Getting Started with Secure Software

Class 1: Introduction to Platform Security Architecture (PSA)

April 20, 2020

Jacob Beningo

DesignNews

CEC CONTINUING EDUCATION CENTER

Presented by: Digi-Key ELECTRONICS

Course Overview

Topics:

- **Introduction to Platform Security Architecture (PSA)**
- Performing a Security Threats Analysis
- Architecting a Secure Solution
- Secure Boot and the Root-of-Trust
- Secure Frameworks and Ecosystems

The Lecturer – Jacob Beningo

Jacob Beningo

President

Social Media / Contact

- E : jacob@beningo.com
- T : 810-844-1522
- : Jacob_Beningo
- : Beningo Engineering
- : JacobBeningo
- EDN : Embedded Basics

ARM Connected Community

Consulting

- Advising
- Coaching
- Content
- Consulting
- Training

BENINGO EMBEDDED GROUP

www.beningo.com

DesignNews

Presented by:

CEC CONTINUING EDUCATION CENTER

Digi-Key ELECTRONICS

Jacobs CEC Courses

CEC 2013 – 2016

- Fundamentals of Embedded Software (2013)
- Mastering the Software Design Cycle (2014)
- Python for Embedded Systems(2014)
- Software Architecture Design (2014)
- Baremetal C (2015)
- Mastering the ARM Cortex-M Processor (2015)
- Writing Portable and Robust Firmware in C (2015)
- Design Patterns and the Internet (2015)
- Bootloader Design for MCUs (2016)
- Rapid Prototyping w/ Micro Python (2016)
- Debugging (2016)
- Professional Firmware (2016)

CEC 2017 - 2018

- API's and HAL's February 2017
- Baremetal to RTOS April 2017
- Designing IoT Sensor Nodes July 2017
- From C to C++ October 2017
- Connecting Edge Devices (March 2018)
- Building an IoT Connected PLC (April 2018)
- Securing IoT Devices using Arm TrustZone (Nov 2018)
- Minimizing Defects (Dec 2018)

CEC 2019

- Machine Learning for Embedded (April 2019)
- Designing Embedded Systems using MicroPython
- Launching a Product (Nov 2019)

CEC 2020

- Getting Started with Secure Software
- Building Machine Vision Applications using OpenMV (June)
- Techniques for Interfacing with Modern Sensors (October)
- Designing Embedded Systems using the ESP32 (December)

CEC CONTINUING EDUCATION CENTER

Presented by: Digi-Key ELECTRONICS

Session Overview

- Introduction
- Platform Security Architecture (PSA)
- Stage 1 – Analyze
- Stage 2 – Architect
- Stage 3 – Implement
- Stage 4 – Certify
- Best Practices

Introduction

Why is security important?

Hackers Remotely Kill a Jeep on the Highway – With Me in It

Source: Andy Greenberg / Wired

Hacking risk leads to recall of 500,000 pacemakers due to patient death fears

FDA overseeing crucial firmware update in US to patch security holes and prevent hijacking of pacemakers implanted in half a million people

Source: Abbott / St Jude Medical, theguardian.com

LILY HAY NEWMAN SECURITY 07.16.19 02:13 PM

THESE HACKERS MADE AN APP THAT KILLS TO PROVE A POINT

Source: Lily Hay Newman / Wired

Introduction

Security is not optional anymore

Billions of IoT devices

Data integrity, security and privacy

Potential losses from hacks and breaches

Government regulation

Introduction

What are we protecting?

