Building a Complete IoT Security Solution with a Hardware Root of Trust

Ben Levine
Senior Director, Product Management
Rambus Inc.
blevine@rambus.com

June 4th, 2019

Rambus
Data · Faster · Safer

IoT Attacks on the Rise

NEWS

Chinese firm recalls camera products linked to massive DDOS attack

Hangzhou Xiongmai Technology is recalling earlier models of four kinds of cameras due to a security vulnerability

Source: http://www.pcworld.com/article/3133962/chinese-firm-recalls-camera-products-linked-to-massive-ddos-attack.html

SF Muni Hack a Wake-Up Call for Public Systems

By Richard Adhikari
Nov 28, 2016 3:28 PM PT

Fare payment machines at underground stations were out of order, resulting in free rides on the subway and light rail system known locally as "SF Muni."

Source: http://www.technewsworld.com/story/84112.html

The FTC has sued D-Link over unsecure routers and webcams

Part of an ongoing effort to secure the Internet of Things

by Andrew Liptak | @AndrewLiptak | Jan 7, 2017, 1:00pm EST

Source: http://www.theverge.com/2017/1/7/14199232/ftc-sued-d-link-unsecure-routers-webcams-cybersecurity

Foscam Security Cameras Full of Security Flaws

by PAUL WAGENSEIL Jun 8, 2017, 8:48 AM

We've said it before, and we'll say it again: Don't buy cheap Chinese-made security cameras, because their security may just be terrible.

Source: http://www.tomsguide.com/us/foscam-camera-flaws,news-25254.html

Ukraine's Power Grid Gets Hacked Again, a Worrying Sign for Infrastructure Attacks

Russian hackers may be behind attacks leveled at the nation's power grid and artillery. The West should take note.

DHS Releases Strategic Principles For Securing The Internet Of Things

Release Date: November 15, 2016

Source: https://www.dhs.gov/securingtheIoTits

Finding of IoT Security Research Study

Analyzed IoT devices from manufacturers of:

- TVs
- Webcams
- Home thermostats
- Remote power outlets
- Sprinkler controllers
- Door locks
- Home alarms
- Garage door openers
- Hubs for controlling multiple devices

Source: IoT Research Study, HP

- **A majority** of devices included some form of cloud service.
- **70%** of the most commonly used IoT devices contain serious vulnerabilities
- **70%** of devices used unencrypted network service
- **25** vulnerabilities were found per device on average

IoT Device Threat Landscape

- IoT devices tend to be cost-sensitive and have limited power budgets
- Despite limited resources, IoT devices need strong security
- Compromise of IoT devices can have serious consequences

Privacy
Tap security cameras or baby monitor video stream

Repurpose
Device is repurposed to be used in a botnet

Sabotage
Reprogram an appliance firmware to cause a failure

Vandalism
Control an IoT device to cause a property damage

False Alarm
Initiate false alarm from smoke detector or security system

Physical Security
Disable the burglar alarm, unlock your front door

Trojan
Used to attack other devices in your home network

Infrastructure
Distributed attack to bring the power grid down

Security Evolves Throughout the Device Lifecycle

- IoT devices remain in the field for many years demanding flexible security:
  - Respond to and protect against new threats and new attack techniques
  - Adapt to new functionality and new use cases for existing devices
  - Meet the requirements of new security standards and protocols
  - Must be able to update software throughout the device lifecycle

- IoT chips may be leveraged across many market verticals each requiring different security capabilities:
  - Applications have different threat vectors and require different protection levels
  - Applications require different security protocols and cryptographic algorithms
  - Applications require different types of security protection at different stages in the lifecycle of the device

Software or Hardware-Based Security?

Software Security

- Inexpensive
- Anyone can write software
- Flexible – can perform any function
- Easy to update – adapt to new threats or add new functionality
- Insecure, easy for attackers to modify for malicious purposes

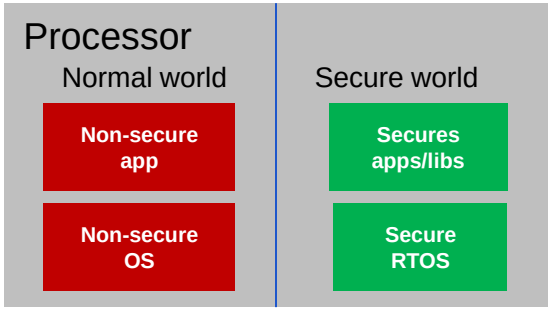
Trusted Execution Environments add additional security to software, but still rely on software for security.

