Towards Secure, Scalable, Efficient IoT of Scale

ARM

Remy Pottier
Director of Strategy, Incubation Businesses

Valence
June 22nd, 2016

Agenda

- Some security principles & solutions
- Arm Building blocks
- Q&A

Security in the Internet of Things is in a bad shape

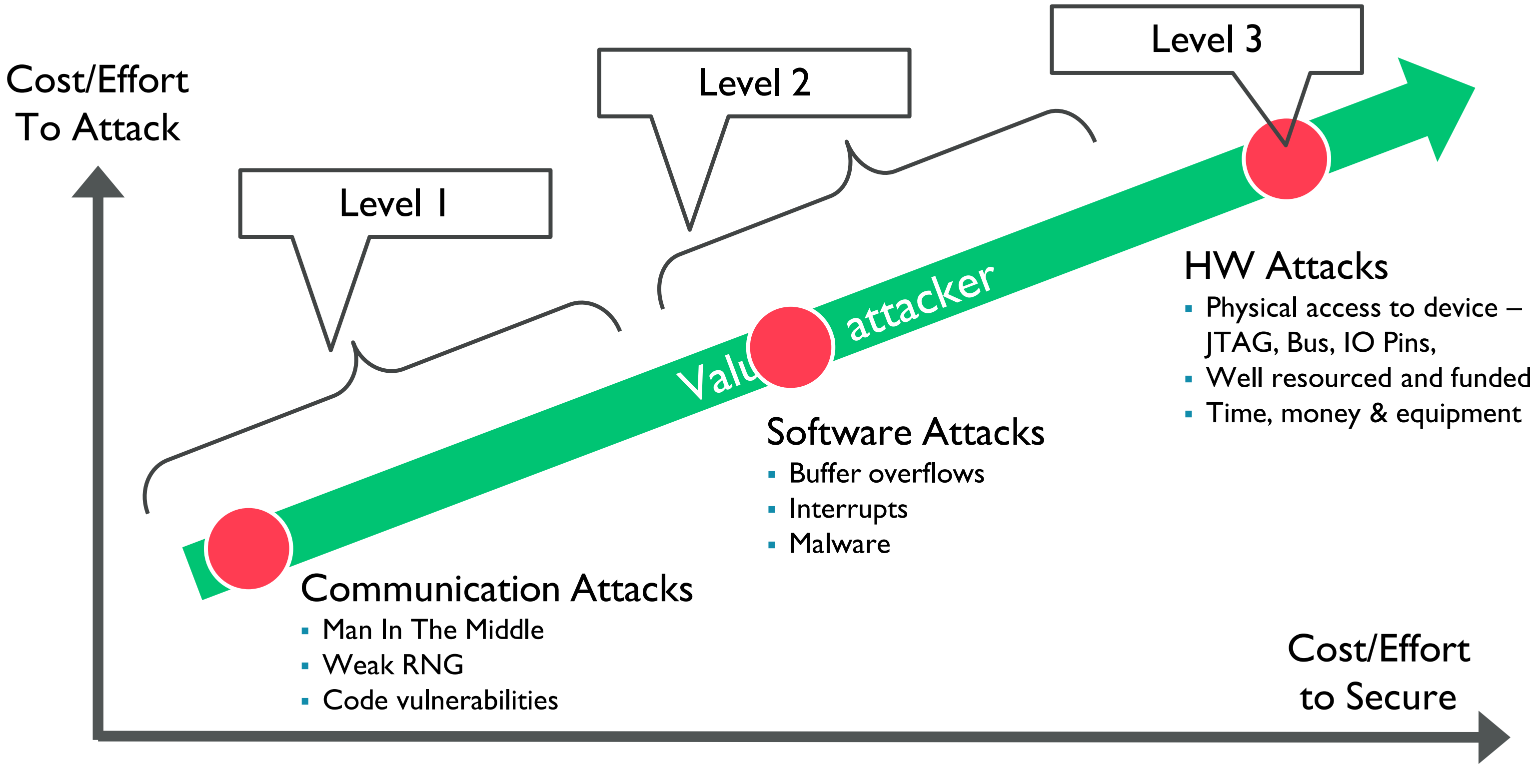
ARM

Key finding highlights

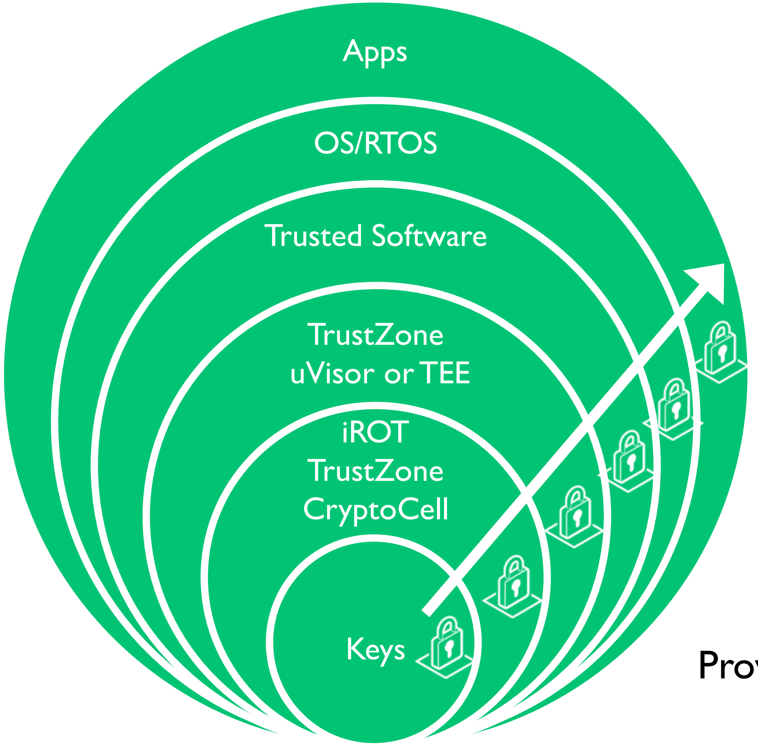
- 1 in 5 did not use secured connections to the cloud
- None provided mutual authentication
- Many of the cloud services contained common web vulnerabilities
- 10 vulnerabilities for 15 web interfaces to control the devices
- Most did not use secure firmware updates

We failed to learn from our history

Security Requirements: How Much Security?



