Rambus

Protecting a Connected World with Secure Silicon IP

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

April 9th, 2019

Connected Device Threat Landscape

- Connecting devices opens a wide range of new attack vectors
- Compromise of connected devices can have serious consequences
- Connected devices need strong security

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 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 down the power grid

Complex System Components Create Vulnerabilities

SCIENTIFIC AMERICAN

Meltdown and Spectre Expose the Dark Side of Superfast Computers

Windows Central

Meltdown and Spectre exploits impact Intel, ARM, and AMD processors

The New York Times

Researchers Discover Two Major Flaws in the World's Computers

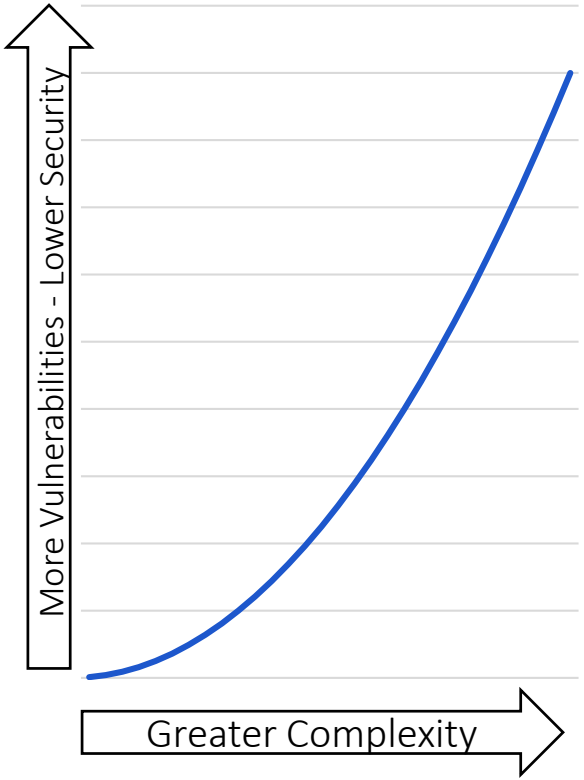
FORTUNE

Meltdown and Spectre Security Attacks Haunt Chip Industry

- Complex system components, like modern CPUs, are inherently weak against attackers
- The more lines of code or the more gates, the more possible security vulnerabilities
- Design optimization for characteristics such as performance without adequate consideration of security also increases vulnerabilities
- Security favors simpler, focused components that do less, but do important functions securely.

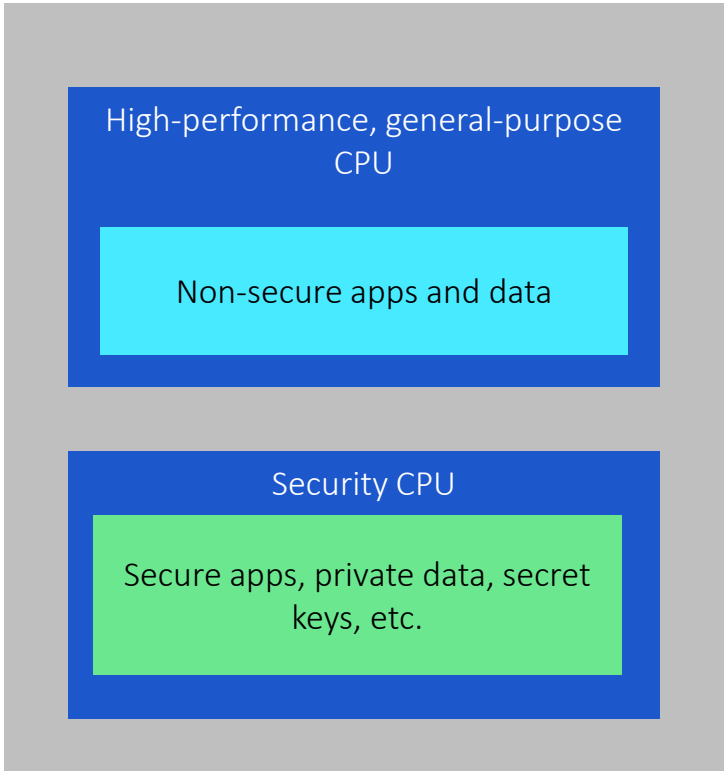
Conflict Between Security and Complexity