Hardware Security

- More secure than software when designed correctly
- Harder to tamper with
- Can be expensive
- Harder to develop than software, requires specialized skills
- Can't be changed or updated in field

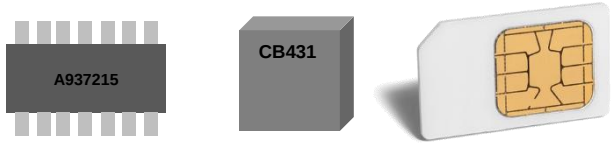
Fixed-function (state-machine) security hardware can be made very secure, but is inflexible.

Trusted Execution Environment



- A trusted execution environment typically consists of an execution mode in a processor that provides security to applications using a secure RTOS, memory/peripheral isolation, and application integrity checks.
- Benefits:
  - Readily available for standard processor IP
  - Integration is straightforward (although poor integration can compromise security)
  - Apps use standard development tools
- Drawbacks
  - No security isolation between secure apps – all can access same keys and secure assets
  - Communication between normal and secure worlds is complex and can expose weaknesses
  - Secure RTOS difficult to get right in practice and is foundation for TEE security
  - Large attack surface – bad software compromises security, minimal hardware protection
  - There are numerous known exploits of commercials TEEs and likely many more unknown ones

Discrete Security Chips



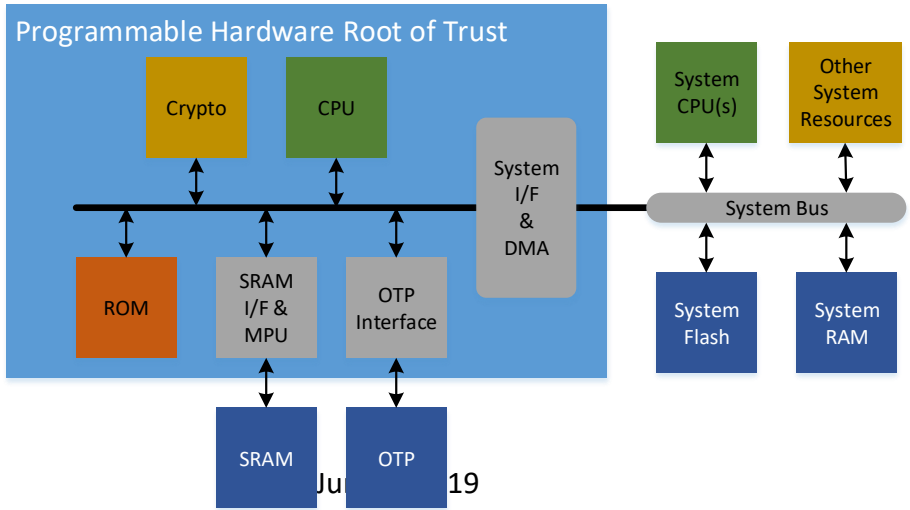
- A specialized security hardware component, such as a Secure Element (SE) or Trusted Platform Module (TPM), can be added to a device to provide security functions.
- Benefits:
  - Uses specialized hardware designed specifically for security applications
  - Can incorporate strong anti-tamper logic and other security functionality
  - Can be programmed for specific applications, although processing capabilities may be limited
- Drawbacks
  - Expensive
  - Requires separate part be added to device, requiring additional area on PCB or in package
  - Security functionality moved out of main SoC, providing greater attack surface and limiting overall security of system

Programmable Hardware Root of Trust

A programmable hardware root of trust (HRT) is an embedded security core that combines a processor with secure storage, cryptographic accelerators, and other security hardware. It can be integrated into any chip. It provides the security of hardware with the flexibility of software.

