PSA & Attestation

Yogesh Deshpande, Abeezar Burhan, Thomas Fossati

Feb’20

Agenda

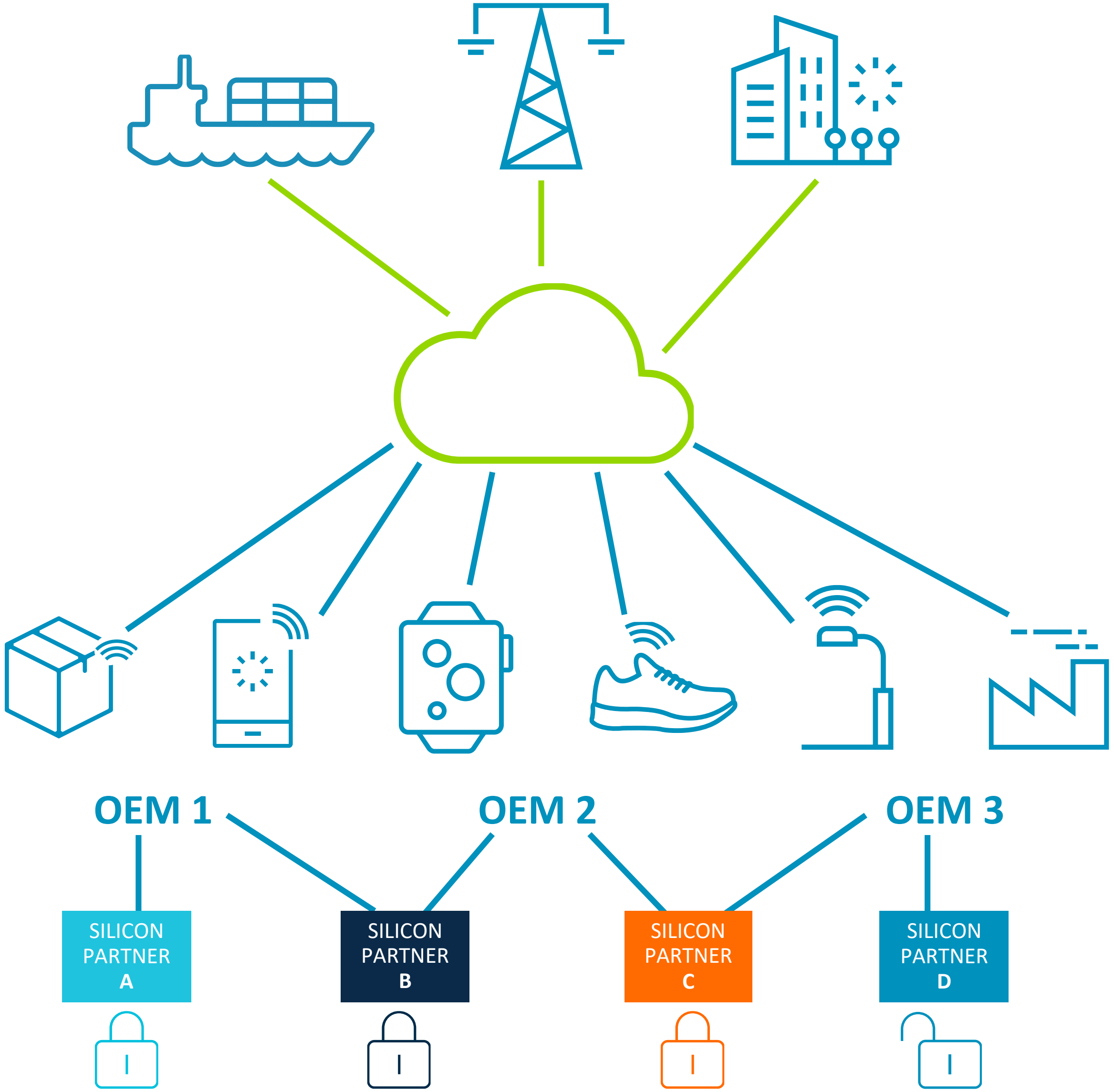
- IoT Security – Challenges
- PSA – building trust in IoT
- PSA Attestation
- Practical use cases of attestation
- arm view of reference IoT implementation

IoT Diversity Demands a Different Approach

Many cloud services needing to trust the data & therefore trust the devices

10,000's OEMs

100's of chip vendors with different RoT

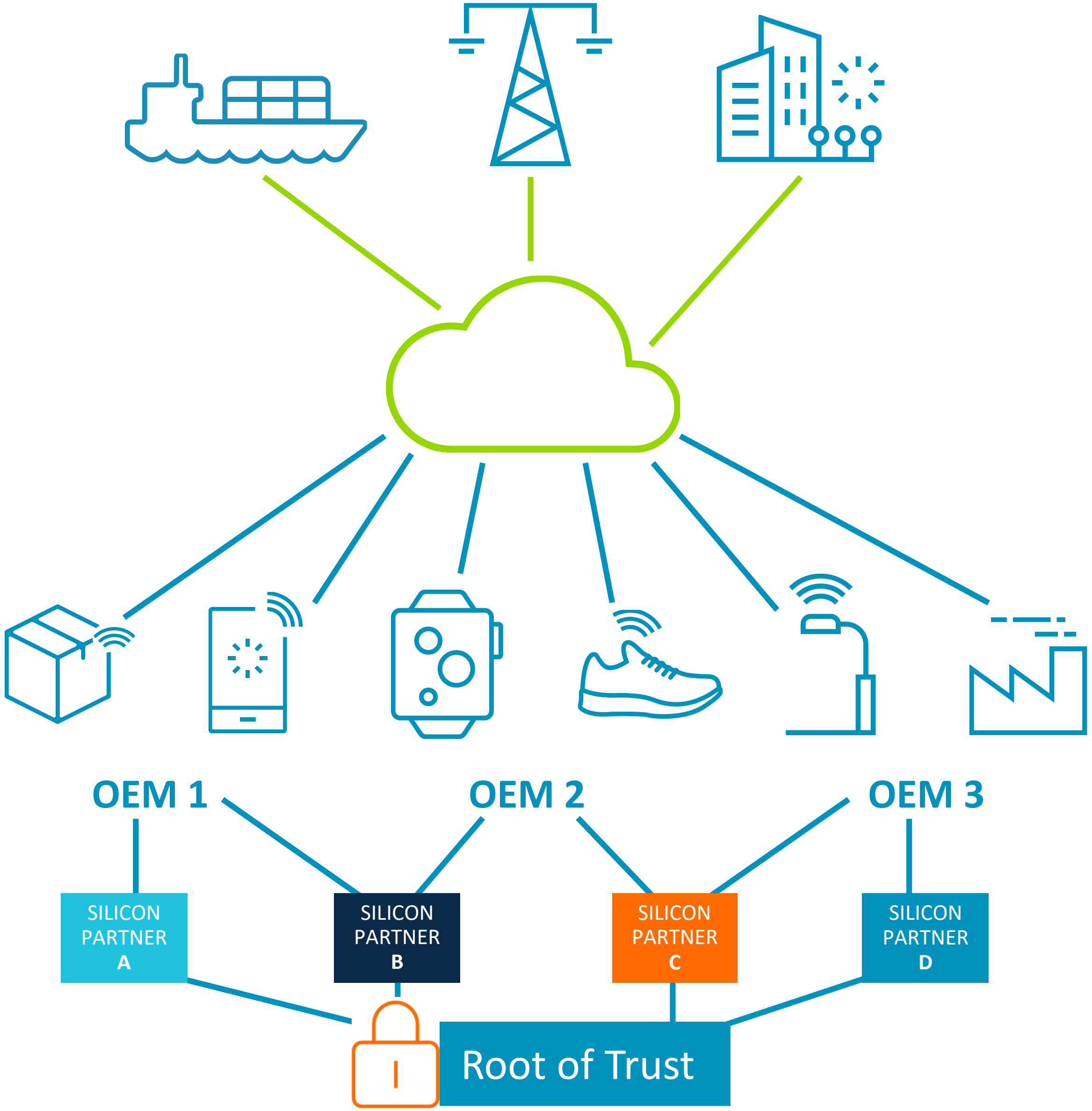


IoT Diversity Demands a Different Approach

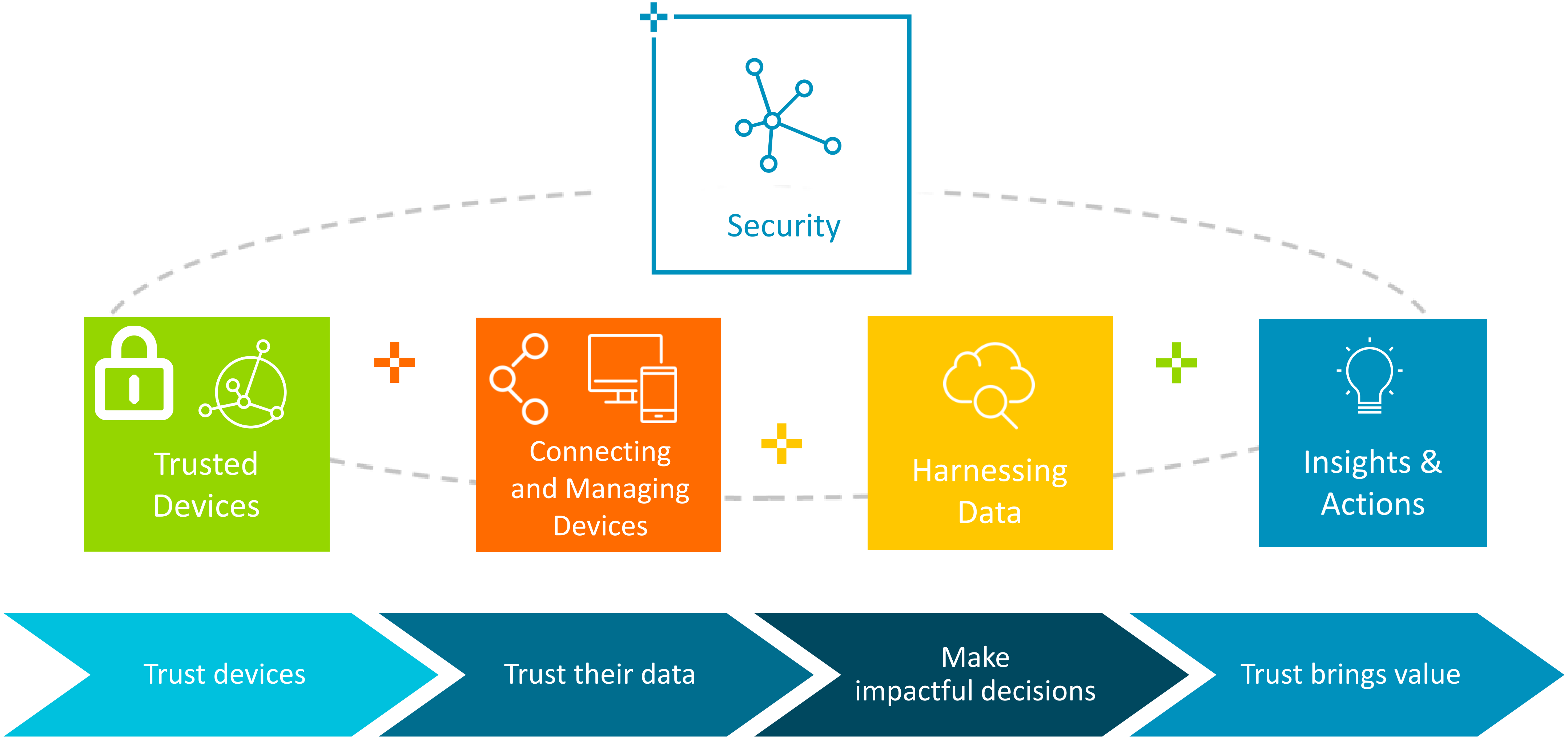
Many cloud services needing to trust the data & therefore trust the devices