Initial Root of Trust & Chain of Trust



RTOS

Trusted Apps/Libs

Extended Root of Trust e.g. TrustZone based uVisor or TEE

Initial Root of Trust: Dependable Security functions

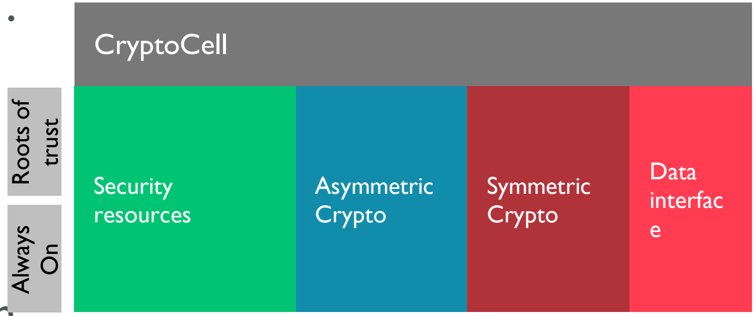
Provisioned keys/certs

Root Of Trust & Security Components

- Basic system:
  An initial Root of Trust: security functions that you can depend on
  An identity (e.g. key loaded at manufacture)
  Secure boot/ Primary Boot Loader/ Authentication
  TRNG
  Secure storage

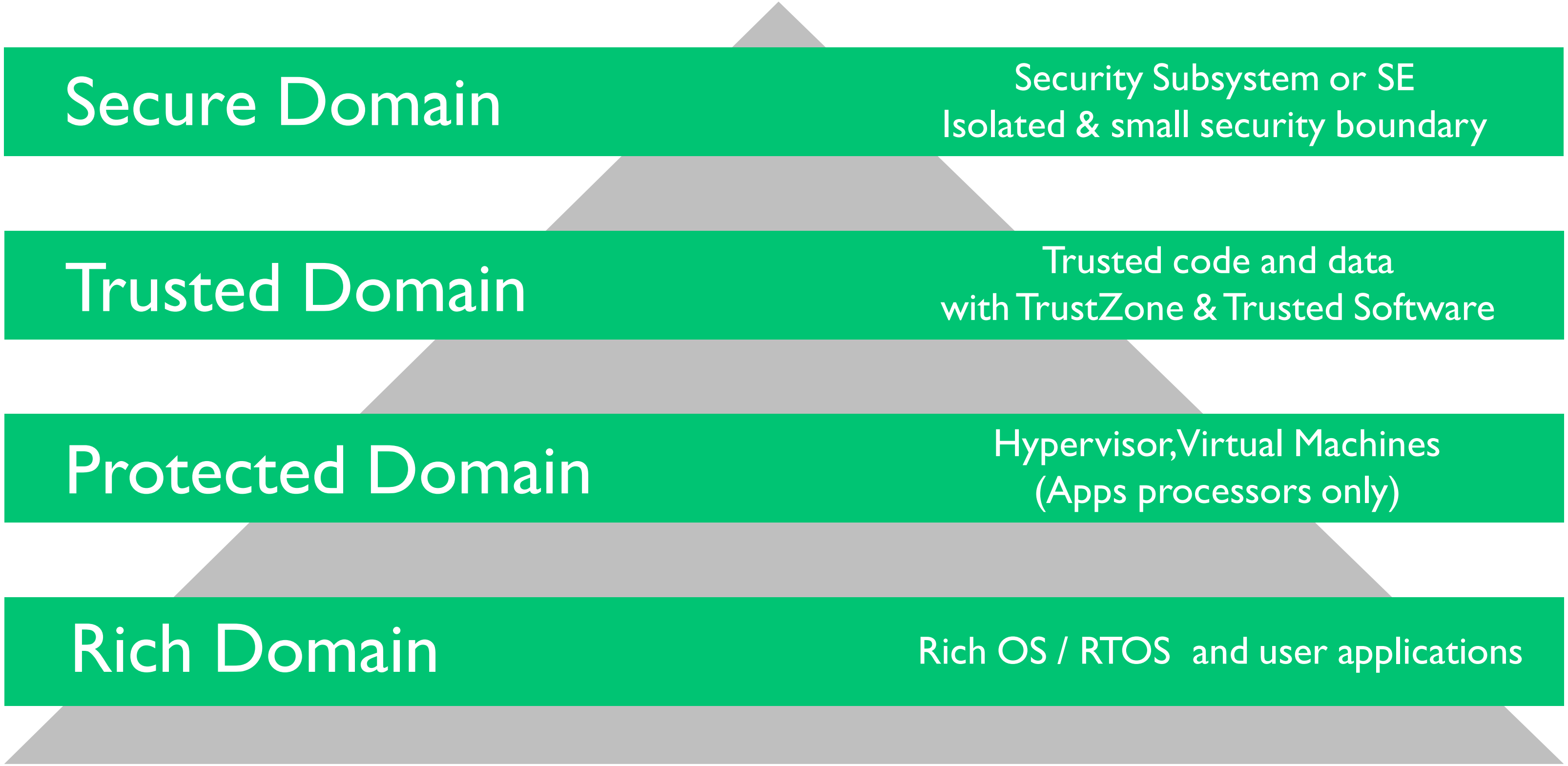
- Extended system:
  Crypto Acceleration e.g. for TLS, RSA…
  Life cycle management
  Secure debug
  Firmware updates