- Security flaws often result from different components interacting in unexpected ways.
- As connected devices become more complex, the number of components increases
- The number of interactions between two components goes up with the square of the number of components.
- As device complexity increases, the number of possible interactions goes up exponentially, meaning the number of security vulnerabilities goes up exponentially
- Even when security is considered to be important in complex systems (and it often isn't), finding and eliminating security vulnerabilities quickly becomes prohibitive
- Device makers need to eliminate order N^2 vulnerabilities; attackers only need to find one



Siloed Execution

- So how do we make complex systems secure? As seen, this is very hard.
- Let's do something different:
  - Separate operations that need to be secure from operations that need to be fast
  - Run them on two physically separate CPUs.
  - Keep the security CPU simple and optimize for security
  - Make the general purpose CPU as fast (and necessarily) as complex as needed
  - Keep keys and other security assets and functions in the security CPU so they are not compromised even if the general purpose CPU is hacked



Hardware Security Cores

- In addition to a secure CPU, other hardware is usually needed to handle keys and data that must be protected, and to provide a complete security solution. This includes:
  - Cryptographic accelerators
  - Private memory
  - Non-volatile key storage
  - Secure test and debug interfaces
  - Anti-tamper and other security logic
  - Random number generator
- An assembly of a secure CPU and these related components is often referred to as a Hardware Security Core (HSC)
- HSCs can run a wide range of security applications

HSC Use Cases

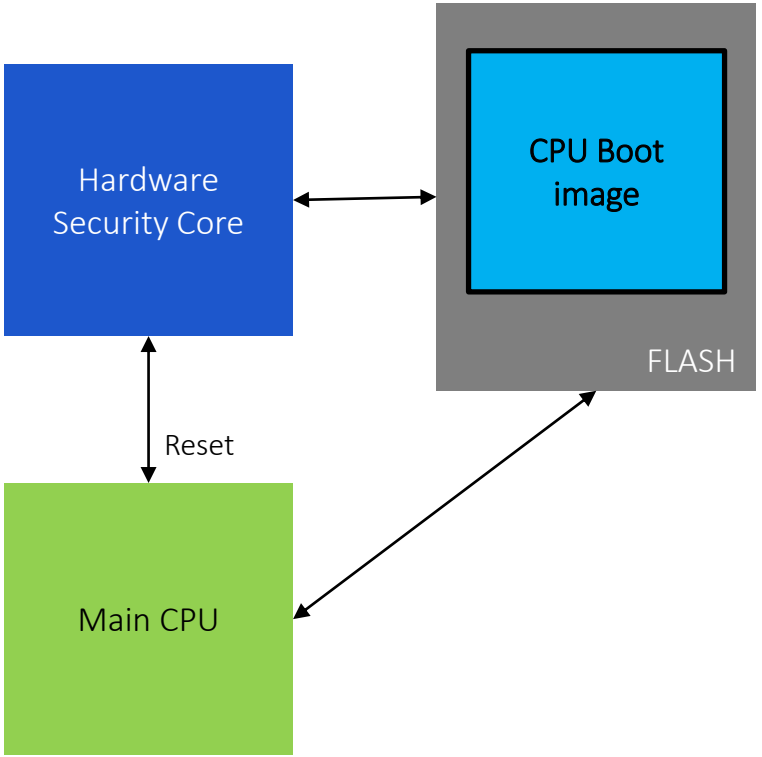
Integrating a hardware security core enables a wide range of security use cases:

- Secure booting of system SW
- Runtime integrity of system SW
- Secure system monitor
- Secure firmware updates
- Device personalization
- Key and data provisioning
- User data privacy (Unique Device Base Key)
- Secure data storage
- Secure key storage
- Authentication (Local and Remote)
- Attestation (SW & HW states, SW update confirmation)
- Secure communication (TLS, MKA/MACsec, etc)
- Cryptographic algorithm acceleration (AES, SHA, RSA, etc.)
- Secure debug / RMA
- Feature/Configuration/SKU management (Ex: Enable Features in Field)

HSC Application: Main CPU Secure Boot

The HSC boots itself securely and can then ensure that the main CPU boots securely.

