QorIQ Platform's **Trust Architecture Overview**: Add Trust to Networked and Networking Systems

AMF-SNT-T1041

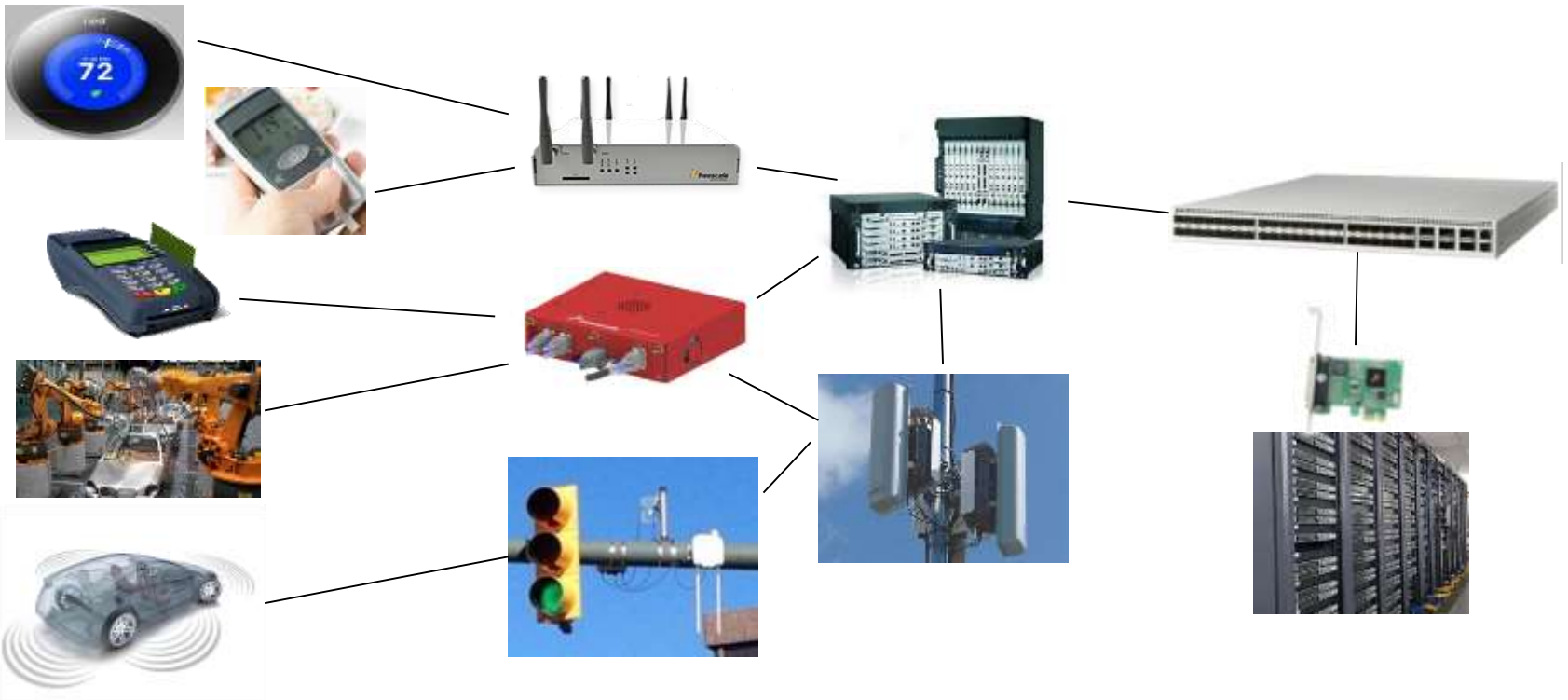
Geoff Waters | Systems Engineer

M A R . 2 0 1 5

Agenda

- What is a Trusted Platform?
- What does Trust Architecture provide?
  - Secure Boot
  - Memory Access Control/Strong Partitioning
  - Persistent Storage
  - Security State Monitoring
  - Master Secrets
  - Security Violation Detection
  - Secure Debug

Freescale Solutions for the Internet of Tomorrow



Securing the Internet of Tomorrow

- Trusted Platform for consumer privacy

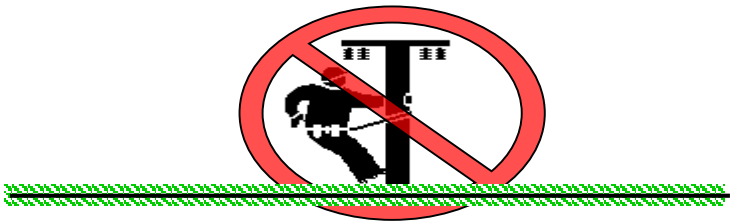
- Trusted Platform for PCI & HIPAA compliance

- Trusted Platform + Strong Functional Safety
- Low (& Determininstic) Latency Secure Message Exchange

- Software Defined Multi-Protocol Security
- High Performance Secure Data Transport
- Trusted Platform
- Virtualization and acceleration ease of use

- Data at rest protection
- High Secure Connection Rates

Internet of Things End Points | **Software Defined Network Infrastructure** | **Cloud Data Centers**



Trusted Platform

Protects against:
- Theft of user & 3rd party data
- Theft of functionality
- Theft of uniqueness (cloning)

Using:
- Secure Boot
- Secure Storage/Key Protection
- Tamper Detection
- Secure Debug
- Virtualization/Strong Partitioning

Secure Data Transport

Protects against:
- Masquerading
- Eavesdropping
- Data Manipulation
- Replay

Using:
- IPsec
- SSL/TLS
- MACSEC
- PDCP
- SRTP

Trusted Platform

Freescale's definition:

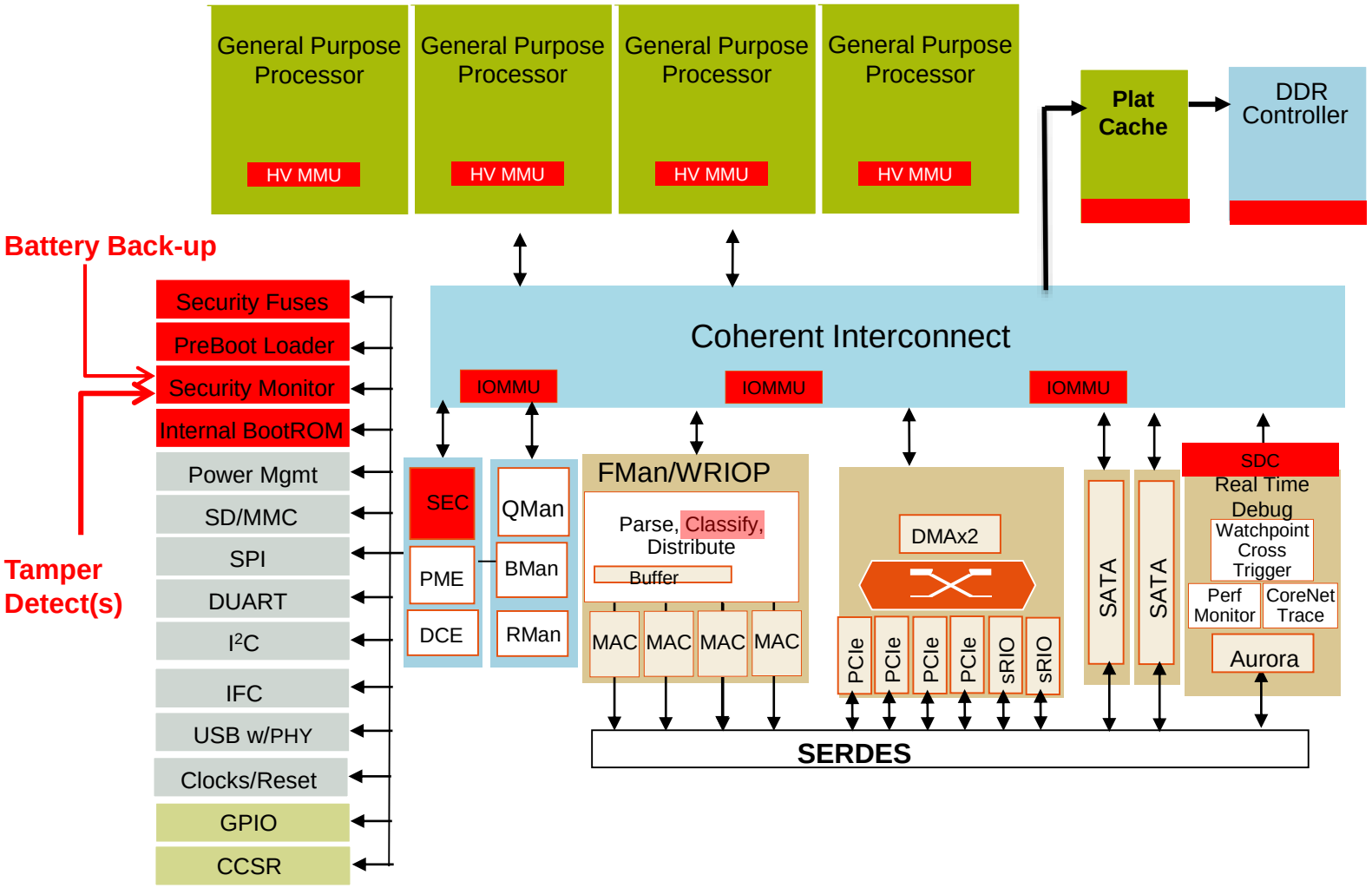
A Trusted Platform is a system which does what its stakeholders expect it to do, resisting attackers with both remote and physical access, else it fails safe.

Freescale Trust Architecture SoCs provide OEM controlled silicon features which simplify the development of trusted platforms. The Trust Architecture is an opt in scheme, with OEM controlled trade-offs in cryptographic strength, debug visibility, sensitivity of tamper detection, and anti-cloning mitigation.

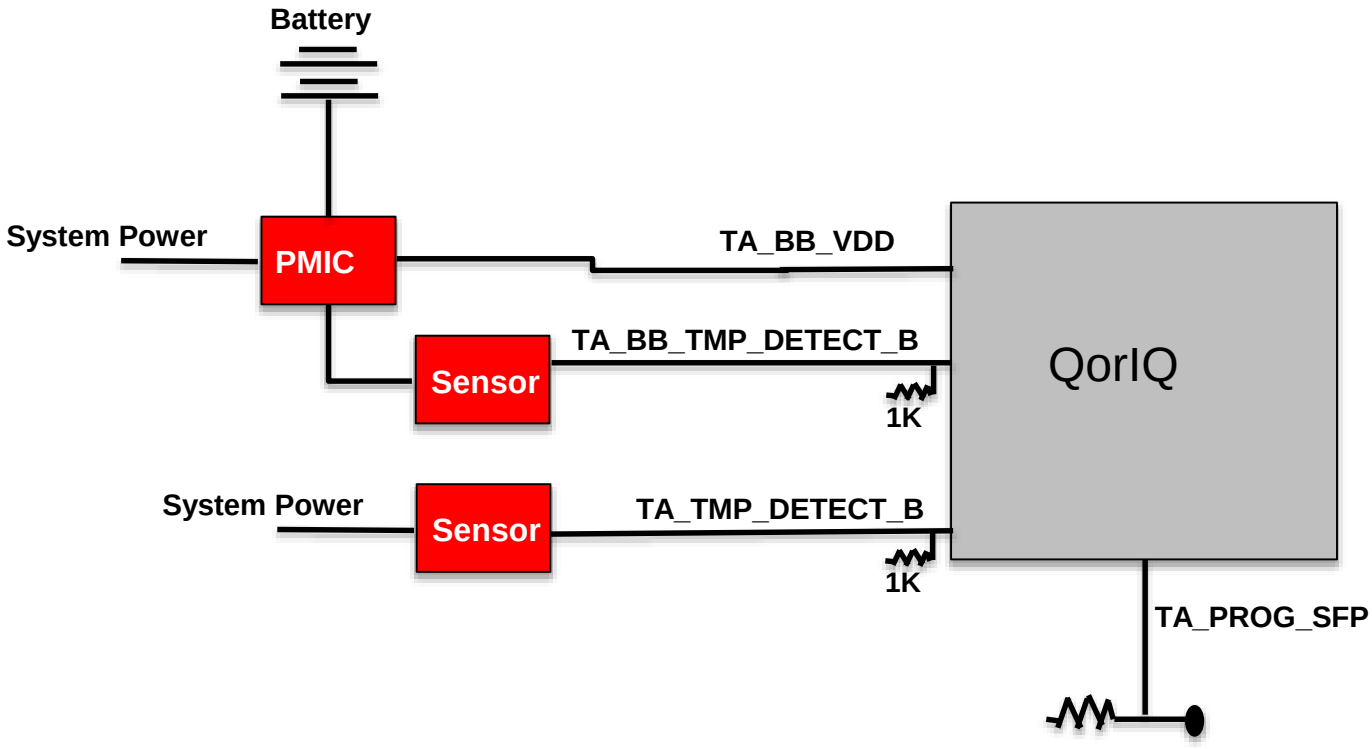
Trust Architecture: Out of Scope

1. **Preventing advanced physical attacks**
   - Deprocessing chip
   - Memory remnance attacks
   - All Side Channel Attacks
     - We do have side channel attack mitigation in our AES and Public Key accelerators.
   - Glitching with out of spec operating conditions
2. **Providing absolute partitioning/isolation**
   - We support strong partitioning is based on access control mechanisms. If a resource is private to partition 1, partition 2 must not be able to access that resource, directly or indirectly.
   - This is different from absolute partitioning or isolation, in which partition 2 is unable to interfere with the operation of partition 1. From internal and external bus bandwidth consumption to unfiltered buffer releases, Freescale knows of scenarios in which partition 2 can interfere with partition 1.
3. **Operating as a single edged sword**
   - Denial of service by triggering security violations is possible.