A good programmable hardware root of trust core should:

- Use a CPU specifically targeted for security applications
- Provide strong isolation of keys and resources between users of the core
- Protect against a wide range of attacks, including side-channel attacks like DPA
- Avoid unnecessary complexity and focus on strong security
- Allow easy development of new secure applications and protocols



Security Architecture Comparisons

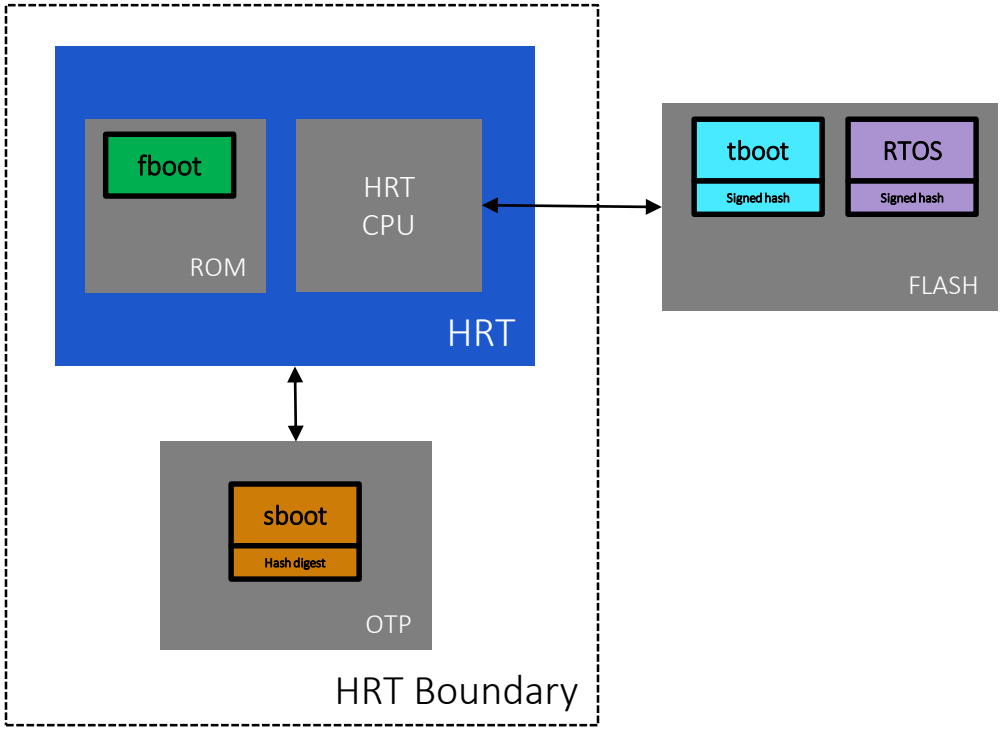
A programmable hardware root of trust provides very high security and a high degree of flexibility at a low cost

Security Options:	**Cost**	**Security**	**Flexibility**
Software only	Low	Low	High
Software in TEE (Trusted Execution Environment)	Low	Low/Medium	Medium
Application-specific security hardware	Medium	High	Low
TPM or Secure Element	High	High	Medium
Programmable Hardware Root of Trust	**Low**	**Very High**	**High**

Example: HRT Secure Boot

1. HRT CPU loads fboot (1st stage bootloader) from internal ROM. Fboot code is invariant and trusted.
2. Fboot then loads sboot (2nd stage bootloader) from OTP and verifies its integrity. If valid, sboot can run.
3. Sboot then loads tboot (3rd stage bootloader) from system flash and verifies its integrity. If valid, tboot can run.
4. Tboot then loads the RTOS from flash and verifies its integrity. If valid, the RTOS can run and normal operation can begin.

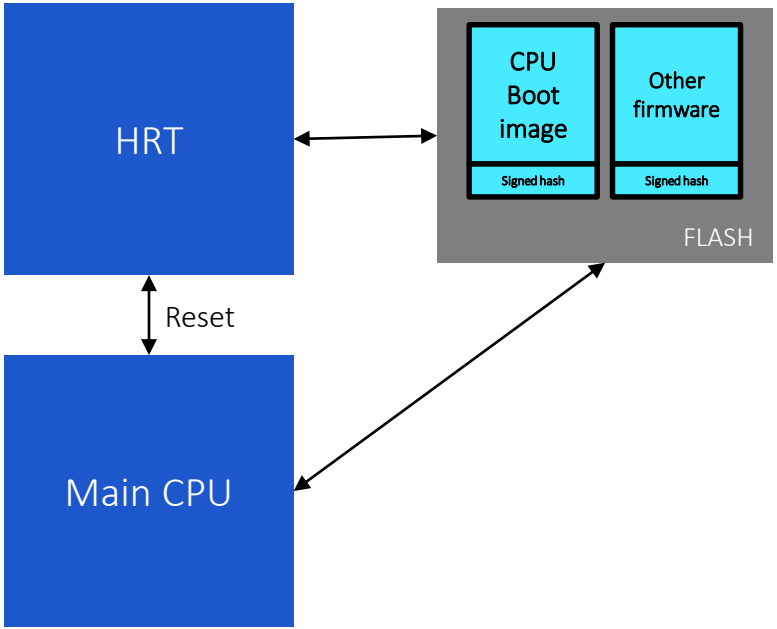
Each bootloader stage is responsible for ensuring the HRT is correctly initialized for the next stage.



