Sensors Expo Workshop

Securing the IoT and Embedded Systems with TCG and Embedded Computing Design

Workshop Speakers

- **Brandon Lewis**, Technology Editor, Embedded Computing Design, with a focus on IoT Design, Industrial AI & Machine Learning, and Automotive Embedded Systems and IoT Insider columnist
- **Josef Kohn**, Regional Marketing Manager, Infineon
- **Dennis Mattoon**, Principal Software Development Engineer, Microsoft
- **Lee Wilson**, Product Development Engineer, Onboard Security

Sensors Meeting Security

How TCG and Trusted Computing Can Help Secure Sensor Innovations

Josef Kohn, Infineon

THE SENSOR WORLD IS PHENOMENALLY INNOVATIVE

..but: *How do I secure my products? - How much security is needed?*

How to efficiently implement good security? - What to look out for?

Security is complex, can be confusing and overwhelming; how to navigate and find guidance, best practices, proven solutions?

The **Trusted Computing Group** is offering help to secure your innovations!!
Let‘s learn more about it and have a closer look at

- the core security elements of Trusted Computing: TPM, TSS, DICE
- available resources and support for you

THE INTERSECTION OF SENSORS AND SECURITY

- What are the most important security assets to be protected?
- A quick look at the actual threatscape and how to prepare for the future
- Properties of good product security design and how trusted computing can be designed into products

The many daily news about IoT Hacks are a denial-of-service attack to our attention

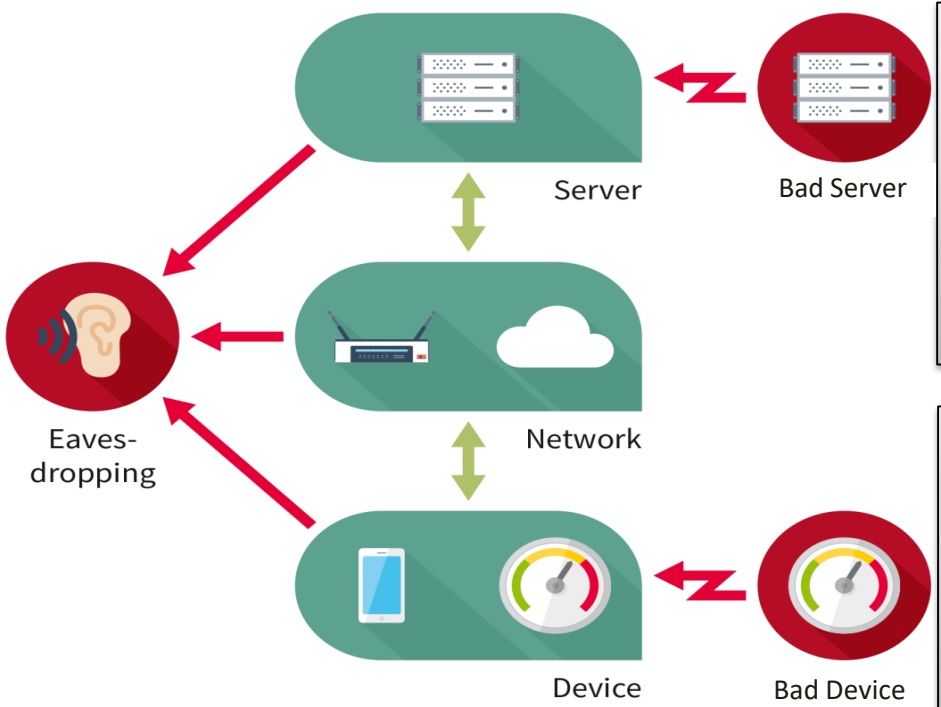
The Threatscape for Connected Devices

A sobering reality:

- Some spectres are no longer a ghost
- Botnet infections of IoT devices: reality, too
- Cryptomining on IoT devices: very lucrative
- Poorly protected edge nodes are entry vectors to high-value targets and sophisticated attacks

In Connected Systems, Each Layer is a Target for Attacks

An **Eavesdropper** listening in on data or commands can reveal confidential information about the operation of the infrastructure.



A **Bad Server** sending incorrect commands can be used to trigger unplanned events, to send some physical resource (water, oil, electricity, etc.) to an unplanned destination, and so forth.

A **Bad Device** injecting fake measurements can disrupt the control processes and cause them to react inappropriately or dangerously, or can be used to mask physical attacks.

Primary Classes of Security Assets Under Attack

- Physical safety when connected to machines
- Confidentiality and privacy of user data
- Value of transactions
- Trustworthiness and availability of services

Risks, damages can be existential - directly and indirectly!

...not just to users and owners but also to the makers

Trusted Computing:
Important Properties of Good Product Security Design

- Defense in Depth: Strong Isolation of Security Boundaries
- Small Surface of Attack
- HW-Based Root of Trust
  - Trusted Boot: Securely Measure, Store, Report Platform Integrity Metrics
  - Strong Attestation (local and remote)
- Protected Capabilities:
  Secure KeyGen + Storage, Crypto-Primitives and –Services
- Evaluated, Certified Security (CC, FIPS)
- Secure Firmware Update Capability

More: see Trusted Platform Module details

For More Than 15 Years, the TCG Has Helped the Embedded World ecure Infrastructure, Platforms and Use Cases

With global Standards and products for Trusted Computing:

- **Trusted Platform Module (TPM2.0)**
  ISO/IEC 11889-1:2015
- **TSS – Trusted Software Stack**
- **DICE**

- Publicly available specifications
- Protection Profiles

With Resources and Support for Developers

- Developer Community
- Guidances to secure different types of devices
- Webcasts
- Workshops
- Compliance Tests
- Open Source Software
- Tiered Memberships w. low entry fees

In nearly 20 work groups, experts are developing standardized security solutions

Cloud Security
DICE Architectures
Embedded Systems
Industrial
Infrastructure
Internet of Things
Mobile
Network Equipment
Certification
PC Client
Regional Forums
Server
Software Stack
Storage
Trusted Network Communications
Trusted Platforms
Virtualized Platforms

The Trusted Platform Module 2.0: multiple valuable Security and Commercial Benefits

A Common Criteria EAL4+ and FIPS 140-2 certified security Module

Benefits: SECURITY

- **Independently evaluated** and certified security
- **High Resistance** against
  - Fault attacks
  - Side-Channel and semi-invasive attacks
  - Invasive attacks (tamper resistance)
- Engineering **risk mitigation**
  - Leveraging TPM functions **reduces** the need to implement these on the main application uC and the **risk to inject security flaws** there

Trusted Platform Module:
Securely implemented + certified *Toolbox*

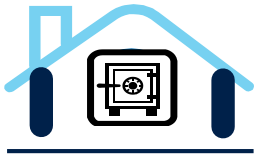
Hash: SHA	Endorsement Hierarchy
Asymmetric: RSA, ECC	Platform Hierarchy
Symmetric: AES	Storage Hierarchy
HMAC	NULL Hierarchy
Opt.: SM1/2/3 + Agile	Monotonic Counters
KeyGen,Encr/Decr/Sign/Ver	Tick Counters/Clock
CRTM/R, PCRs	Dict. Attack Lockout
Sealing, Binding	Enhanced Auth. Policies
Protected NV Storage	Encrypted Sessions
TPM Firmware Update	Audit Sessions
RNG/Entropy	Self-Test
ECert,Keys, Primary Seeds	GPIO

