Secure Bootloader Signing Workflow for Arm-Based Embedded Systems

Enabling Compliance, Integrity, and Scalable Production

Presenter: Xin Qiu Aurora Networks™

Authors: Ron Birkett (TI), James Ni, Xin Qiu, Ting Yao, Lisa Yin

PKI Center™ www.pki-center.com

Xin Qiu, Ph.D.

Head of Security Solutions and PKI Center, Aurora Networks

Dr. Xin Qiu brings over 25 years of expertise in Public Key Infrastructure (PKI), device and software security, with a portfolio of 90+ patents.

She leads cross-functional teams delivering trusted, scalable security services to device manufacturers and network operators worldwide.

Agenda

- Secure boot fundamentals and evolving requirements
- Signing infrastructure and operational challenges
- Reference signing architecture
- TI AM6x case study
- Lessons learned and key takeaways

Secure Boot: What and Why

What is secure boot?

- Ensures devices start **only with approved, trusted software**
- Only software signed by an authorized key is allowed to run on the device

**Like a lock on the front door, only the person who has the right key can enter*

What Happens Without Secure Boot?

- Devices can be permanently compromised
- No reliable recovery or updates
- Risk of recalls, fines, and blocked market access

Why Signing Keys Are Critical?

- The security in Secure Boot depends on **how signing keys are handled**
- If mishandled, attackers can create "trusted" malware
- EU CRA expects strong, auditable key protection

ARM-Based Secure Boot Flow

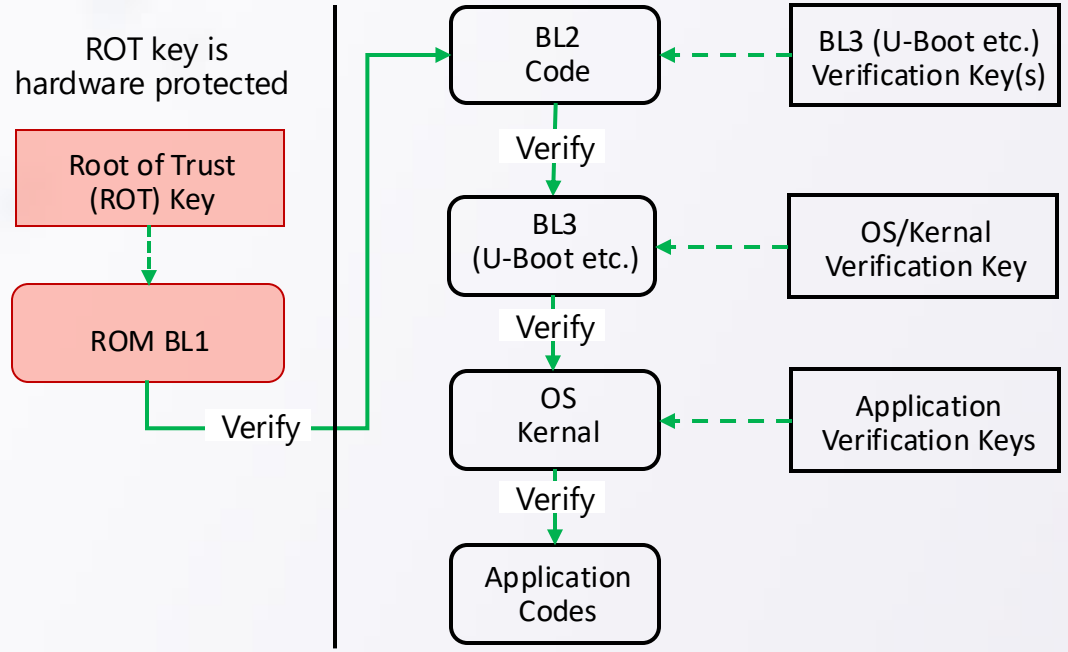
Trust is anchored in immutable hardware (ROM + Root of Trust key)

Each stage verifies the integrity of the next stage

Cryptography signatures enforces the chain of trust

Signing workflow and key management - not cryptography itself - are the weak point