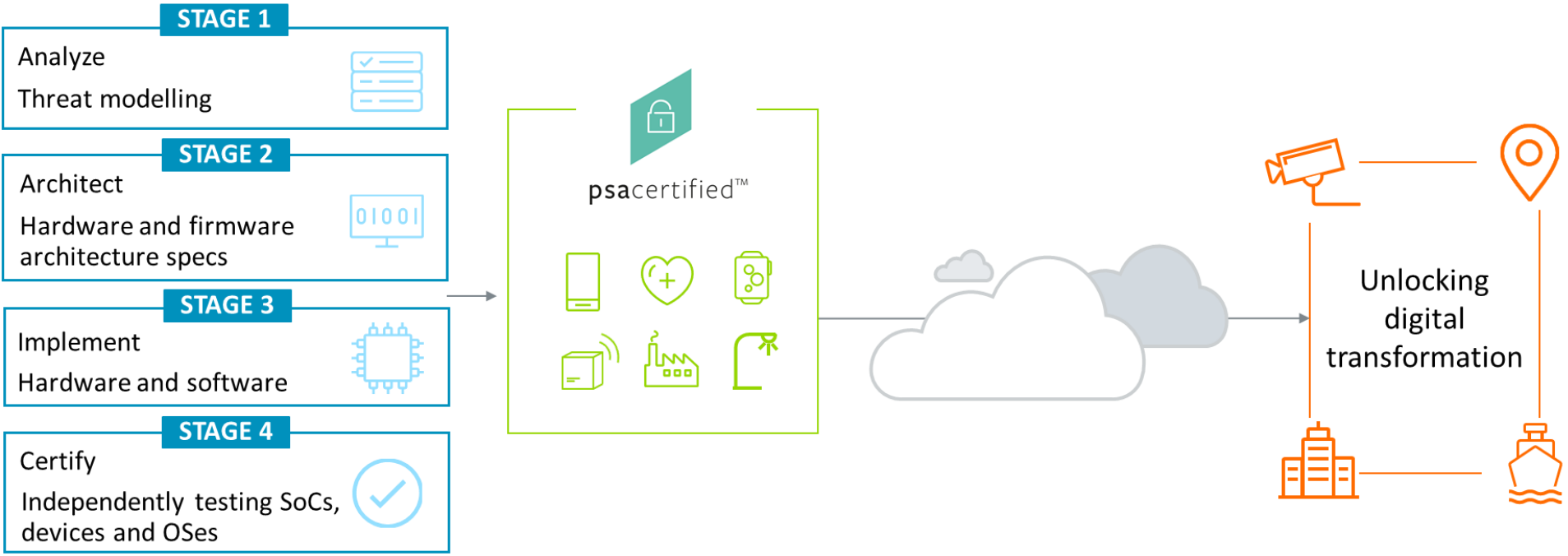
General Data Assets

- The firmware (embedded software)
- Unique device ID
- Passwords (device, user, etc.)
- Encryption keys (device control, secure communication, cloud access, etc.)

Device Specific Data Assets

- Sensor data
- Image data
- Control data

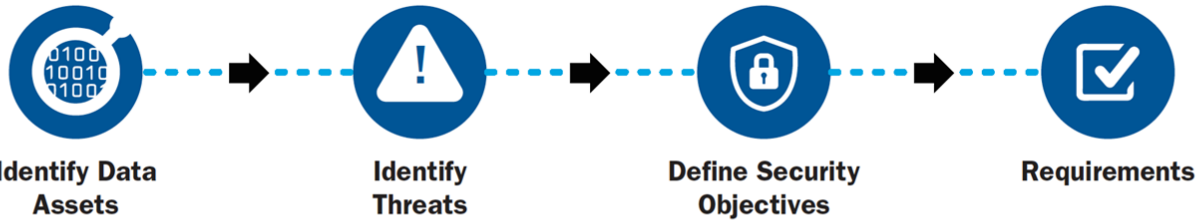
Platform Security Architecture (PSA)



PSA: enabling right-sized device security

Stage 1 - Analyze

Threat-based Analysis Method



System requirements drive implementation, including microcontroller selection

Example:	*Firmware*	*Malware Attack*	*Device Attestation*	*Isolated Root of Trust*