- TrustZone CryptoCell provides a general purpose security HW subystem



ARM

Compartmentalisation & Least Privilege – Hierarchy of Trust



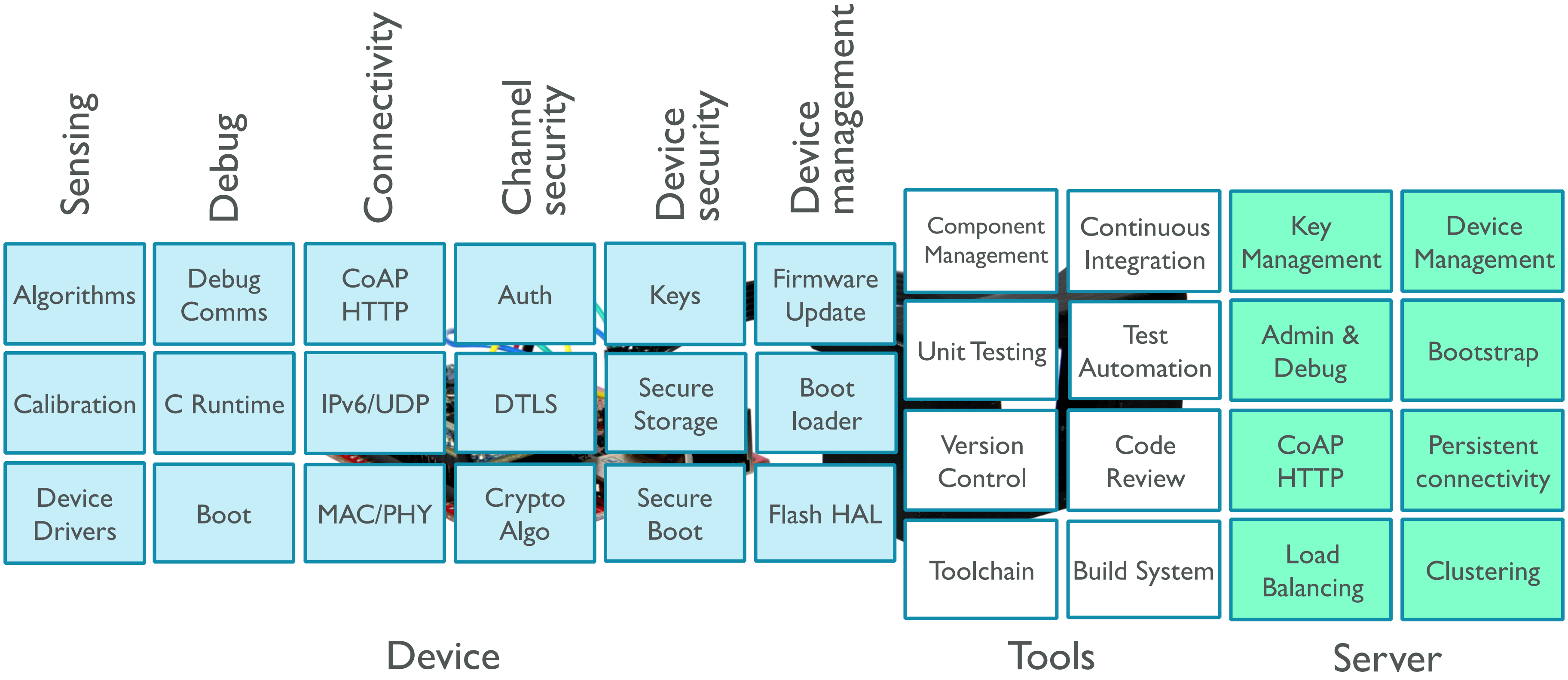
ARM

Now we need to enable secure < $1 microcontroller designs done by people with absolutely no security experience

Traditional Embedded Development vs. IoT

- Historically closed systems
  - Very little code reuse or design commonality between systems

RTOS	Bespoke Software Stack and…


Sensing
Algorithms
Calibration
Device Drivers
Debug
Debug Comms
C Runtime
Boot
Connectivity
CoAP HTTP
IPv6/UDP
MAC/PHY
Channel security
Auth
DTLS
Crypto Algo
Device security
Keys
Secure Storage
Secure Boot
Device management
Firmware Update
Boot loader
Flash HAL
Component Management
Unit Testing
Version Control
Toolchain
Continuous Integration
Test Automation
Code Review
Build System
Key Management
Admin & Debug
CoAP HTTP
Load Balancing
Device Management
Bootstrap
Persistent connectivity
Clustering
Device
Tools
Server

Traditional Embedded Development vs. IoT

- Historically closed systems
  - Very little code reuse or design commonality between systems