Example: Main CPU Secure Boot

Once the HRT is securely booted, it can then ensure that the main CPU boots securely. One method of doing this:

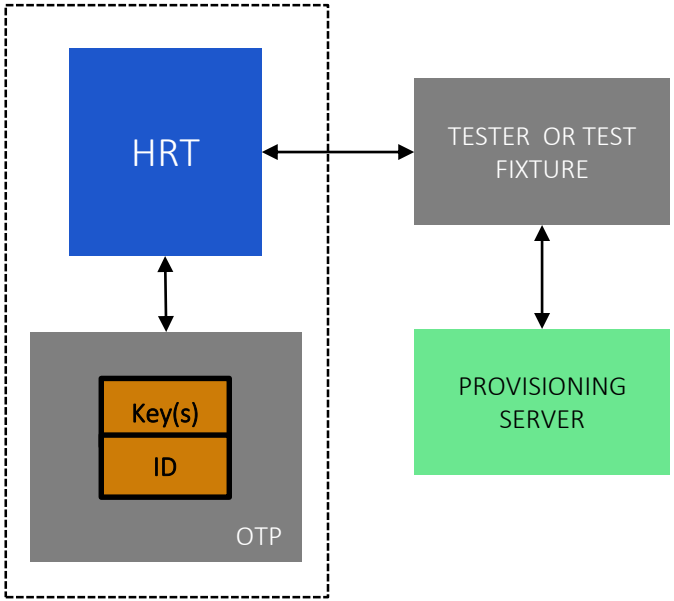
1. HRT holds Main CPU in reset after powerup
2. HRT reads CPU boot image from flash and verifies signed hash.
3. If image is valid, HRT releases CPU reset line and CPU boots using verified boot image
4. If image is invalid, HRT can direct main CPU to boot from “golden” default boot image in flash or OTP
5. CPU bootloader can then verify integrity of other firmware in the system



Example: Unique ID

Numerous options exists for provisioning unique IDs and key to secure devices.

- Provisioning can be done during chip or device testing. Usually HRT is connected to tester or test fixture via JTAG interface
- Pre-generated unique IDs can be provided by provisioning server or randomly generated by HRT. Unique ID (and possibly other IDs, such as OEM or product ID) are stored in HRT OTP and cannot be changed or tampered with.
- Secret leys are provisioned or generated by HRT. Secret keys are stored in HRT OTP and can only be accessed by HRT crypto blocks.

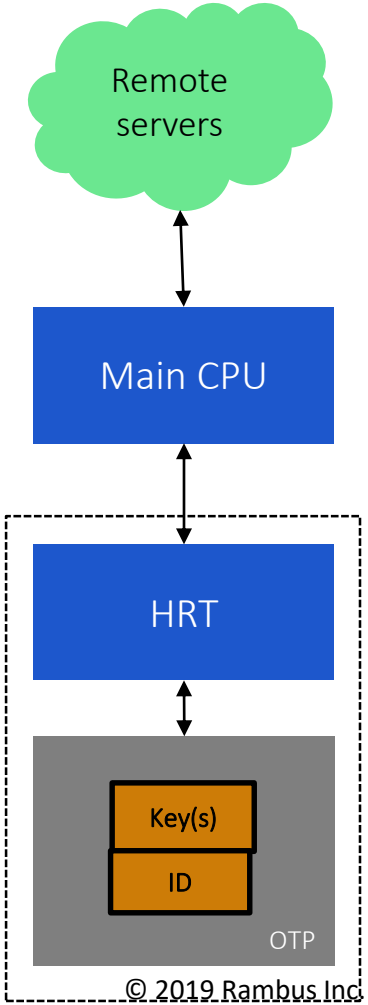


Example: Device Authentication

Once provisioned, the HRT can use device-specific keys and IDs to authenticate the device with remote servers. Strong authentication will prevent cloned or counterfeit devices from connecting to servers. The HRT can also be used to authenticate the server as well, typically using a server certificate. Server authentication will prevent devices from connecting with insecure servers.