10,000’s OEMs

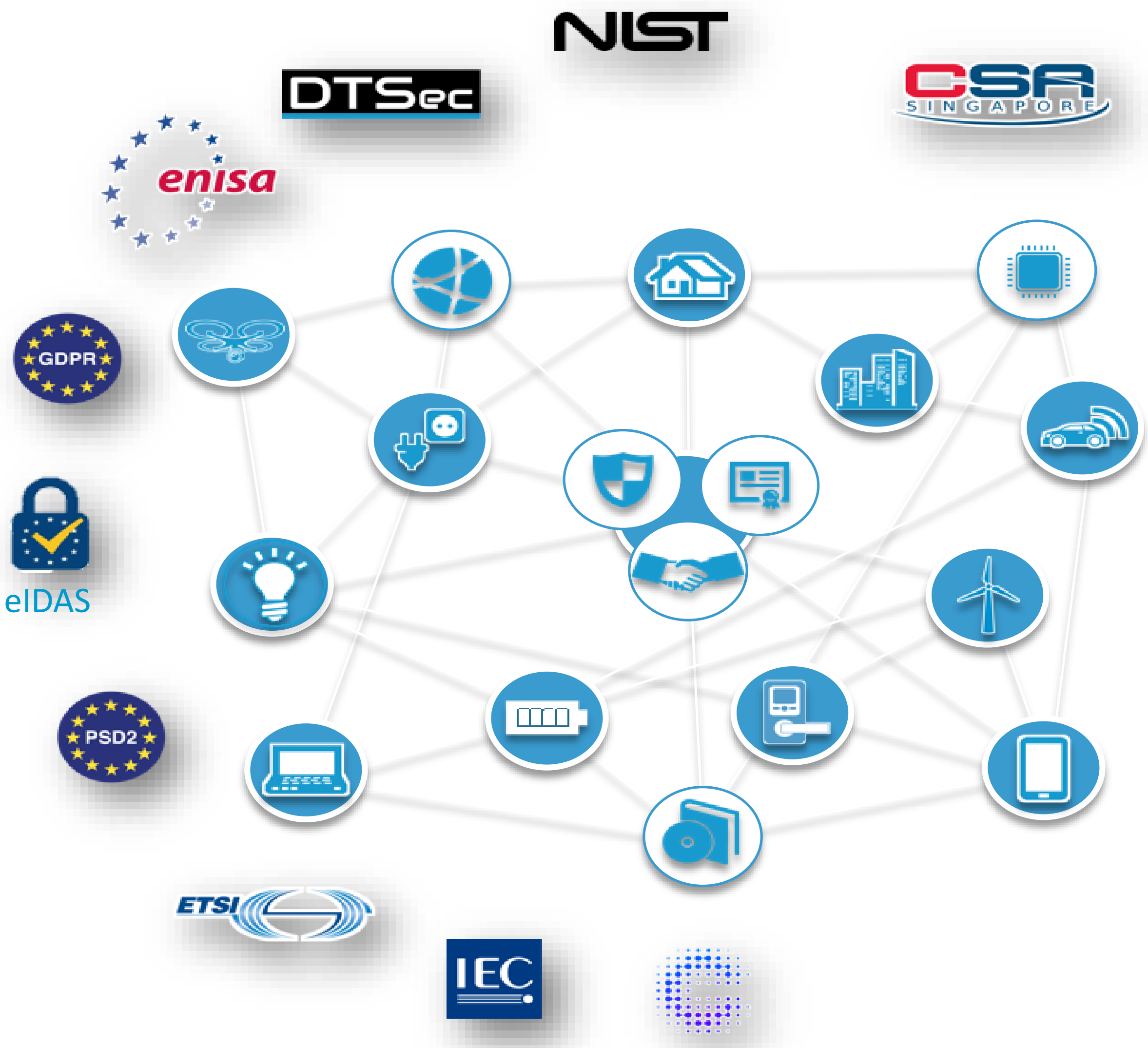
100’s of chip vendors with different RoT



Trust is Essential for Digital Transformation



IoT Developers Face Challenges Such As

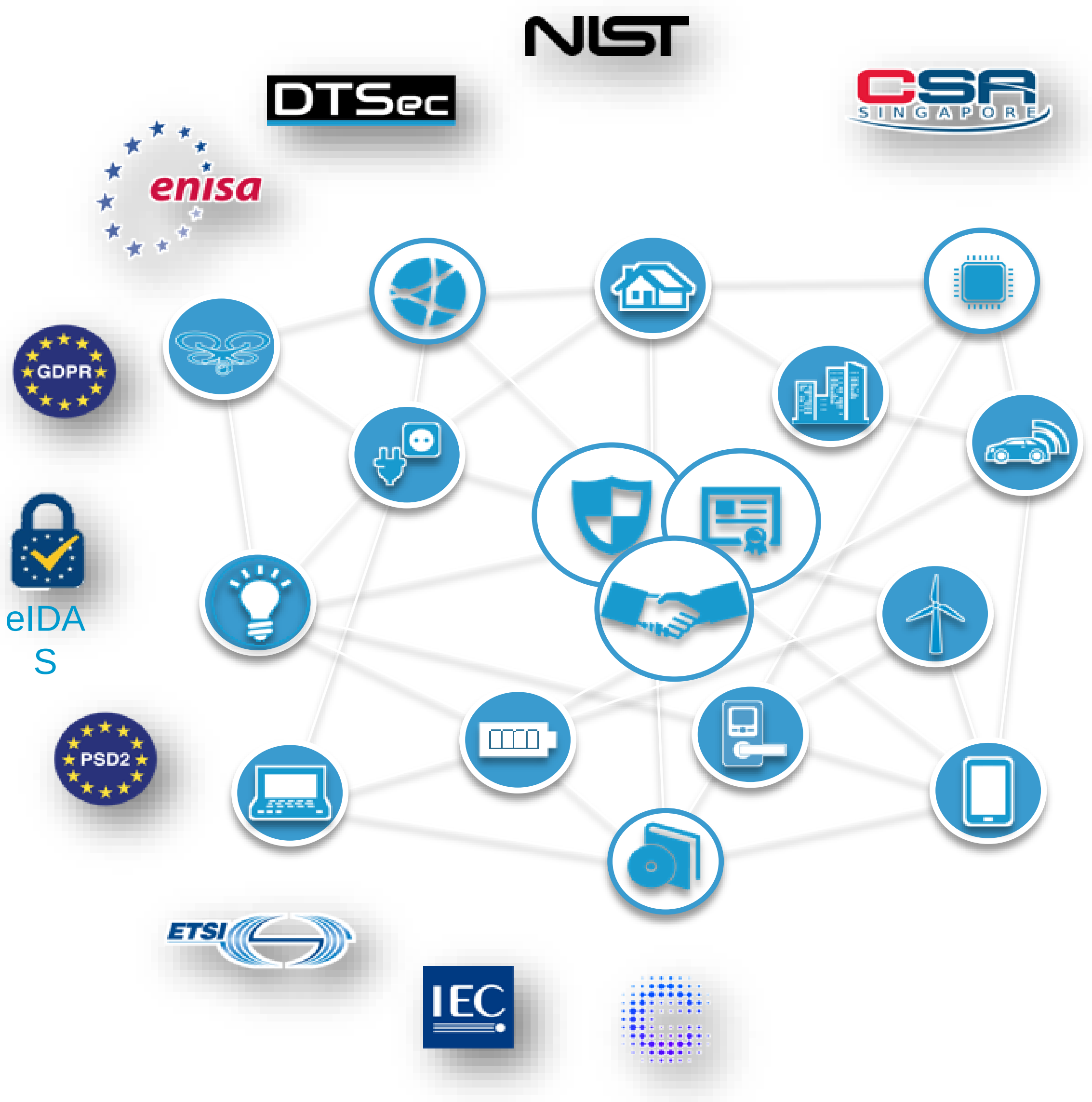


Differentiate by means of proven security functionality showing **accountability**

Protect themselves from **liability** claims and recalls

Meet private and public **compliance** requirements as pre-condition for access to market

Typical Challenges of the IoT Industry



IoT developers are **experts on services and product** execution, not on security.

Hardware and software providers need to differentiate gaining visibility and recognition in the IoT ecosystem.