Generic Trust Architecture SoC



Trust Architecture Pins



- Bed of nails tester
- GPIO to FET

* Older devices use somewhat different pin names

Major Security Fuses

FSL Section

1b - FSL Section Write Protect

32b - FSL Unique ID

OEM Section

1b - OEM Section Write Protect

1b - Intent to Secure

Nb - Key Revocation (Trust 2.x+ only)

3b - Debug mode

Open

Conditionally closed w/o notification

Conditionally closed w/ notification

Locked

Root of Trust for Verification

256b – Super Root Key(s) Hash (Trust 2.x+ for list)

64b - Debug Challenge Value

Persistent device secrets

64b - Debug Response Value

256b - One Time Programmable Master Key

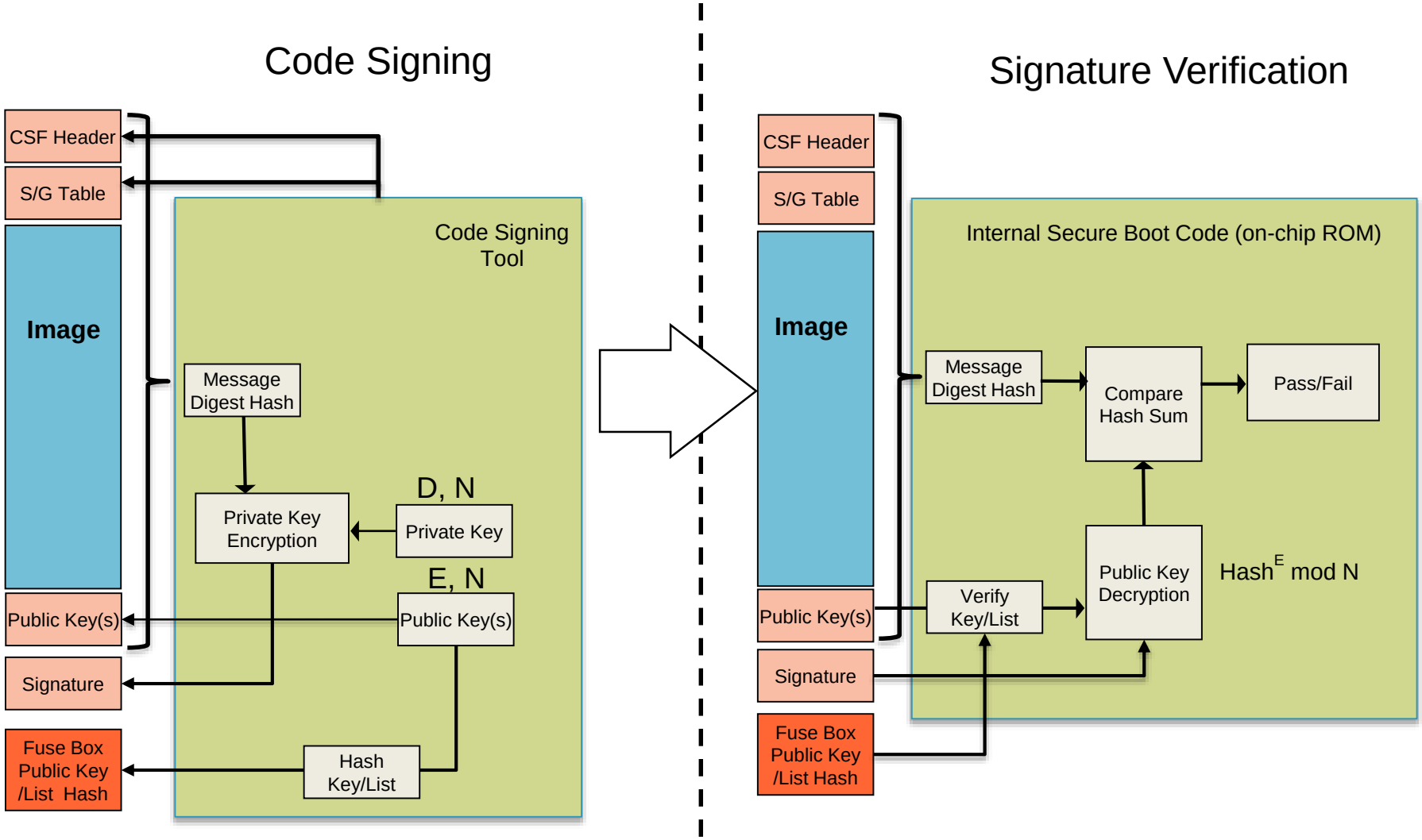
32b - OEM Unique ID

32b - OEM Scratchpad

32b - OEM Scratchpad

Secure Boot

Secure Boot Phases

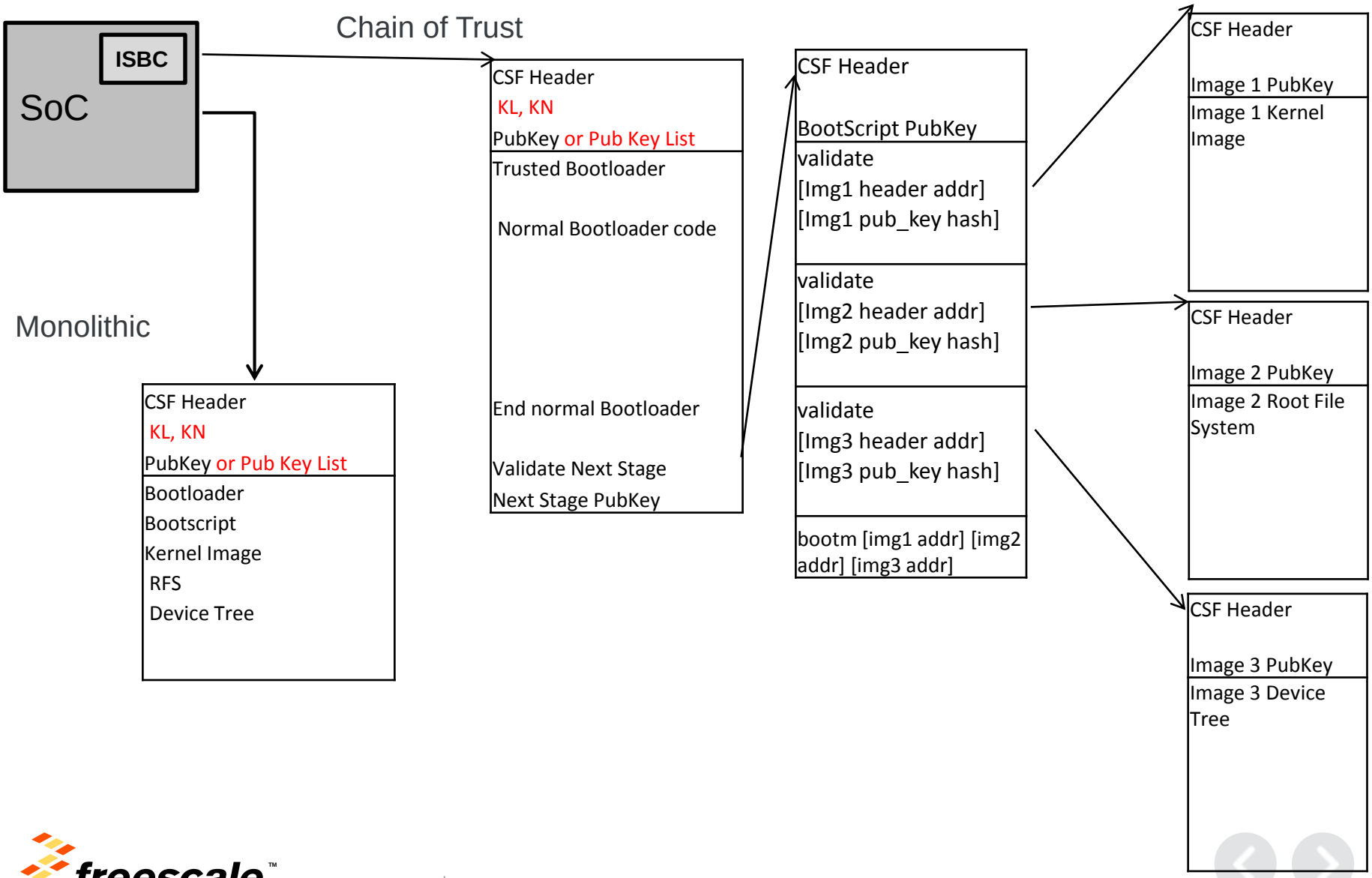


Code Signing Tool

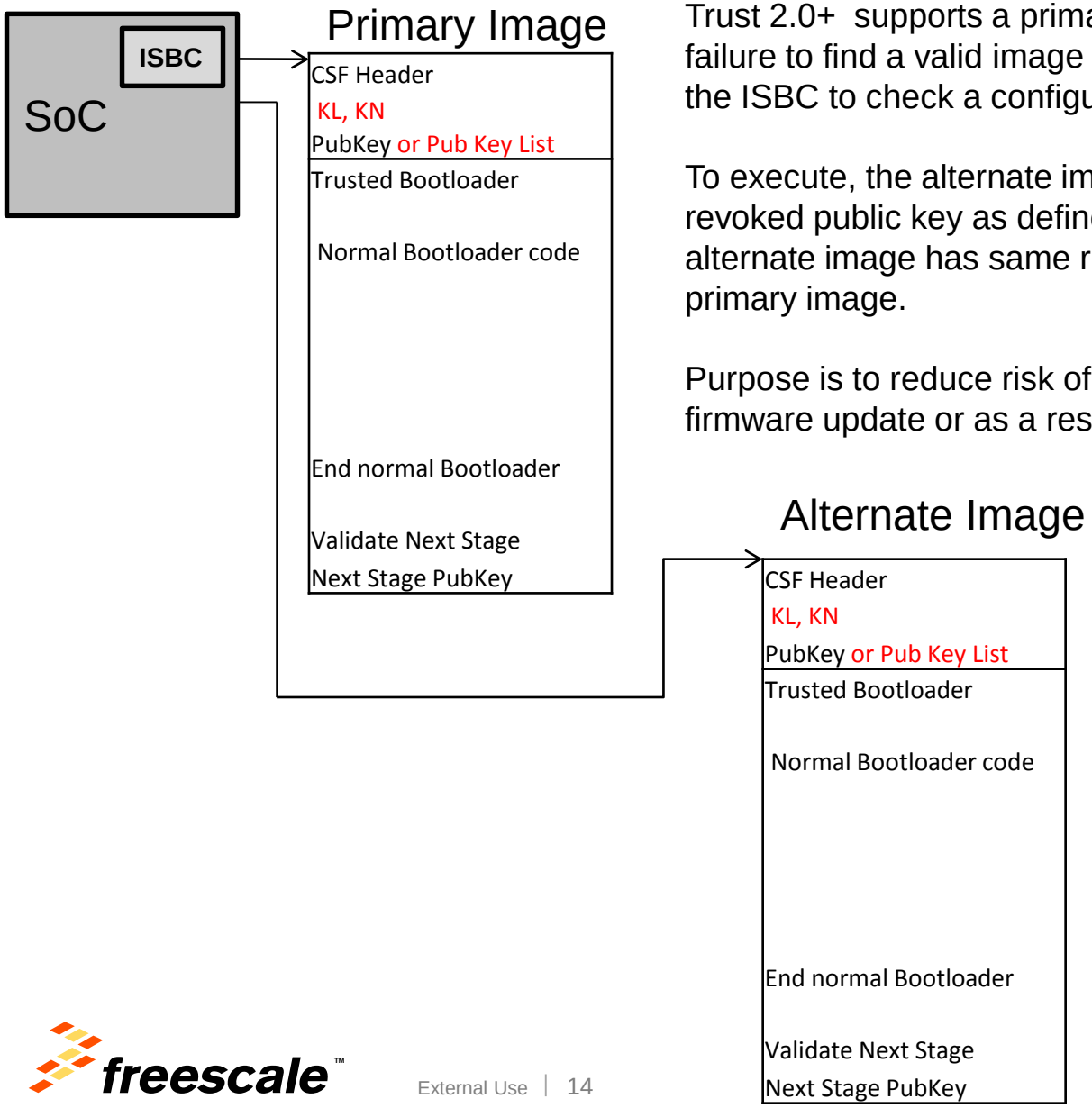
- The Freescale Code Signing Tool does some things you probably don't want to do for yourself:
  - Adding Command Sequence File (CSF) Header to the defined image
  - Embedding the hamming code in the OTPMK and DRV
  - Calculating SHA-256 hashes
  - But if you want, you can develop your own code to do this from our documentation!