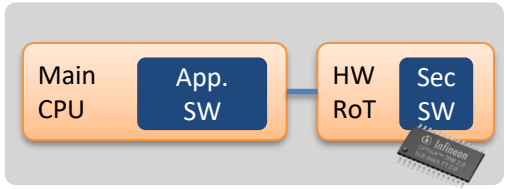
Completely Standardized and preprogrammed functional module with many functions (Trusted Computing Group, TCG TPM v.20)

Benefits: COMMERCIAL

- **Compliance** tested for functionality and security
- 15 years of **proven** and matured technology
- Vendor agnostic, **interoperability**
- COTS: high volumes, **cost efficient**
- High Reuse of TPM-aware SW: **engineering efficiency**

Comparing Hardware and Software Trust Anchors: same functionality – major differences in security





Crypto functionality	(✓)	✓
Tamper resistant	STOP	✓
Attack resistant	STOP	✓
Security certified	STOP	✓
Secure manufacturing	STOP	✓
Secured shipment	STOP	✓

For More Than 15 Years, the TCG Has Helped the Embedded World ecure Infrastructure, Platforms and Use Cases

With global Standards and Products for Trusted Computing:

- ✓ **Trusted Platform Module (TPM2.0)**
  ISO/IEC 11889-1:2015
- ☐ **TSS – Trusted Software Stack**
- ☐ **DICE**

- Publicly available specifications
- Protection Profiles

With Resources and Support for Developers

- Developer community, develop.trustedcomputinggroup.org
- Guidance docs to secure different types of devices
- Webcasts
- Workshops
- Compliance tests & certification
- Open source software support
- Tiered Memberships w. low entry fees

Trusted Computing Value, Use Cases and Software

Putting the TPM to Work

Lee Wilson, OnBoard Security

How Has Cybersecurity developed, deployed and evolved over the last 50 years?

Hardware Based Security for High End Systems

- *Financial Institutions, Critical Industries, Large DataCenters, etc.*
- *Heavily regulated security*
- *Very expensive HSMs required – not optional*

All the Other (More Cost Sensitive) Platforms in the World

- *Store keys and certificates in file systems.*
- *Hope firewalls, etc. (software only solutions) do the job.*
- *Cross your fingers.*

Per the DoD Orange Book - A TCB, system measurement, etc. need to be added to the model

Hardware Based Security for All (Trusted Computing!

- *No more critical keys stored in file systems*
- *Strong device identity for remote management, code update, etc.*
- *Much better attack detection – stop rootkits and bootkits.*
- *Add hardware protected software measurement and TCB creation for total solution*

Netting it out: Computers did not start out with good cybersecurity. It has evolved along the way and now the needed model is available to all.

Firm Requirements Now… Focus on Bottom Up Platform Security

Note:
These charts were presented at the Winter 2012 TCG Members Meeting by the US Govt. The US Govt. is developing the SP800 standards to codify this and they will extend these efforts to server firmware and systems mgmt.

Motivation

- Major malware outbreaks spread via OS vulnerabilities (e.g., Blaster, Nimda).
- Targets have moved to application layer.
  - In 2009, 49% of web-based attacks targeted PDF vulnerabilities [Sym10].
- Future attacks could move down the stack to firmware.

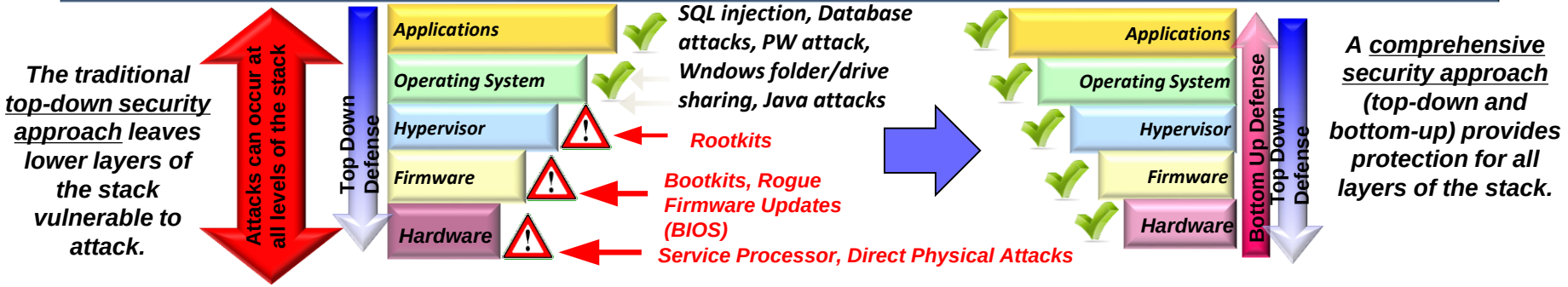
[Sym10] *Symantec Global Internet Security Threat Report- Trends for 2009*. April 2010

BIOS Concerns

- 1998 – Chernobyl (CIH) Virus
- 2004 – NiBiTor (NVIDIA BIOS Editor)
- 2006 – ACPI BIOS Rootkit
- 2006 – Persistent BIOS Infection
- 2007 – Hacking the Extensible Firmware Interface
- 2008 – UEFI Hypervisors
- 2009 – Deactivate the Rootkit (Computrace)
- 2009 – Attacking Intel BIOS
- 2011 – Mebromi

Security Holes Missed by Software-Only Security – Problem Solved by Trusted Computing

Antivirus programs, Firewalls, Compliance Management, etc. Provide "Top Down" Security.



