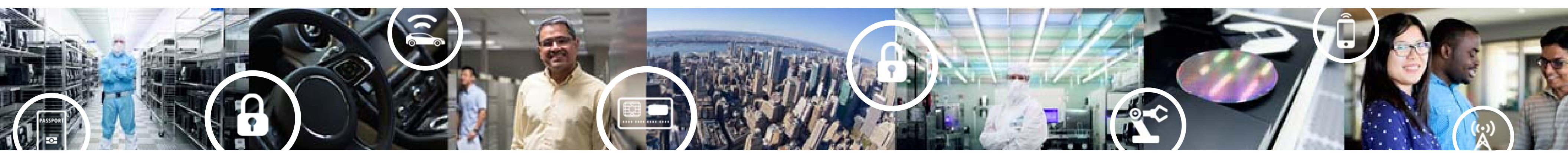
ADVANCE YOUR IoT SECURITY LEVERAGING HARDWARE PROTECTED KEYS

DONNIE GARCIA
NXP IoT SECURITY SOLUTIONS
JUNE 2019



PUBLIC

SECURE CONNECTIONS FOR A SMARTER WORLD

Hardware Protected Keys Webinar Series

This webinar meets 3 times.

Tue, Apr 16, 2019 10:00 AM - 11:00 AM CDT
Tue, May 21, 2019 10:00 AM - 11:00 AM CDT
Tue, Jun 18, 2019 10:00 AM - 11:00 AM CDT

Part 1: Utilizing hardware protected keys on broad market Microcontrollers Recording

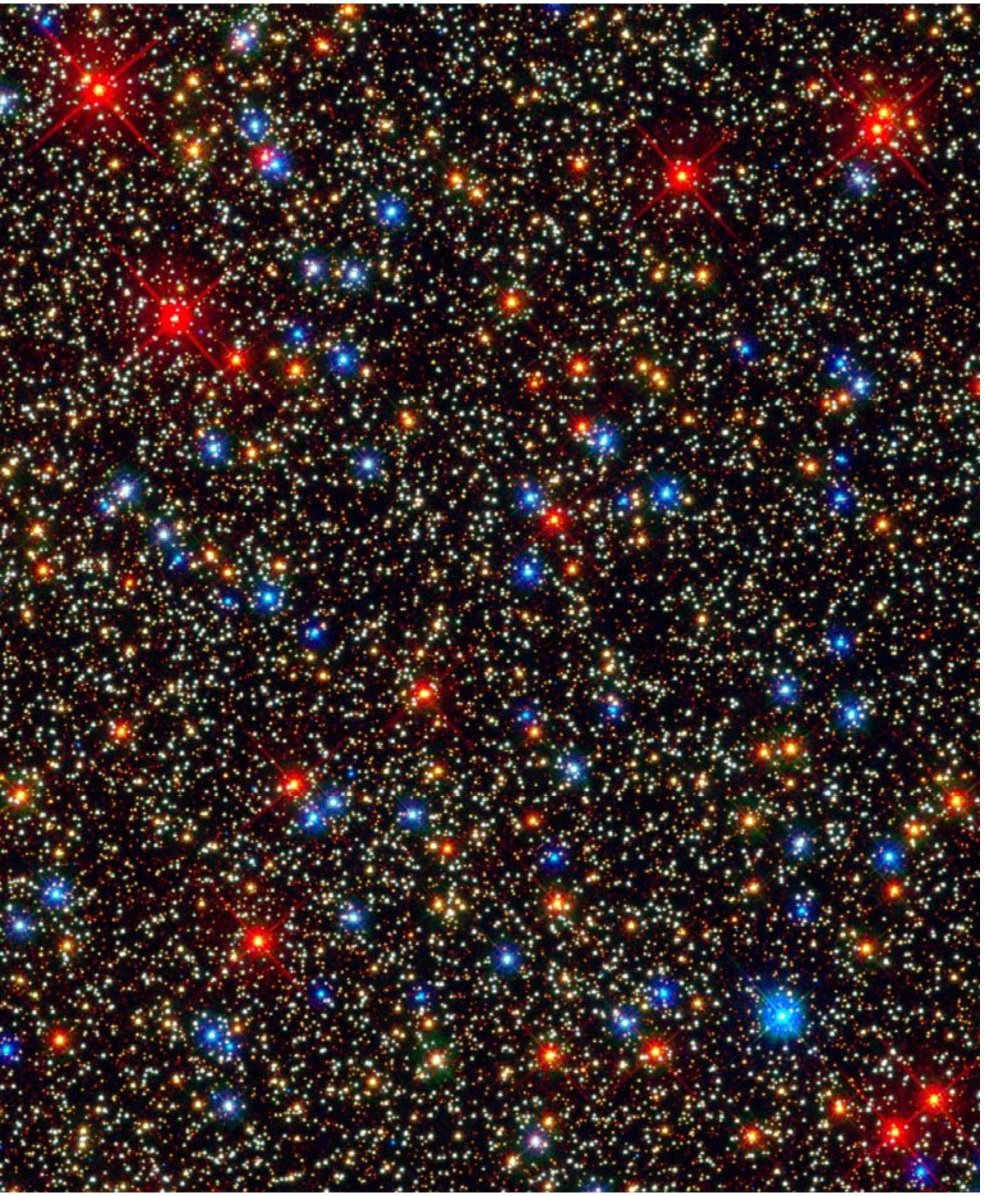
For the IoT Edge device, the cryptographic keys used to perform the services such as encrypted boot, onboarding, and over the air updates are critical components that must be protected. Chip level hardware protected keys are the standard for achieving strong security protection for embedded designs. This session will define what a hardware protected key is and show several examples of how these keys are realized on NXP processors. The i.MX RT 1050 family of devices will be used as a real world example of how Intrinsic ID Broadkey® SRAM based PUF can advance your IoT Security.

Part 2: Using hardware protected keys on state of the art Microcontrollers Recording

For the latest microcontrollers addressing IoT applications, hardware protected keys address critical security functions to protect application integrity, software confidentiality and encrypt data at rest. This session will explore the ability of the recently launched NXP IoT microcontroller, LPC5500 series. This family of devices will work as the main processing unit for a broad range of IoT applications and integrates breakthrough capabilities with regards to security. Along with Arm TrustZone technology the SRAM PUF based key management makes security easy to use and easy to deploy.