RTOS | Bespoke Software Stack and…

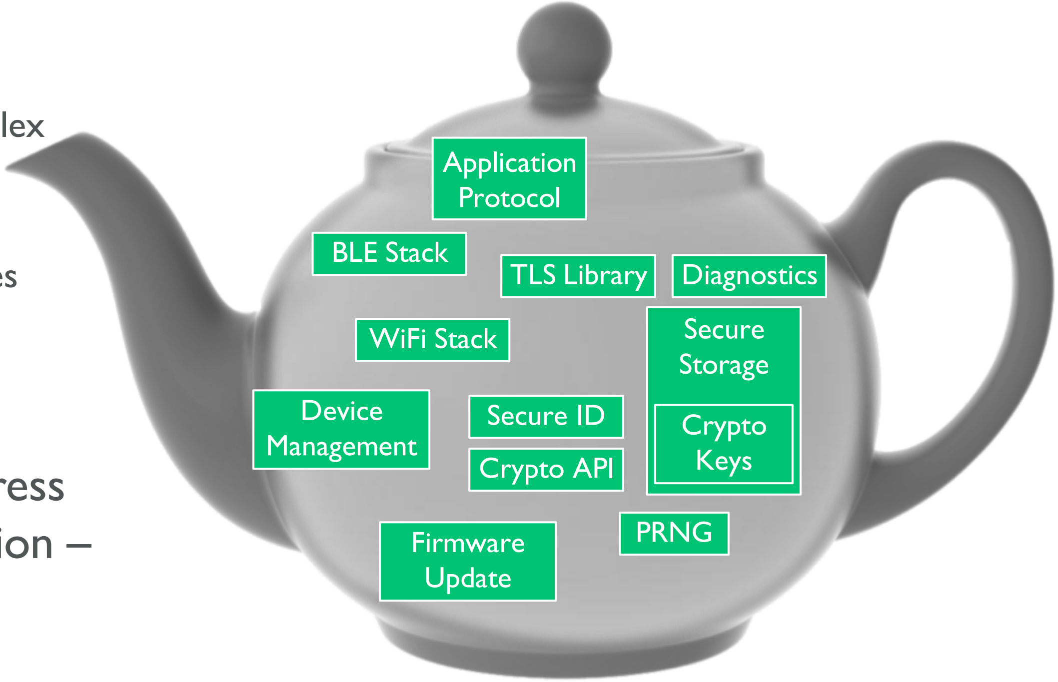
- Very few developers have strong experience in creating secure systems
- Need a platform with built-in security and strong guidance on best practices
  - Increased productivity: “Common denominator” security functionality ready to go

mbed OS | Application

IoTeapot "Hello World" example – the attacker view

- Even simple IoT products require complex components
  - Secure server communication over complex protocols
  - Secure firmware updates over the air
  - Unclonable cryptographic device identities
  - Cryptography APIs and random number generation

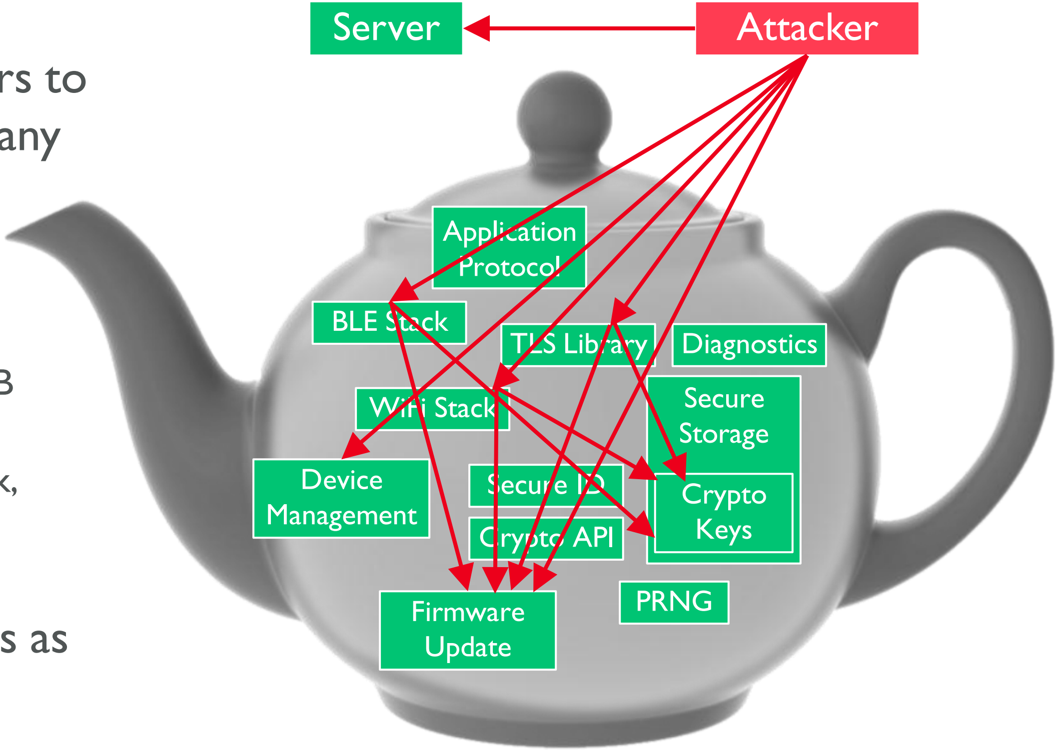
- Existing IoT solutions use flat address spaces with little privilege separation – especially on microcontrollers



ARM

IoTeapot "Hello World" example – the attacker view

- Flat security models allow attackers to break device security by breaking any system component
- Common attack entry points:
  - Complex protocols like TLS, Wi-Fi or USB device configuration
  - Firmware update functions (USB, network, CAN…)
- Impossible to recover from attacks as firmware update functions can be compromised by the attacker