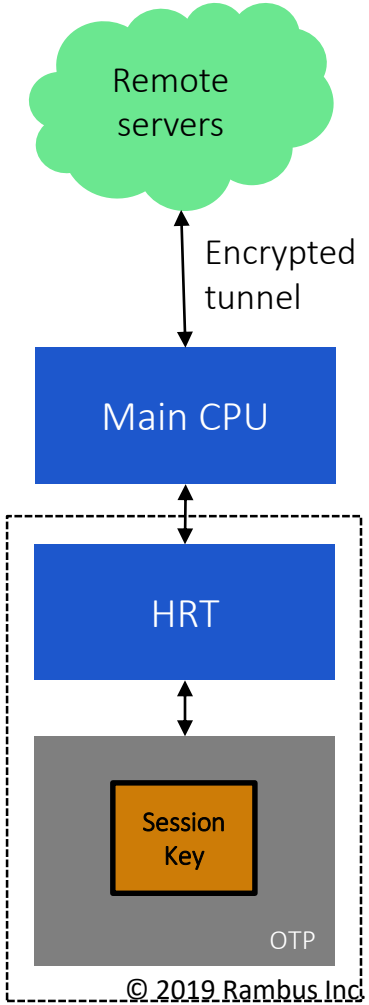
Device authentication example:

1. Main CPU contacts server with device ID retrieved by HRT
2. Server has database of device IDs and corresponding device-specific symmetric keys from provisioning infrastructure. Server encrypts random challenge and sends to device.
3. HRT decrypts challenge, perform predetermined function on challenge, encrypts result and sends response to server.
4. Server decrypts response and verifies correct result



Example: Secure Communication

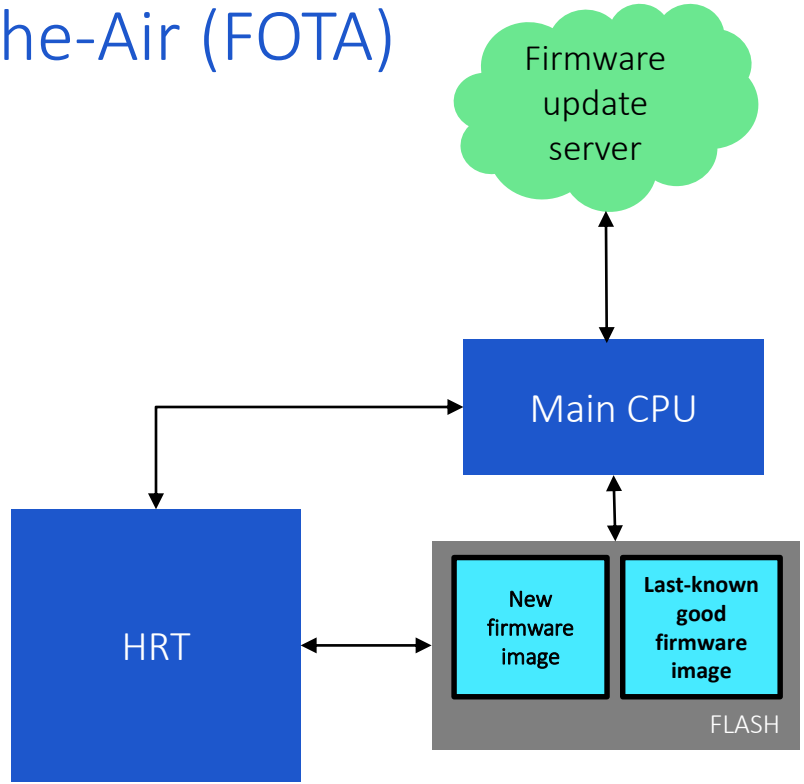
Once the device has been authenticated, the HRT can assist secure communication between the device and the server. After authentication, or during the authentication process, a shared one-time session key can be established between the server and the HRT. This key can be stored temporarily and securely in the HRT. The HRT can then encrypt data going to the server and decrypt data coming from the server. Standard protocols such as TLS can be supported.



Example: Secure Firmware Over-The-Air (FOTA)

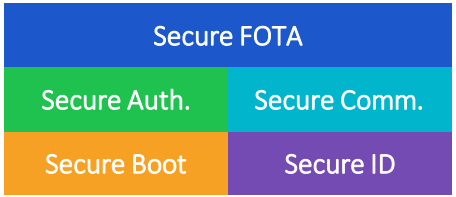
The HRT can work with the main CPU to ensure that firmware updates are legitimate and unaltered.