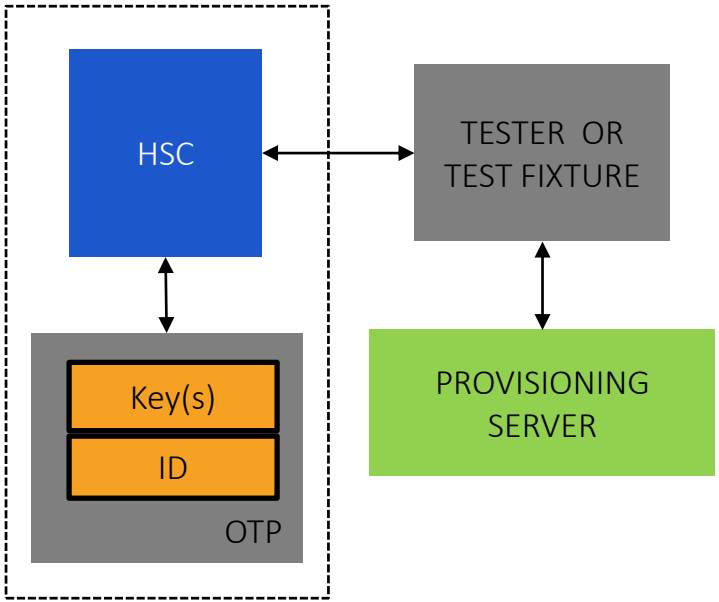
1. HSC holds Main CPU in reset after powerup
2. HSC reads CPU boot image from flash and verifies signed hash.
3. If image is valid, HSC releases CPU reset line and CPU boots using verified boot image
4. If image is invalid, HSC can direct main CPU to boot from “golden” default boot image in flash or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



