progress



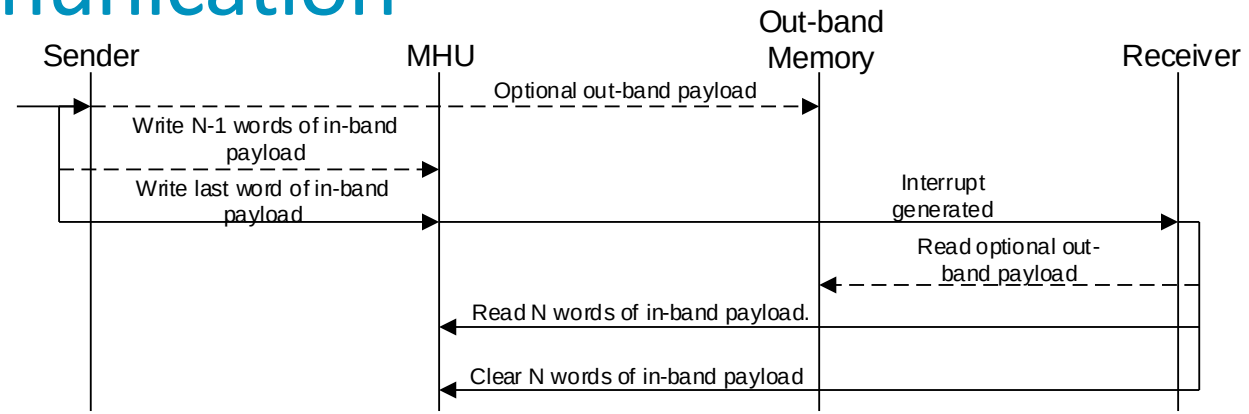
© 2023 Arm

Shielded locations

- RSE owns the system's OTP
  - With countermeasures against physical attacks
- AP firmware needs OTP for
  - Provisioned public keys for boot authentication
  - Anti-rollback counters for boot image versions
- RSE provides APIs based on usecase, rather than OTP read/write primitives
  - psa_export_public_key(psa_key_id_t key=*TFM_BUILTIN_KEY_ID_HOST_CCA_ROTPK*, …)
  - tfm_platform_nv_counter_increment(…)
  - tfm_platform_nv_counter_read(…)
  - Above are all existing TF-M APIs, made permissible to access over MHU
- Trusted Firmware-A upstream supports using RSE interface for counters
  - Using ROTPKs from RSE for trusted boot still todo

© 2023 Arm

AP to RSE communication



- Short RSE requests/responses are sent “in-band” between the host and RSE using MHUv3
  - Bidirectional communication using interrupts and memory mapped registers
- Longer messages are copied “out-band” between host RAM and RSE local SRAM using the RSE DMA-350
  - RSE maps windows of host memory into its own memory space using the Address Translation Unit (ATU)
- RSE side of communication is interrupt-driven
  - Client-side choice whether to block waiting for response
- MHU message protocol is a lightweight serialization of psa_call() function from the PSA Client APIs
  - Fixed-size parameters are packed into C struct, which is serialized in byte-order
  - Variable-length iovecs follow in variable-length trailer (or out-band memory)
  - Protocol ID in header specifies in-band or out-band protocol

© 2023 Arm
arm

More resources

- Documentation: https://tf-m-user-guide.trustedfirmware.org/platform/arm/rss/index.html
- Source Code: https://git.trustedfirmware.org/TF-M/trusted-firmware-m.git
- Integration with Trusted Firmware-A: https://trustedfirmware-a.readthedocs.io/en/latest/design_documents/rss.html

© 2023 Arm

Thank You
Danke
Gracias
Grazie
谢谢
ありがとう
Asante
Merci
감사합니다
धन्यवाद
Kiitos
شكرًا
ধন্যবাদ
תודה

© 2023 Arm

The Arm trademarks featured in this presentation are registered trademarks or trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. All rights reserved. All other marks featured may be trademarks of their respective owners.

www.arm.com/company/policies/trademarks

© 2023 Arm