Trusted Computing Hardware/Firmware/Software Required for Bottom Up Defense

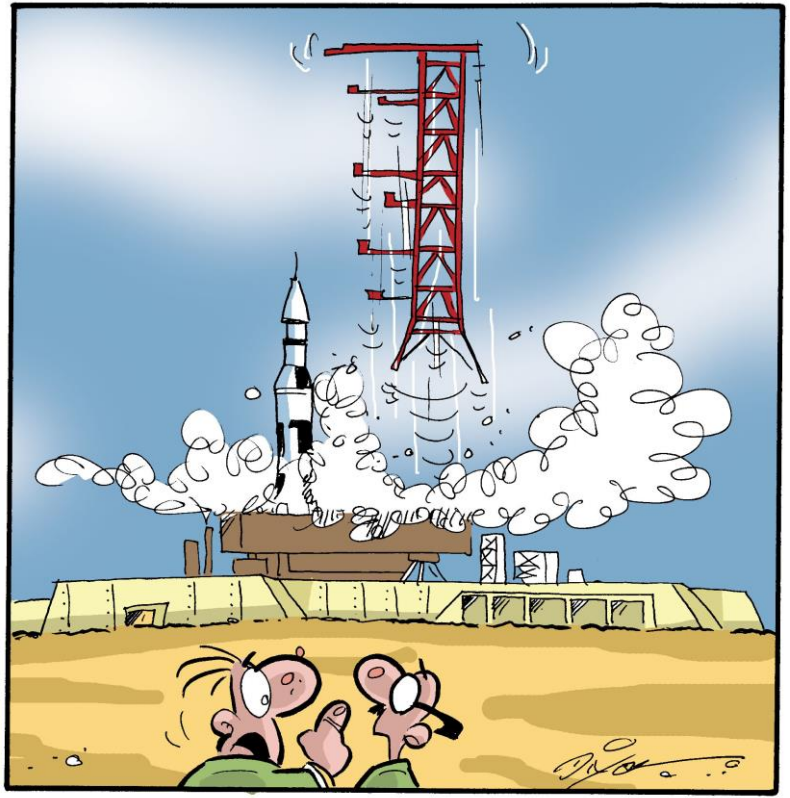
- TPM v2.0/TSS v2.0 provide support infrastructure for trusted computing
- Trusted computing-enabled firmware building a "transitive trust chain" and establishing systems measurements off of a CRTM (Core Root of Trust for Measurement) launch bottom-up security.
- OS'es, RTOS'es,firmware, trusted applications… seal secrets designed to protect overall platform security to the TPM PCRs (Programmable Configuration Registers) and use the TPM as an HSM where appropriate.

Top Down Defense

- Build a defensible barrier at the system's external attack surface. Top down defense is typically "software only" and launches later in the boot up of a system.
- Keep sensitive/secret stuff in.
- Keep bad stuff out

Bottom Up Defense

- Insure the “right” code (correct and trusted) launches in the system.
- Measure the code so it can be checked by a trusted third party and also use those measurements to keep secrets safe (enhanced authorization).
- Provide a hardware vault and key wrapping to protect essential system secrets (keys)
- Build a Trusted Computing Base (TCB) - a self-defending kernel of code from which a system can recover if attacked.



“Now I’m pretty sure that’s not supposed to happen!”

CartoonStock.com

The Cost of an IIOT Security Failure

Case Study: The Stuxnet Attack

https://en.wikipedia.org/wiki/Stuxnet

- “Stuxnet reportedly compromised Iranian PLCs, collecting information on industrial systems and causing the fast-spinning centrifuges to tear themselves apart.”
- “Stuxnet’s design and architecture are not domain-specific and it could be tailored as a platform for attacking modern supervisory control and data acquisition (SCADA) and PLC systems (e.g., in factory assembly lines or power plants), the majority of which reside in Europe, Japan and the US.[5] Stuxnet reportedly ruined almost one fifth of Iran's nuclear centrifuges.[6] Targeting industrial control systems, the worm infected over 200,000 computers and caused 1,000 machines to physically degrade.

Putting Trusted Computing into the IOT

How is IOT/IIOT Different that IT?

- PCs/Laptops/Servers/etc. have IT support – field engineers.
  - If your machine dies, a real person comes to help you.
  - This is not the case for IOT/IIOT. They have to have devices which can be remotely controlled and which can survive a security threat without the need for human intervention. **If a field engineer has to show up to recover a "bricked" IIOT device it is a financial disaster for an IIOT (or IOT) company.**
- IIOT devices have long life cycles
  - IIOT designers have to foresee not only today's cybersecurity threats, but also those that will arise fifteen and twenty years from now.
- Security is a new subject to many in the IOT world

Getting IOT Where It Needs to Be on Security

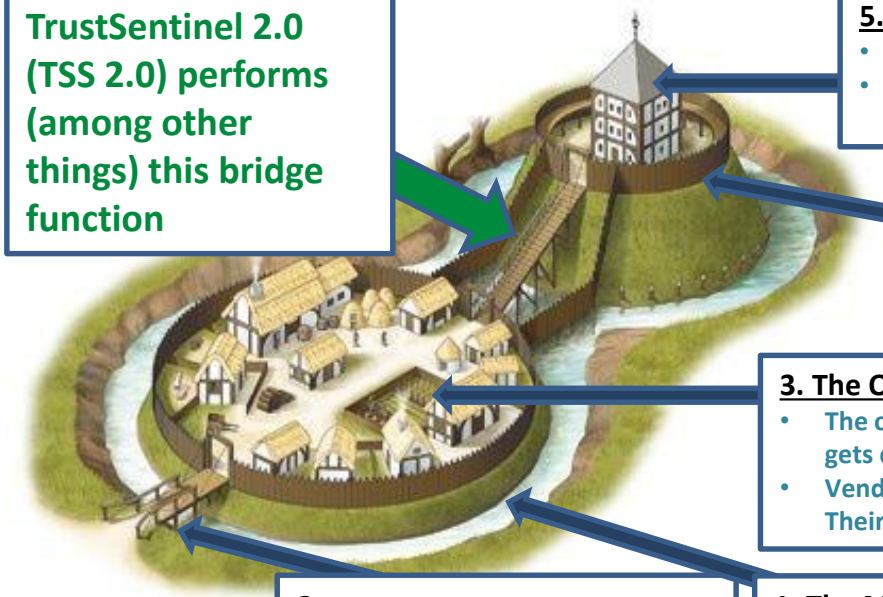
- Moving from environments where connectivity is non-existent or very highly constrained to full internet attachment is a huge hurdle. You cannot take it in incremental steps.
- Trusted computing is a **great** fit for IIOT. It provides security and surviveability.

IOT TODAY

IOT TOMORROW

Trusted Computing *(Modern Cybersecurity)* : Like the Architecture of a Medieval Castle

TrustSentinel 2.0 (TSS 2.0) performs (among other things) this bridge function



5. The Keep
- The keep is the trusted computing base (TCB) – the TPM is here.
- It includes the hardware which is used to store the most important secrets, measure the system and defend it against catastrophic failure.