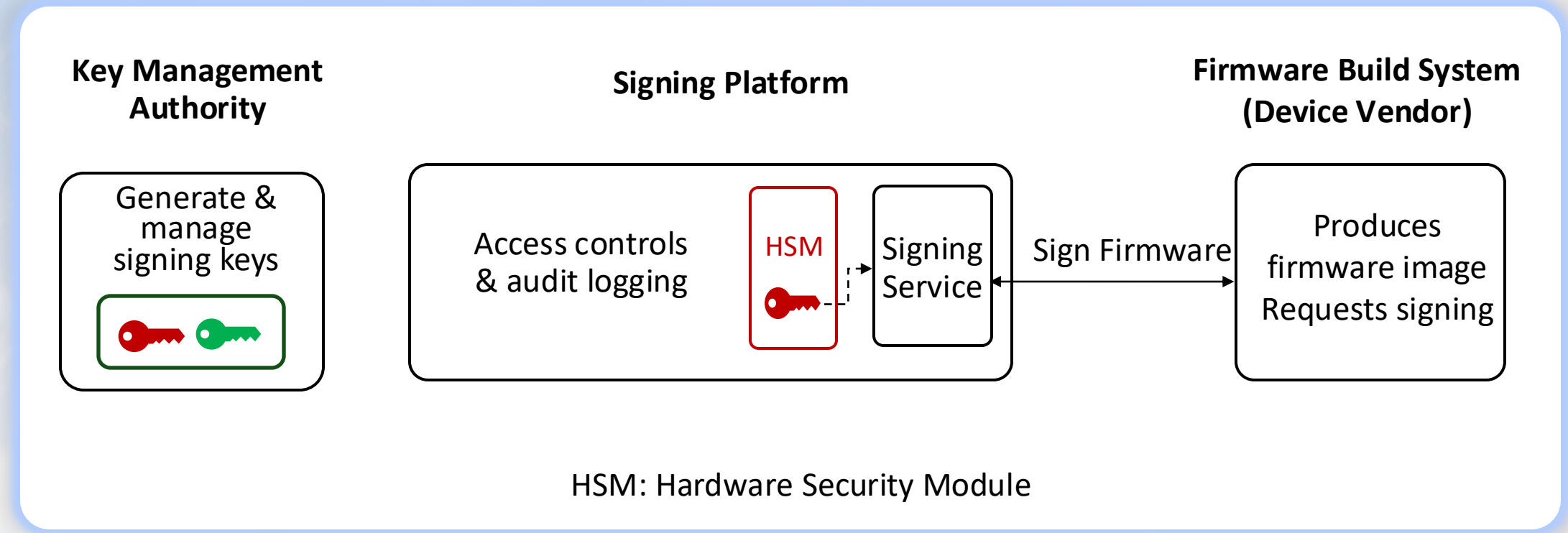
Typical ARM boot chain



Why Secure Bootloader Signing Matters Now

- Embedded devices are increasingly long-lived, connected, and remotely updateable
- Software integrity failures now translate into safety, liability, and regulatory risk
- The EU Cyber Resilience Act (CRA) raises expectations from best effort to demonstrable control, emphasizing **verifiable integrity**, secure updates, and lifecycle accountability
- CRA makes secure boot implementation auditable and enforceable
- Secure bootloader signing has become a foundational and mandatory system requirement

Key Management & Signing Infrastructure for Bootloader



Compliance-grade signing infrastructure entails significant engineering and operational effort

Secure Boot Signing Challenges: *DIY*

CRA-Compliant Key Protection

- **DIY**: Chip vendors' tools start with software-based key. Moving to **HSM** protected key requires cryptographic expertise & custom integration.

Operational Burden

- **DIY**: Ongoing key custody, access control, audits, recovery, and continuity — not a one-person's job and responsibility.

Scaling Complexity

- **DIY**: Secure signing processes don't scale easily across multiple products, factories, and development CI/CD pipelines.