- The CST also does some things which you may want to do for yourself:
  - Generating random numbers (which can be used as OTPMKs, ZMKs, and DRVs)
  - Generating RSA Public & Private Keys
    - But if want to generate your own key pairs, the CST will accept keys as inputs
      - Input keys to CST in PEM format
  - Signing Images
    - But if you want to sign images with your own signing facilities, the CST can simply add CSF header & export hash of image for signing
      - You still need to provide the public key in PEM format (this is part of the hashed image)
      - You also need to append PCKS#1v1.5 signature to signed image at location defined by CSF header.

Monolithic Secure Boot or Staged Chain of Trust



Alternate Image



Trust 2.0+ supports a primary and alternate image, where failure to find a valid image at the primary location will cause the ISBC to check a configured alternate location.

To execute, the alternate image must be validated using a non-revoked public key as defined by its CSF Header. A valid alternate image has same rights and privileges as a valid primary image.

Purpose is to reduce risk of corrupting single valid image during firmware update or as a result of Flash block wear-out.

FAIL!

- Symptoms of a secure boot failure: (ITS=1)

  1. System hangs (spins)
  2. System continually resets

- The ISBC code makes a reset request after writing the failure code to the error code register… A FPGA (with input from a switch) on the development system determines whether that reset request results in a reset.

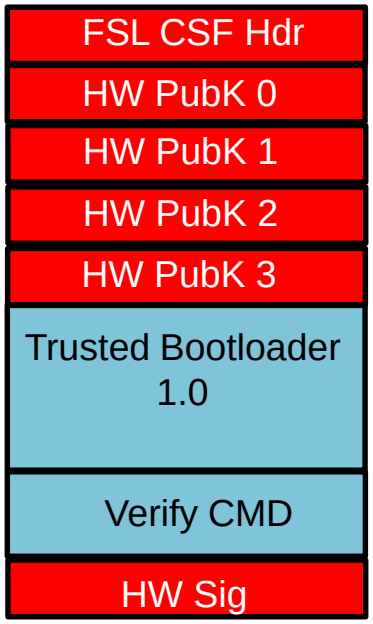
- Failures are detected and logged by secure boot software, including the ISBC and ESBC.

- The ISBC logs error reason codes in either the SCRATCHRW or Internal Boot ROM Error Code Registers (IBRECR), depending on whether the device implements a hardware or software PBL. Some error codes are specific to hard and soft PBL implementations.

- The ESBC/Trusted Bootloader logs the error reason code on the Uboot console.

- Trust 1.x and Trust 2.0 have similar error reason codes, however the Trust 2.0 list isn't a superset. Don't assume a value you've seen on a Trust 1.x device has the same meaning on a Trust 2.0 device.

What is Roll-Back?



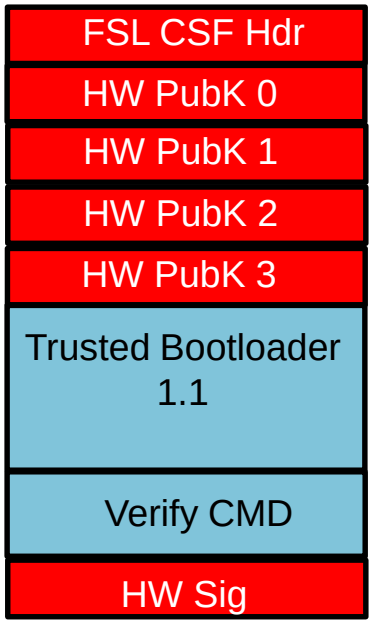
Authentic but flawed

OEM creates and signs the image on the left (1.0) with HW Private Key A. Image is validated by the ISBC using HW Public Key A.

Later, the image is determined to contain a security flaw, and is updated to the image on the right (1.1).

An attacker can't create his own image, but he can 'roll-back' the system's firmware to rev 1.0 in order to exploit this security flaw.

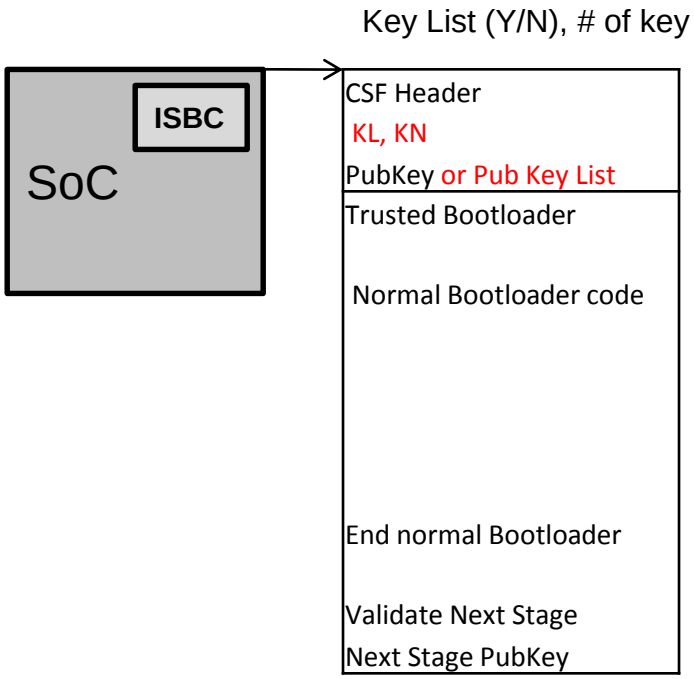
That is, unless anti-rollback methods are implemented.

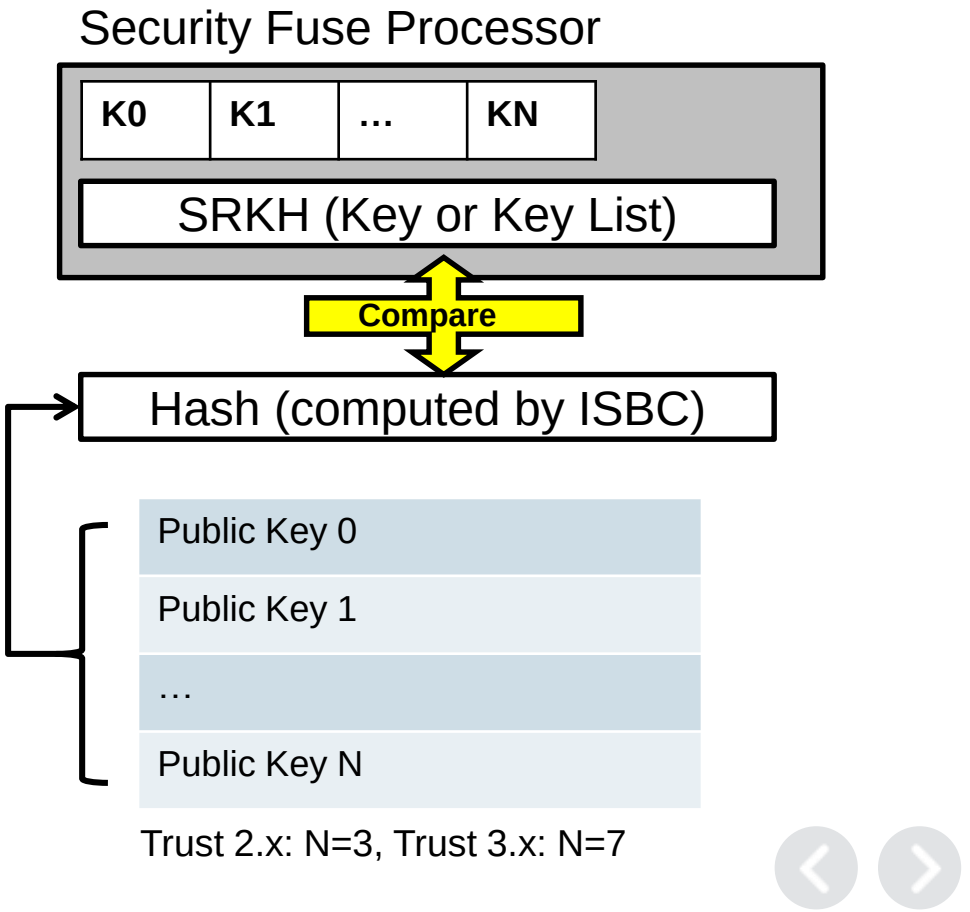


Authentic and fixed

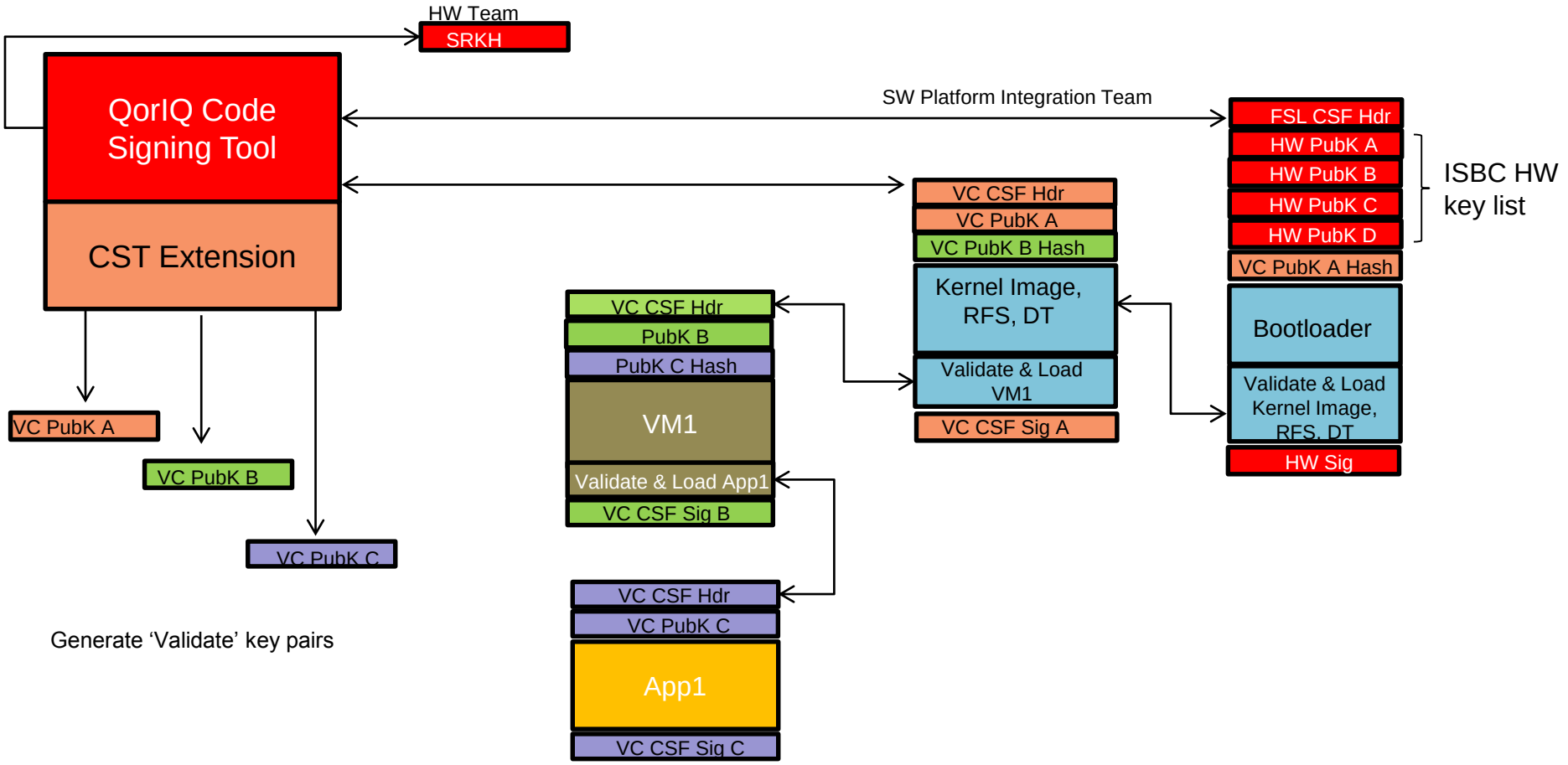
Key Revocation for Anti-Rollback

- Trust 2.x+ devices support key revocation. This feature provides rollback protection. ‘Valid’ but buggy images can be prevented from passing secure boot by revoking the public key used to validate it.