Joint white paper at:
www.cypress.com/psoc6security

arm CYPRESS EMBEDDED IN TOMORROW™

Stage 1 - Analyze

Analysis Leads to Requirements

System description

↓

Assets

↓

Threats

↓

Security Objectives

↓

Security Requirements

Arm delivered representative IoT device security analyses & requirements

Example – Smart Meter

Asset: metering data to be protected in integrity & confidentiality

↓

Threat: Remote SW attacks

↓

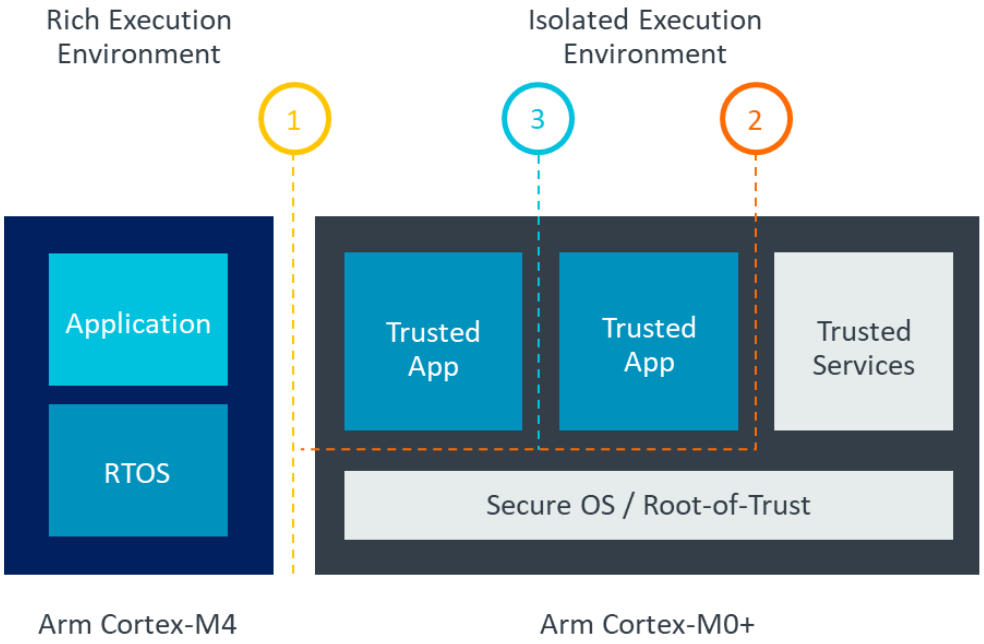
Security Objective: Strong Crypto

↓

Security Requirement: Hardware based key store

Stage 2 - Architect

Security through isolation – Option #1



Hardware-based Isolation within PSoC 64 Secure MCUs

Hardware based isolation within PSoC 64 Secure MCUs enables secure element functionality and reduces the attack surface

Three levels of isolation

1. Secure execution environment (SEE) isolated from rich execution environment
2. Root-of-trust and trusted services isolation within SEE
3. Application isolation within SEE