Part 3: Advanced IoT application key management based on hardware protected keys

The recently launched NXP IoT microcontroller, LPC5500 series, works as the main processing unit for a broad range of IoT applications. Along with Arm TrustZone® technology the chip supports SRAM PUF based key management. The product includes a software development kit (MCUXpresso SDK) that contains prebuilt applications to demonstrate edge to cloud connections out of the box. With the integrated security technology and software enablement, the LPC5500 makes security easy to use and easy to deploy. Join this session for a quick run through the demo applications available to kickstart your next IoT designs.Less



Agenda

- Quick recap and highlights
- Example IoT Device and Enablement
- Key Use Exploration
  - Secure boot
  - Software IP protection
  - Encrypted Execution
  - Device Data Confidentiality
  - Secure Connections
  - Cloud based OTA
  - Authenticated Debug
- Key Management Table Summary
- Conclusions

QUICK RECAP & HIGHLIGHTS

System Level Security Goals Depend on Cryptography

System Level Requirements

Establish a trusted supply chain	★ Protect symmetric and private keys with hardware	Actively monitor physical and logical conditions for intrusion	Update software when vulnerabilities are discovered	Monitor and report unauthorized Wireless networks
★ Protect system integrity with an Authenticated Boot	Only use validated RNG and Cryptographic Accelerators	Enforce the principle of least privilege for users and processes	★ Use cryptography to protect connections and data at rest	Limit sensitive functions and services (ie wireless network connections)
★ Encrypt sensitive software functions to prevent reverse engineering	Protect manufacturing with key injection from a trusted source	Support fall back images to sustain minimal functionality	Maintain audit logs that are read only	Support decommissioning to remove all sensitive data

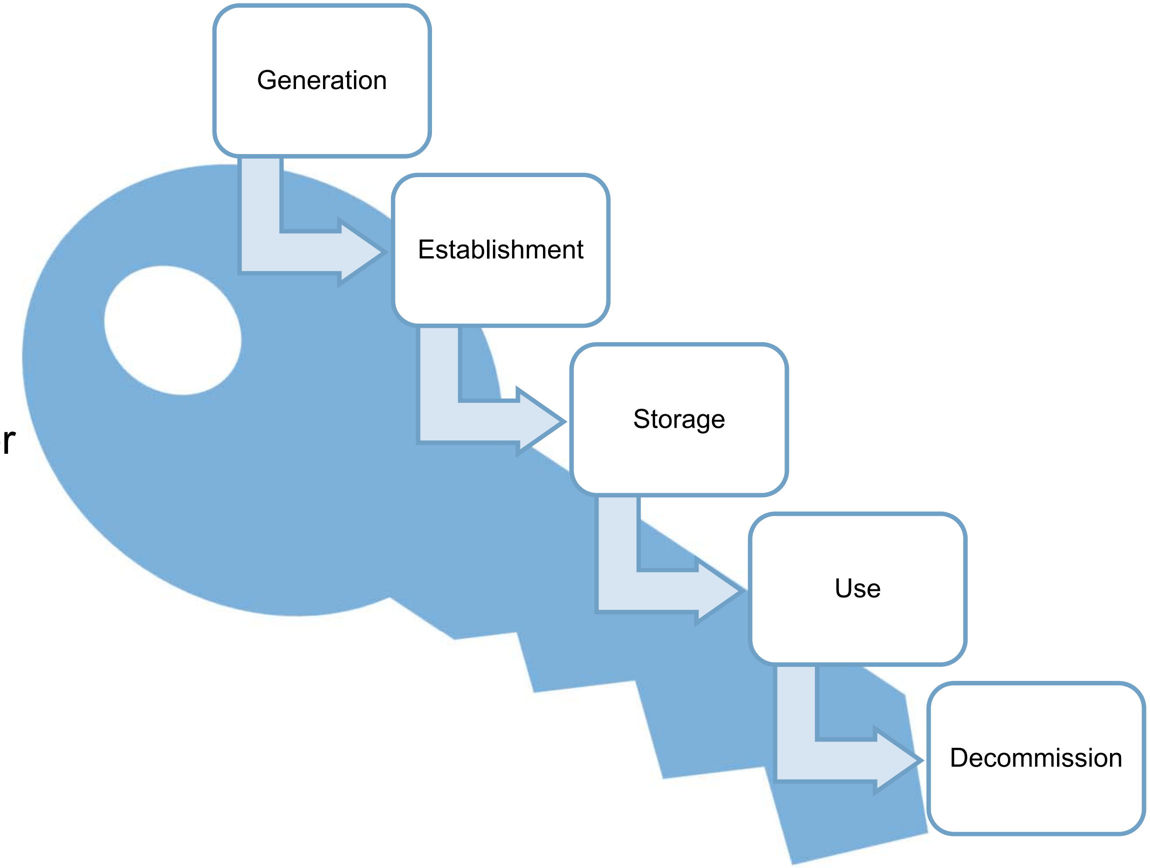
- **Cryptography is a fundamental capability needed to address edge device security**
  - Basis for protecting data at rest and in transit
  - Provides robust identity for the end device by cryptographic authentication
- **The key material used for cryptographic operations must be protected by hardware**
  - Attacks against Confidentiality/Integrity/Authenticity are aimed at attaining the Cryptographic Key

★ *Requirements which depend on Cryptography*

PROTECTED

Protected over the lifecycle* of the Cryptographic keys

- Key Lifecycle
  - Generation
    - Who/what creates the key material
  - Establishment
    - How the key material is shared or signed between entities
  - Storage
    - Where the key material is placed for future access
  - Use
    - How the key is utilized during the cryptographic processing
  - Decommission
    - Revocation and destruction of key material



*Key Lifecycle https://community.nxp.com/docs/DOC-333095

SRAM PUF Overview

Leverages the intrinsic entropy of the silicon manufacturing process

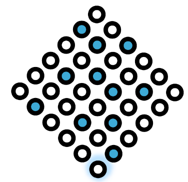
Device unique, unclonable fingerprint derived on every activation of the PUF

PUF master key is used to protect other secrets

1 

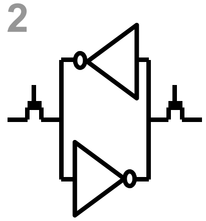
Process Variation

Naturally occurring **variations** in the attributes of transistors when chips are fabricated (length, width, thickness)



3 Silicon Fingerprint

The start-up values create a **random** and repeatable pattern that is unique to each chip

2 

SRAM Start-up Values

Each time an **SRAM block** powers on the cells come up as either a 1 or a 0

4 

SRAM PUF Key

The silicon fingerprint is turned into a **secret key** that builds the foundation of a security subsystem

Exploring Protected Key Options

1 External Security IC

NXP IoT Security ICs:
- A71CH
- A100x Authenticator
- SE050

- **Strongest protection across all key life stages**
- **Uses:**
  - Device identity and establishing TLS/onboarding
  - NXP Trust provisioning reduces overhead for key generation and establishment

Uses may overlap

2 Security with OTP Keys

Security Hardening on MCU/MPU

- **Provides runtime application security**
- **Uses:**
  - Secure boot
  - Bulk data protection
  - Enforces security policies (Roles)
  - Firmware updates

3 Software SRAM PUF

Security Hardening on MCU/MPU with Software PUF (Intrinsic ID BroadKey)

- **Assist with early key life stages and improves protection for keys**
- **Uses:**
  - Key Generation and establishment
  - Device identity
  - Assist with TLS/onboarding