ARM

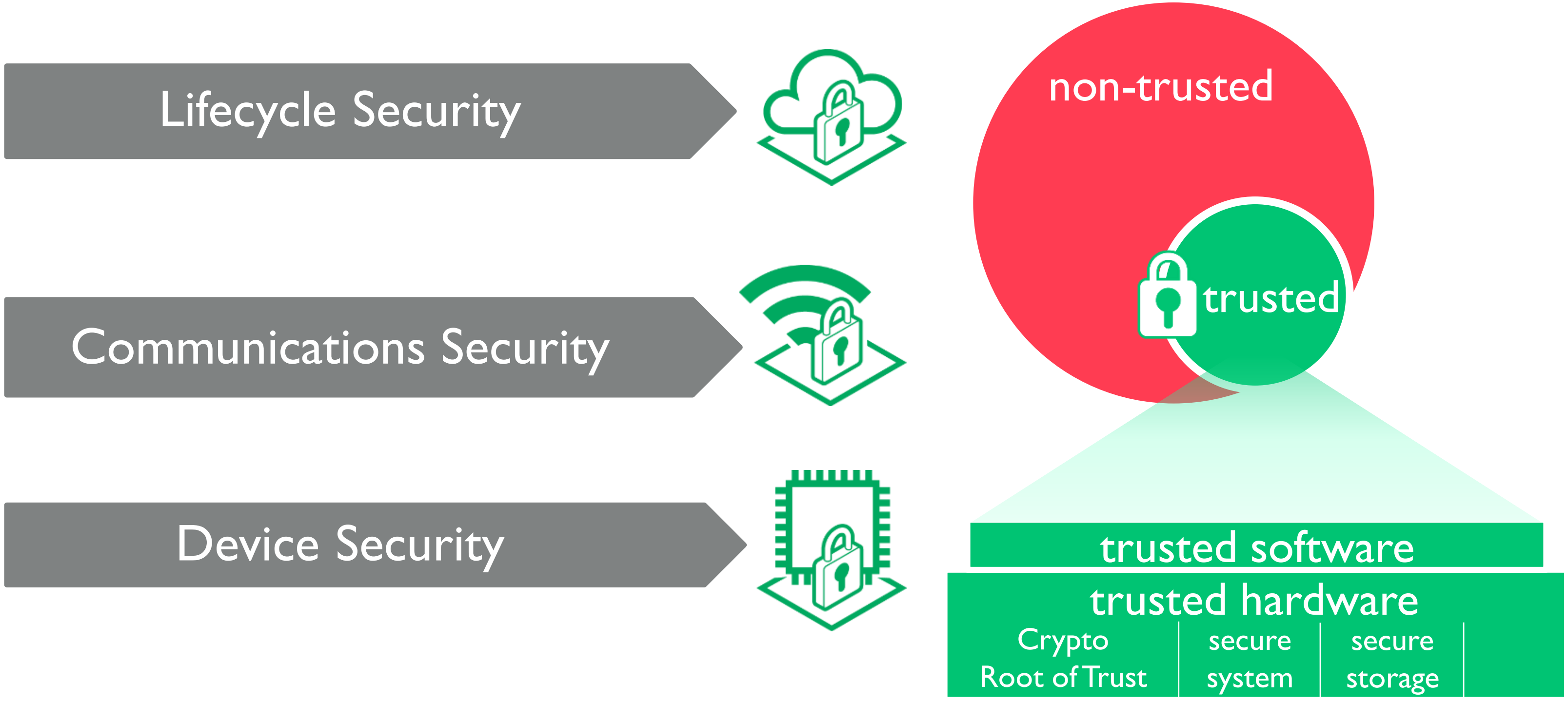
Security plus time equals comedy: plan for the worst case

- System security is dynamic over device lifetime
- Devices last longer than expected
- Likelihood of attacks underestimated as a result
- If your system is successful, it will be hacked
- Deployment costs of firmware updates in case of hacks often surpasses costs of new devices. As a result known-broken systems are kept in use
- Developers must ensure secure, reliable and "cheap" update possibilities
- Devices must be capable of remote recovery from an untrusted state back to a trusted state