Lack of IoT product security comes at a price: hundreds of norms and regulations introduced around the world

arm

PSA

Building Trust in IoT

Platform Security Architecture

A complete security offering – openly published. **Independently tested.**

Analyze

Threat models
& security analyses

Architect

Hardware & firmware
architect specifications

Implement

Firmware
source code

Certify

Independently
tested

psacertified™

arm

Platform Security Architecture

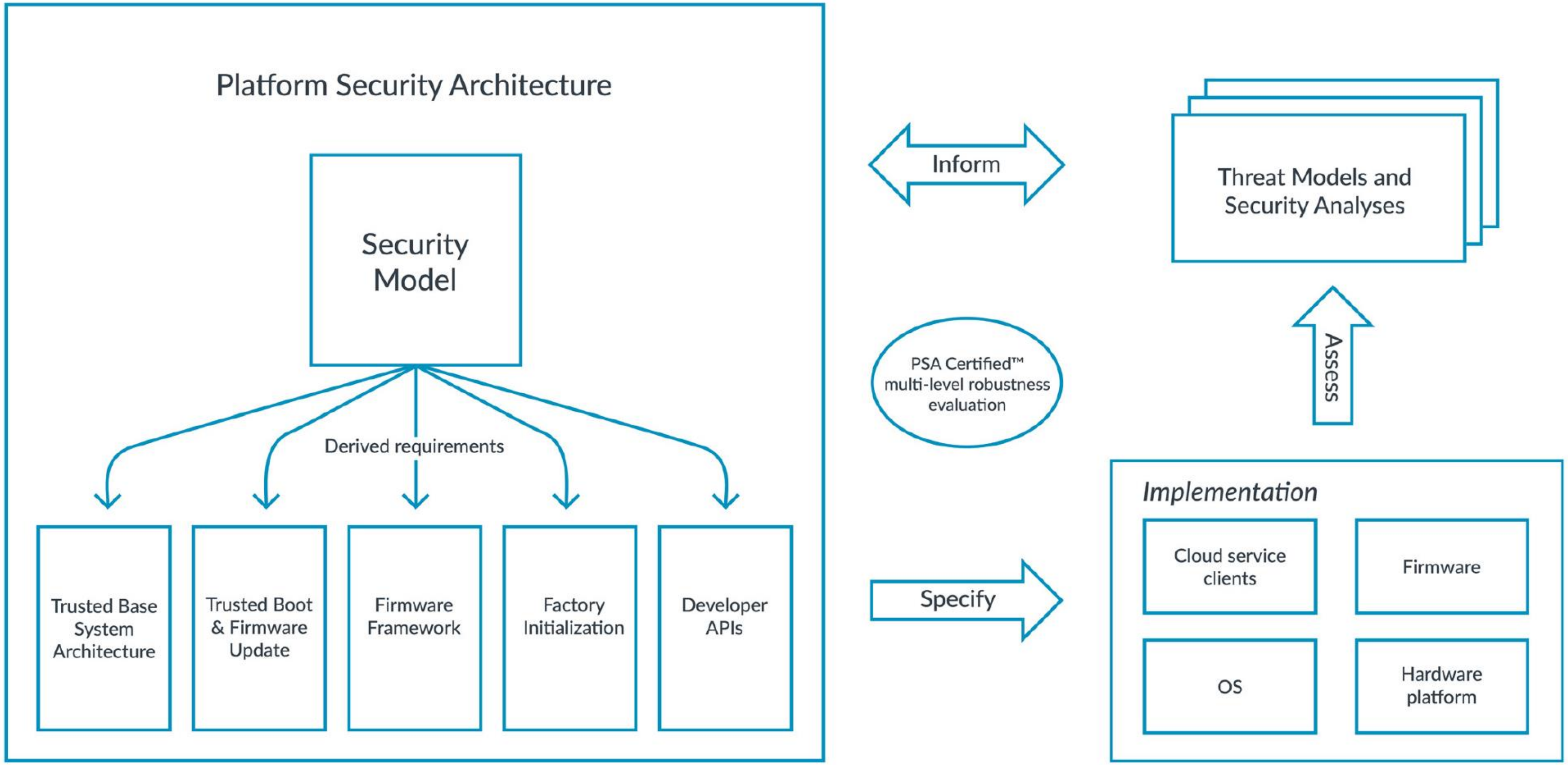
- Analyse with Threat Models and Security Analyses
  - Identify the assets that needs protection
  - All potential threats
  - Scope and Severity of these threats
  - Different Types of attacker and the methods they might use to exploit vulnerabilities
  - Define security objectives and create security requirements.

- Create a System Architecture that meets security requirements and is according to PSA Architecture specification. Adheres to following specifications:
  - PSA Security Model
  - Factory initialization
  - Hardware platform requirements
  - Firmware Framework

- Implement with Trusted Firmware-M
- Certify with PSA Certified and PSA Functional API Specification

Platform Security Architecture

PSA Components



PSA – Example of Analyse

Threat Models and Security Analyses

System description	Smart Metering
Assets	Metering data to be protected in integrity and confidentiality
Threats	Remote software attacks
Security objectives	Strong crypto
Security requirements	Hardware-based keystore

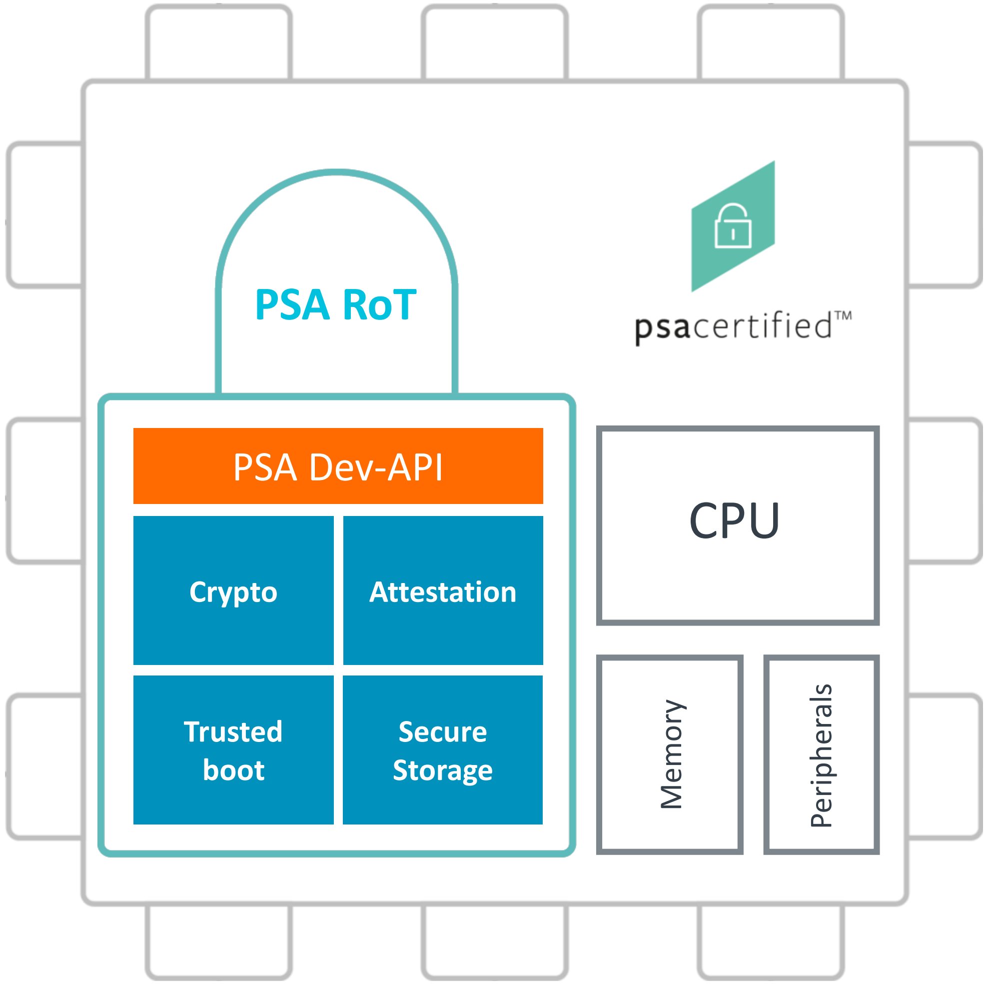
PSA – Root of Trust

Source of integrity and confidentiality

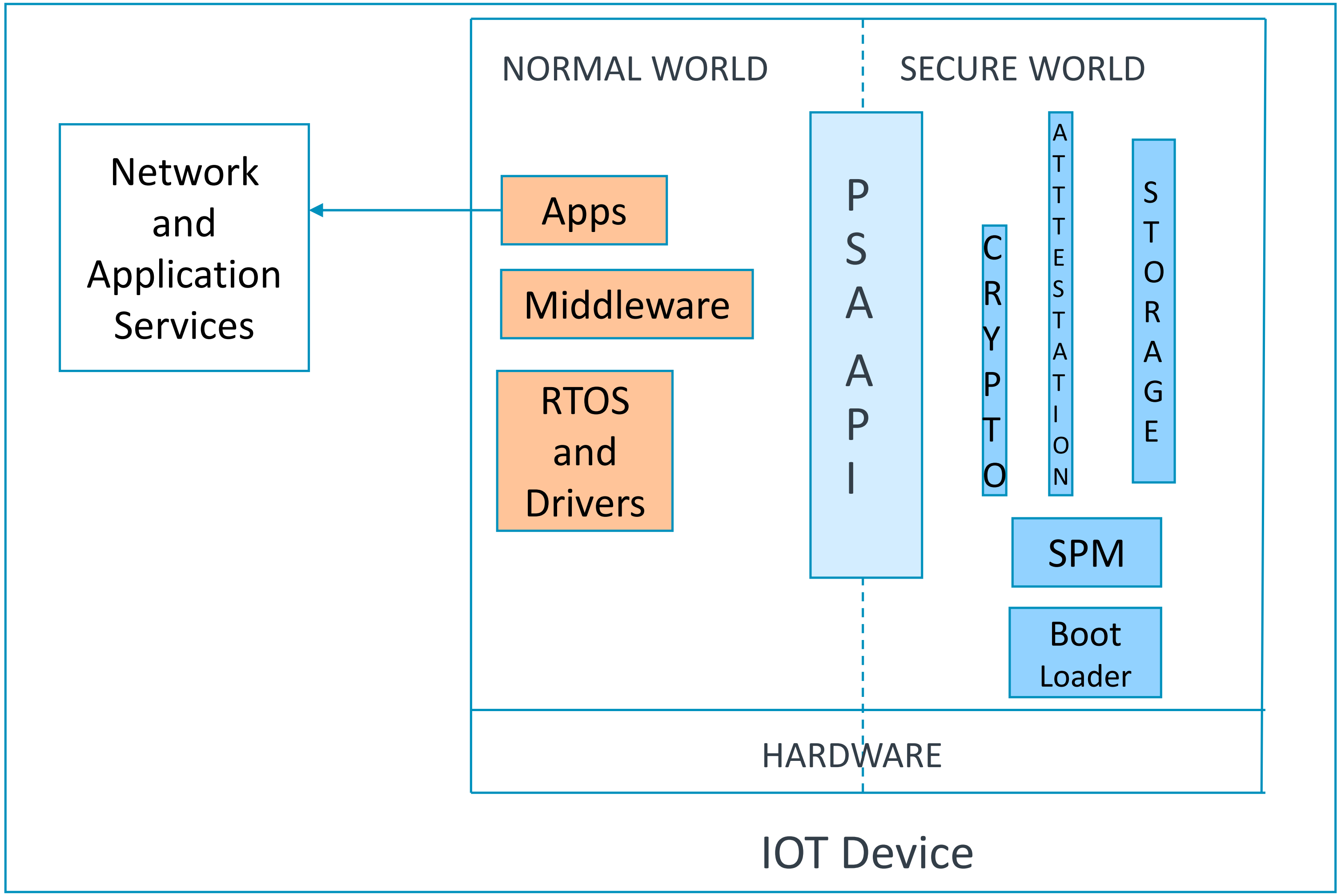
Separates critical security functions in a Secure Processing Environment (SPE) from rest of system