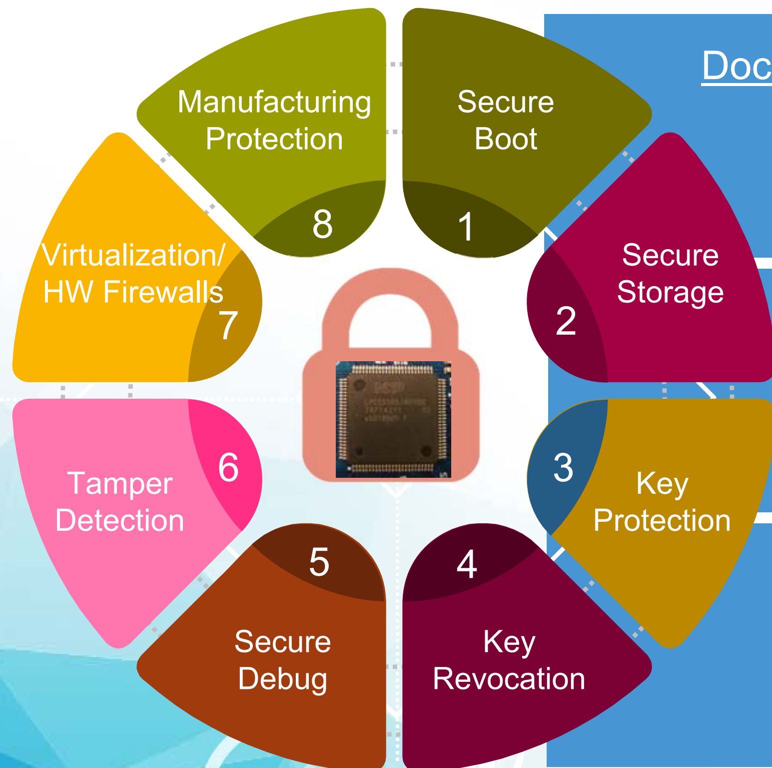
Uses Incremental

4 Security w/SRAM PUF

Hardware PUF (Intrinsic ID QuiddiKey): LPC5500 Family

- **Links advantages of PUF to runtime application security**
- **Uses:**
  - PUF protected keys used for secure boot, etc.
  - PUF for Key generation and establishment protects early life stages

NXP Security Technology



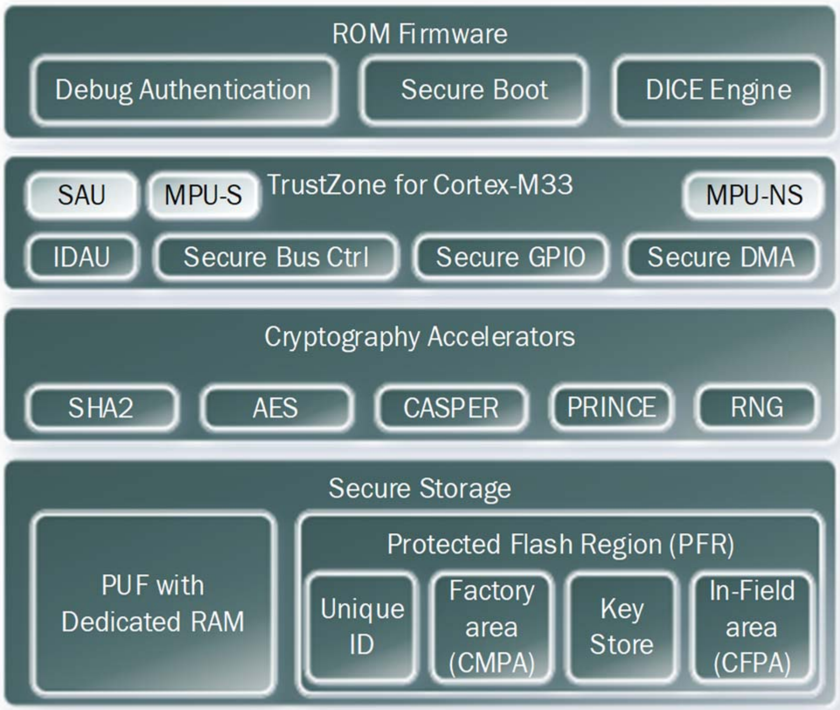
Documentation & Certifications
Security Users Manual
App. Notes
Community

Software &Tools
MCUXpresso
Code Signing Tools
Manufacturing Tools
Serial Download Tools

Hardware
Casper, Prince
MPC, PPC
PUF

SECURITY SUBSYSTEM OVERVIEW

- **ROM supporting**
  - Secure Boot, Debug Authentication & DICE Engine
- **TrustZone for Cortex-M33**
  - Arm's Security Attribution Unit (SAU)
  - Arm's Memory Protection Unit (MPU): Secure & Non-Secure
  - NXP's (implementation) Defined Attribution Unit (using IDAU interface)
  - NXP's Secure Bus, Secure GPIO & Secure DMA Controllers
- **Cryptography Accelerators**
  - Symmetric (AES-256) & Hashing (SHA2) engine
  - On-the-fly flash encryption/decryption engine (PRINCE)
  - Asymmetric engine for RSA and ECC (CASPER)
  - Random Number Generator (RNG)
- **Secure Storage**
  - Physically Unclonable Function (PUF)
    - Device unique root key (256 bit strength), 64-4096 bit key size
  - Protected Flash Region
    - RFC4122 compliant 128-bit UUID per device
    - Customer Manufacturing Programable Area (Boot Configuration, RoT key table hash, Debug configuration, Prince configuration)
      - PUF Key Store (Activation code, Prince region key codes, FW update key encryption key, Unique Device Secret)
    - Customer Field Programable Area (Monotonic counter, Prince IV codes)



LPC5500 EXAMPLE IOT DEVICE

NXP LPC5500 MCU SERIES

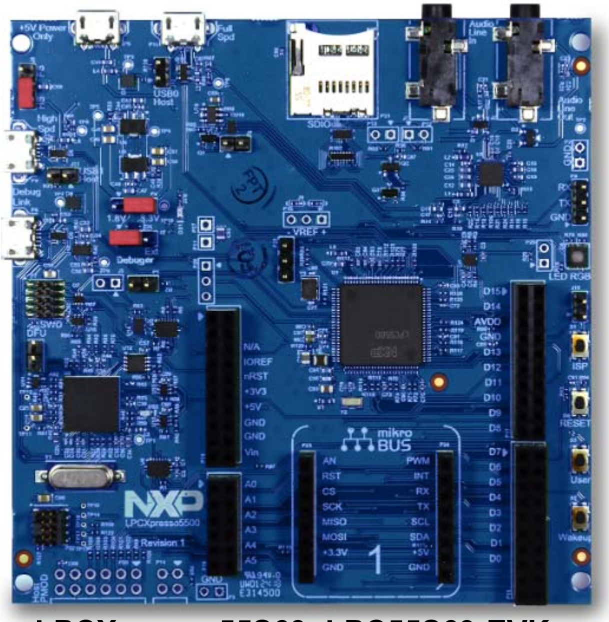
MCUXPRESSO SOFTWARE & TOOLS ECOSYSTEM

Complimentary with Extensive Support

- MCUXpresso SDK
- MCUXpresso IDE
- MCUXpresso Config Tools

Hardware Platform for Ease of Development

- On-board debug circuit
- PCB Layout, Schematic and Board Files Available



LPCXpresso55S69: LPC55S69-EVK

ARM KEIL Microcontroller Tools

IAR SYSTEMS

SEGGER

Simplify secure embedded development; Reduce time to market.