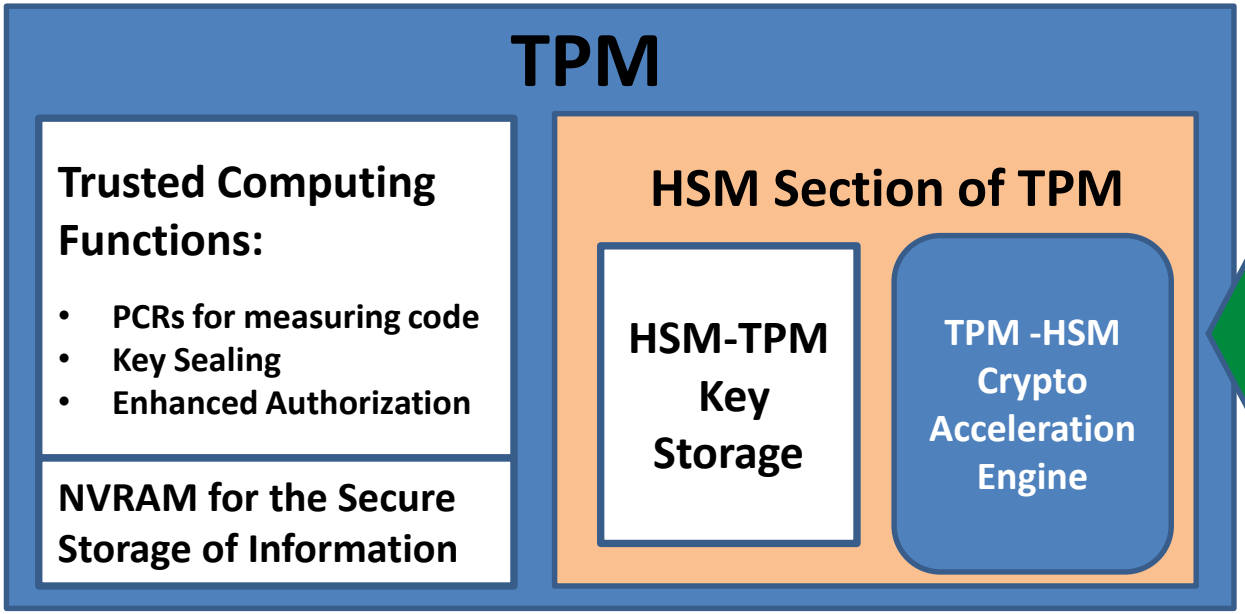
4. The Inner Perimeter
- This is where kernel space and the trusted computing base (keep) reside – and possibly some trusted user applications. This is where the most secure and valuable things reside.

3. The Outer Perimeter
- The outer perimeter is what we call user space. It's where the business of the kingdom gets done.
- Vendors receive police protection (isolation) but must also provide their own security. Their "bank" should be in the keep (but often isn't).

2. The Bridge
- The bridge is where you authenticate (prove who you are) and enter if authorized.

1. The Moat
- The firewall is basically the moat. It dramatically limits points of entry.
- All communications should be secure beyond here.

A TPM Is Best Described In Two Pieces



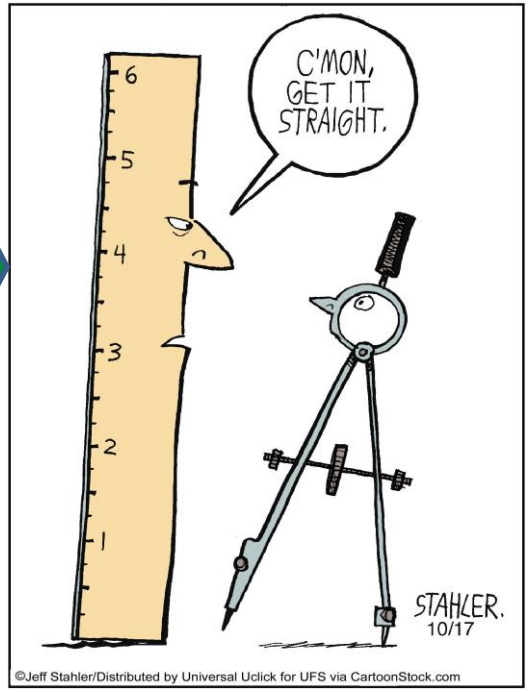
A TPM can be thought of as a code measurement / key sealing system combined with an HSM – <u>at a MUCH lower cost point.</u>

TPM HSM Functions: Protect Keys

The TPM's Hardware Security Module (HSM) functionality allows the protection of keys form your potentially infected system. If they steal the treasure (the data), it is no good to them if they can't get the keys.

TPM Trusted Computing Functions: Allow Secure Storage of Software Measurements

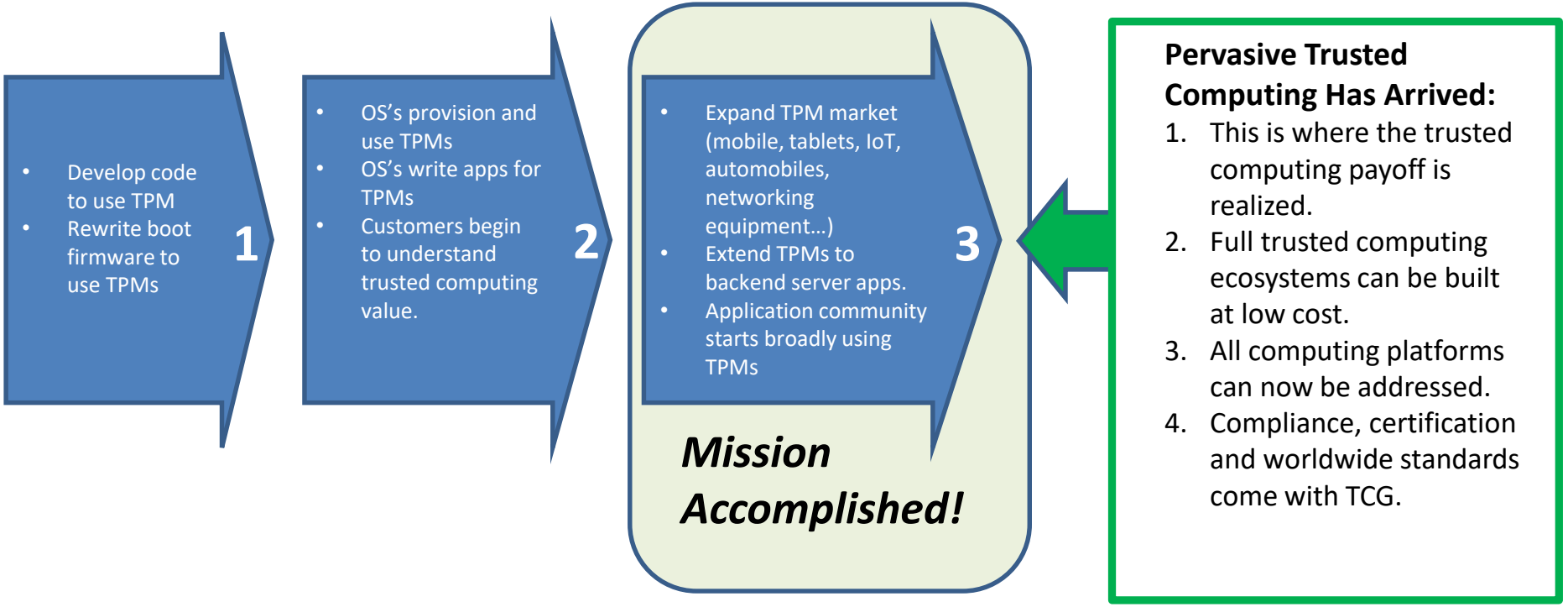
The TPM's Programmable Configuration Registers (PCRs) allow the secure storage of comprehensive software measurements. These can be used to protect keys against infected systems, "attest" the health of the system, etc.