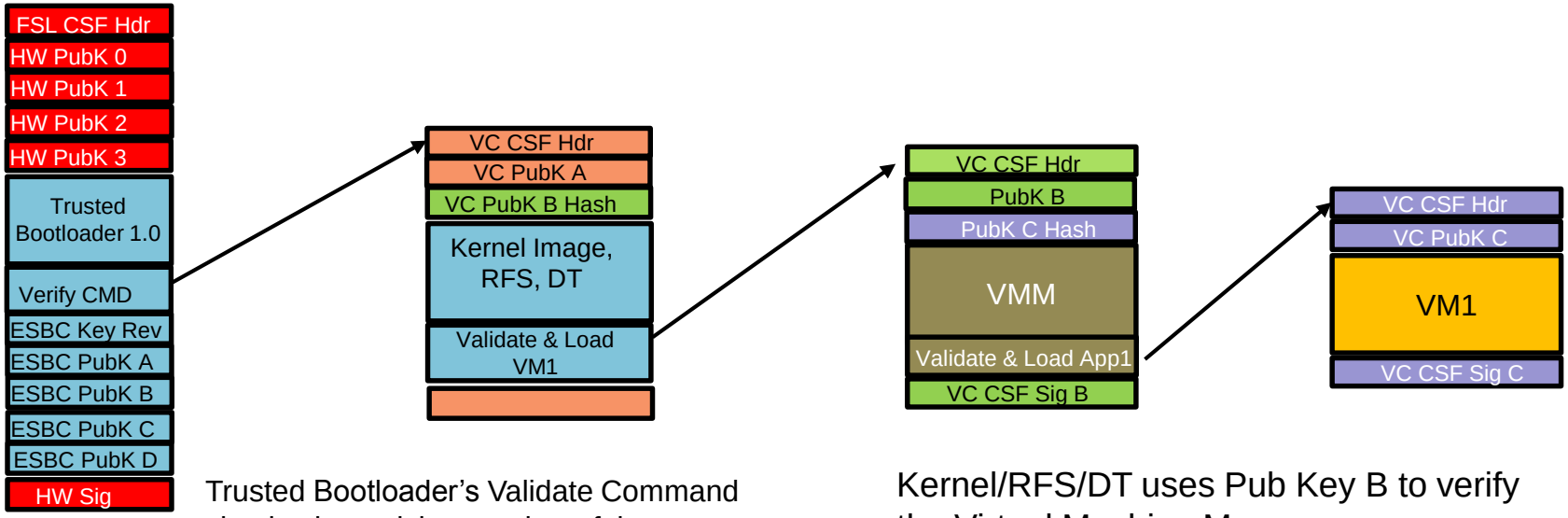




Code Signing Hierarchy



Extending Anti-Rollback via Chain of Trust & Monotonic Counter

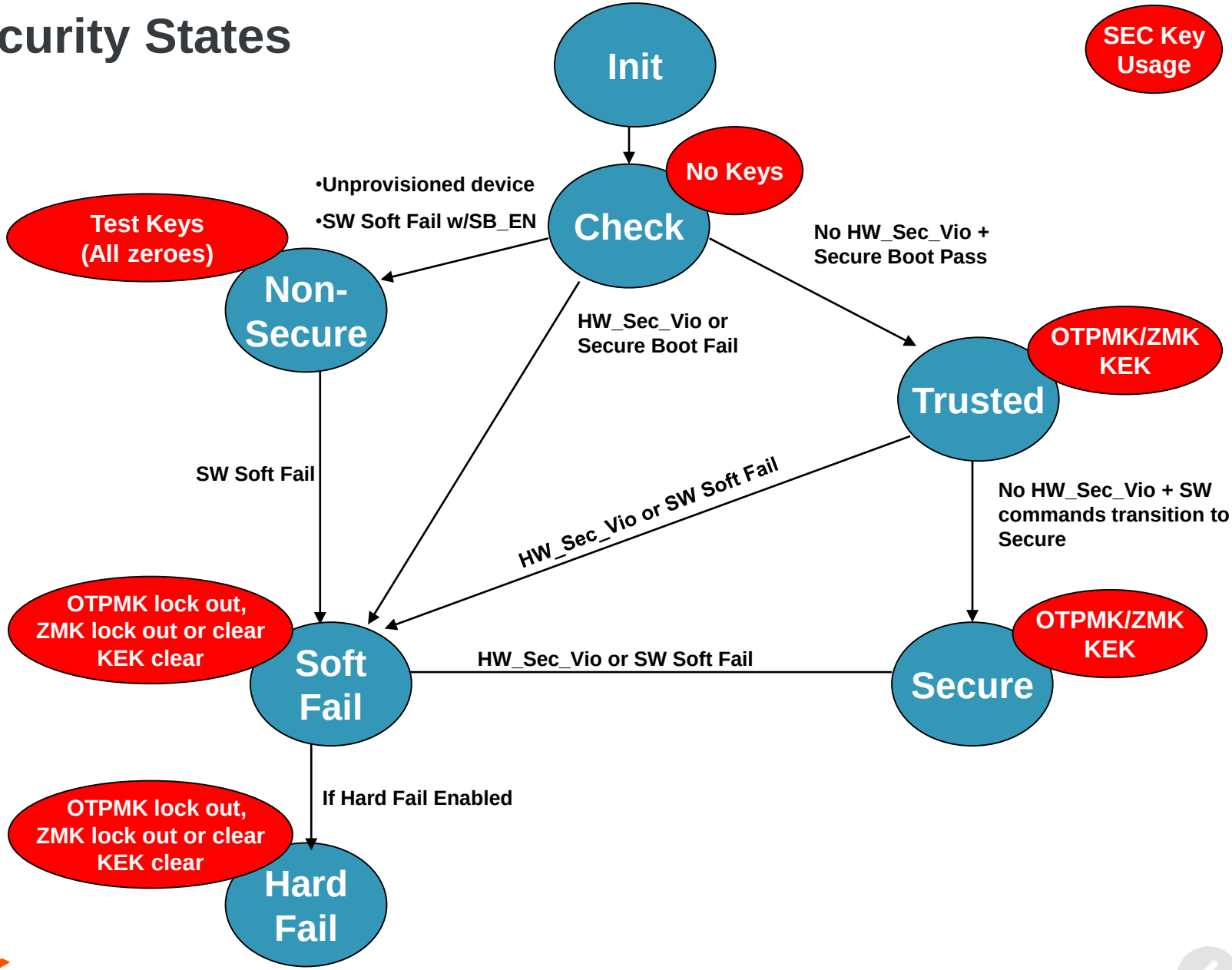


Trusted Bootloader's Validate Command checks the revision number of the Kernel/RFS/DT. If the Kernel/RFS/DT revision is >= the value of the monotonic counter, and the digital signature verification passes, the Kernel/RFS/DT is allowed to execute.

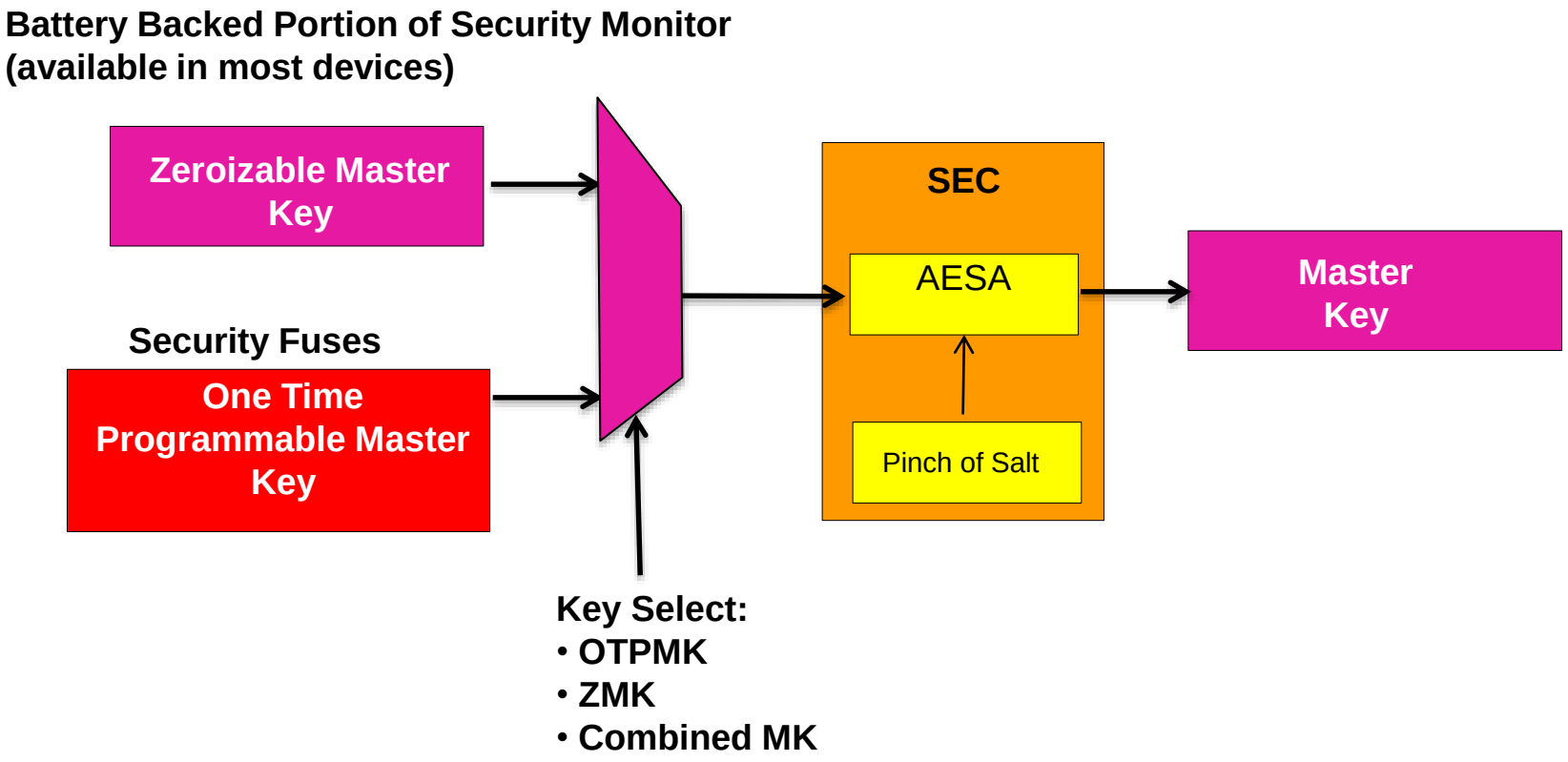
Kernel/RFS/DT uses Pub Key B to verify the Virtual Machine Manager. Kernel/RFS/DT's validate command includes the lowest acceptable revision number for the VMM. Same process is used to validate VM1.