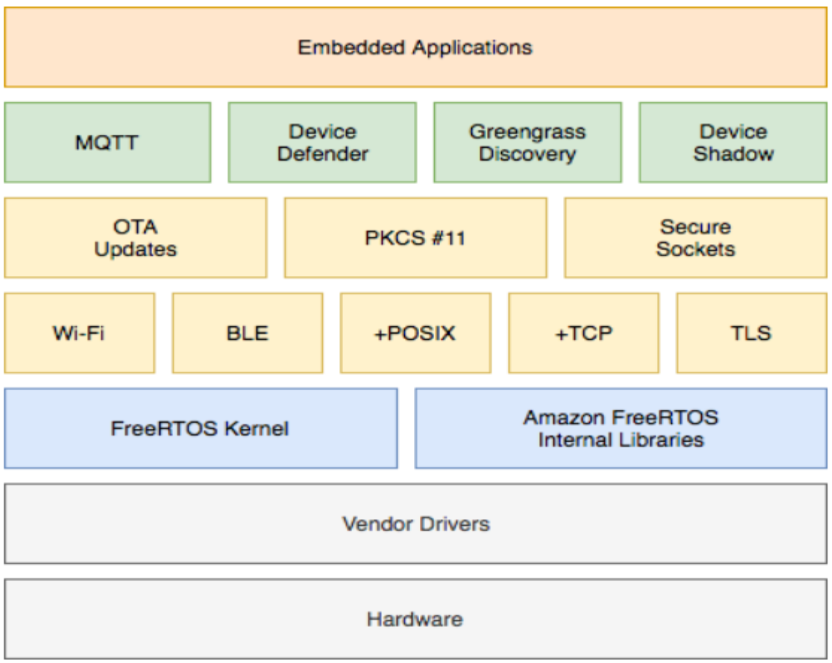
LPC5500 MCU Series

IDE SDK CFG

NXP

Amazon FreeRTOS at the Device

aws The FreeRTOS kernel is now an AWS open source project, and these pages are being updated accordingly. AWS are pleased to announce immediate availability of the MIT licensed Amazon FreeRTOS operating system, built on the FreeRTOS kernel v10.

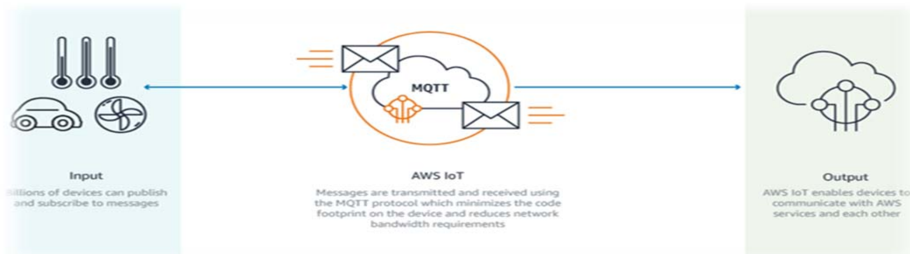


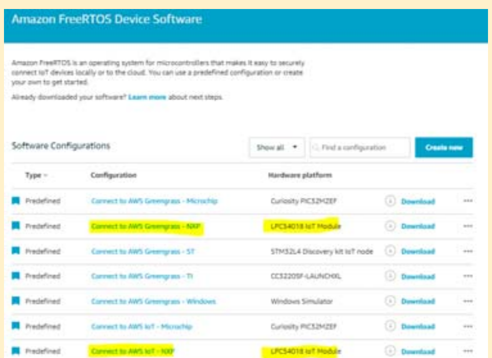
i.MX RT Series
High Performance
Crossover processors with real-time functionality and MCU usability for next generation consumer and industrial IoT applications.

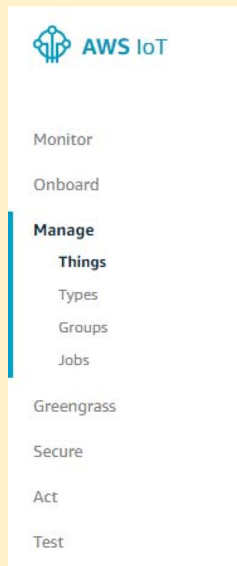
LPC54000 Series
Power-Efficient
A power-efficient, mainstr series for everyone.

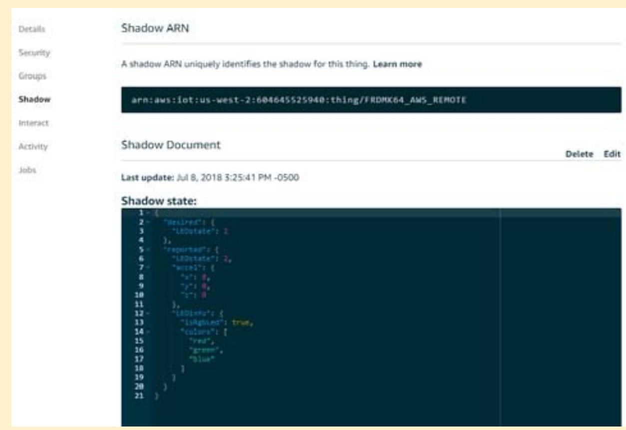
Kinetis K Series
Performance
190+ high performance MCUs with up to 2MB of embedded Flash and 1MB SRAM, advanced security and connectivity such as Ethernet, USB and CAN.