When to Use a TPM…

- If you need the following characteristics for your security solution then use the TPM:
  - If you need your security solution to conform to international standards as now commonly specified in RFPs. (Note: The TPM is an ISO standard (ISO/IEC 11889:2015). To understand the importance of this see this blog https://cloudblogs.microsoft.com/microsoftsecure/2015/06/29/governments-recognize-the-importance-of-tpm-2-0-through-iso-adoption/
  - If your device must ship worldwide without import/export controls. (Note: Since it is an ISO standard, world trade organization members cannot put import/export controls on it)
  - If you need to be able to measure your systems software, seal secrets to these measurements, authorize key use with them and run remote health checks of your system.
  - If you need the functions of an hardware security module (HSM) in a very low cost solution
  - If you need a way to create a strong, permanent identity for your machine when you manufacture it.

Achieving Trusted Computing's Full Promise with an Ecosystem of Solutions



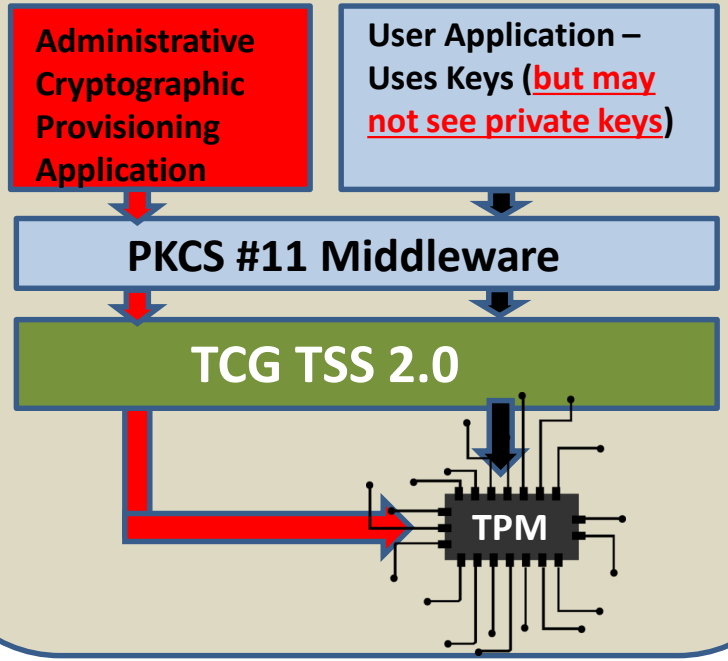
Why Is Trusted Computing An Even Better Pick for IOT Than It Was for PCs/Servers?

- Attestation is easier with IOT devices:
  - They are typically “computer appliance” meaning that they have fixed code loads which customers may not modify. This makes their measurements and event logs very predictable making remote health check via attestation of the TPM much easier.
- IOT devices absolutely need strong identity
  - Since there are effectively no field engineers for IOT, their network identity has to be very strong so they cannot be impersonated by another device or have their identity changed by an attacker. TPMs are very good at providing strong identity. This is also important for PCs, but it is crtical for IOT.
- The low cost of TPMs for PCs was a big benefit but it is a necessity for IOT.

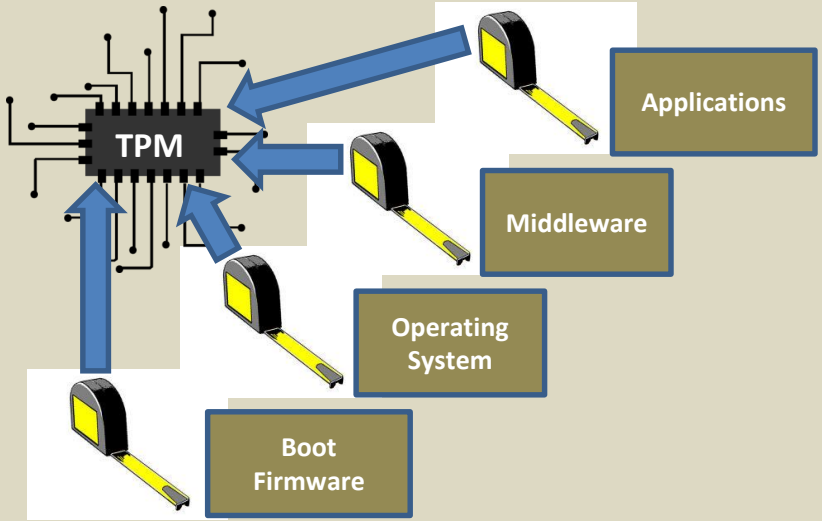
Trusted Computing Use Cases

Major Trusted Computing Use Cases - 1&2

Use Case 1: HSM-Style Key Store and Use – But Using a TPM



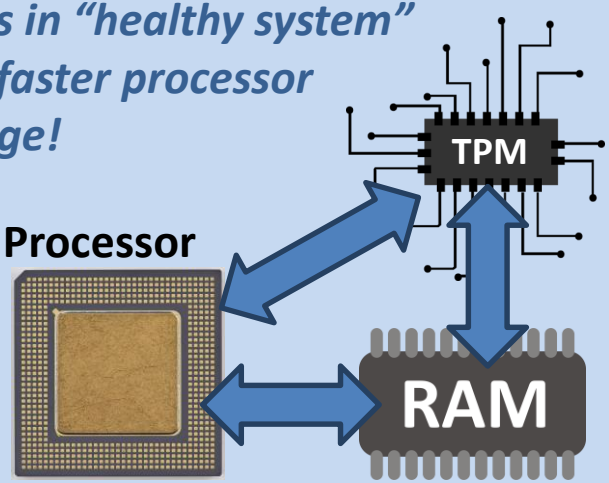
Use Case 2: Measure your software – Seal keys, detect attacks, endpoint management…



Major Trusted Computing Use Cases - 3&4

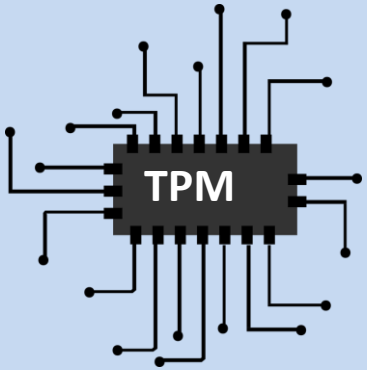
Use Case 3: Trusted Computing Specific Key Use Model: Key Sealing

Unseal TPM-protected keys in "healthy system" for faster processor usage!