Secure Storage/Key Protection

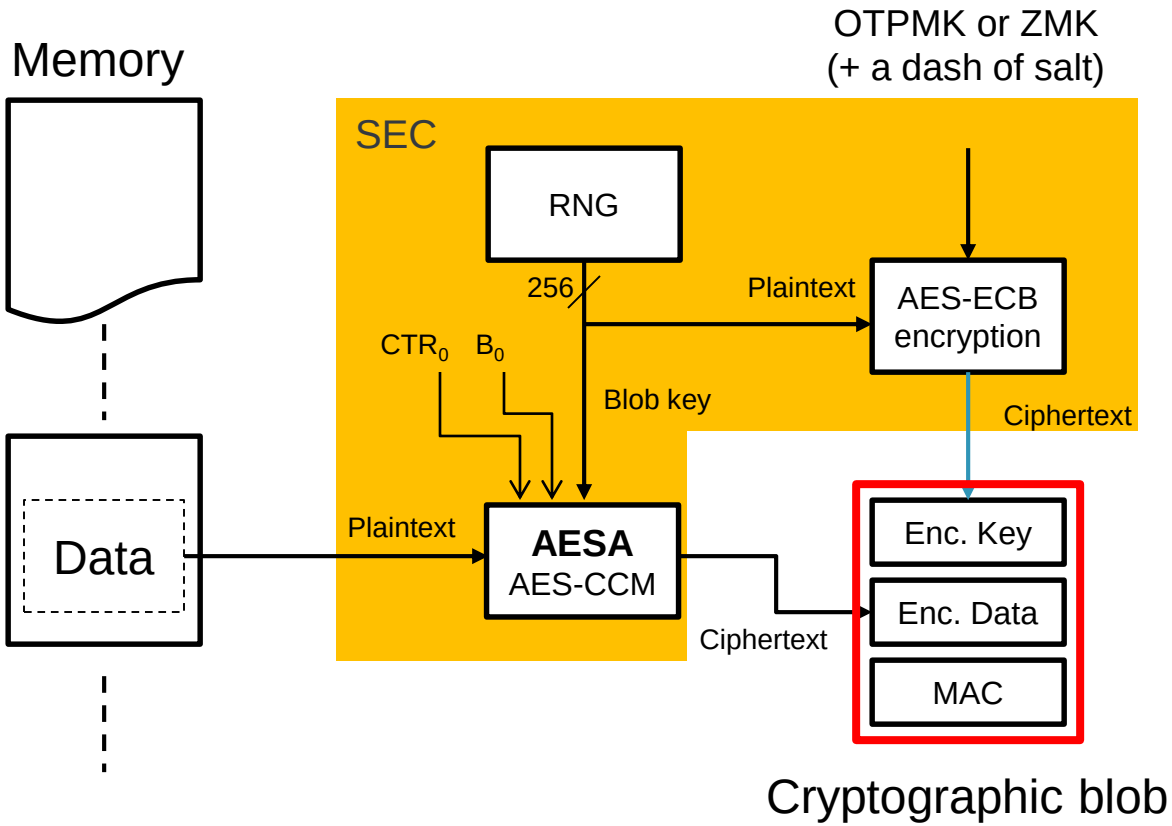
Security States



Persistent Master Keys

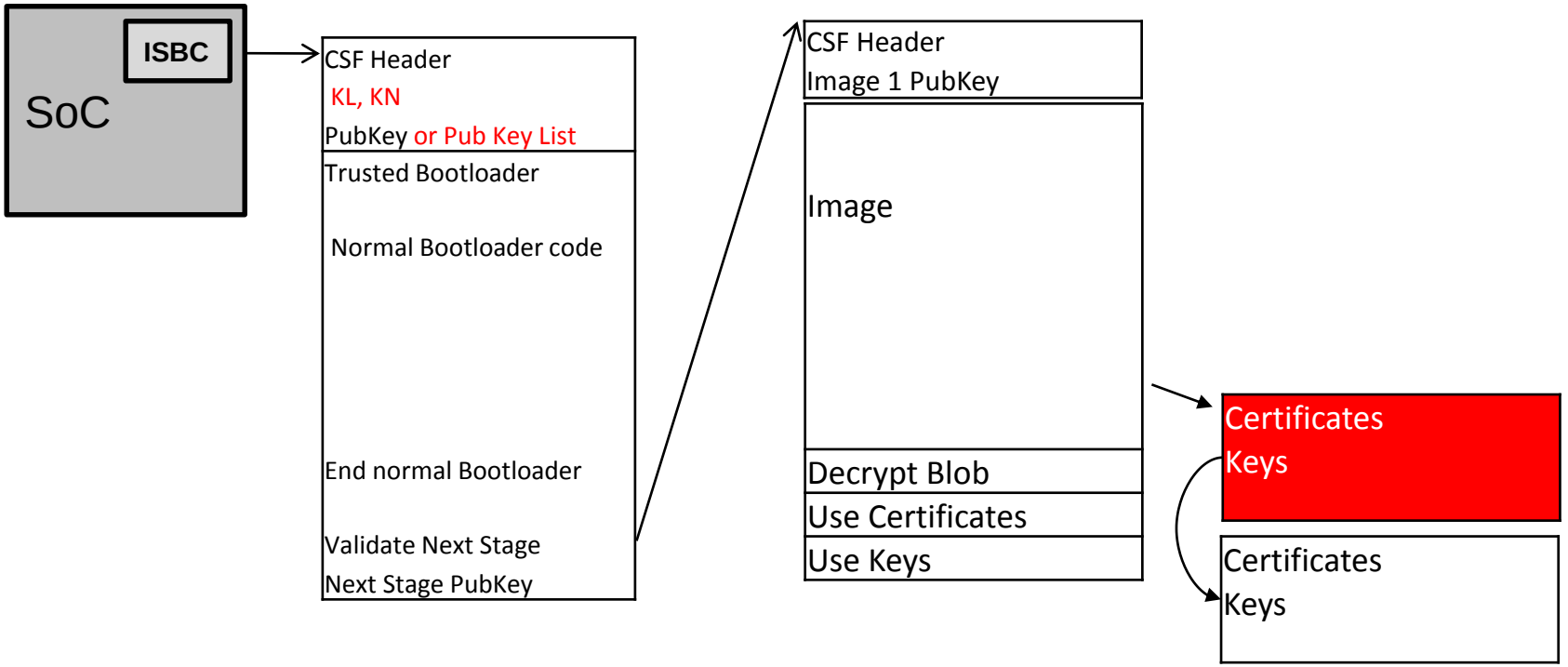


Secure Storage with Blobs

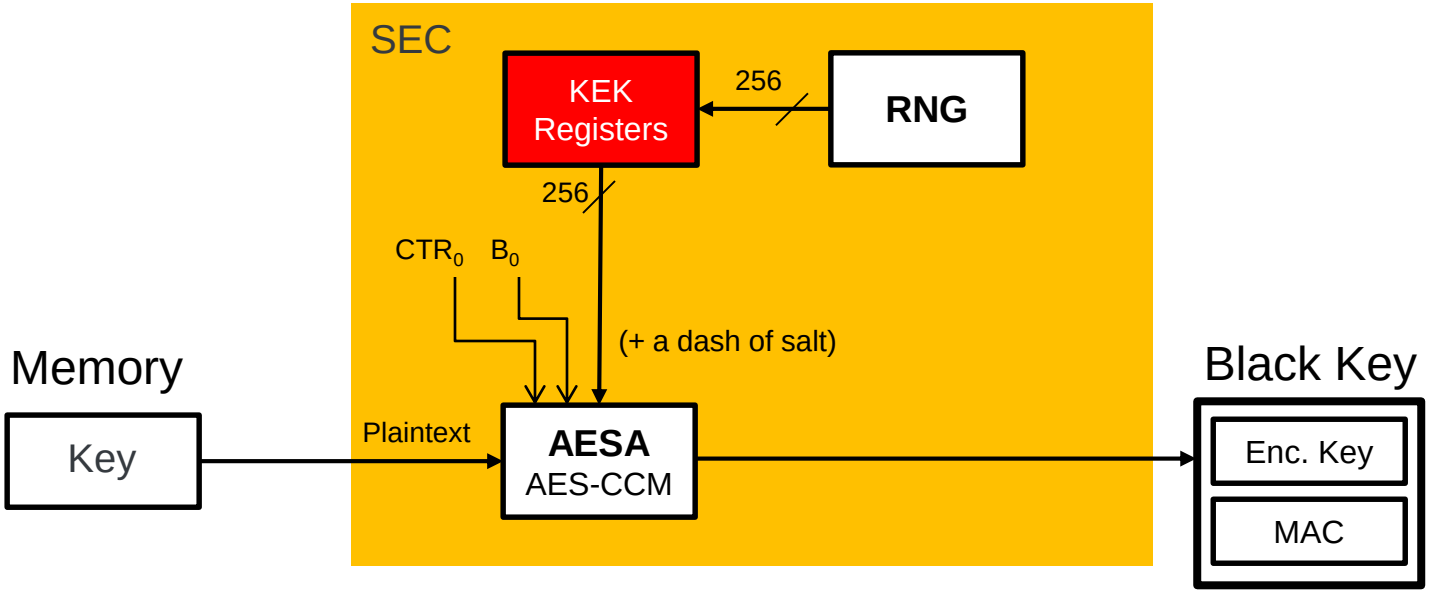


- Following successful secure boot, the SEC can be commanded to create blobs or decrypt them.
- There are data blobs (user specified input/output pointers) and key blobs.
- Key blobs encrypt the contents of a key register or decrypt the blob into a key register.

Secure Boot with Blobs



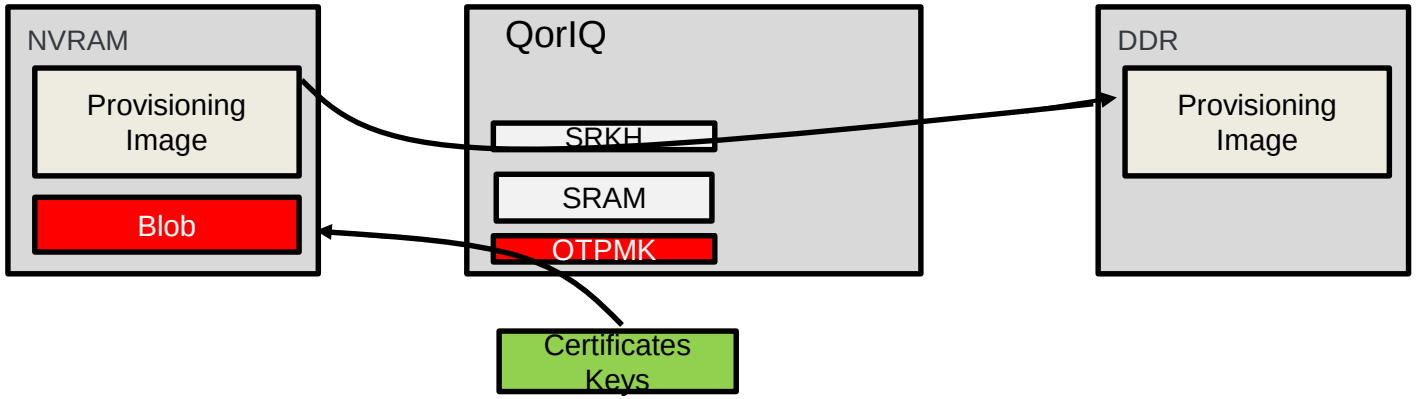
Key Protection



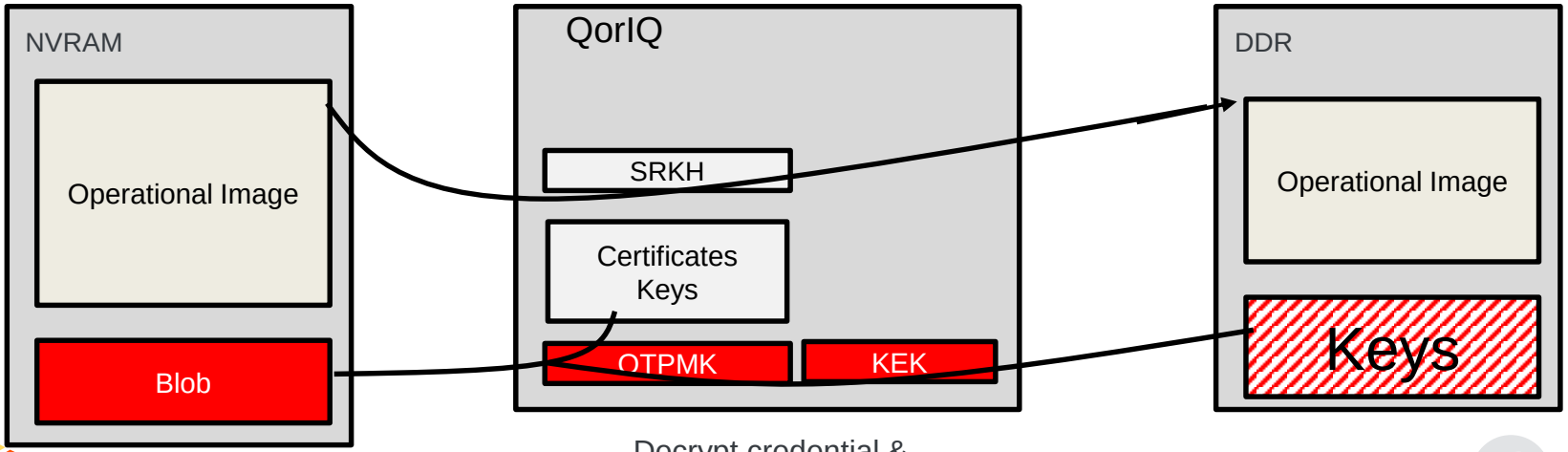
- Following successful secure boot, the SEC can be commanded to provision a Key Encryption Key (KEK).
- The KEK registers are loaded from the RNG.
- Once a KEK is provisioned, SEC descriptors can load a plaintext key and store an encrypted black key. Descriptors can also decrypt a key blob and re-encrypt as a black key. This allows provisioned keys to be moved from NVRAM to DDR.
- Black keys can be used by descriptors for normal operations, like IPsec,. Black keys are always decrypted into SEC key registers, within a minimal performance impact.