Typically used for secure boot, storing secrets, crypto, attestation, audit logs…

Developed by chip vendors
(for example, by porting Trusted Firmware-M open source software to secure hardware)



PSA Compliant - Software Architecture – IOT Device



PSA Certified – An Overview

Building trust through independent testing

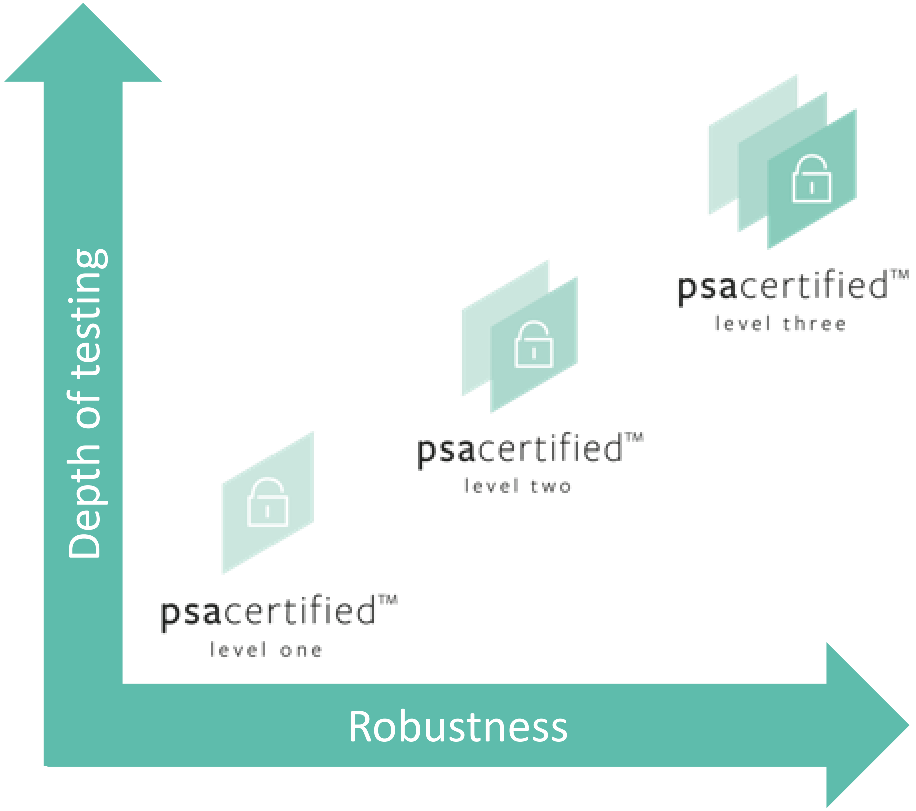
- Dedicated to PSA-RoT enabled chips, devices and platforms
- Builds on IoT threat models, PSA docs, Government best practice & protection profiles
- Simple three-level scheme
- Scalable to IoT ecosystem
- Backed by reputable experts
- Supporting complementary vertical evaluations

PSA Certification - How it Works

- PSA Certified provides three progressive levels of security assurance/robustness: PSA Certified Level 1, 2 and 3
- PSA Functional API Certified enables ecosystem through a consistent high-level interface to the PSA-RoT

psacertified™
functional API

PSA Certified Levels



PSA Certified Levels

PSA Certification Level	Silicon	OS	OEM
L3 Months	✓	Third-party evaluation schemes	
L2 1 month	✓		
L1 1 day	✓	✓	✓

Three assurance levels

Level 1: Document & Declare with lab check

- Security Model goals, government requirements
- IoT threat models – Security Functional Requirements
- Lab check of questionnaire

Level 2: Mid Level assurance/robustness

- Time-limited white box testing
- Protection Profiles, eval methodology and attack methods

Level 3: Substantial

- More extensive attacks

  e.g. Side Channel, perturbation
- Higher assurance

PSA Attestation

Building Trust in IoT

PSA Attestation

- Attestation Tokens are small reports that are produced by a device upon request. These tokens are collection of “Key/Value” pairs known as **claims**.

- Claims can relate to device own pedigree, or health or pretty much anything one wants the device to attest about.

- Collected data can originate from the Root of Trust, or any protected area (secure element, TrustZone, container), or from non-protected areas, in which case they are clearly marked as such.

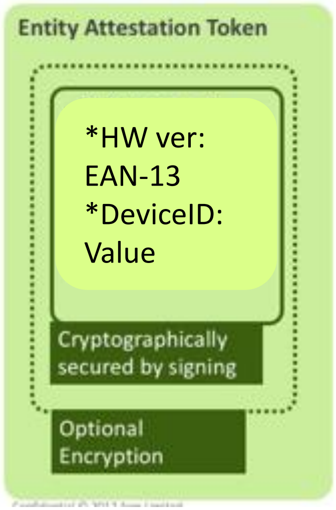
- Tokens are attested because they are signed by devices using device specific unique cryptographic key.

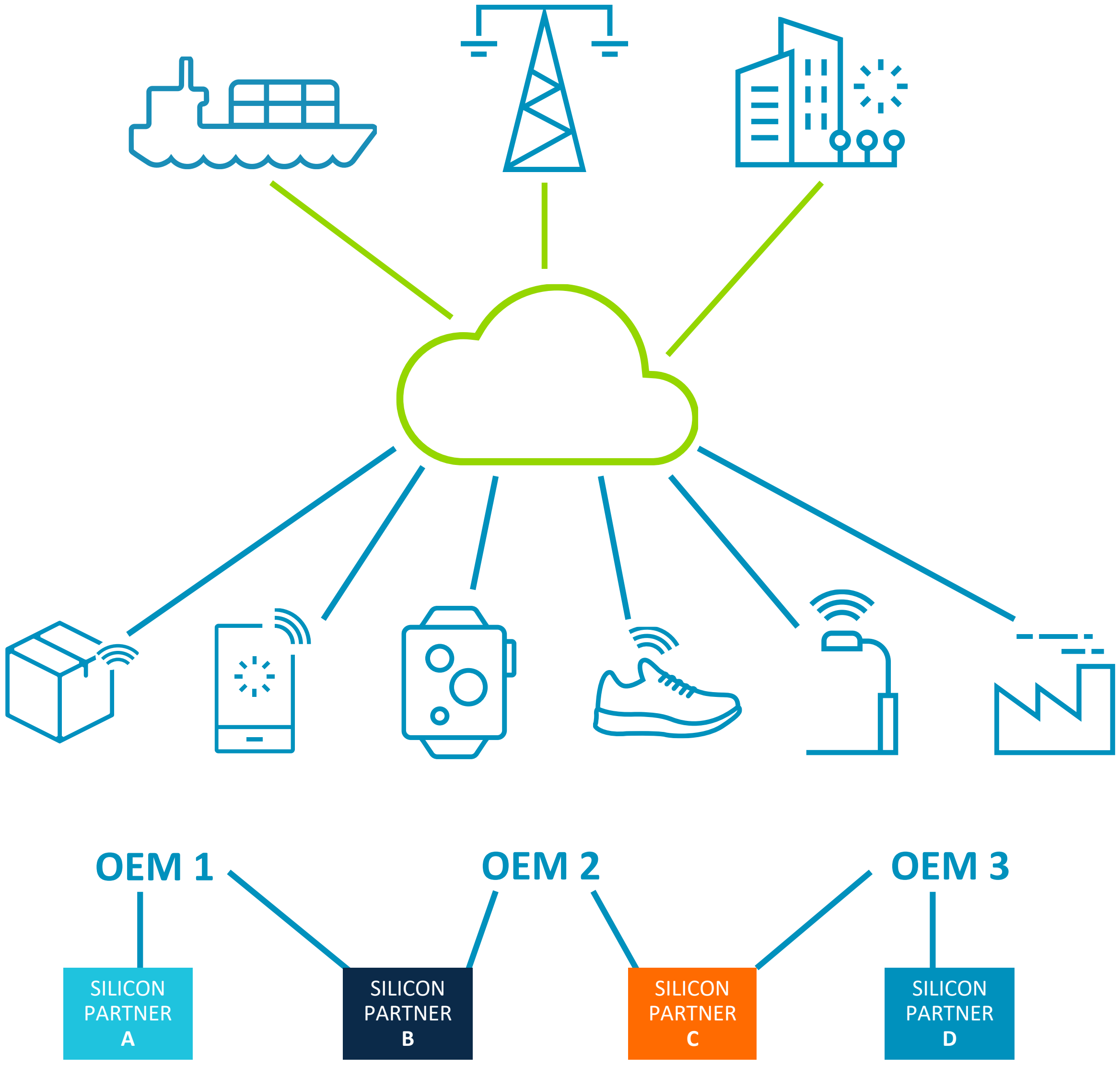
PSA Certified Devices Support Attestation Tokens