Production Considerations

Security

- Hardware-protected private keys
- No ad-hoc or bypass signing paths

Operations

- Automation-friendly
- Role-based access control
- Full audit trail

Scalability

- Works across products, teams, and lifecycles

Signing platform side

- Centralized signing service
- Policy-driven authorization

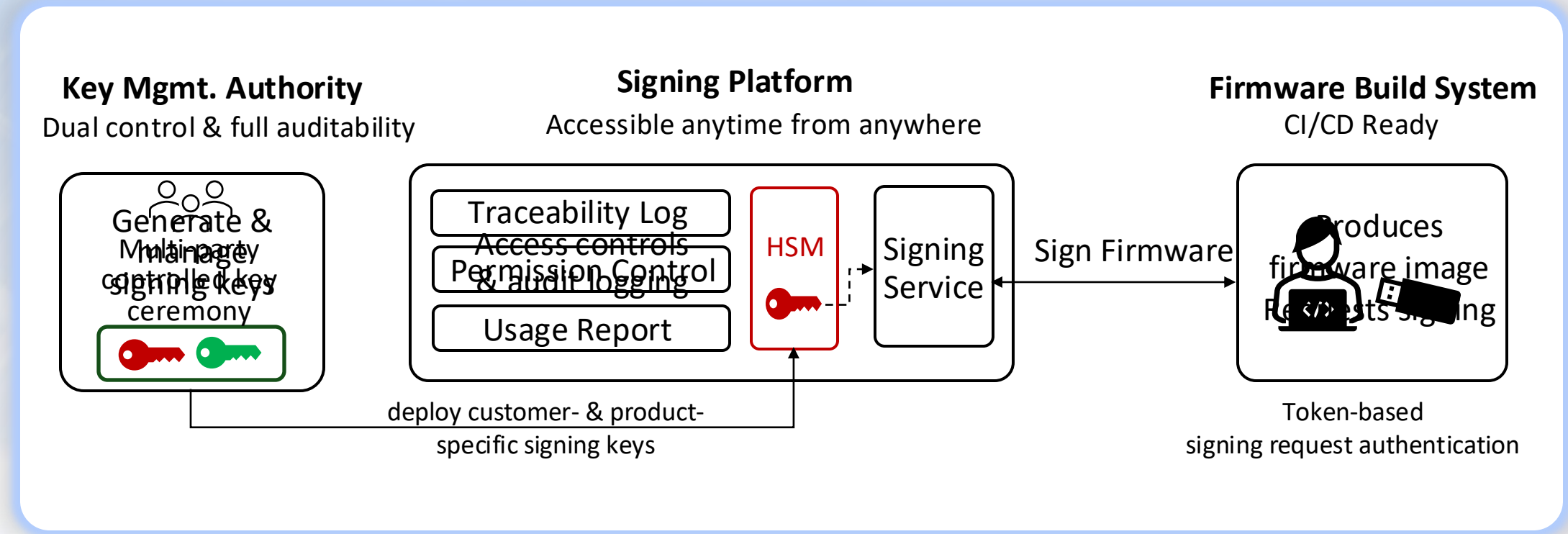
Build integration

- Build produces unsigned artifacts/code images
- Signing is a controlled, separate step
- Build → sign → release in an integrated CI/CD

Device side

- Verification enforced at every boot stage
- Unbroken chain of trust from ROM upward

Reference Architecture: End-to-End View



 |

A Case Study: Why a Real SoC Matters

ARM platforms differ in details and constraints

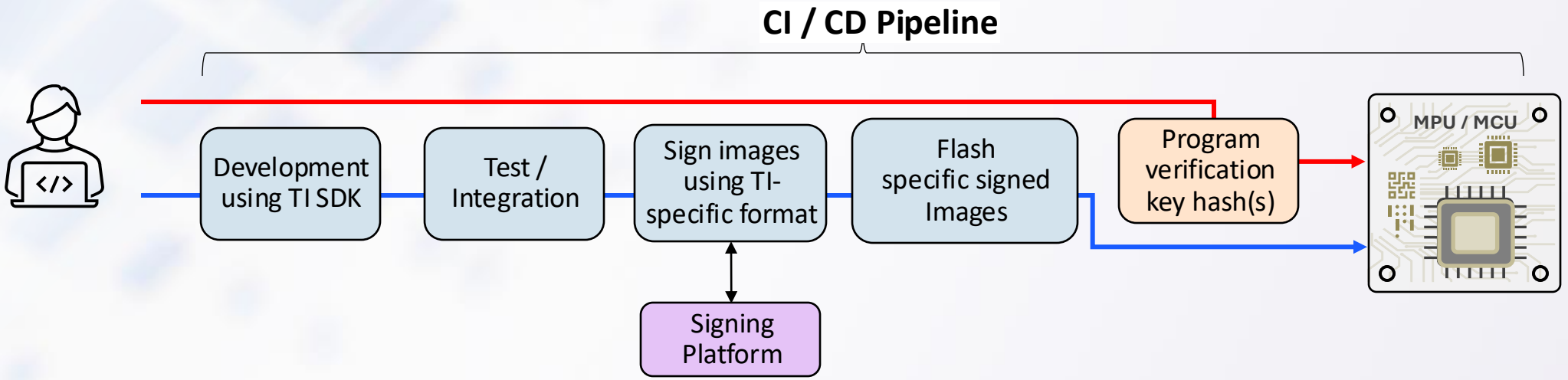
Only a real SoC exposes the practical friction in secure boot and signing workflows