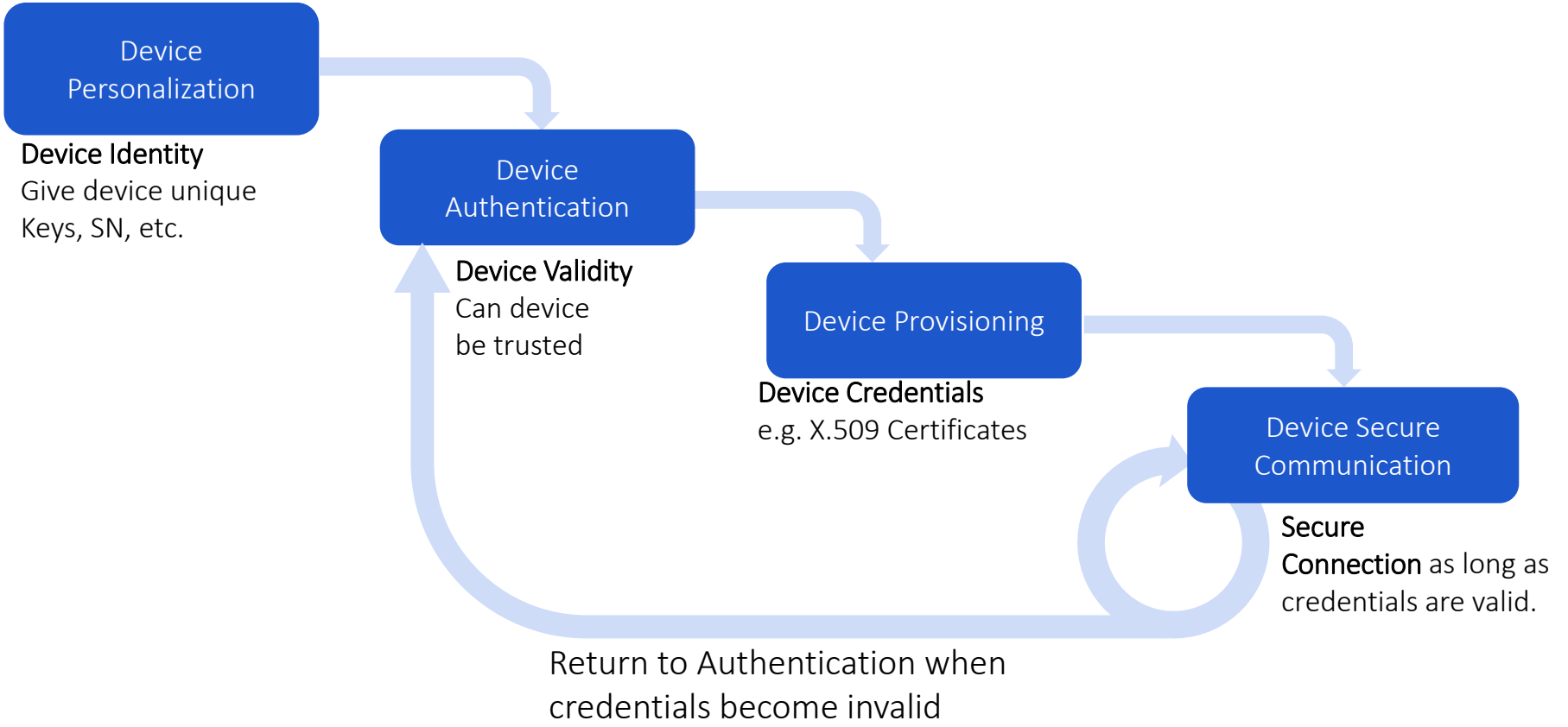
HSC Application: Secure ID

Numerous options exists for provisioning unique IDs and keys to secure devices using an HSC: **Rambus offers not only an HSC, but also a complete provisioning infrastructure solution supporting multiple options**

1. Provisioning can be done during chip or device testing.
2. Unique ID(s) are stored in HSC OTP and cannot be changed or tampered with.
3. Unique symmetric keys can be provisioned by the server or generated by HSC.
4. A unique asymmetric key pair can be generated by HSC. Secret keys are stored in HSC OTP and can only be accessed by HSC crypto blocks.

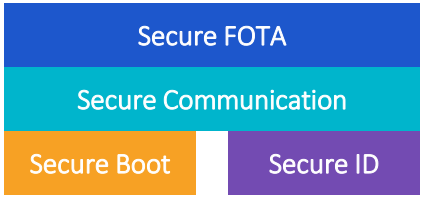


HSC Application: Secure Communication

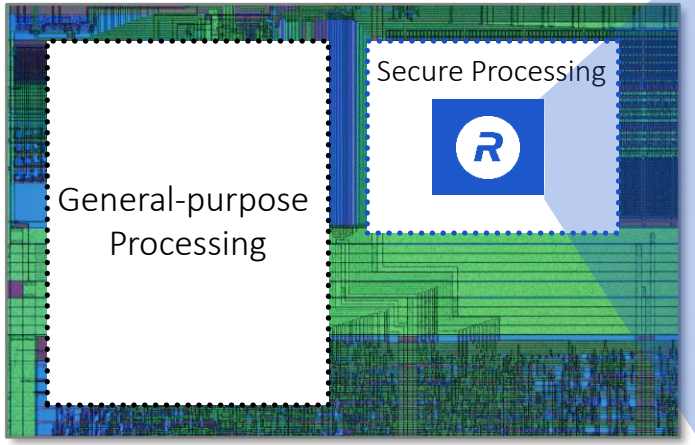


Using an HSC for Connected Device Security

- 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 communication:
    - Devices need to connect only with authorized servers and servers with real devices
  - Secure communications, secure ID, and secure boot enable secure firmware over-the-air updates (FOTA)