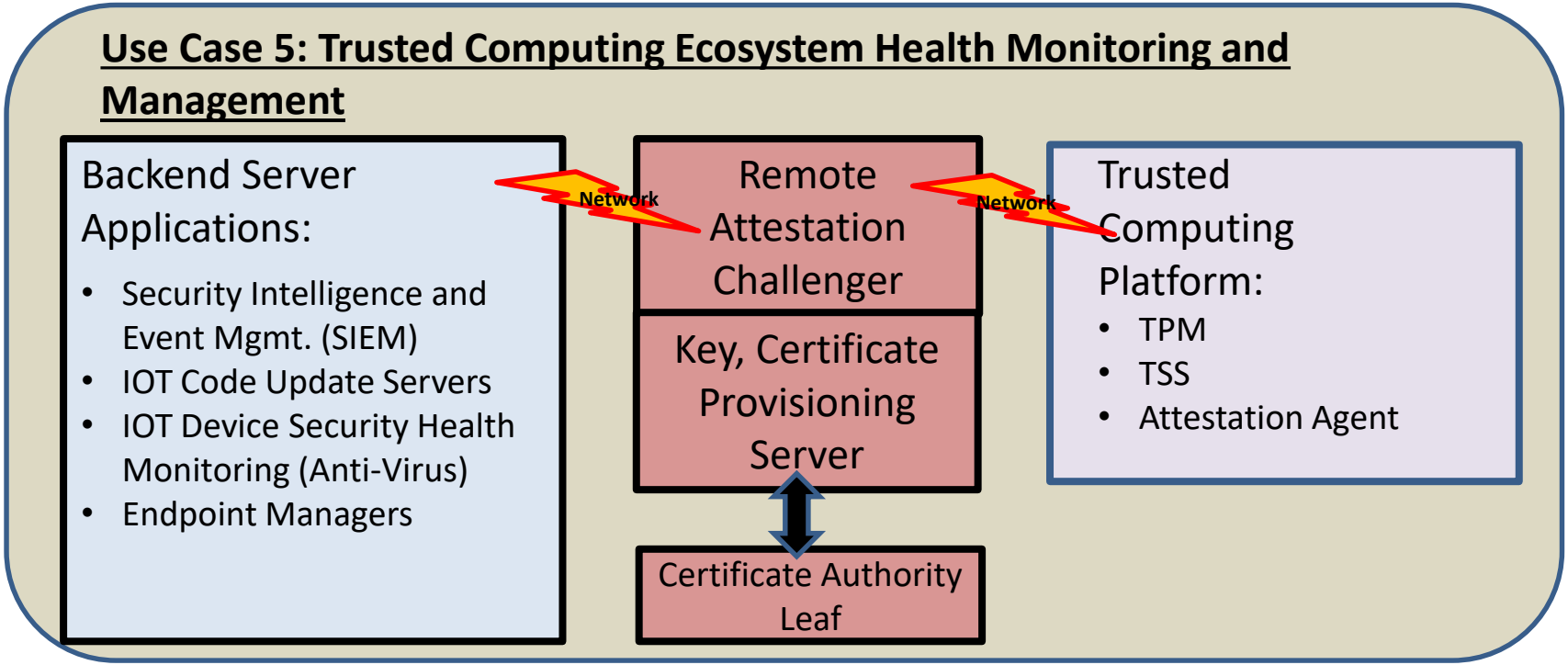
Use Case 4: "Strong Device Identity and Authentication" Using TPMs

Trusted Computing Enabled Platform (TPM Rigidly Attached As Required)



Inexpensive TPMs are required to be permanently melded to their platforms

Major Trusted Computing Use Cases - 5



TSS 2.0

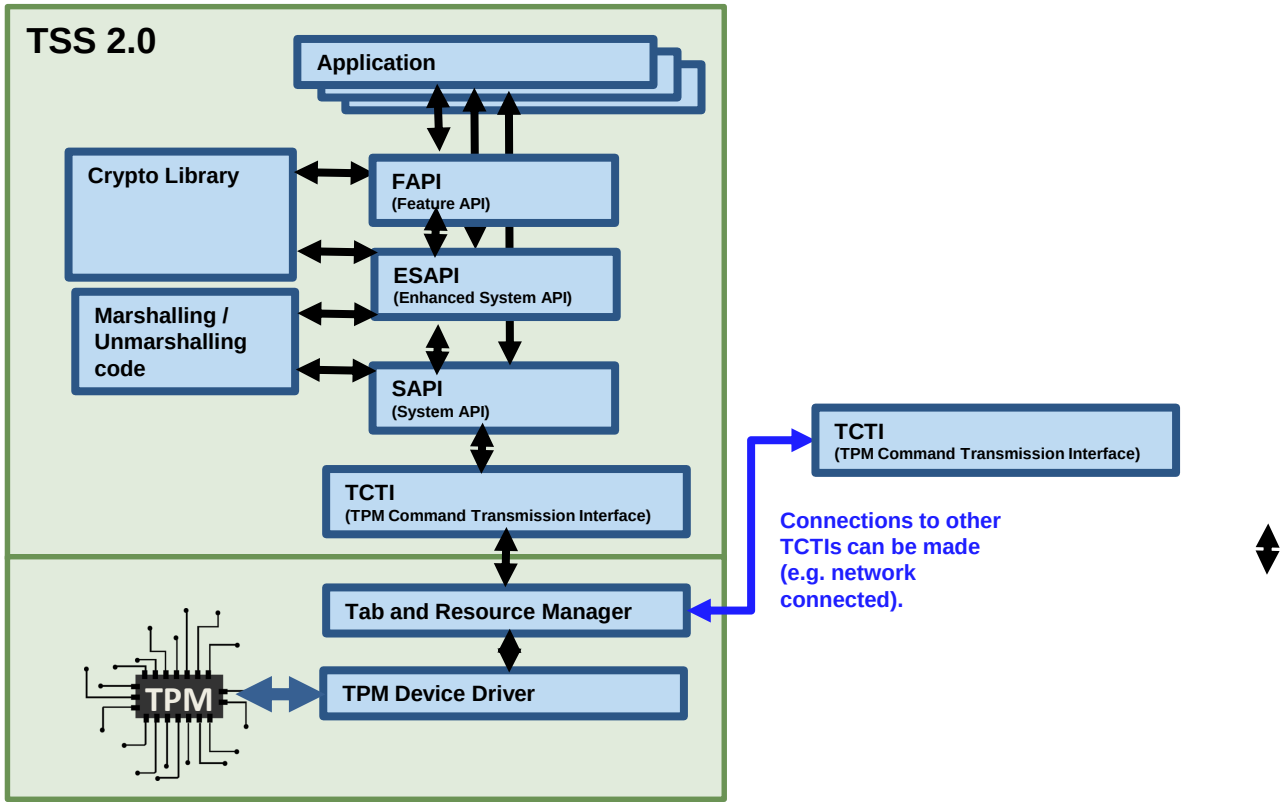
Why Choose the TPM and TCG TSS 2.0?

- TPM 2.0 is an ISO standard
  - It is THE international standard for building hardware based roots of trust.
  - WTO members cannot put import/export controls on an ISO standardized security device.
- TSS 2.0 is TCG's Standard API for using the TPM
  - Will Be/Is specified as required in RFPs now.
  - Governments, critical infrastructure etc. view it as an international standard.
  - It allows cross platform support – you can easily move security applications across platforms.