https://commons.wikimedia.org/wiki/File:Cret_Comedy_and_Tragedy.JPG

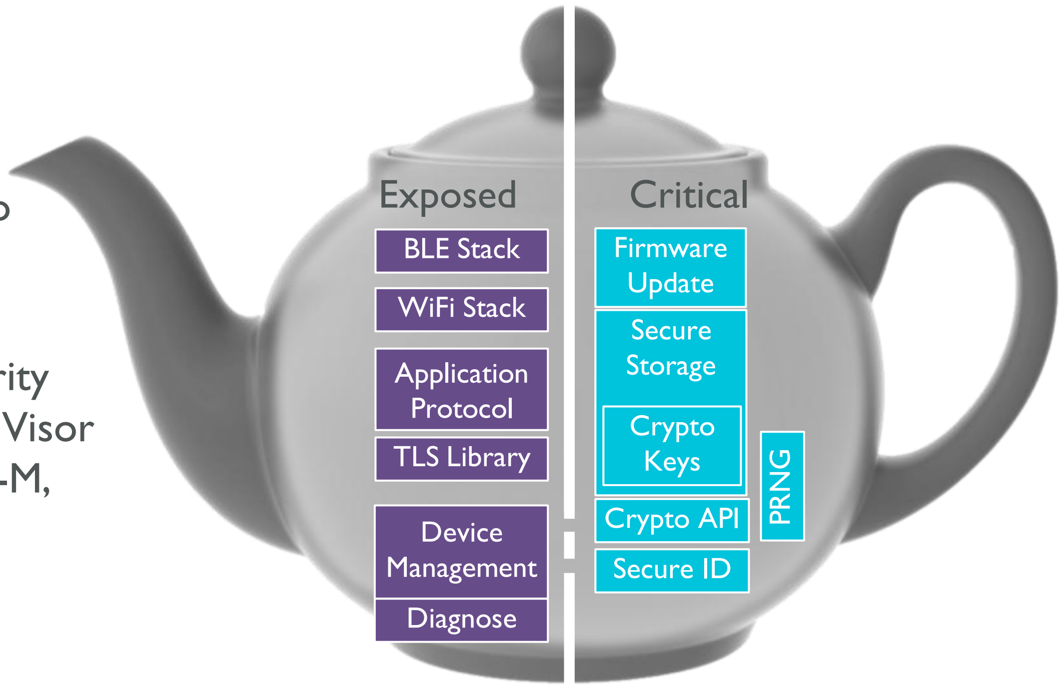
ARM

Applying the Lessons From 20 Years of Mobile to IOT



IoTeapot "Hello World" example – mitigation strategies

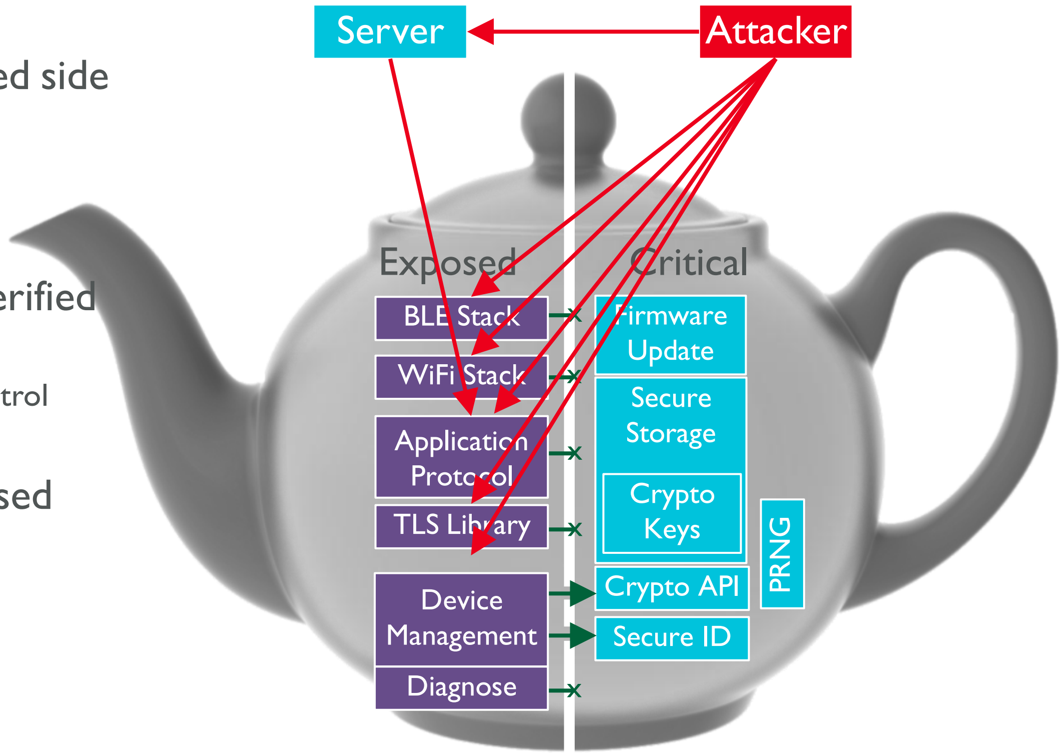
- Split security domains into
  - exposed uncritical code
  - protected critical code
- Keep footprint of critical code small to enable verification
- Protect key material and system integrity using hardware memory protection (uVisor using ARMv7-M MPU, ARM TrustZone-M, TrustZone-A)



ARM

IoTeapot "Hello World" example – mitigation strategies

- Attackers can compromise the exposed side without affecting critical code
- Using cryptographic hashes the integrity of the exposed side can be verified
  - Triggered on server request
  - Protected security watchdog allows remote control
- Protected side can reliably reset exposed boxes to a clean state
- The device attack surface is massively reduced as a result



ARM

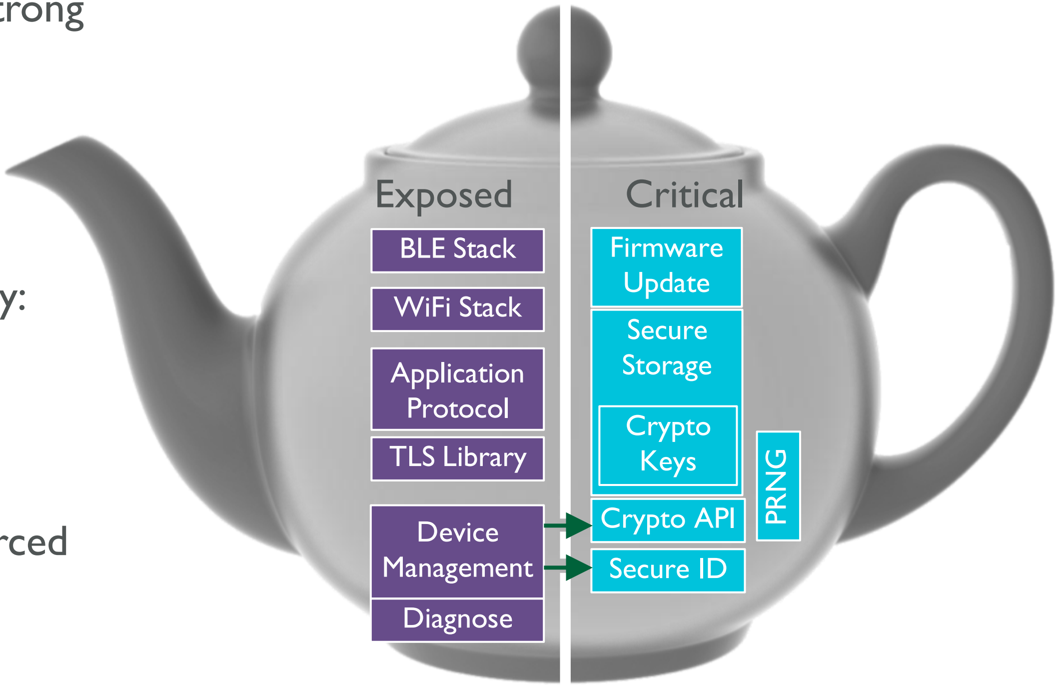
Enable fast innovation

- Modules on the critical side require strong security coding practices

- The critical code rarely changes

- Exposed code can be developed rapidly:
  - Faster time to market
  - Quick innovation cycles for the exposed boxes
  - Still a secure product

- Firmware updates can be reliably enforced even over broken or malware-infected firmware



ARM

"Even the easiest to develop type of Endpoint device must behave in a reliable, high quality, and secure manner because it is expected to participate in a network that could eventually span up to millions of devices in size"