CryptoManager Root of Trust (CMRT)



CryptoManager Root of Trust

Custom RISC-V CPU

Secure Memory

Secure Functionality:

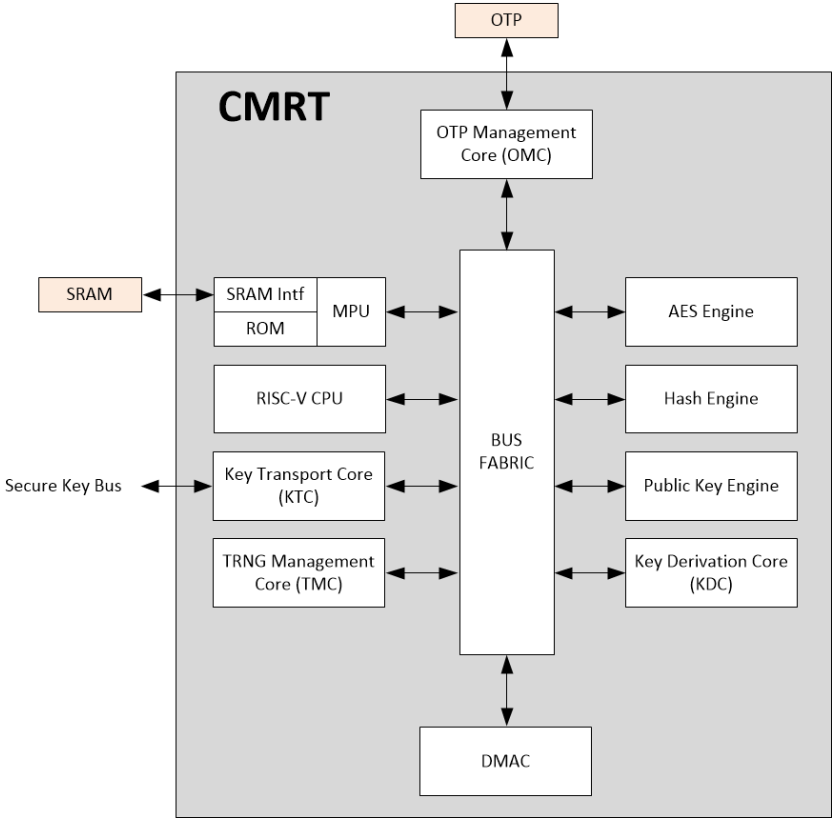
- Secure boot
- Secure IDs
- Secure communication
- Runtime integrity
- Etc.

Crypto Accelerators
(AES, SHA, others…)