Provisioned Secrets

Factory Provisioned Credentials & Keys



Field Use of Credentials & Keys

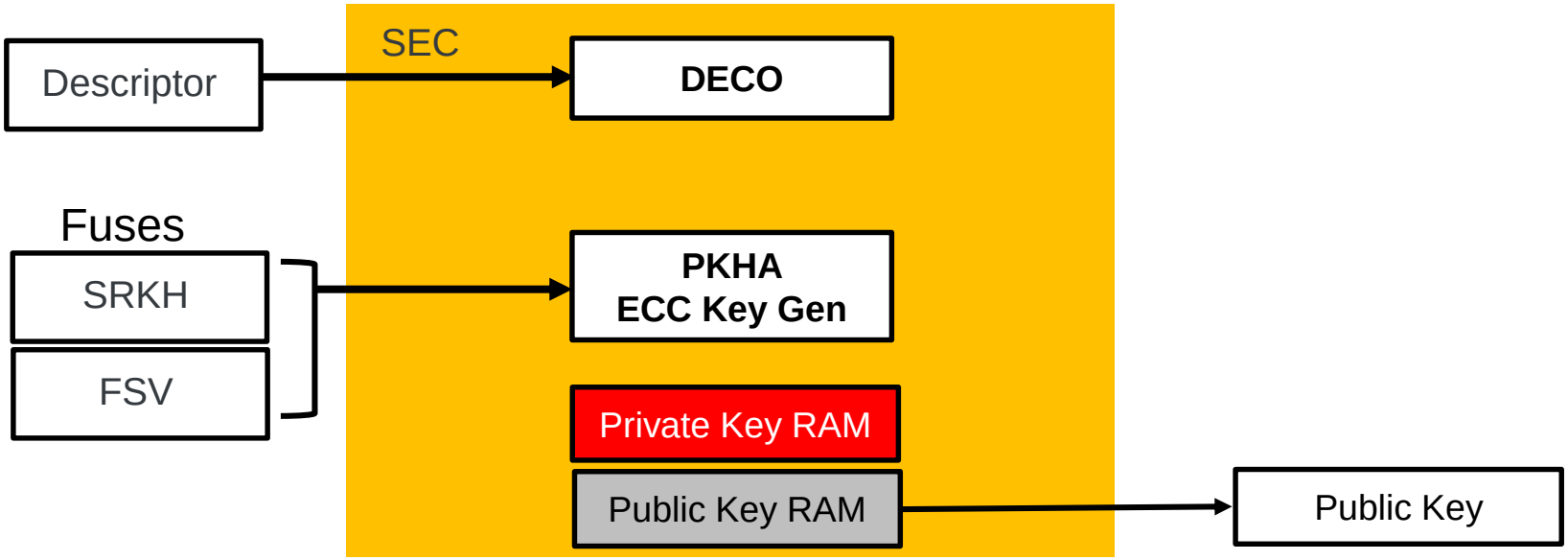


Decrypt credential & keys into on-chip SRAM

Hardware Key Pair

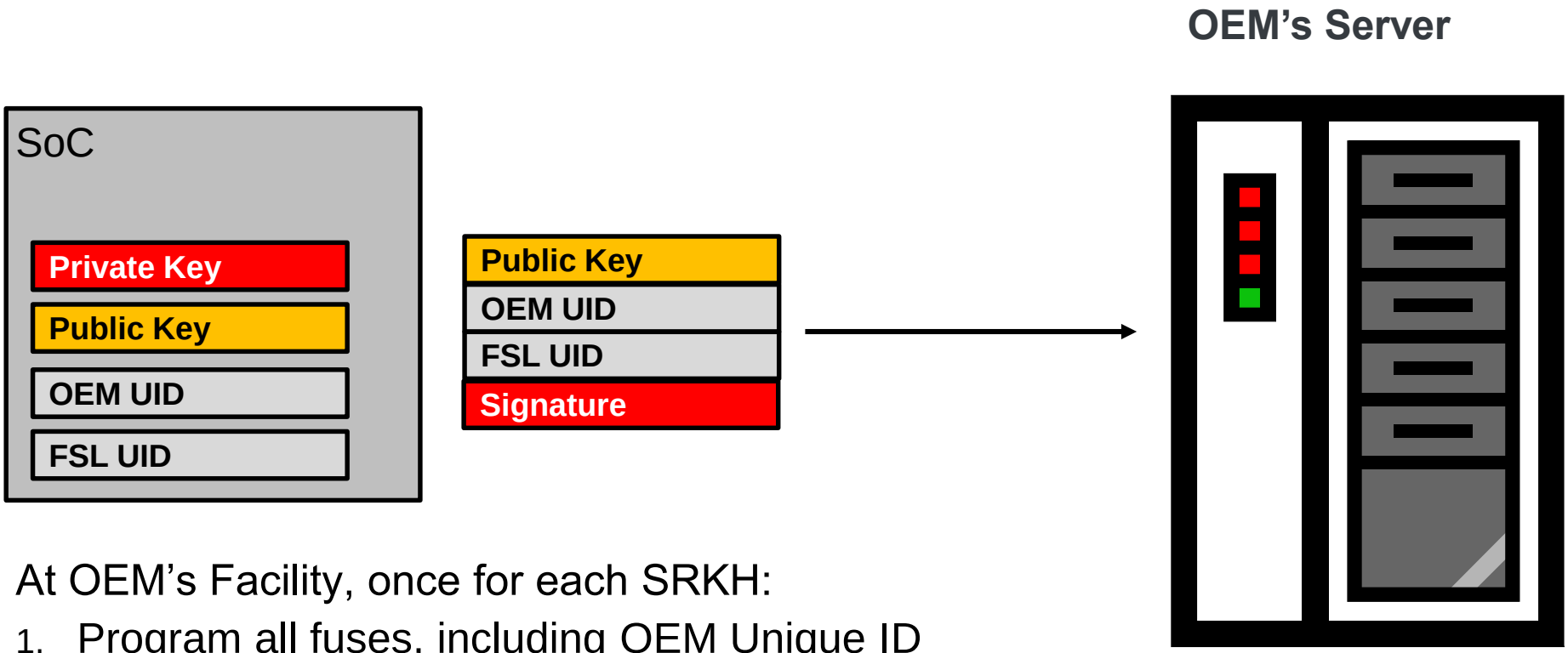
(aka Trusted Manufacturing)

Hardware Key Pair Generation



- Following successful secure boot, the SEC can be commanded to generate an ECC public/private key pair.
- The OEM programmed Super Root Key Hash and a Freescale Secret Value are the inputs to the Key Gen process.
- Once the Hardware Key Pair is generated, the Public Key is optionally output. The Private Key isn't readable by software, and cannot be output. It can only be used by the SEC.
- The same Hardware Key Pair is generated each time the Hardware Key Pair Generation is executed. The Keys are locked out & cleared in response to a security violation.

Trusted Manufacturing – Part 1

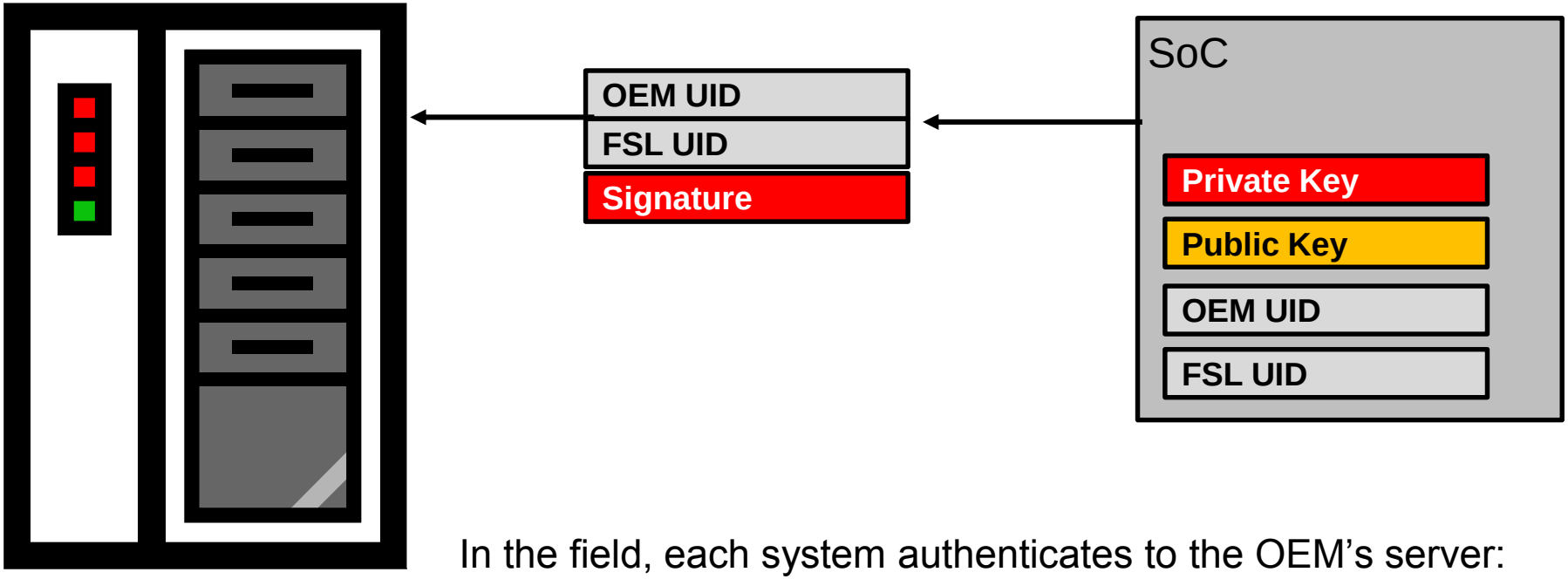


At OEM's Facility, once for each SRKH:

1. Program all fuses, including OEM Unique ID
2. Perform the Hardware Key Pair Generation operation
3. Export a file containing OEM & FSL Unique ID and public key, sign file with private key

Trusted Manufacturing – Part 2

OEM's Server



In the field, each system authenticates to the OEM's server:

1. Sends signed authentication message including OEM & FSL Unique IDs
2. Server verifies the message, confirms IDs match the public key used to verify the message
3. Server completes systems provisioning (TFTP of additional software), checks this system off list to detect cloning attempts