EAT is a crypto signed "report card" with useful claims

Can be consumed by higher level attestation schemes

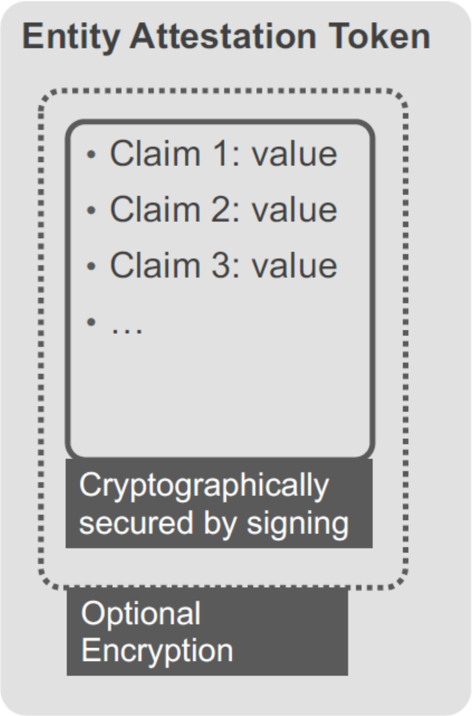
"HW Version" claim used as a chip class ID that can be used to look up PSA Certified level on www.psacertified.org





https://git.trustedfirmware.org/trusted-firmware-m.git/tree/docs/user_guides/services/tfm_attestation_integration_guide.rst

PSA Attestation – Token Encoding



- "Concise Binary Object Representation" (CBOR, http://cbor.io)
- Compact code and data representation for IoT
- Standards based (RFC 7049), quite mature

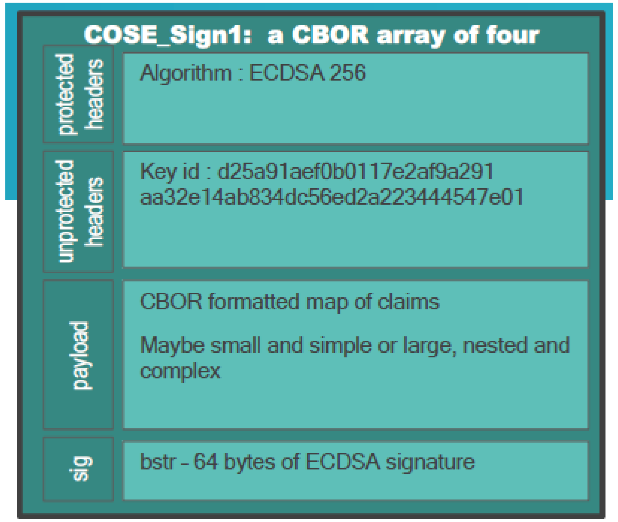
Handles multiple data types, with open source implementations and tools

Data types are simple & powerful – a claim can be a simple integer or have a complex internal structure; allows for optional data

Four Aspects of Standardization

1. General Structuring and Representation of Claims
   - Labeling of claims
   - Optionality of claims
   - Flexible data representation – integers, strings, binary…
2. Meaning of Individual Claims
   - Interoperability between devices and servers from different vendors
3. Signing Format
   - Accommodate different schemes and algorithms
4. Encryption Format (optional)
   - Accommodate different algorithms

PSA Attestation – Token Signing



- CBOR Object Signing and Encryption ("COSE")
- An IoT-oriented format for signing and/or encrypting a payload
- Much simpler and more compact than PKCS #7, CMS and JOSE
- COSE provides structuring of payload, algorithm identification, key identification and signature
- COSE signed tokens are small, self-secured data blobs
- Standard format (RFC 8152) allows use and development of standard / open source tools

PSA Attestation – Utilized Claims

Claim	Mandatory	Description
Auth Challenge	Yes	Input Object from caller. This can be a cryptographic nonce
Instance ID	Yes	Unique identifier of the instance. Hash of Public(IAT) Key
Implementation ID	Yes	Uniquely identifies the underlying immutable RoT
Client Id	Yes	Represents the partition ID of the caller. Signed integer, where –ive ID represent NSPE Call and +ive ID represent SPE call
Security LifeCycle	Yes	Represent current life cycle state of the PSA RoT
Boot Seed	Yes	Represents the random value created at system boot time
Software Components	Yes	A list of Software components that represent all the software loaded by PSA RoT.
No Software Measurements	Yes	Mandatory claim, only if SW Components are missing!
Verification Service Indicator	No	A hint used by RP, to locate a validation service for the token
Profile Definition	No	Name of document that described the profile of the report
Hardware Version	No	Provides metadata linking the token to the GDSII that went to fabrication

arm

PSA Attestation Practical Use Cases

Building Trust in IoT

How IoT world can benefit from PSA Attestation

Smart Home/City

- Example Smart Door locks/sensors
- Smart Lighting

Industrial IOT

- Attest signals monitoring state or measurement of each equipment
- Fleet Management
- Asset tracking