CMRT RT630

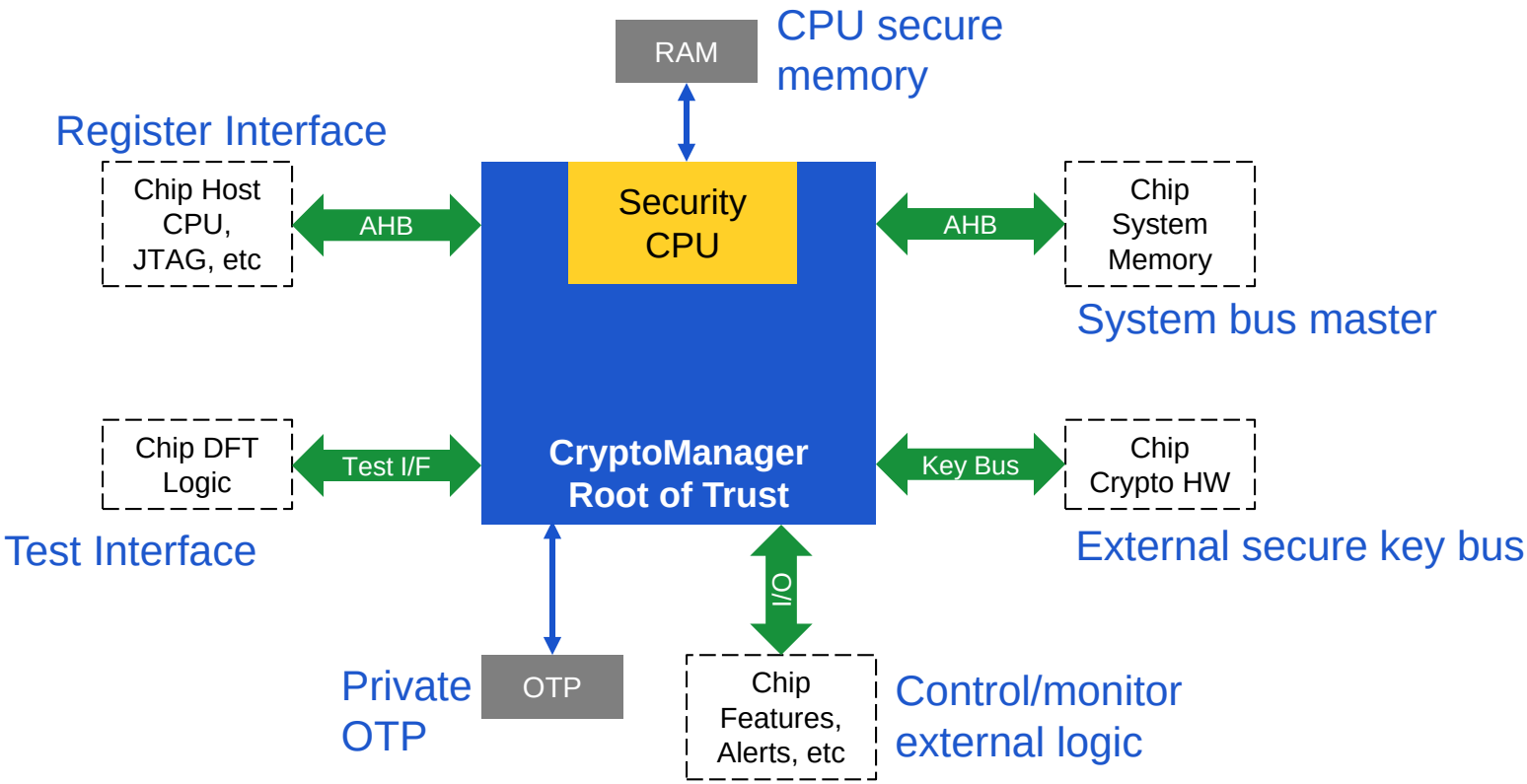
Processor based CryptoManager hardware root of trust soft IP core delivered as Verilog RTL for inclusion in any chip or FPGA design. Currently shipping. Provides:

- Flexible and extensible functionality
- Modular architecture; can add wide range of crypto accelerators and other functions
- Layered security model provides security of hardware with flexibility of software
- Ability to execute complex algorithms and protocols in within the security perimeter
- Strong hardware-enforced security, including anti-tamper features and other secure logic



CMRT diagram is simplified

CMRT System Interfaces



Summary and Conclusions

- Connected devices need strong security.
- As devices become more complex, security vulnerabilities increase
- One way to reduce security vulnerabilities is to move functions requiring security into simple but secure processing domains
- Hardware security cores combine secure CPUs with supporting logic and storage to provide a security domain for secure processing.
- Hardware security cores can perform a wide range of security applications needed for a complete security solution.

Thank You!

For More Information:

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/

Contact Information:

Ben Levine
Senior Director, Product Marketing
blevine@rambus.com

Steve Singer
Worldwide Director of Field Applications
ssinger@rambus.com

Tim Smith
VP, Sales
tsmith@rambus.com

Rambus
Data · Faster · Safer