Tamper Detection

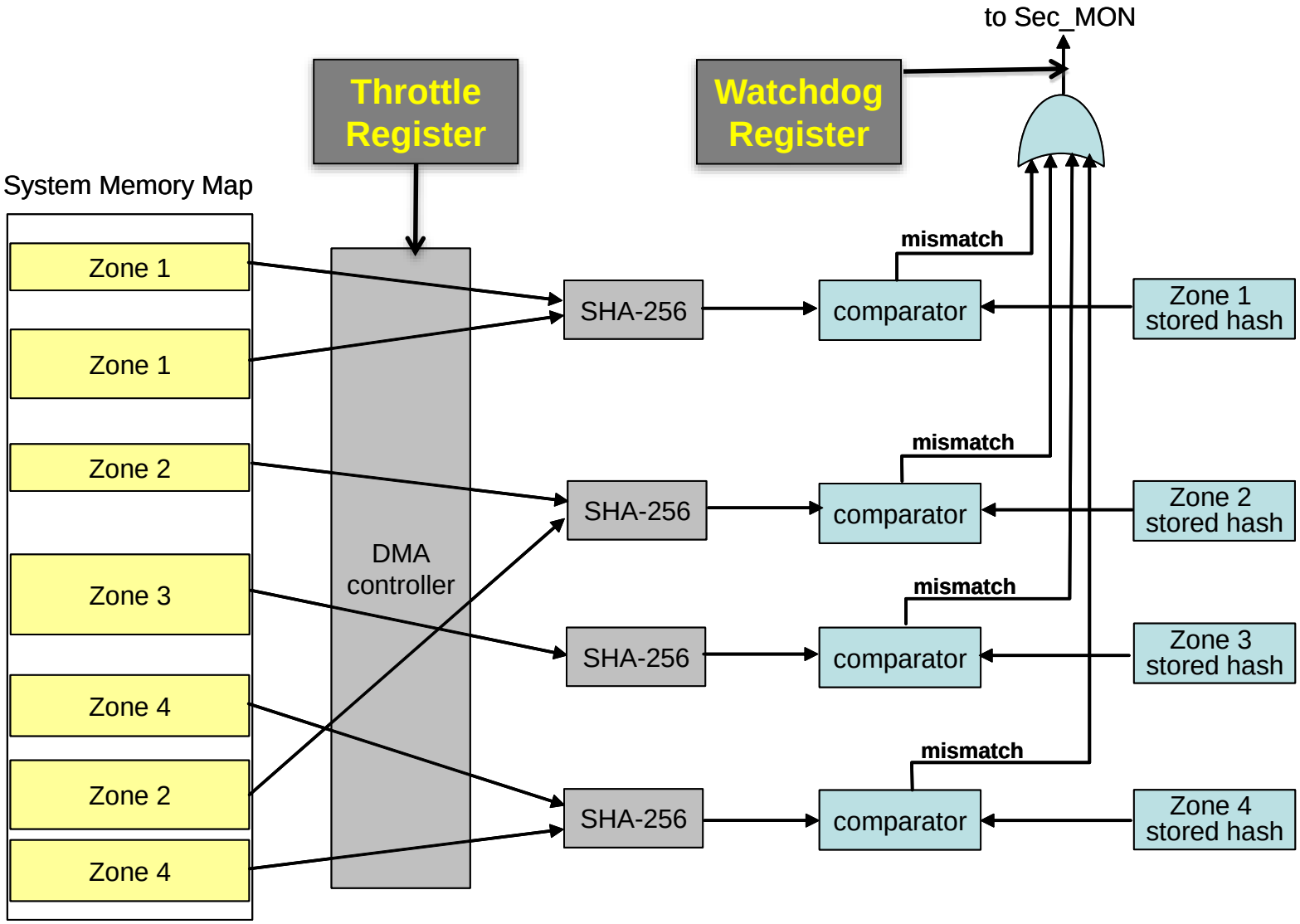
Tamper Detection Sources

- Hardware:
  - External Tamper Detection via TMP_DETECT and LP_TMP_DETECT
  - Secure Debug Controller (if set to Conditionally Closed with Notification)
  - Run Time Integrity Checker (in SEC)
  - Security Fuse Processor (if fuse array read fails, including hamming code check)
  - Security Monitor (OTPMK and ZMK hamming code check)
  - All sensitive flops upon detection of scan entry and exit (expert mode debug)
  - Power Glitch
  - In Trust 2.0:
    - Monotonic counter roll-over
- Software:
  - ISBC (Boot 0)
  - ESBC/Trusted-Uboot (Boot 1)
  - Any SW with write access to the Security Monitor can declare a security violation.

Secure Debug

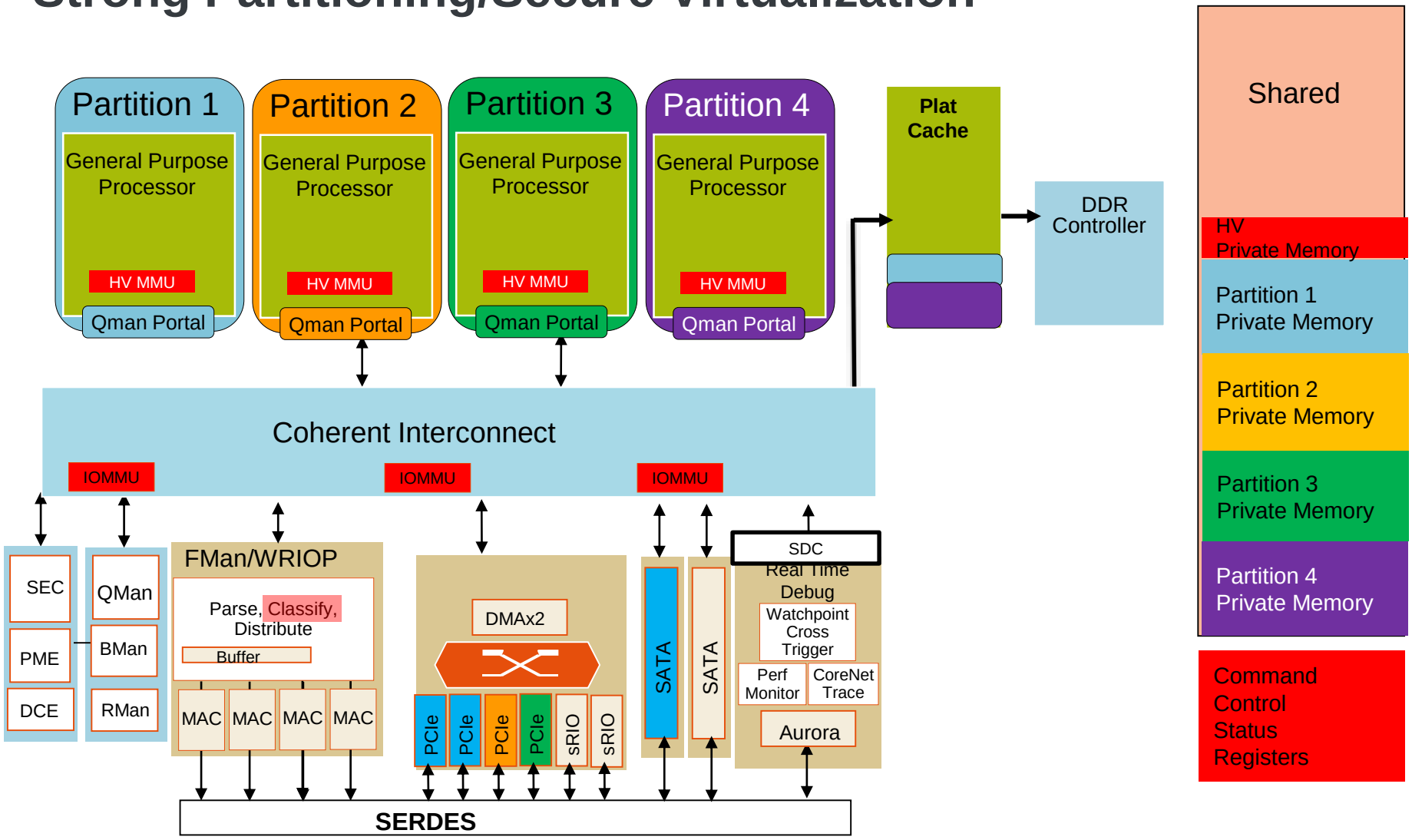
1. Open – Debug interfaces have full access to the QorIQ memory space. If the device is already in Secure state, device secrets remain usable. This setting is only appropriate in a lab environment.
2. Conditionally Closed without Notification – Debug interfaces are blocked until the user passes a challenge/response sequence.
   - PASS = full debug access, as in the Open case
   - FAIL = Access denied. 3 fails locks out chal/resp mechanism and reports Sec_Vio to Sec_Mon.
3. Conditionally Closed with Notification - Debug interfaces are blocked until the user passes a challenge/response sequence.
   - PASS = Sec_Mon notified of active debug, ephemeral device secrets cleared, persistent secrets locked out, followed by full debug access, as in Open case.
   - FAIL = Access denied. 3 fails locks out chal/resp mechanism and reports Sec_Vio to Sec_Mon.
4. Locked – All debug operations are blocked. The JTAG interface can still be used for boundary scan physical interconnect testing.

Runtime Integrity Checking



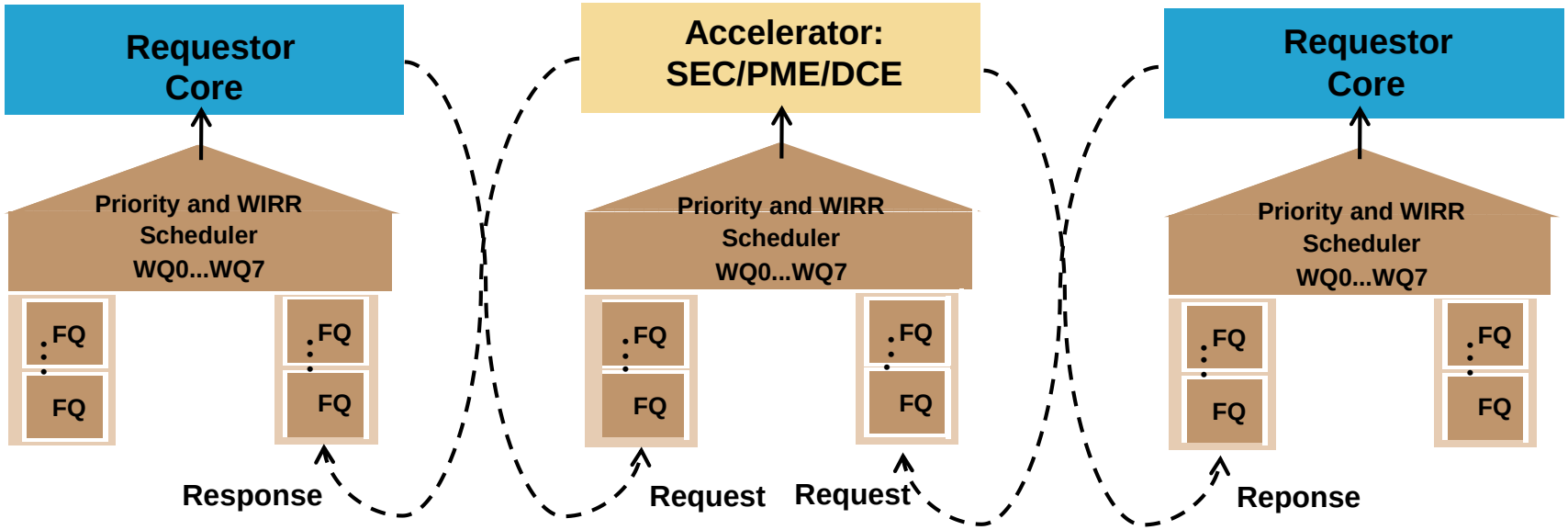
Strong Partitioning/Virtualization

Strong Partitioning/Secure Virtualization



Virtualized Accelerator Interface

- SEC, PME, and DCE are integrated into the DPAA
  - Acquire/release buffer pointers from/to BMan
  - Dequeue and enqueue frames from QMan
- QMan “virtualizes” these hardware accelerators
- QMan provides processing “context” and instructions with dequeued frames
  - e.g. crypto keys, IVs, ciphersuite
  - Simplifies software’s use of accelerators



Trust Architecture Deployment

Feature	Trust 1.0	Trust 1.1	Trust 2.0
Devices	P4080, P1010	P3041, P5020, P5040, P2041, BSC913x	C29x, T4240, T2080, T1040, B4
Secure Boot	Yes	Yes	Yes
HW Acceleration of Secure Boot	No	No	Yes. Only 'E' devices support secure boot.
Alternate Image	No	No	Yes, failure of primary image leads to validation attempt for alternate image.
Key List & Key Revocation	No	No	Yes, SRKH is hash of a list of up to 4 public keys, where up to 3 can be revoked with fuses.
Blobs based on Master Key	Yes, only Master Key option is OTPMK.	Yes, Master Key can be either OTPMK or ZMK. ZMK not available in BSC913x.	Yes, Master Key can be either OTPMK or ZMK. ZMK not available in B4.
Ephemeral Key Encryption Keys	Yes	Yes	Yes
Secure Debug Controller	Yes	Yes	Yes
Security Monitor High Power Section	Yes, including security state tracking and HW_Sec_Vio inputs from RTIC, SDC, SFP, & TMP_DETECT_B .	Yes, including security state tracking and HW_Sec_Vio inputs from RTIC, SDC, SFP, TMP_DETECT_B , and SecMon LP section.	Yes, including security state tracking and HW_Sec_Vio inputs from RTIC, SDC, SFP, TMP_DETECT_B , and SecMon LP section.
Security Monitor Low Power Section	No	Yes, including ZMK, and HW_Sec_Vio detection from Power Glitch, LP_TMP_DETECT_B. Not present in BSC913x.	Yes, including ZMK, 4 GPRs, and HW_Sec_Vio detection from Power Glitch, LP_TMP_DETECT_B , and Monotonic Counter Roll-Over. Not present in T1040 or B4.
Monotonic Counters	No	No	1 (Not present in T1040 or B4)
CPU Memory Access Control	Power ISA MMU w/HV (HV level not available in P1010)	Power ISA MMU w/HV (HV level not available in BSC913x)	Power ISA MMU w/HV (HV level not available in C29x)
IO Memory Access Control	Platform MMU (PAMU) in P4080. CCSR Access Control and PCIe ATMU in P1010.	Platform MMU (PAMU) in QorIQ. CCSR Access Control and PCIe ATMU in BSC913x.	Platform MMU (PAMU) in QorIQ. CCSR Access Control and PCIe ATMU in C29x.