AWS IoT Device and Cloud Views

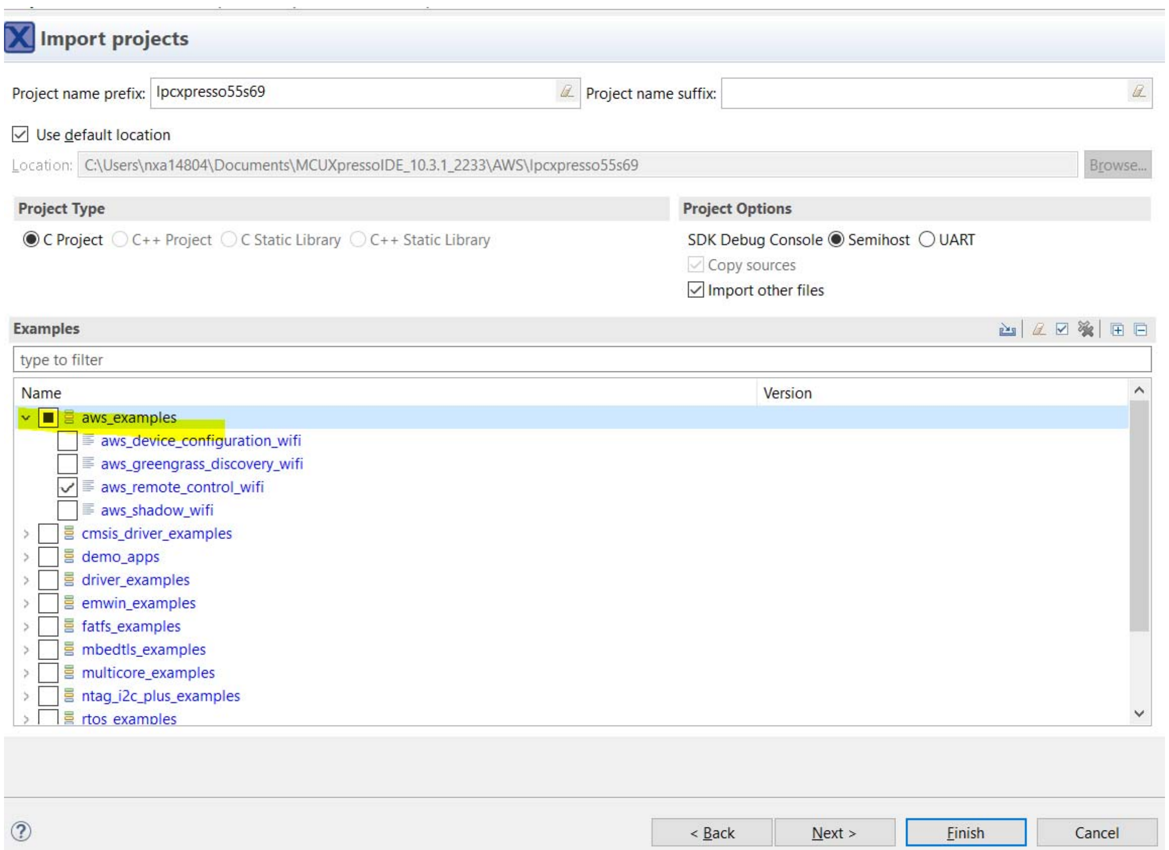






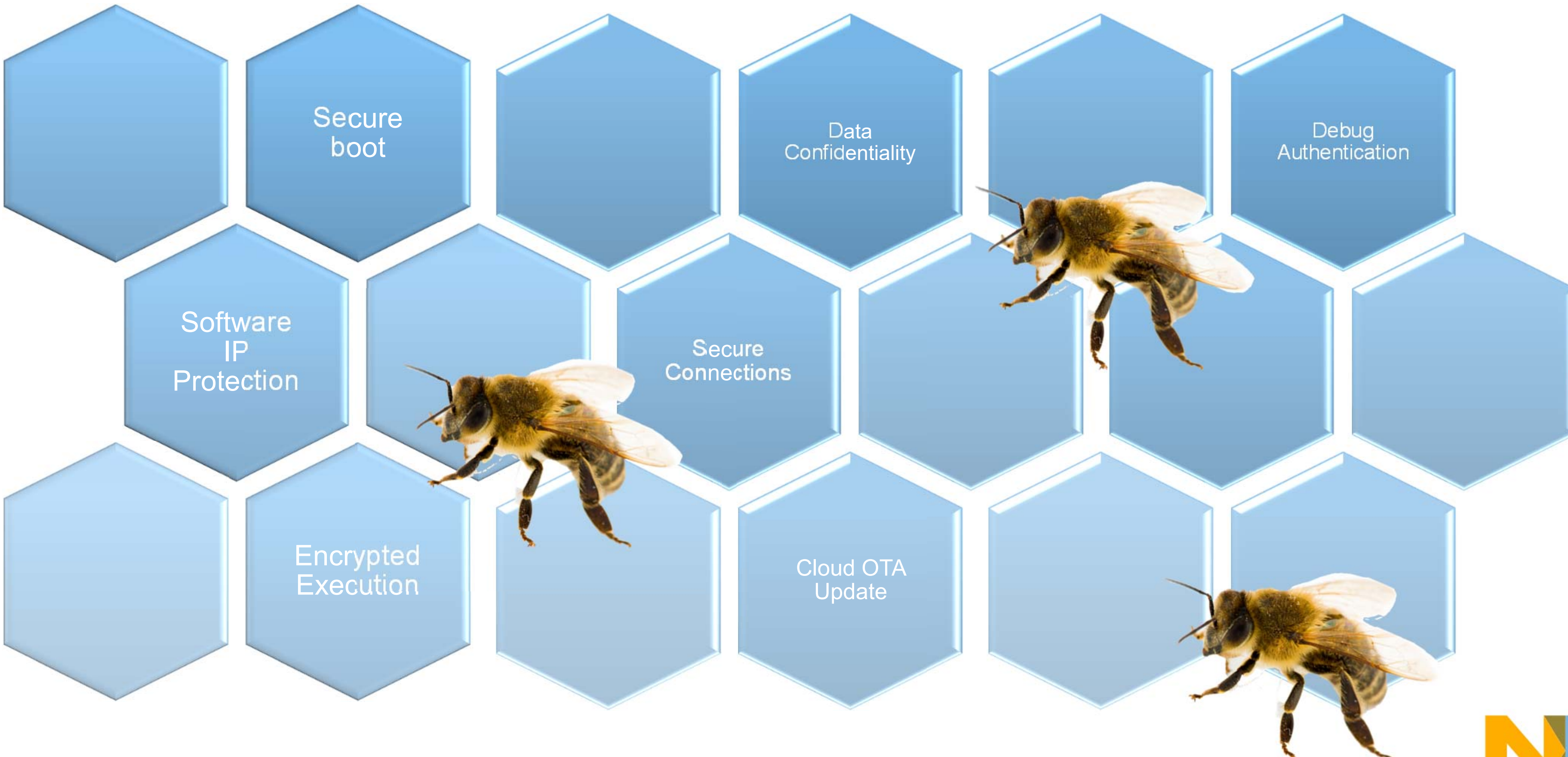


MCUXpresso SDK Examples



KEY USE EXPLORATION

Exploring Embedded Cybersecurity Functions & associated Keys



Secure boot

Cryptographic validation of application code before allowing execution

Attacks mitigated, Security policy and Benefits

- Protection from malware injection from local or remote attacks
- Enforce authenticated boot, authenticated debug and secure OTA process

Creator, storage, and protection

- Created by product owner on a host machine (e.g. HSM)
- Public key data is part of boot image. Protection is achieved through cryptographic validation. In addition Root of Trust Public key is bound the device through Protected flash

Name, type and functions

- Minimum of 2 asymmetric key pairs, Root of Trust and Image key pairs (4 total)
- Private keys are used by host machines. RoT is used to sign Image Certificates, Image is used to sign image data

Software IP Protection

Protection of software in transit by use of symmetric cryptography

Attacks mitigated, Security policy and Benefits

Interception of software binaries in transit (at manufacturing or in the field)

Protect Software Intellectual Property, Prevent product clones

Creator, storage, and protection

PUF chip master key is created by LPC5500, other subordinate keys are created by a host machine

PUF key is ephemeral and maintained by the LPC5500 key store in Protected Flash. This key is used to protected others pre-shared keys.