Connected Health Care

- Attest Distress Signals
- Catching Counterfeits during medical surgeries

Others

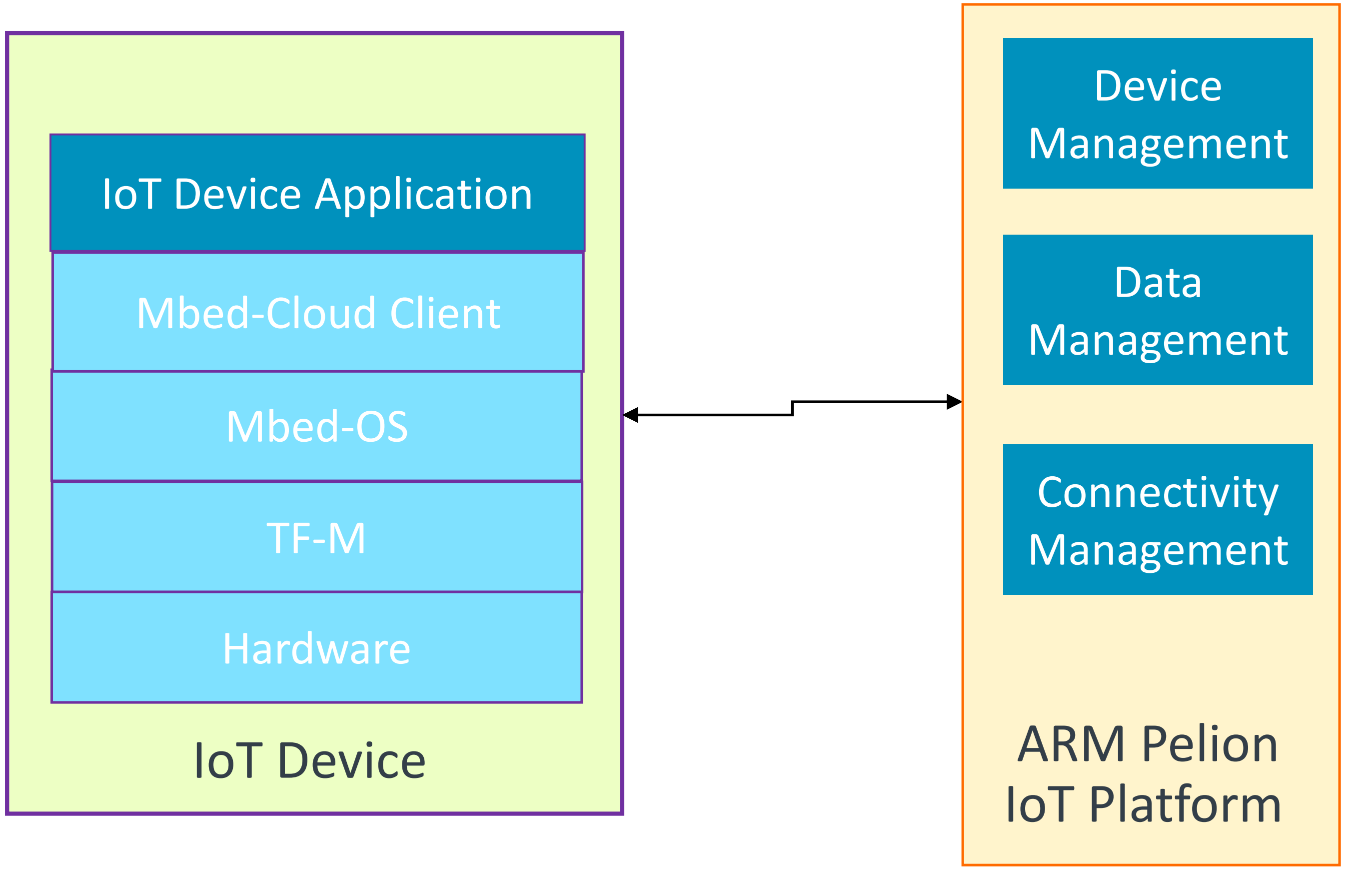
- Military & Défense
- Miscellaneous

PSA Attestation

ARM view of reference IoT Implementation

Building Trust in IoT

PSA Compliant – ARM reference IoT platform

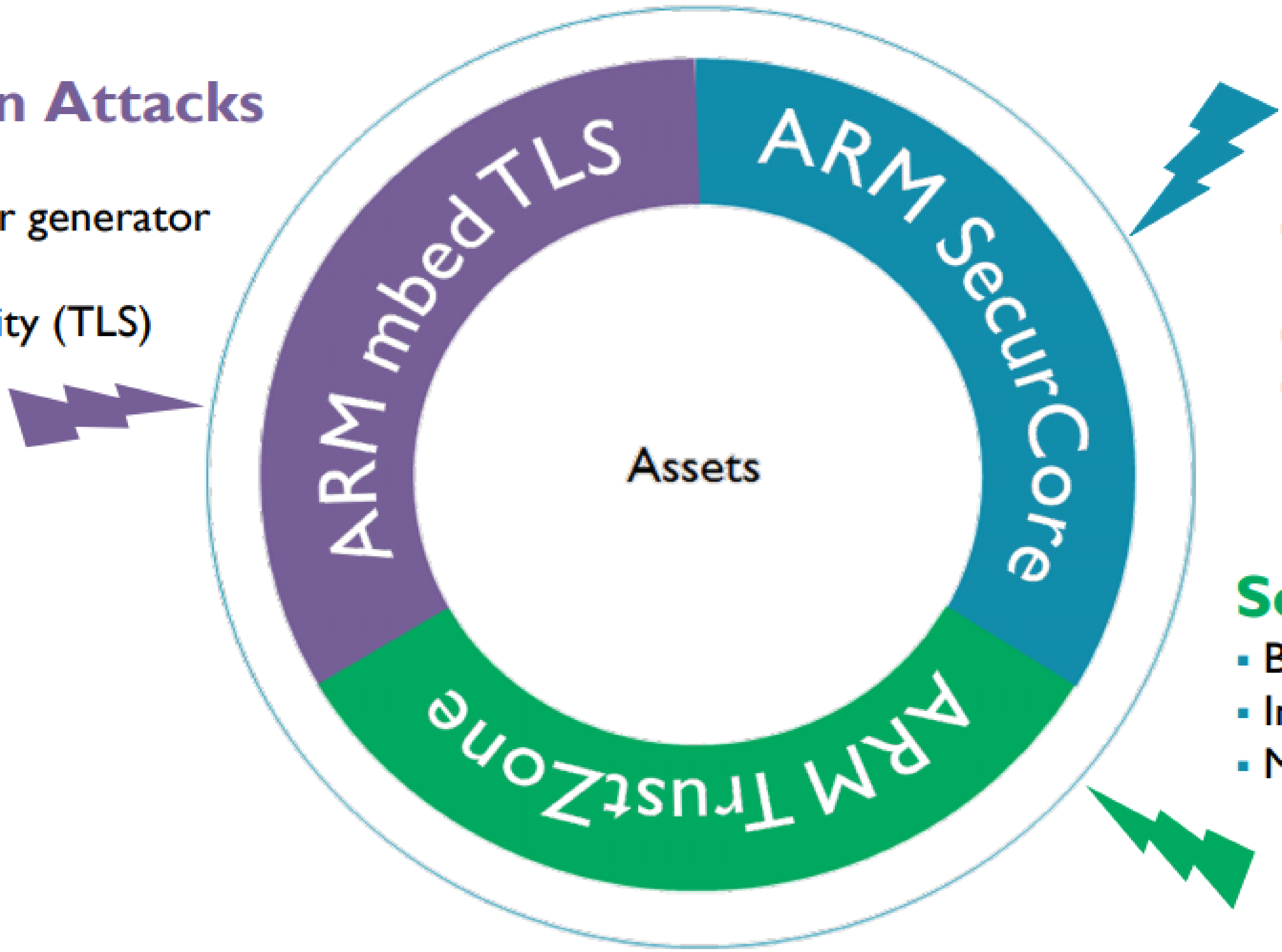


Trusted Firmware

Building Trust in IoT

Attack types

Communication Attacks
- Man in the middle
- Weak random number generator
- Code vulnerabilities
- Transport layer security (TLS)



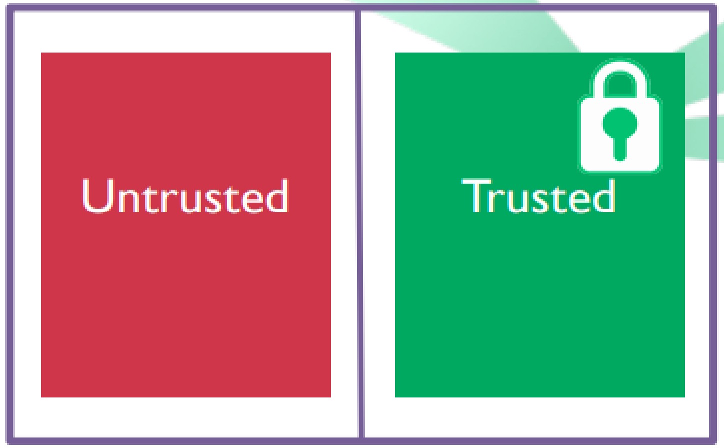
Physical Attacks
- Fault injection: clock or power glitch, alpha ray
- Side channel analysis
- Probing, focused ion beam

Software Attacks
- Buffer overflows
- Interrupts
- Malware

Target: Security for all embedded applications

Root of trust applications - IoT

- Trusted software
- Trusted hardware
- Crypto | Secure system | Secure storage | TRNG*



IP Protection

- Valuable firmware
- Trusted drivers
- Trusted hardware

Sandboxing

- Certified OS / functionality
- Trusted drivers
- Trusted hardware

Standard, affordable

Developer friendly

Ecosystem friendly

Trusted Firmware - https://www.trustedfirmware.org/

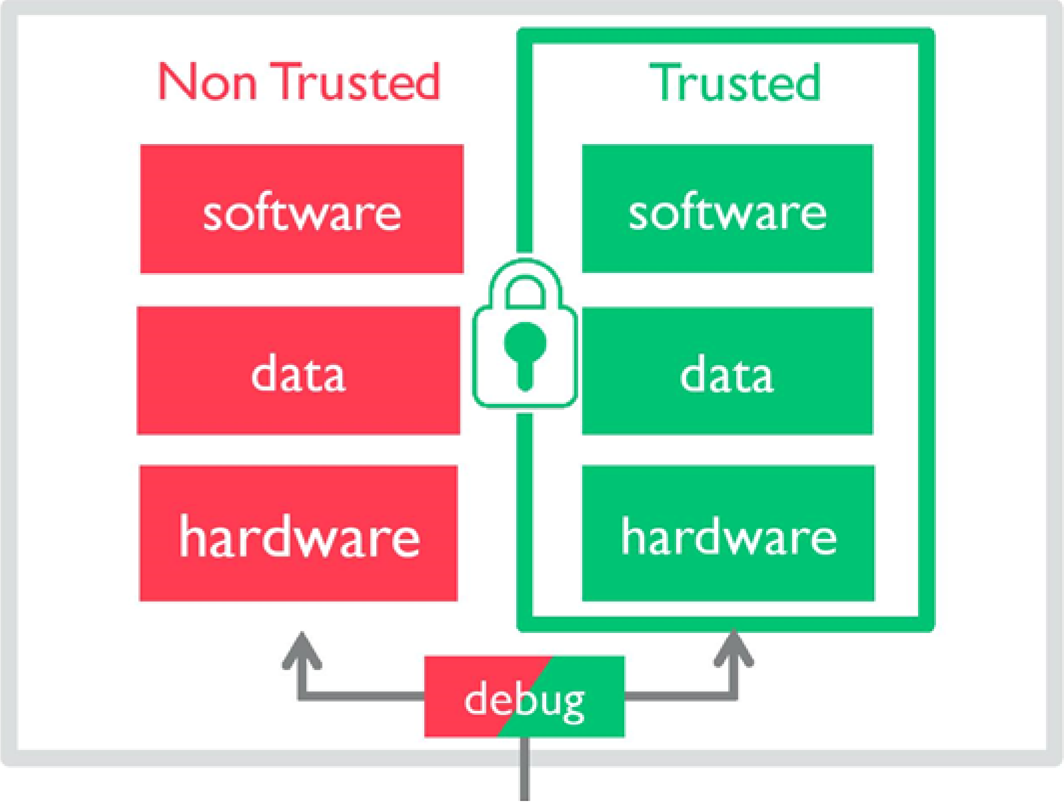
- **Why choose Trusted Firmware?**
  - Trusted Firmware provides a reference implementation of secure world software for Armv8-A and Armv8-M. It provides SoC developers and OEMs with a reference trusted code base complying with the relevant Arm specifications.
  - This firms the foundations of a Trusted Execution Environment (TEE) on application processors, or the Secure Processing Environment (SPE) of microcontrollers.

- **Availability of Trusted Firmware**
  - Support for Armv8-A / Trusted Firmware-A (TF-A)
  - Support for Armv8-M / Trusted Firmware-M (TF-M) and relationship with Platform Security Architecture (PSA)
    - PSA provides a common security foundation for the whole IoT ecosystem. It includes many elements, including architecture specifications and threat models. An important part of PSA is open source firmware. ***This is available in the form of Trusted Firmware-M*** for Arm Cortex-M23 and Arm Cortex-M33 processors, which use Arm **TrustZone** technology.

- **Trusted Firmware-M**
  - Secure Firmware for Arm v7-M and v8-M Systems
  - Provides a Trusted Execution Environment (TEE) for Arm v7-M and v8-M devices. For v8-M devices, it leverages, Arm TrustZone technology. It is the reference implementation of Platform Security Architecture (PSA). PSA is a recipe for building secure connected devices from analysis to implementation. PSA consists of four elements - Threat models and Security Analyses, Architecture Specifications, Open Source Reference Implementation (TF-M) and Certify.
  - TF-M implements PSA Specifications and APIs that can be found here.
    - https://developer.arm.com/architectures/security-architectures/platform-security-architecture