Feature	Trust 2.0	Trust 3.0
Devices	C29x, T4240, T2080, T1040, B4	LS102xA, LS208xA
Secure Boot	Yes	Yes
HW Acceleration of Secure Boot	Yes. Only 'E' devices support secure boot.	Yes. Only 'E' devices support secure boot.
Alternate Image	Yes, failure of primary image leads to validation attempt for alternate image.	Yes, failure of primary image leads to validation attempt for alternate image.
Key List & Key Revocation	Yes, SRKH is hash of a list of up to 4 public keys, where up to 3 can be revoked with fuses.	Yes, SRKH is hash of a list of up to 8 public keys, where up to 7 can be revoked with fuses.
Blobs based on Master Key	Yes, Master Key can be either OTPMK or ZMK. ZMK not available in B4.	Yes, Master Key can be either OTPMK or ZMK.
Ephemeral Key Encryption Keys	Yes	Yes
Secure Debug Controller	Yes	Yes, plus TrustZone 'Secure World' additional protections
Security Monitor High Power Section	Yes, including security state tracking and HW_Sec_Vio inputs from RTIC, SDC, SFP, TMP_DETECT_B , and SecMon LP section.	Yes, including security state tracking and HW_Sec_Vio inputs from RTIC, SDC, SFP, TMP_DETECT_B , and SecMon LP section.
Security Monitor Low Power Section	Yes, including ZMK, 4 GPRs, and HW_Sec_Vio detection from Power Glitch, LP_TMP_DETECT_B , and Monotonic Counter Roll-Over. Not present in T1040 or B4.	Yes, including ZMK, 4 GPRs, and HW_Sec_Vio detection from Power Glitch, LP_TMP_DETECT_B , and Monotonic Counter Roll-Over.
Monotonic Counters	1 (Not present in T1040 or B4)	1
CPU Memory Access Control	Power ISA MMU w/HV (HV level not available in C29x)	ARM ISA MMU w/HV and TrustZone
IO Memory Access Control	Platform MMU (PAMU) in QorIQ. CCSR Access Control and PCIe ATMU in C29x.	Platform MMU (SMMU) in QorIQ Layerscape.
Hardware Key Pair (aka Trusted Mfg)	No	Yes

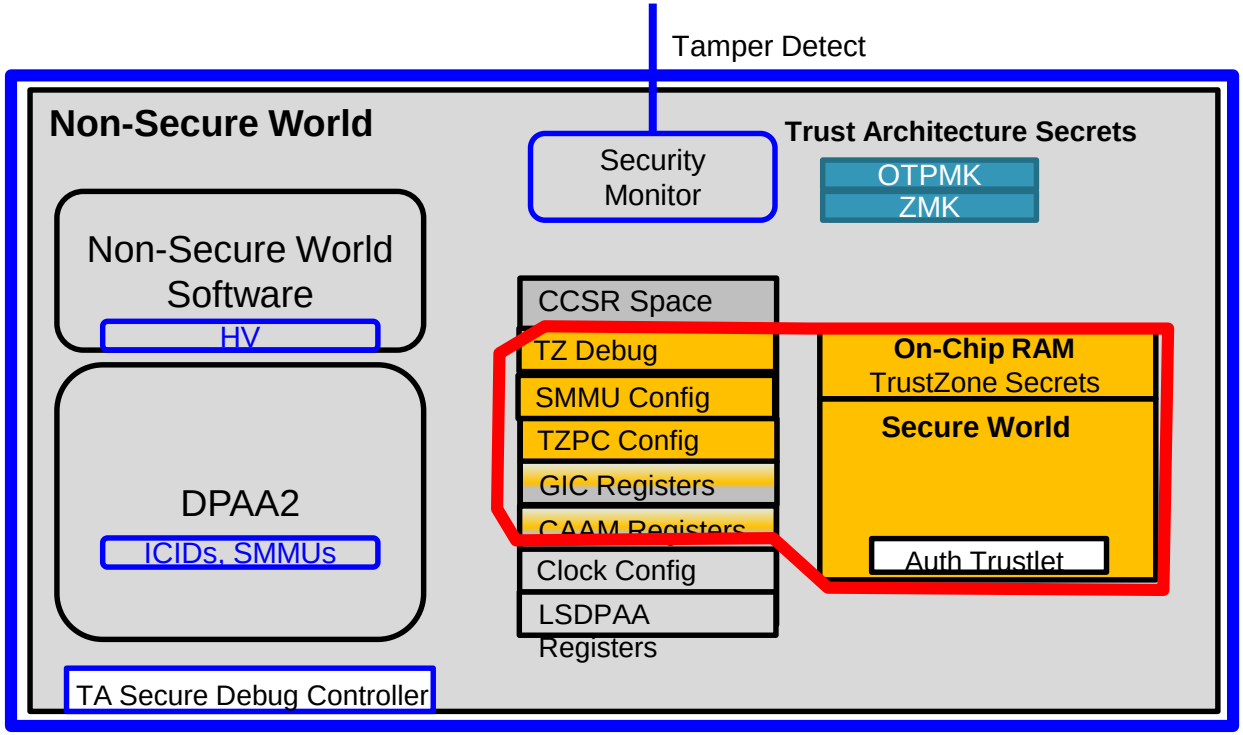
Major Enhancements and their use

Enhancement	First Appears	Benefit	Impact when used
Zeroizable Master Key	Trust 1.1	OEM can elevate the consequences of a security violation.	Requires battery back-up of portion of SoC (SecMon LP section), additional configuration registers and LP_TMP_DETECT
HW Acceleration of Secure Boot	Trust 2.0	Makes secure boot time closer to non-secure boot time	Trust only available in 'E' devices
Alternate Image	Trust 2.0	Adds resiliency to the secure boot process	Requires signing of 2 images, additional PBI commands
Key Revocation	Trust 2.0	Permanently revoke flawed images which were signed with a super root key.	Need to manage more pub/pri keys Need to develop key revocation images
Monotonic Counter	Trust 2.0	Prevent 'roll back' to a flawed image without revoking a super root key	Requires battery back-up of portion of SoC (SecMon LP section), need to include anti-rollback check in chain of trust
Hardware Key Pair (aka Trusted Mfg)	Trust 3.0	More intrinsic method of provisioning a device public/private key	Requires additional steps to generate key pair, export pub key. Requires database of IDs + public keys.
ARM TrustZone	Trust 3.0	Creates secure container (Secure World) where trusted applications can perform tasks on behalf of Non-Secure World applications. 3rd party software offerings.	Must still use Trust Arch to validate TrustZone software. Additional images to sign.

Trust Architecture + ARM TrustZone

Trust Arch provides a secure perimeter for trusted software.

TrustZone provides an inner keep for especially trusted software.



Resources

Trust Architecture Documentation

- Whitepapers
- App Notes
  - An Introduction to the QorIQ Platform's Trust Architecture (Doc# QORIQTAWP)
  - Manufacturing Guidelines for the QorIQ Platform's Trust Architecture (Freescale Trust Arch Mfg Guidelines NDA r3_4.pdf)

- User's Guides
  - User_Enablement_SecureBoot_PBL_platforms (SDK1.4).pdf (Code Signing, Error Codes)
  - SEC Reference Manual; SEC configuration and descriptor programming, including blobs

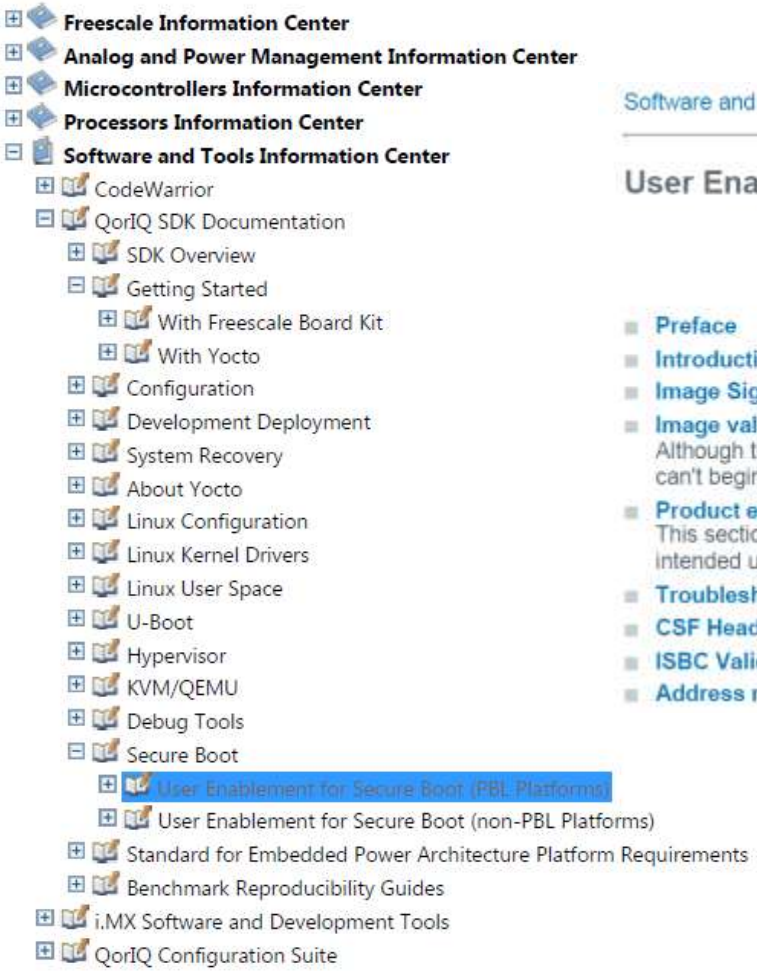
- SoC Reference Manual
  - Product specific
  - Trust Chapter contains;
    - Basic overview
    - Functional description
    - Detailed address map and registers of;
      - Security Fuse Processor
      - Security Monitor
  - Separate chapters for functions which are also used for non-secure operations
    - Access control/PAMUs
    - Boot Sources, RCW, & PBL

Software and Tools Information Center > QorIQ SDK Documentation > Secure Boot

http://www.freescale.com/infocenter/index.jsp?topic=%2FQORIQSDK%2F3069688.html ,

Follow the documentation/instruction in the SDK iso image. e.g.

/2915748.html



Trust Acronyms

- OEM: Original Equipment Manufacturer; the purchaser of the QorIQ device
- IBR: Internal Boot ROM
- ISBC: Internal Secure Boot Code
- ESBC: External Secure Boot Code
- PBL: Pre-Boot Loader
- SFP: Secure Fuse Processor
- RCW: Reset Configuration Word
- SRK: Super Root Key
- SRKH: Super Root Key Hash
- SecMon: Security Monitor
- SEC: Security Engine (crypto accelerator)
- OUID: OEM Unique ID
- FUID: Freescale Unique ID
- ITS: Intent To Secure
- OTPMK: One Time Programmable Master Key
- DCD: Device Configuration Data
- PAMU: Peripheral Access Management Unit
- ITS: Intent To Secure

Summary

- The QorIQ platform's trust architecture provides OEMs with the hardware anchor points they need to develop a trusted system.
- Trust features are found on a full spectrum of QorIQ, Qonverge, and Layerscape-based devices from low-power to highest performance.
- 3 generations of Trust Architecture have steadily enhanced the features and capabilities.
- Trust configurations are under OEM control, and support a support a tailored trade-offs between trust and serviceability.
- Trust features are supported with a Code Signing Tool and reference chain of trust.

www.Freescale.com