Name, type and functions

Four (4) Symmetric Keys, PUF Chip master key, SB Key, SB MAC Key, and SB Data encryption key

PUF is used to decrypt SB Key from protected Flash, SB key is used to decrypt MAC and Data keys passed in binaries.

LPC5500 Series: Secure Binaries

- The Secure Binary (SB) image format is a command-based firmware update image
- The SB 2.0 and 2.1 file format also uses AES encryption for confidentiality and HMAC for extending trust from the signed part of the SB file to the command and data part of the SB file. These two keys (AES decrypt key and HMAC key) are wrapped in the RFC3394 key blob, for which the key wrapping key is the SBKEK key

> - User application receives an encrypted SB file containing new firmware and stores it in external SPI flash, or a similar memory.
> - Use API to authenticate SB file.
> - Use API to decrypt and load the SB file.
> - If also using secure boot, the API can be used to authenticate the new firmware in flash before rebooting into it. If this final authentication fails, the new firmware should be made non-executable by erasing and writing over critical regions of it such as the vector table. Even if not using secure boot, the code written to flash can still be signed to support this final authentication step.

Encrypted Execution

Protection of software *at rest* by use of symmetric cryptography

Attacks mitigated, Security policy and Benefits

Chip reverse engineer by physical means. Extraction of software through logical interfaces.

Protect the confidentiality of software property at rest

Creator, storage, and protection

PUF chip master key is created by LPC5500, PRINCE symmetric key is created by LPC5500

PUF key is ephemeral and maintained by the LPC5500 key store in Protected Flash. This key is used to protected the PRINCE keys stored in protected flash

Name, type and functions

Minimum 2 symmetric keys (PUF Chip Master and PRINCE region key)

PUF is used to decrypt PRINCE Key from protected Flash, PRINCE key is instantiated by ROM to support encrypted execution

Data Confidentiality

Protection of data with hardware diversified keys

Attacks mitigated, Security policy and Benefits

Protect from data extraction via logical interfaces or reverse engineering

Strong protection for application data (e.g. Sensor data, WiFi credentials, passwords)

Creator, storage, and protection

PUF chip master key is created by LPC5500, User Data Encryption key is created by LPC5500

PUF key is ephemeral and maintained by the LPC5500 key store in Protected Flash. This key is used to protected the User Data Encryption Key

Name, type and functions

Two, (2) Symmetric Keys, PUF Chip master key and the User Key

PUF is used to decrypt User Key and User Key is used to encrypt sensitive data for the device (Passwords, WiFi credentials, etc.).

Secure Connections

Achieve Transport Layer Security for device to cloud connections

Attacks mitigated, Security policy and Benefits

Protection from snooping and man-in-the middle attacks

Trusted services through validation of the authenticity of the device to cloud connection and confidentiality through key agreement

Creator, storage, and protection

Depends on multiple entities to create the Public Key Infrastructure (CA, Server, Client). With PKI in place, key agreement protocols are used to create session keys.

For the device (LPC5500), Pub/Priv key material is protected by secure boot and Software IP protections.

Name, type and functions

Minimum three (3) asymmetric key pairs, CA, Server and Client. One Symmetric key that is reached by key agreement

LPC5500 has access to CA public key to validate server public key. Also LPC5500 uses Client Private key to respond to server challenge and reach key agreement

TLS Handshake (Source: AN12131)

https://www.nxp.com/docs/en/application-note/AN12131.pdf

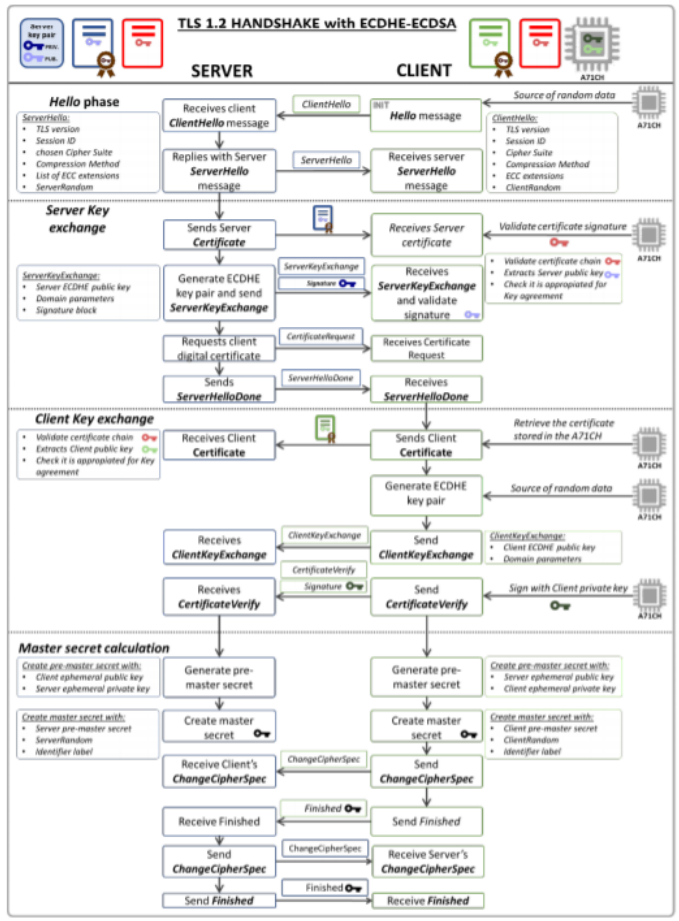


Fig 11. TLS 1.2 Handshake diagram with ECC

Cloud OTA Update

Cloud based fleet management services for secure OTA

Attacks mitigated, Security policy and Benefits

- Prevent tampering of device firmware updates
- Implement a secure OTA for group of devices