GSMA IoT Security Guidelines, 2016

ARM

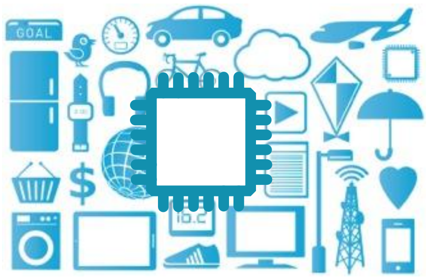
Trusting little data

Little Data Enables Big Data

ARM mbed

IoT Device Platform

mbed OS



IP and Web to the edge

mbed Device Server

IoT Service or App

End-to-End Security, Web, Data Objects & Management

Little Data

BIG DATA

Lifecycle Security

Communication Security

Device Security

Directory and Subscription | Security, Admin and Multi-tenancy

Data Flow Management – *RESTful and Publish/Subscribe*

Device Management – *Lightweight M2M (LWM2M)*

mbed Device Interface – *Open Web Standards*

Application Transfer Protocols – *CoAP, HTTP, MQTT*

mbed TLS

mbed Device Server

IPv4 | IPv6, 6LoWPAN

mbed TLS

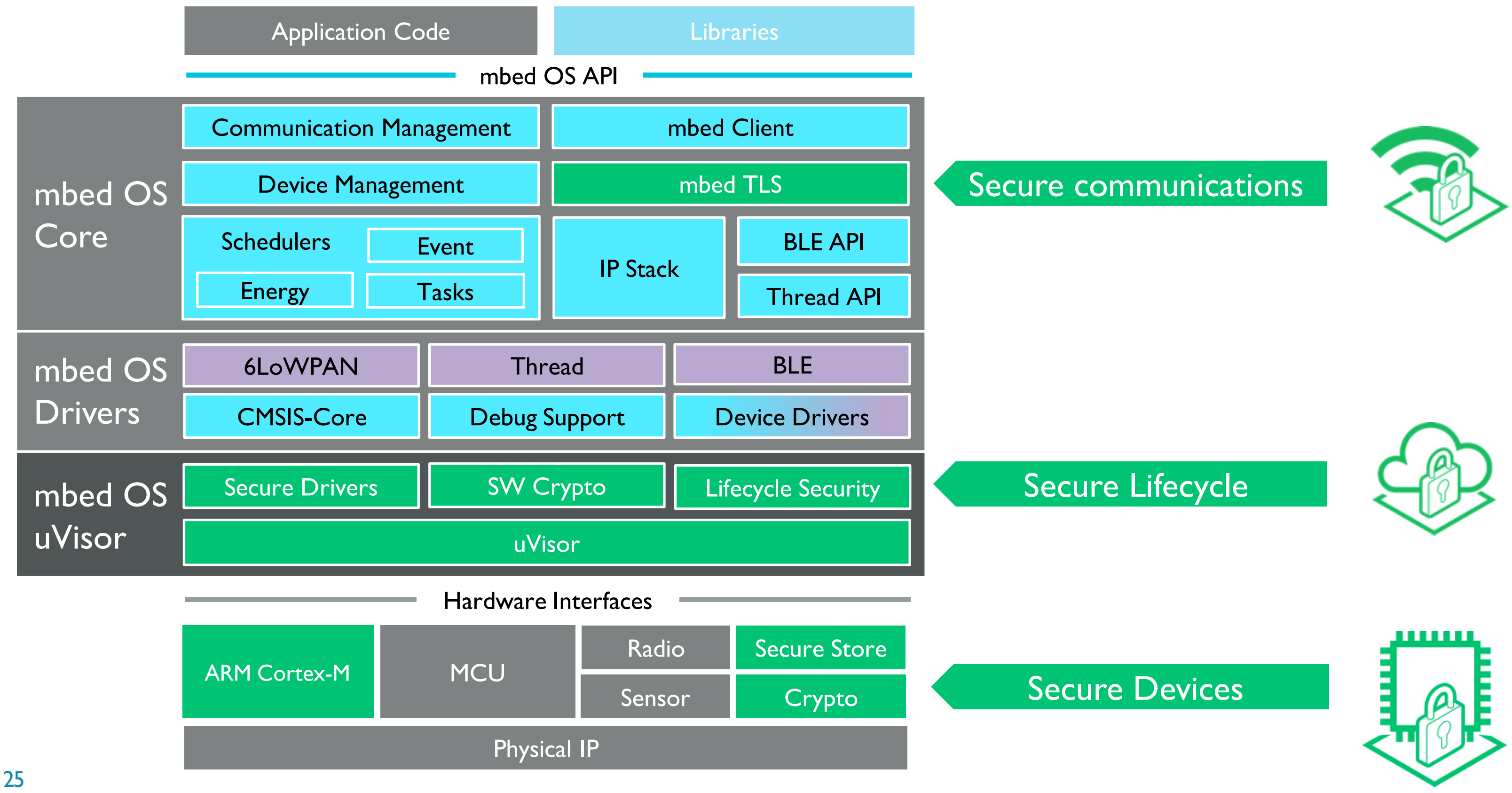
Device Management: LWM2M

Secure Device Services

mbed OS

ARM

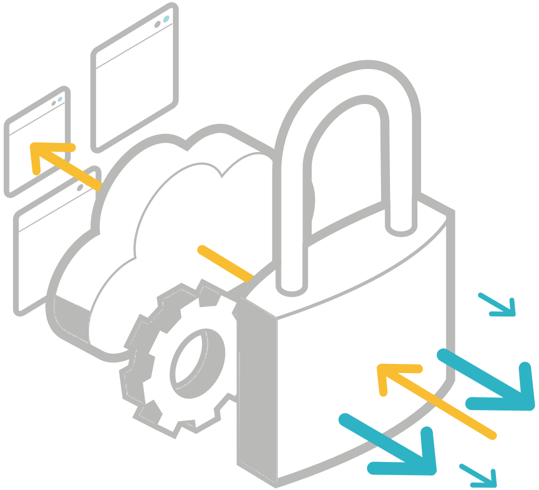
Device security from Silicon to Services



mbed TLS

- mbed TLS makes it trivially easy for developers to include cryptographic and SSL/TLS/DTLS capabilities in their embedded products, with a minimal code footprint.
- mbed TLS makes it easy to disable any feature during compilation thus reducing the footprint to only those features absolutely needed.
- Small Footprint and API’s with full documentation available
- Open Source under Apache 2.0 license at https://tls.mbed.org/
- Suitable for use on Cortex-M and Cortex-A targets