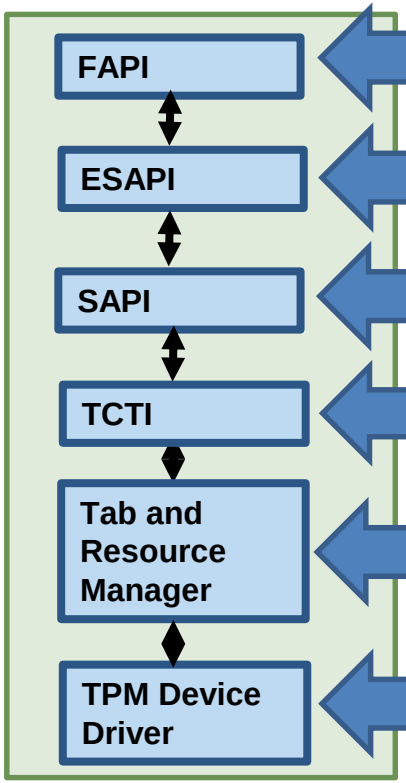
Why Do Security Programmers Need the TCG Software Stack (TSS 2.0) Needed?

- Handles the marshaling/unmarshaling needed when you communicate with a TPM – handles multiple TPM applications.
- Provides synchronous and asynchronous function call models for communicating with the TPM.
- Encrypts the data stream from the software to the TPM stopping side-channel (hardware probing) attacks (EAL 4++).
- Simplifies context and session management needed when applications work with TPMs.,
- Provides varying levels of abstraction (depending on the TSS layer you use) simplifying the task of using the TPM.
- Provides “scalable solutions” allowing different code footprints from the smallest IOT device up to server applications.

TCG Software Stack (TSS 2.0)



Descriptions of TSS 2.0 Layers



FAPI provides new ease of use not available for TSS 1.2. **It allow programmers to interface to the TPM without having ot be TPM experts.**

ESAPI has easier context management and provides the ability to encrypt the data stream to the TPM **stopping sidechannel attacks (essential to EAL4++)**

SAPI doesn't need a file system or a heap. **It can be integrated with your boot firmware** or used in the smallest IOT devices.

The TCTI is an enormous help to development programmers. It allows you to target TPMs other than the hardware TPM on your platform (eg. soft TPMs)

The Tab and Resource Manager will vary depending on the operating system. It allows multiple applications and the kernel to share TPM resources.

Drivers are available today in Linux, Windows – your particular IOT platform may need a modified or custom driver.

The TSS 2.0 API Has These Characteristics

- Adheres to industry recognized best software practices and have “high semantic content”:
  - See Bob Martin’s book Clean Code: A Handbook of Agile Software Craftsmanship (very famous man and book – his nickname in the software community is “Uncle Bob”) to further understand modern requirements for writing and maintaining good code over the complete lifecycle of a product.
  - See Joshua Bloch’s presentation on good API design ➔ http://www.newt.com/java/GoodApiDesign-JoshBloch.pdf Here’s an additional excellent resource ➔ https://www.infoq.com/articles/API-Design-Joshua-Bloch
  - What does it mean to have “clean programming” techniques
    - No function overloading ***High Semantic Content!***
    - Strong type checking ***No variadic variables!***
    - High semantic content (Others – including yourself – will be able to read your code, understand it and maintain it over the lengthy product lifecycle we must support.
    - No global variables, etc.

The TSS 2.0 API Has These Design Characteristics

- MISRA Compliant:
  - For the world of industrial IOT, Automotive, etc. you must meet MISRA coding standards.
- Strong versioning and revision control
  - Designed so that if the underlying implementation behavior changes it is obvious to the user.
  - It is clear from the versioning and revision of the code what changes were made and when.
  - Barring a version change – backwards code compatibility is maintained.

Foundational Trust for IoT

Dennis Mattoon, Microsoft

INTRODUCTION

- Modern cyber-attacks are often sophisticated and relentless in their continual efforts to seek out vulnerabilities in modern technology-based solutions
- At the same time, market segments like IoT are driving architectures and solutions with challenging power, security, resource, and other constraints. These constraints make an optimal security posture much more difficult to create and maintain

INTRODUCTION

- To effectively address these challenges a security architecture must be:
  - Free or very cheap, and not just in BOM cost
  - Adaptable, with minimal silicon requirements
  - Scalable to millions of endpoints per solution
  - Standards-based, i.e., interoperable
- And most importantly, it takes a combination of hardware support and software techniques

WHY HARDWARE SUPPORT?

- There are problems with software-only solutions
- Device Identity
  - If a bug leads to disclosure of Device Identity secret then how do we securely (and remotely) recover and re-provision a device?
- Device State and Attestation
  - Cannot trust software to report its own health
- Roots of Trust, data encryption, entropy, etc.
  - How do we securely extend trust chain, store keys, etc.?

BEWARE SIMPLISTIC HW SOLUTIONS

- Why not just store Device Identity key/secrets in fuses?
  - If malware can manage to read the fused key then you are no better off than with a software-based key
- TPMs are great but, especially in IoT solutions, systems and components probably won't have TPMs or similar silicon-based capabilities (cost, complexity, physical space on the MCU/SoC)
- We need something different

DICE AND RIOT

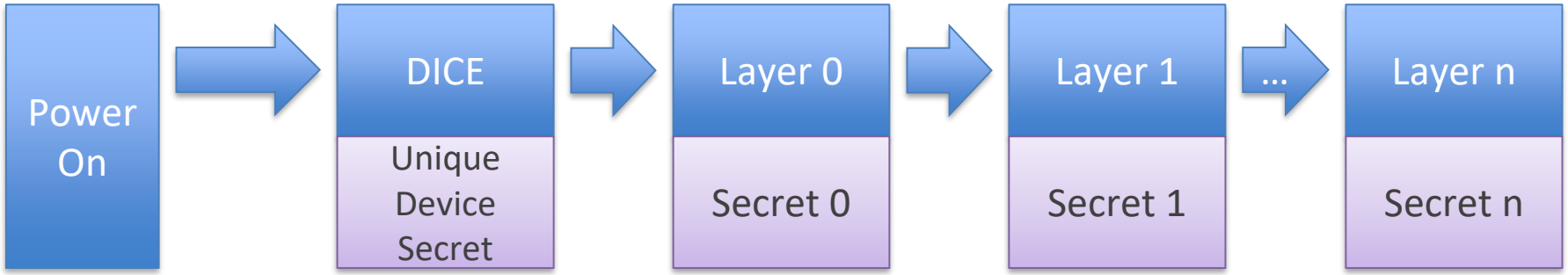
- Device Identifier Composition Engine (DICE, TCG)
- Robust, Resilient, Recoverable IoT (RIoT, MSFT)
- New specification from the DICE Architectures Workgroup in the Trusted Computing Group (TCG)
- Foundational security for IoT at near zero cost
- Simple hardware requirements mean DICE is adaptable to most any system or component
- Provides HW-based identity and attestation, and a foundation for sealing, data integrity, device recovery and update

THE DICE MODEL

- In a DICE Architecture device startup (boot) is layered
- Beginning with a Unique Device Secret (UDS), secrets/keys are created that are unique to the device and each layer and configuration
- This derivation method means that if different code or configuration is booted, secrets are different
- If a vulnerability exists and a secret is disclosed, patching the code automatically re-keys the device