Our reference platform: TI AM6x, a widely deployed ARM-based SoC family

- Supports multiple secure boot
- Representative of industrial and infrastructure devices

TI AM6x Bootloader Development & Signing Process

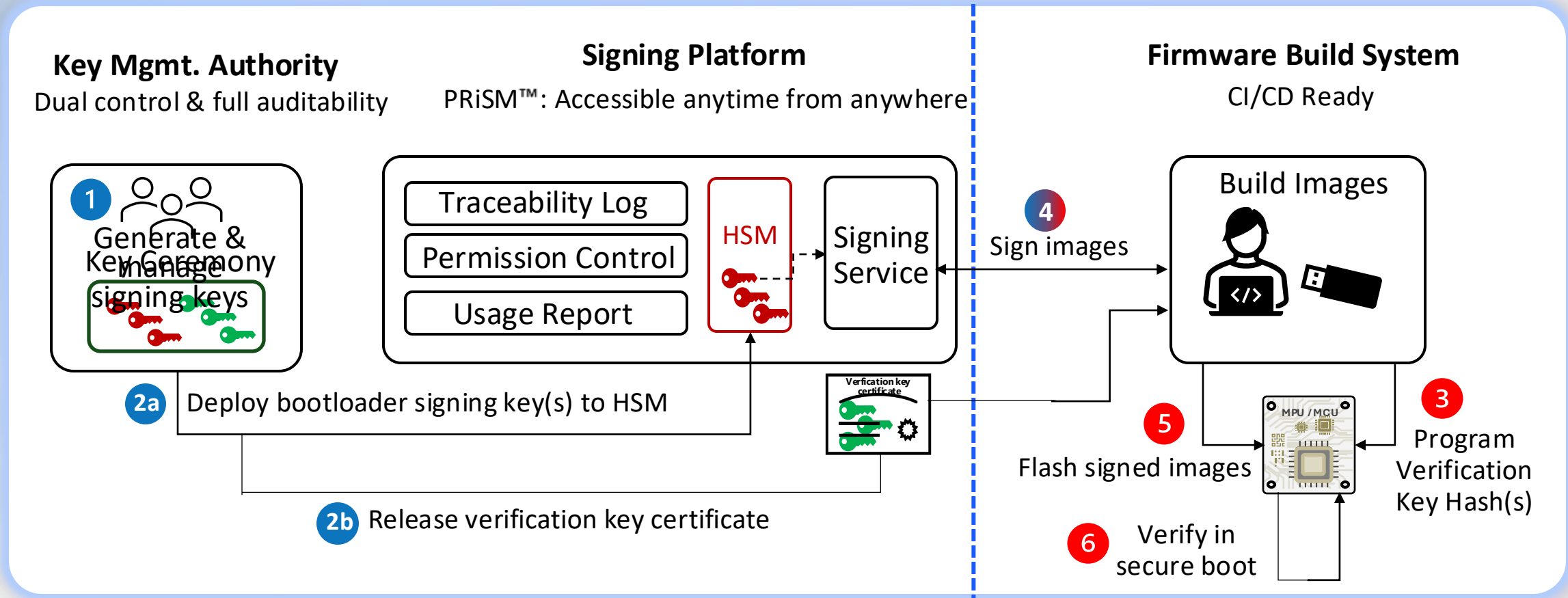
1. Secure boot enforced by ROM using a verification key hash(s) stored in device eFuse
2. Firmware images must be signed using TI-specific signing format
3. Multiple boot artifacts must be signed consistently
4. Hardware and boot firmware enforce signature verification at every stage



1 2a 2b 3 One-time steps

4 5 6 Repeatable steps

TI AM6x Bootloader Signing & Verification



PKI Center™ Aurora NETWORKS A Vistance Networks Business

Texas Instruments

PKI Center™ Aurora NETWORKS A Vistance Networks Business

Practical Lessons from Our Case Study

What we learned

- Complexity lives between tools, not inside them
- Manual “temporary” signing paths become permanent risks
- Clean separation of build and signing pays off quickly

General guidance

- Design for audits early
- Treat signing keys as shared infrastructure assets
- Make the correct path the easiest path

Key Takeaways

Secure-boot signing extends beyond cryptography into operational security and lifecycle management.

Key custody, access control, and auditability dominate long-term effort more than initial bring-up.

Regulatory alignment is easiest when signing workflows are designed with traceability and lifecycle control from day one.

THANK YOU!

Q&A

xin.qiu@auroranetworks.com

PKI-center.com