TrustZone: IoT Security Foundation

Isolates trusted software, data and hardware



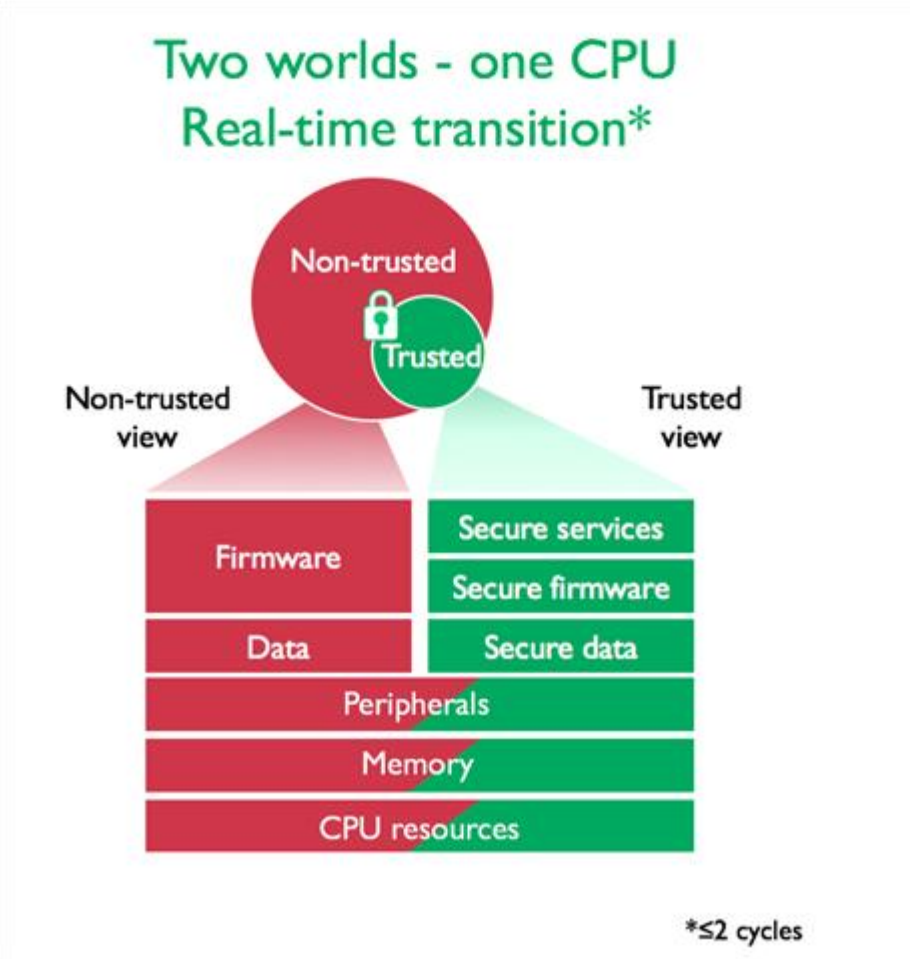
Enables device integrity and system recovery

- Example use cases:
  - Protection of critical assets
  - Safe crypto implementations
  - Secure remote firmware update
  - Firmware IP protection
  - Secure debug

arm

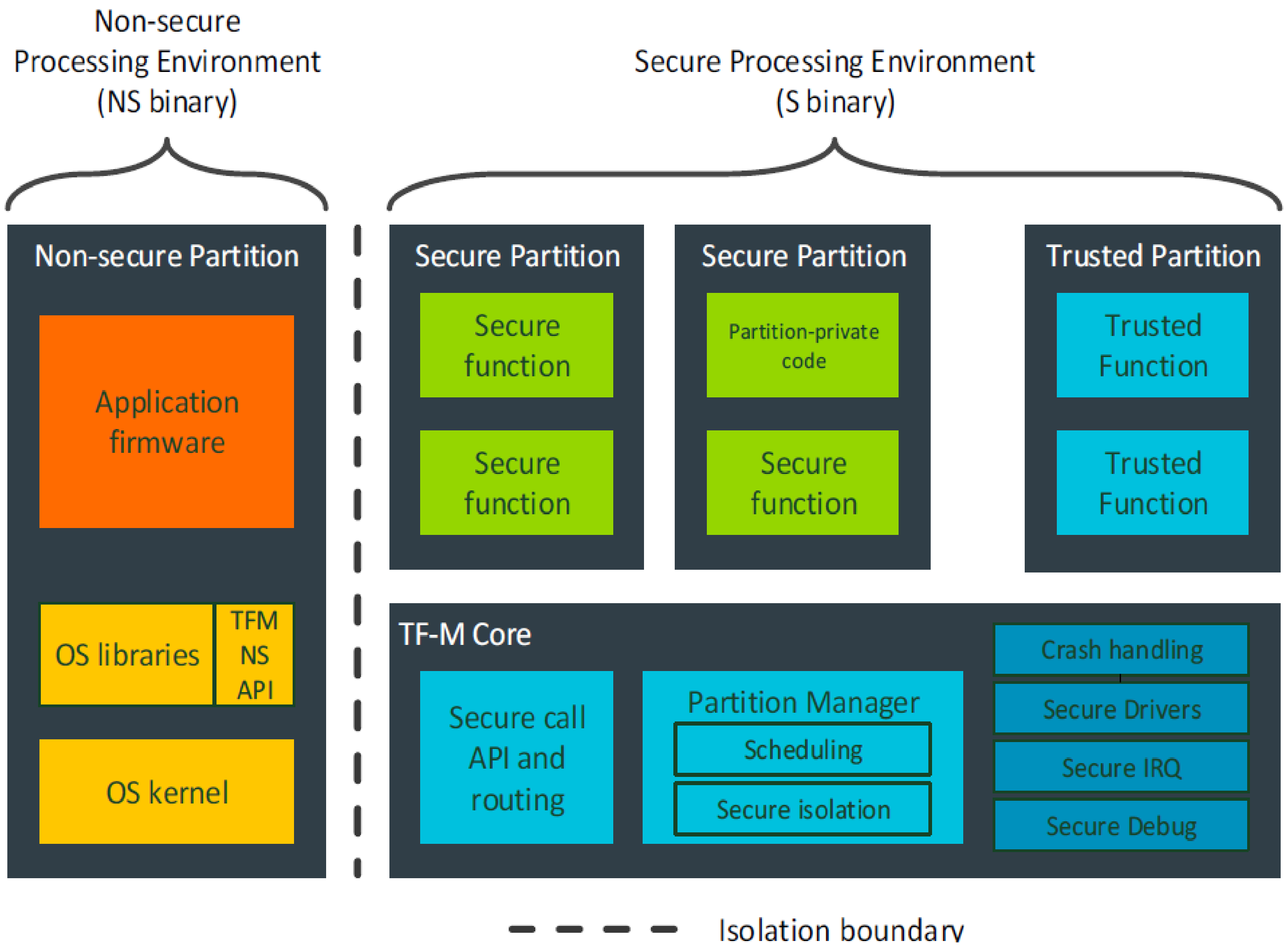
TrustZone Technology for Armv8-M

- The Armv8-M architecture extends TrustZone technology to Cortex-M based systems,
- TrustZone reduces the potential for attack by isolating the critical security firmware and private information, such as secure boot, firmware update, and keys, from the rest of the application.
- TrustZone technology offers an efficient, system-wide approach to security with ***hardware-enforced isolation*** built into the CPU
- Running two domains side-by-side and sharing resources per set configuration.



TF-M Framework

- Secure bootloader
- Secure system init
- Secure Partition Management (SPM)
- Secure function call routing
- Isolation within SPE
- Trusted services, functions
- NSPE API
- Build environment
- Test suite
- ...



Introduction to TrustZone for Armv8-M

Armv8-M architecture includes optional Security Extension

- Branded as Arm TrustZone for Armv8-M

Similar in concept to TrustZone for Armv8-A

- Implementation is optimized for microcontrollers

System may be partitioned between secure and non-secure software

Secure software is highly trusted

- Has access to more system resources
- Protected from access by non-trusted code

To protect the secure software the security extensions provide:

- Isolated Secure memory for code and data
- Secure execution state to run Secure code

ARMv8-M Security Extension

Provides two security domains for code to run in

- Secure and Non-secure
- PEs without the Security Extension behave as though reset into Non-secure

Hardware assists in hiding Secure state from Non-secure code / debuggers

- Debugger can be blocked from accessing PE while Secure code is running
- Hardware pushes and clears registers if non-secure code interrupts secure code
- Stack limit registers added to assist in attack mitigation

Duplicate resources to enable software and hardware isolation

- For example, dedicated stack pointers, SysTick timers and MPUs for each domain
- Non-secure code only able to access non-secure controls