1. Main CPU checks with firmware update server periodically to see if there is a firmware update or firmware updates can be pushed.
2. HRT is used to authenticate communication with the server to ensure server is authentic and confirm with server that device is authentic.
3. HRT can optionally be used to encrypt and decrypt communication with the server
4. If there is an update available update, CPU will download and overwrite oldest firmware image in flash. Last known good firmware image will be kept. Downloaded image can optionally be encrypted and decrypted by HRT as it is being downloaded.
5. Once image is downloaded, HRT verifies its integrity and authenticity, usually using a certificate from the server. Once verified, CPU will use new firmware image.
6. If image is not verified, CPU will fall back to last known good firmware image
7. OTP bits controlled by HRT can be used to prevent rollback of the image.

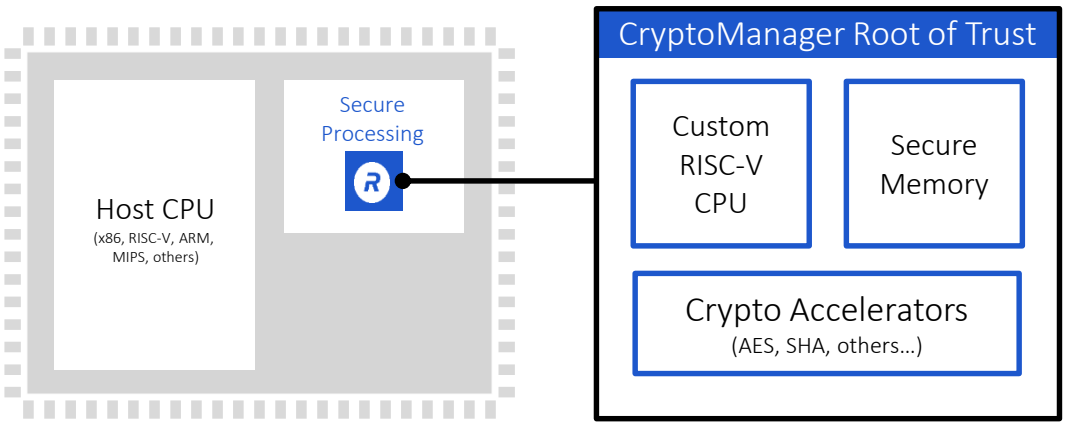


Using an HRT for IoT Security

- Attackers loading and running malicious code is a common attack on IoT devices.
- To prevent this, device needs multiple layers of security:
  - Foundation: Secure boot and secure ID
    - Ensure boot code has not been tampered with. Boot code can then verify other firmware
    - Secure ID: Device needs a secure, unique identifier to enable secure communication
  - Secure ID and secure boot enable secure authentication
  - Secure authentication enables secure communication:
    - Devices need to connect only with authorized servers and servers with real devices
  - Secure communications, secure authentication, secure ID, and secure boot enable secure firmware over-the-air updates (FOTA)



CryptoManager Root of Trust



Secure Functionality:

- Secure data storage
- Secure key storage
- Device personalization
- Key and data provisioning
- Authentication
- Attestation
- User data privacy
- Secure boot
- Secure firmware update
- Secure communication
- Runtime integrity checking
- Cryptographic acceleration
- Secure protocol implementation
- Secure debug
- Feature/configuration/SKU management

A secure Root of Trust that provides a foundation for security throughout a chip and system

IoT Security Presents Unique Challenges

- IoT devices need cost-efficient security which is strong and robust, as well as flexible and adaptable
- Conventional solutions are either too expensive, too rigid, or not secure enough
- New security solutions are needed to meet the security needs of IoT devices now and in the future.
- Programmable hardware root of trust cores provide a strong, flexible solution to a solve wide range of security problems and defend against sophisticated attackers.

Thank You!

CryptoManager Platform

https://www.rambus.com/security/cryptomanager-platform/

CryptoManager Root of Trust

https://www.rambus.com/security/cryptomanager-platform/root-of-trust/

CryptoManager Infrastructure

https://www.rambus.com/security/cryptomanager-platform/cryptomanager-infrastructure

Rambus

Data · Faster · Safer