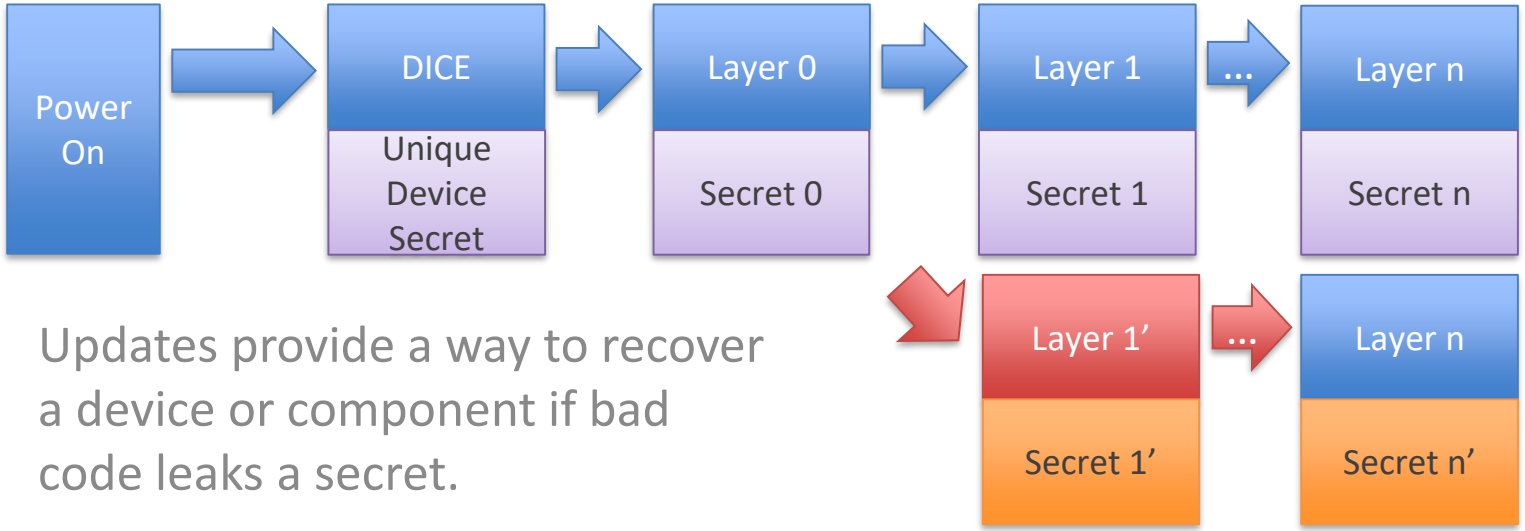
THE DICE MODEL

- Power-on (reset) unconditionally start the DICE
- DICE has exclusive access to the UDS
- Each layer computes the secret for next layer (via OWF)
- In this derivation chain, each layer must protect the secret it receives



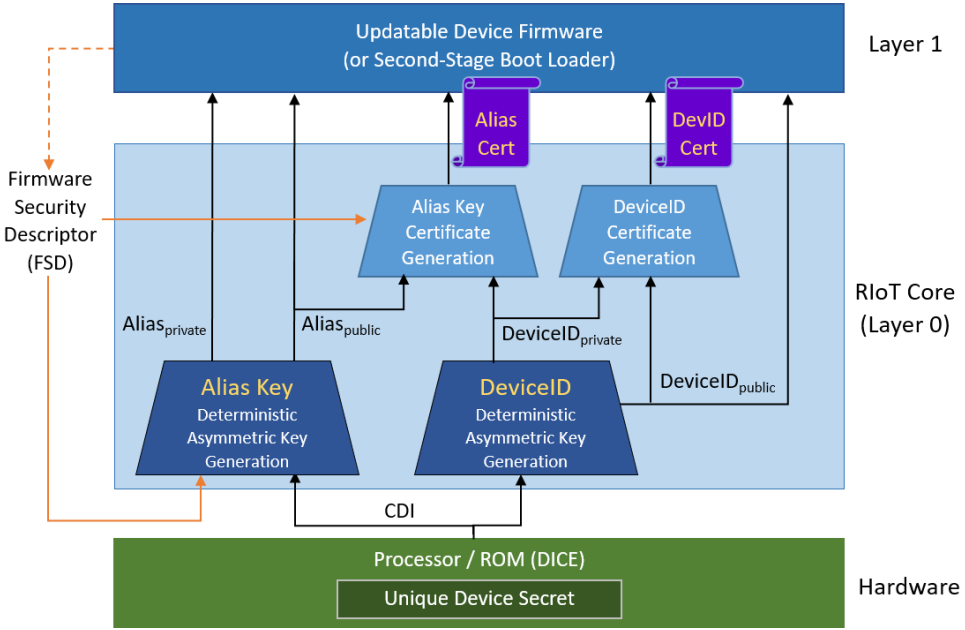
WHEN SOMETHING CHANGES

- The branch illustrates the result of a code/config change



- Updates provide a way to recover a device or component if bad code leaks a secret.

A DICE ARCHITECTURE (RIOT)



- Underlying architecture for HW-based Device Identity and Attestation (Azure)
- DeviceID – Stable and well protected long term identifier for a device or component
- Alias Key – Derived from combination of unique device identity (HW) and identity of Device Firmware (SW)
- Integrates DICE-enabled HW with existing infrastructure

BUT THAT'S JUST ONE EXAMPLE

- We can build on DICE to enable many high-value scenarios
- Secure remote device recovery (Cyber Resilient Platform Initiative)
  - Recover unresponsive (i.e., p0wned, hung, etc.) devices
  - Greatly reduced cost: no need for physical device interaction
- Supply chain management ("DICE for Components")
  - Several recent damaging cyber-attacks were the result of malware introduced in the supply chain
  - DICE attestation lets end-customers trust far less of the supply-chain, e.g., just the storage-subsystem or flash vendor
- Strong cryptographic identity, authenticity, licensing, and many more

DICE TAKEAWAYS

- Flexible security framework, not one size fits all
- Minimal silicon requirements, low barrier to entry
- Foundation for strong cryptographic HW-based device identity and attestation, data at rest protection (sealing), and secure device update and recovery
- Public announcements from SoC, MCU, and flash memory vendors so far with more on the way
- Represents the ongoing work of the DICE Architectures Workgroup (DiceArch WG) in TCG. Come join us!

REFERENCES

- Device Identifier Composition Engine (DICE) specification
- DICE Architectures Workgroup (TCG)
- RIoT – A Foundation for Trust in the Internet of Things
- Cyber-Resilient Platform Initiative
- TCG developer community: develop.trustedcomputinggroup.org
- Partners and Demos:
  - Microchip CEC1702 and SecureIoT1702
  - Micron Authenta
  - Sequitur Labs (i.MX6, SAMA5D2)
  - STMicroelectronics (STM32L4xx)
- Azure IoT:
  - Strengthening IoT Security
  - Zero-Touch Provisioning with Azure IoT