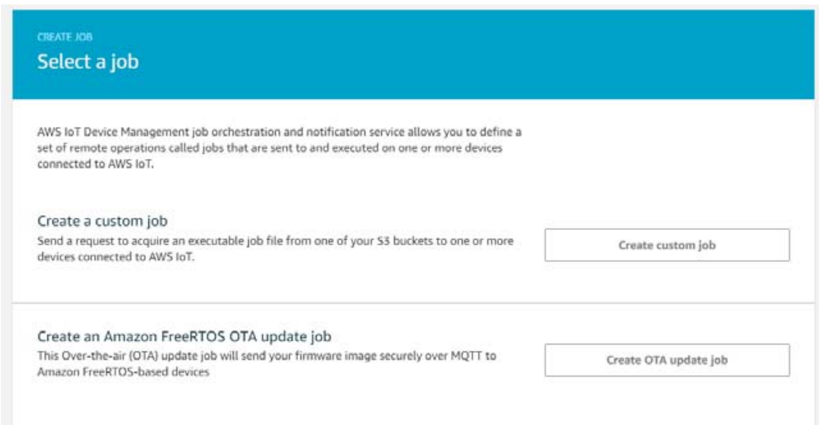
Creator, storage, and protection

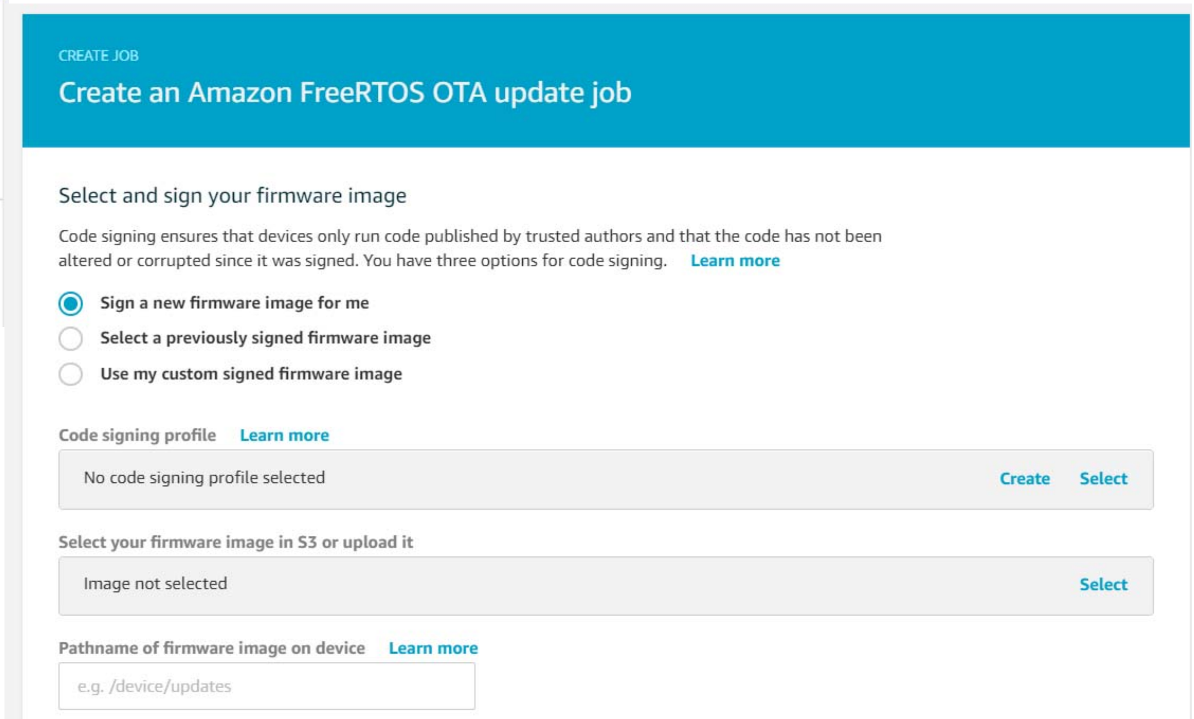
- Host machine is used to create an asymmetric key pair for Cloud OTA
- OTA signing key is protected by host machine, OTA public key is protected by LPC5500 Secure boot

Name, type and functions

- Asymmetric key pair OTA private and OTA public keys
- OTA public key is present in the device so that the OTA image can be validated

OTA Jobs from AWS





Debug Authentication

Use of cryptography to open access to device Debug capabilities

Attacks mitigated, Security policy and Benefits

Prevent firmware tampering at the device or re-profiling of the device with malicious software

Restrict logical interfaces on the device

Creator, storage, and protection

Debug entity creates the keys related authenticated debug

Debug entity must protect the private key, the public key is cryptographically validated

Name, type and functions

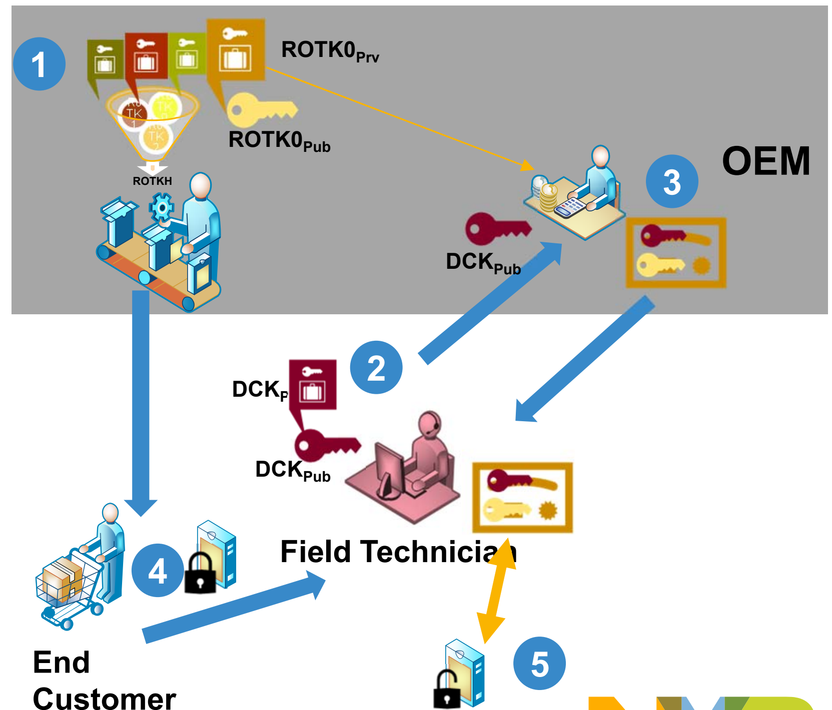
Authenticated debug private key and public key is an Asymmetric key pair.

The Private key is used by a host machine to sign a challenge provided by the LPC5500, the public key is passed to the LPC5500 and validated with the Root Of Trust Public key before use

Secure debug

Debug authentication for RMA use case

1. OEM generates RoT key pairs and programs the device before shipping.
   - SHA256 hash of RoT public key hashes
2. Field Technician generates his own key pair and provides public key to OEM for authorization.
3. OEM attests the Field Technician's public key. In the debug credential certificate he assigns the access rights.
4. End customer having issues with a locked product takes it to Field technician.
5. Field technician uses his credentials to authenticate with device and un-locks the product for debugging.