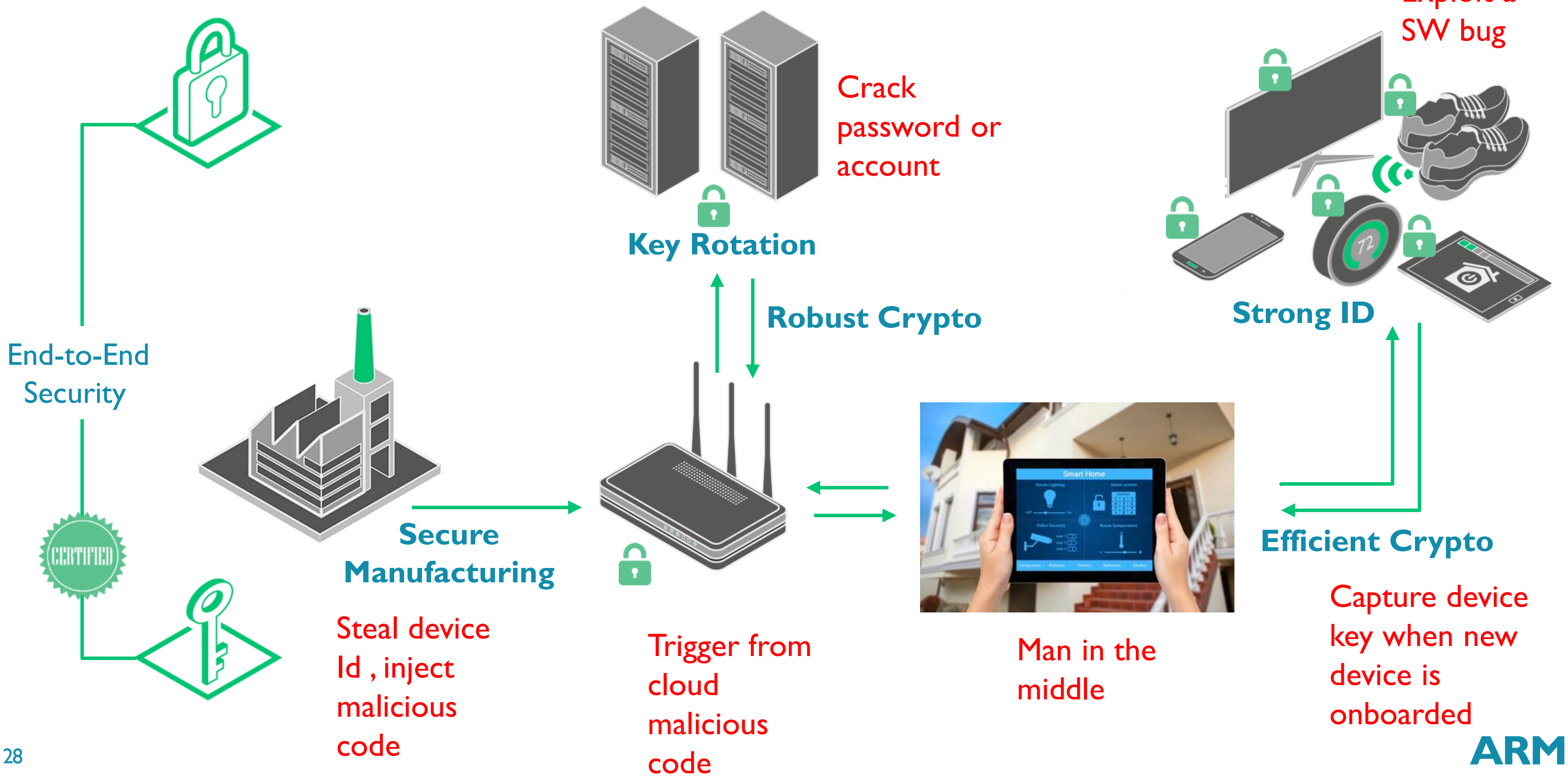
uVisor



- A tiny, hypervisor/microkernel-like security kernel at the foundation of mbed OS
- Creates and enforces secure isolation boundaries within the OS, between different parts of the system
- Enables secrets to be strongly protected against software and network-bourn attackers
- Efficient hardware enforcement through the memory protection unit (MPU)
- Efficient sandboxing for all platforms reduces the need for target-specific modifications

ARM

We need to protect all points in the chain



Summary

- Deployments will not scale without trust
- Embedded IoT development practices have to evolve
- ARM provide building blocks for security :
  - TrustZone for v8-M brings familiar security architecture to lowest cost points
  - CryptoCell provides Root of Trust to system & a toolbox of security functions
  - uVisor
  - Mbed TLS
  - ..

Smart Construction Concrete Monitoring Solution

- Based on mbed Smart City Reference Design using Sub-GHz 6LoWPAN and CoAP mesh networking, mbed OS and mbed Device Connector
- Deployed in field at construction sites by UK's top concrete manufacture
- Solution reports the maturity of a concrete pour in real-time, eliminating delays manual measurements and can save £50k per annum* on project costs

(*Converge data)

Time to Treatment IoT Solutions for Healthcare

In August 2015, the Cardiology Department of Leiden University Medical Center (LUMC), a leading European hospital, completed a 6-month, 100-patient pilot review of the Zebra Time Tracking Solution based on ARM mbed and the solution has now moved into commercialization phase for all AMI patients.