Ability to expose PE's security to system

- ARM's AXI and AMBA5 AHB5 support propagation of NS attribute

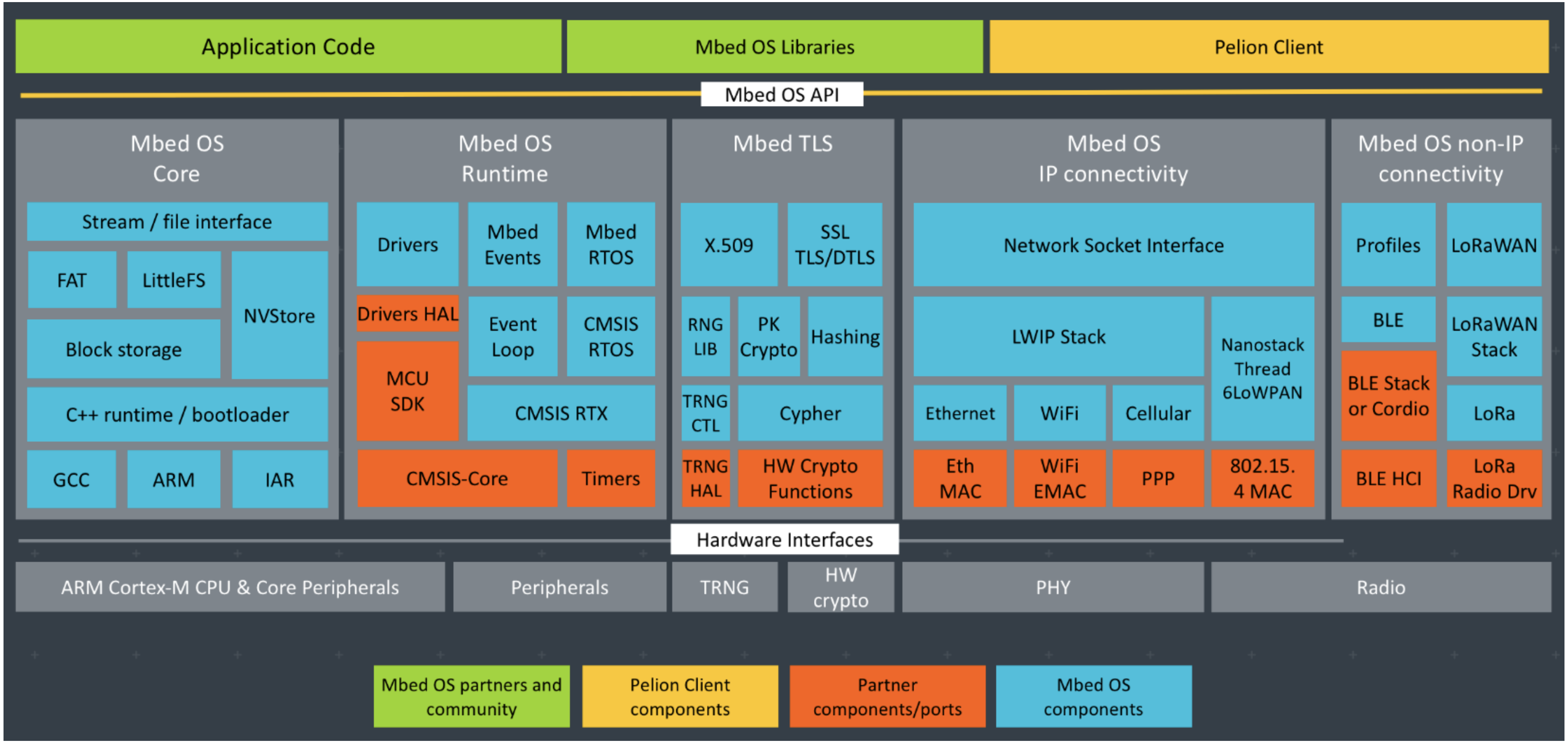
Mbed OS

Building Trust in IoT

Mbed-OS

- Arm Mbed OS is a free, open source embedded operating system that includes all the necessary features to facilitate the development of IOT Connected Products.
- Mbed OS provides an abstraction layer for the microcontroller it runs on
- Mbed OS modules include
  - Standard (PSA Compliant) based security and connectivity stacks
  - RTOS Kernel
  - Middleware for storage and Networking
  - Drivers for sensors and I/O Devices
  - Remote Device Management
- Mbed OS Features
  - Modular, Necessary libraries are included automatically on your device
  - Secure: MultiLayer security helps to protect your IoT solution, from isolated security domains through to Mbed TLS for secure communications
  - Connected: Wide range of communication options with drivers for BLE, Ethernet, WiFi, 6LoWPAN

MBED OS Architecture



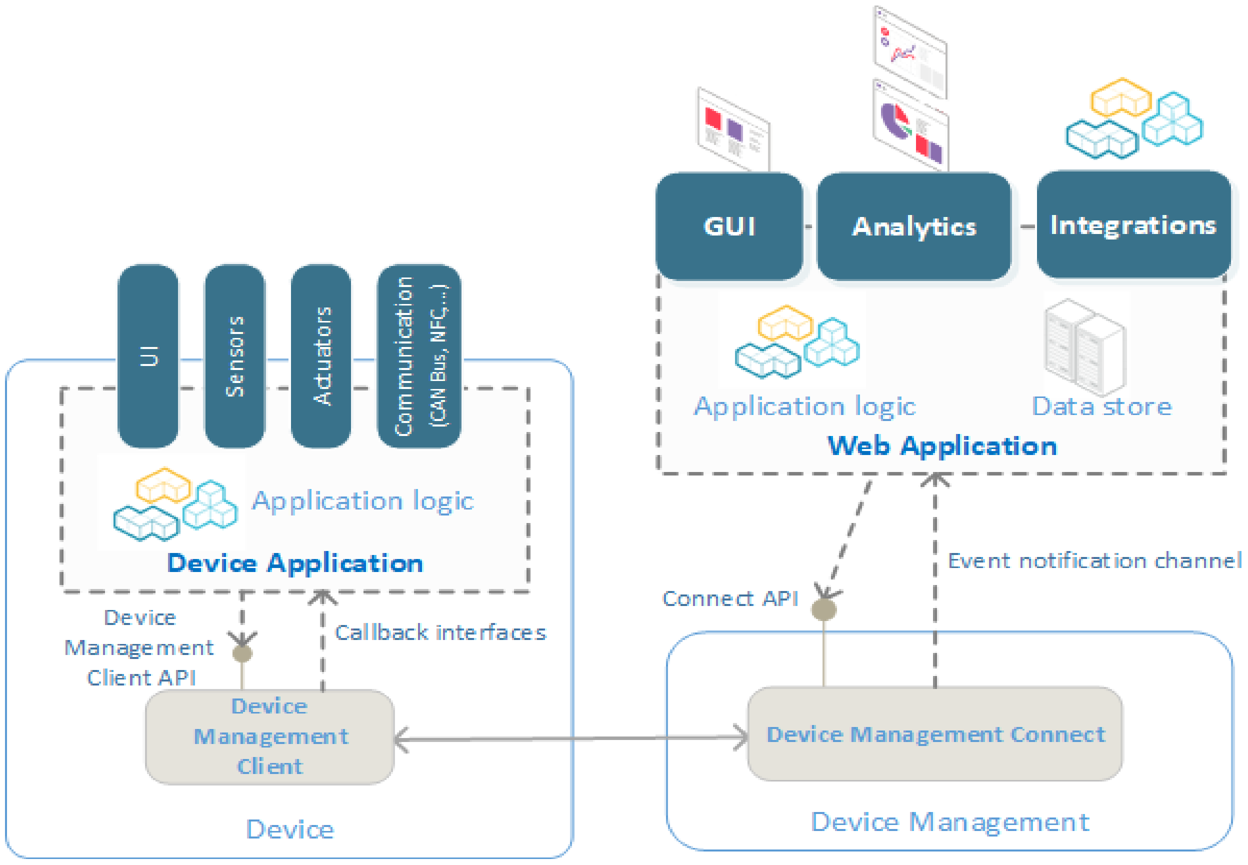
Mbed Cloud Client

Building Trust in IoT

Mbed Cloud Client

- Mbed Cloud Client, (Device management client):
  - Connection Client
  - Update client
  - Provision client
- Mbed Cloud connect service is a secure and energy efficient communication service connecting devices to Device Management.
- Standards-based protocols (OMA LwM2M, CoAP, and TLS/DTLS), designed specifically for low-power devices.

Device Management Connect



Device Management Connect system diagram

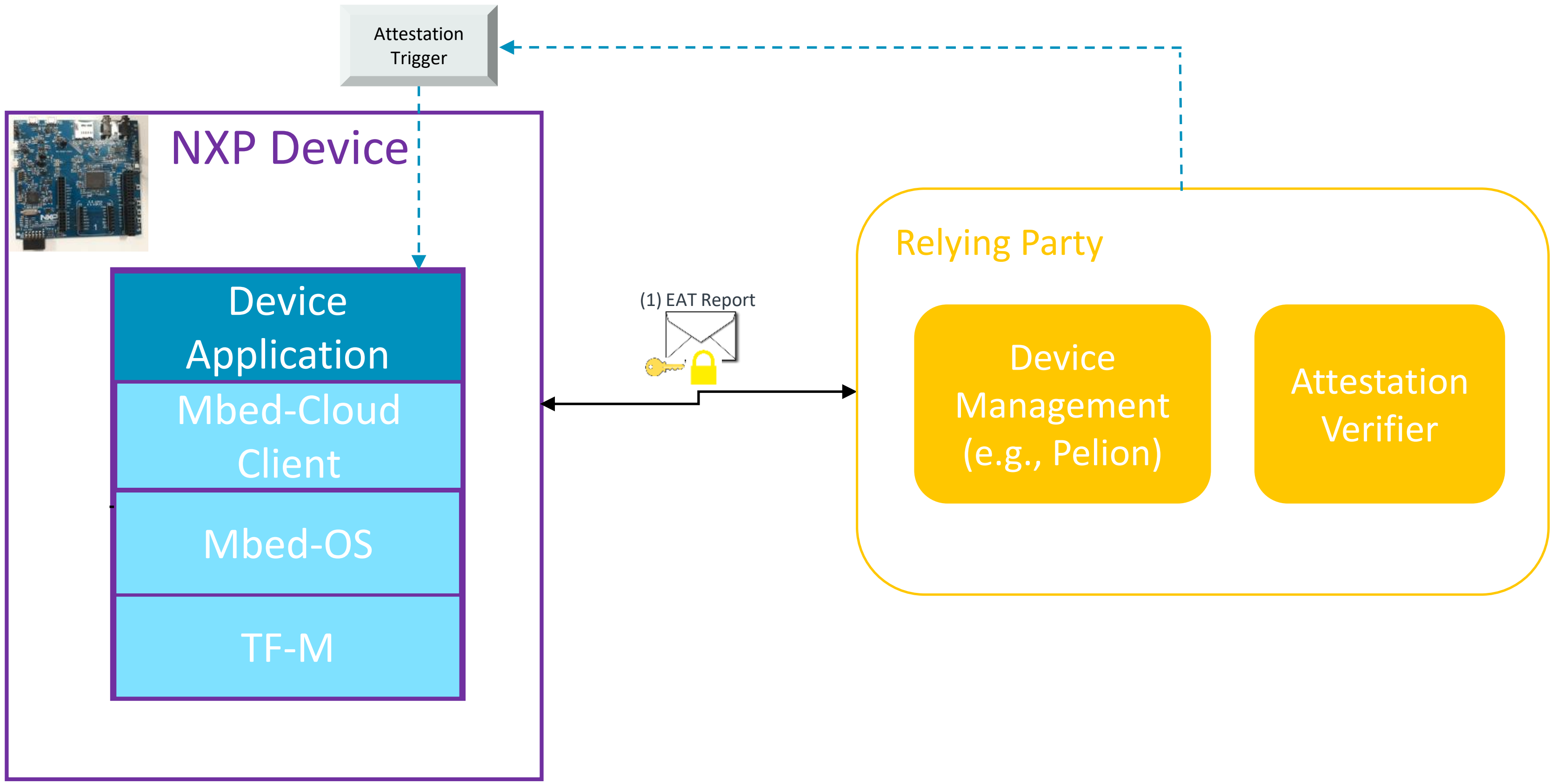
- Device Management uses LWM2M
- Communication using CoAP
- Web application connectivity to the Device Management
- End to End Channel Security

  DTLS/TLS: for the connection between the device and the server.

  HTTPS: for the connection between web applications and the Device Management REST APIs.

- Optimizations for IoT devices.

Attestation Token Validation – Sample Exercise



Thank You
Danke
Merci
谢谢
ありがとう
Gracias
Kiitos
감사합니다
धन्यवाद
شكرًا
ধন্যবাদ
תודה