KEY MANAGEMENT TABLE SUMMARY (REFERENCE)

Secure Boot

Key Type	Key Type Name	Created By	Function	Storage	Used by	Protected by	Benefit	Security Policy	Attacks Mitigated
SECURE BOOT									
Asymmetric-RSA	Root Of Trust Private Keys	Product Owner - Host Machine, OpenSSL, HSM	Sign Image Certificates, Sign Debug credentials	Example: Host machine	Product Owner or their designated entity	Host Machine	Integrity of application code, each time the Secure boot process is done	Authenticated Boot, Authenticated Debug access, Secure OTA process	Local or Remote malware injection. Logical interface attacks (JTAG)
Asymmetric-RSA	Root of Trust Public Keys	Same as above	Validate Image Certificate, Debug credentials	Hash stored in Protected Flash space on the LPC5500	Chip itself for secure boot or authenticating Debug credentials	Chip firewalls			
Asymmetric-RSA	Image Private Key	Same as above	Sign boot data including app code	Example: Host machine	Dat	Host Machine			
Asymmetric-RSA	Image Public Key	Same as above	Validate boot data including application code	Part of boot data	Chip itself for secure boot	Cryptographic Validation using root of trust public key			

Secure Connections

Key Type	Key Type Name	Created By	Function	Storage	Used by	Protected by	Benefit	Security Policy	Attacks Mitigated
SECURE CONNECTIONS									
Asymmetric	Cloud Provider Certificate Authority Public key	Certificate Authority	Validate the identity of the cloud connection	At the device in Application Code image data as a certificate	Application Code TLS handshake	Secure Boot	Validate the authenticity of the cloud connection	Establish Trusted Connections, Protect data in transit	Man in the middle attacks, Snooping
Asymmetric	Cloud Provider Certificate Authority Private key	Certificate Authority	Sign Certificates in the Public Key Infrastructure	Certificate Authority	Application Code for TLS handshake	Certificate Authority			
Asymmetric	Server Private Key	Cloud Service	TLS Handshake	Cloud Service	Cloud	Cloud			
Asymmetric	Server Public Key	Cloud Service	TLS Handshake	Cloud Service	Chip for validating Server and reaching key agreement	Secure Boot-CA Public Key- Validation of Server certificate			
Asymmetric	Client Private Key	Options (Host machine, Chip itself, Cloud Provider)	TLS Handshake	Image Data, encrypted by Chip User Key	Chip for signing server challenges and key agreement protocol	PUF Chip Master Key encryption of Chip User Key - Chip user key encryption			
Asymmetric	Client Public Key	Options (Host machine, Chip itself, Cloud Provider)	TLS Handshake	Image Data	Server for validating authenticity of the client and reaching key agreement	Secure Boot at the device - CA public key validation by the server			
Symmetric	Session Key	Key agreement protocols based on Public Key Cryptography	TLS Data encryption	SRAM	Shared secret for data confidentiality between server and client	Application Software			

Cloud OTA and Authenticated Debug

Key Type	Key Type Name	Created By	Function	Storage	Used by	Protected by	Benefit	Security Policy	Attacks Mitigated
CLOUD BASED OVER THE AIR UPDATE									
Asymmetric	Cloud OTA Image Private Key	Host machine	Sign binaries pushed by the cloud					Secure OTA	Firmware Tampering
Asymmetric	Cloud Based OTA Image Public key	Host machine	Validate binaries received by the cloud	Host machine	Host machine for providing signatures to AWS	Secure boot	Validate the Authenticity of image data sent by the cloud		
AUTHENTICATED DEBUG									
Asymmetric-RSA	Authenticated Debug Private Key	Host machine of the Entity whom will debug	Respond to Chip Authenticated Debug Challenge	Host machine	Host machine for providing signatures to the LPC5500 during debug authentication	Host Machine	Open access to pre-approved debug capabilities	Restrice logical interfaces	Firmware Tampering
Asymmetric-RSA	Authenticated Debug Public key	Host machine of the Entity whom will debug	Sent to LPC5500 in order to be used in a validation of a challenge response		LPC5500 in order to validate a Debug Challenge response	Signing done by Root Of Trust private key on a host machine	Cryptographically validate the debug request		

Software protection, Encrypted Execution and Data Confidentiality

Key Type	Key Type Name	Created By	Function	Storage	Used by	Protected by	Benefit	Security Policy	Attacks Mitigated
SOFTWARE IP PROTECTION - PRODUCT CONTERFEITS									
Symmetric-AES	PUF Chip Master Key	PUF on the chip itself	Key encryption key for other keys	Activation code (non secret) stored in Protected Flash, Ephemeral key creation upon software request.	Chip for decoding other Keys and protected data	Properties of PUF, Software management for initialize/deinitialize	Protection of other keys which provide confidentiality of data (Application code, etc.)	Protect Software IP, Never store key material in plain text, Enforce key diversity (Unique key per chip/device),	Extracting device key material from logical interfaces
Symmetric-AES	SB Key	Product owner, host machine	Decrypt the Binary Key	KEY STORE: Stored as a key code which is encrypted by PUF Chip Master key	Chip for decoding Binary Key during bootloading	Encryption by PUF block to protect confidentiality and integrity	Confidentiality of Software IP	Protect Software IP	Interception of software in transit (at manufacturing or in the field),
Symmetric-MAC	SB MAC Key	Host machine (elftosb)	Check the integrity of the SB file header data	Encrypted by SB Key and part of the SB file	Chip for checking integrity of the Header Data in SB file	PUF Chip Master Key encryption of SB Key and SB Key Encryption of te MAC key	Confidentiality of Software IP	Protect SW IP	attacks on binaries being passed to the device
Symmetric - AES	SB Data Encrytion Key	Host machine (elftosb)	Encrypte image data in SB files	Encrypted by SB Key and part of the SB file	Chip for decrypting image data for loading	PUF Chip Master Key encryption of SB Key and SB Key Encryption of te SB Data Encryption key	Confidentiality of Software IP	Protect SW IP In transit	attacks on binaries being passed to the device
ENCRYPTED EXECUTION									
Symmetric-PRINCE	PRINCE Key	Chip itself using PUF	Encrypted Execution	KEY STORE: Stored as a key code which is encrypted by PUF Chip Master key	Chip for encrypted execution	PUF Chip Master Key encryption of Prince Key only known by the chip	Confidentiality of Software IP	Protect SW IP In use and storage	Chip reverse engineering or data extraction from logical interfaces
DATA CONFIDENTIALITY									
Symmetric- AES	Chip User Key	Host machine or Chip Itself	Protect data managed by the device	KEY STORE: Stored as a key code which is encrypted by PUF Chip Master key	Application Code	PUF Chip Master Key encryption of Chip User Key	Protect data at rest with diversified keys (Sensor data, passwords, WiFi credentials)	Protect data at rest	Extraction of data from logical interfaces

CONCLUSION

Summary

- It has never been so easy to get a device connected and create an IoT Device
  - This is both amazing and frightening at the same time
- Many device types share a common set of assets that must be protected by security functions
  - Secure devices make proper use of cryptography to perform security functions
  - Both Symmetric and Asymmetric cryptography is used
  - Hardware protection of the keys is essential for protecting the device
- State of the art Microcontrollers like LPC5500 series integrate technology to achieve security functions
  - PUF based key management
  - Enabled by ROM

Thanks!

SECURE CONNECTIONS
FOR A SMARTER WORLD