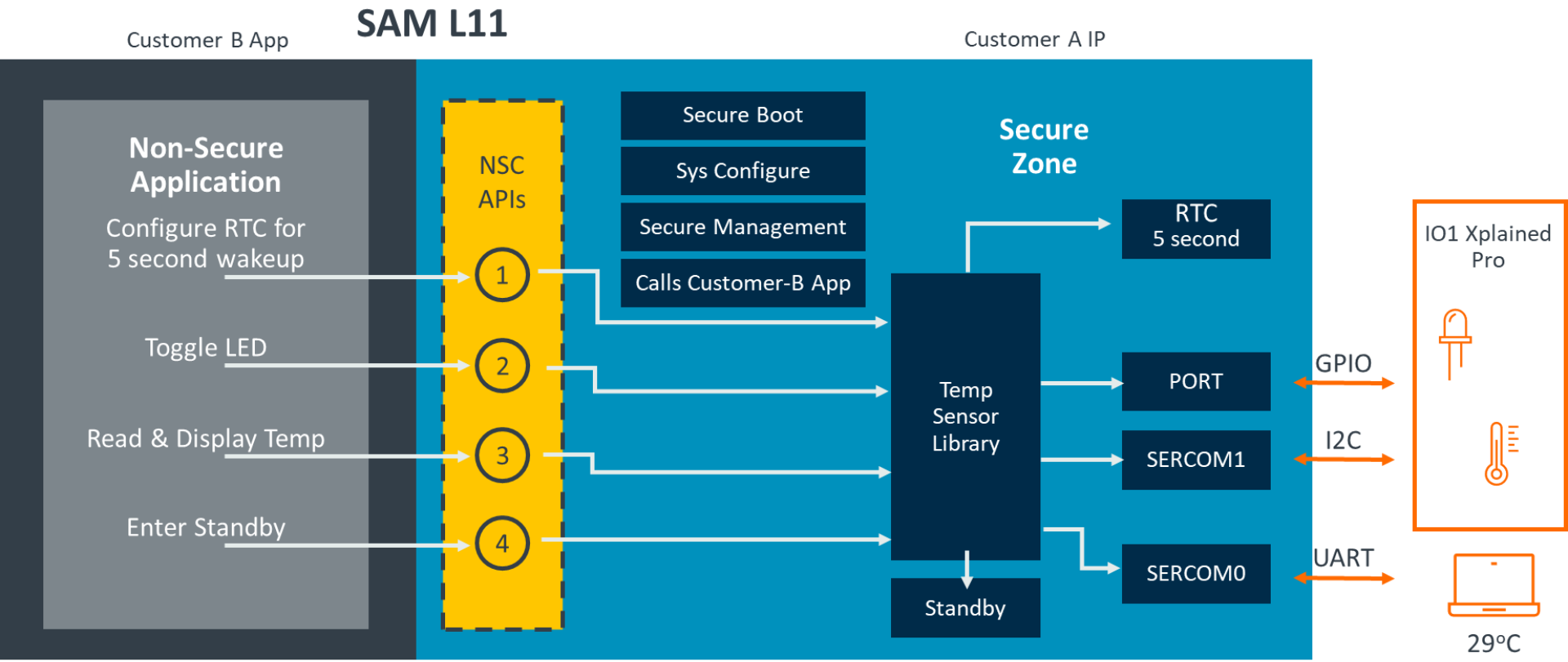
Stage 2 - Architect

Security through isolation – Option #2

arm TRUSTZONE

Normal environment (Non-Secure)		Protected environment (Secure)	
Application Examples • User applications • RTOS • Device drivers • Protocol stacks	Handler Mode		Handler Mode
Normal Resources • General peripherals	Thread Mode		Thread Mode
		Secure Software Examples • Secure Boot • Cryptography libraries • Authentication • RTOS support APIs / RTOS	
		Secure Resources • Secure storage • Crypto accelerators	

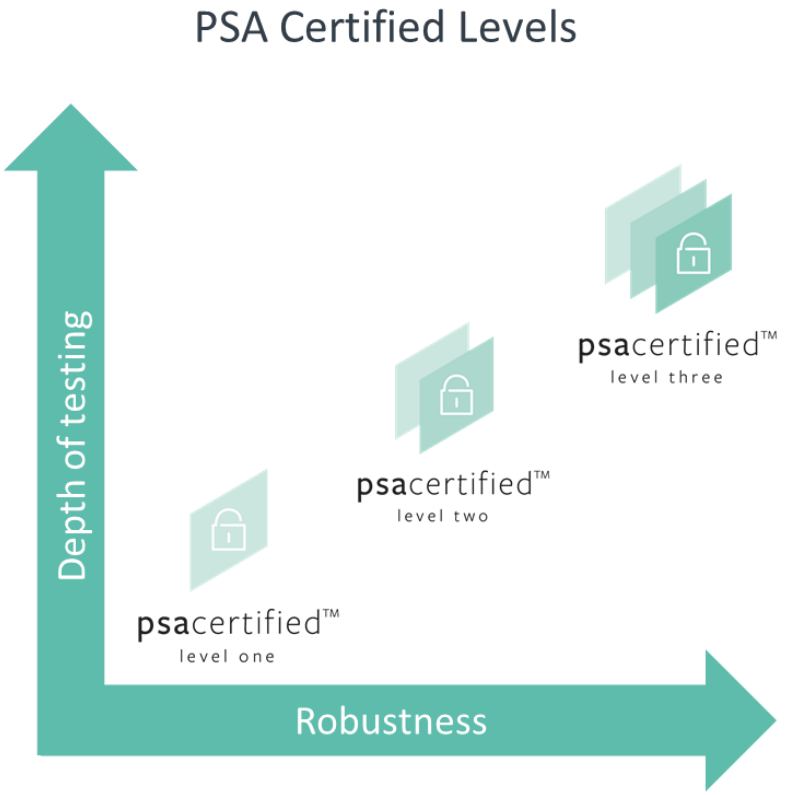
Stage 3 - Implement



Stage 4 - Certify

PSA Certified: 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 Root of Trust (PSA-RoT)



Stage 4 – Certified

PSA Certified Levels

PSA Certification Level	Silicon	OS	OEM
Level 3 Months	✓	Third-party evaluation schemes	
Level 2 1 month	✓		
Level 1 1 day	✓	✓	✓

Three assurance levels

Level 1: Document & Declare with lab check

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

Level 2: Mid-level assurance/robustness

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

Level 3: Substantial - Under development

- More extensive attacks
  e.g. side-channel, perturbation
- Higher assurance

Best Practices for Getting Started with Security

1. Identify the data that you are trying to protect up front!
2. Use a security framework to accelerate and guide system development.
3. Protect your embedded system based on the threats that it will face.
4. Carefully examine the security capabilities of potential microcontrollers to ensure they fit the threats.
5. Create an architecture that uses multiple levels of isolation.
6. Use the memory protection unit (MPU) to provide additional safety and security mechanisms.
7. Identify Secure and Non-secure code elements during the architecture phase, not during implementation.
8. Use hashes and signatures to securely boot your MCU and develop a root of trust.
9. Download and walk through security examples from multiple vendors to get a rounded understanding about security.
10. Review the PSA website and leverage existing threat models to accelerate development.

Additional Resources

- Beningo.com
  - Blog, White Papers, Courses
  - Embedded Bytes Newsletter
    - http://bit.ly/1BAHYXm
- Platform Security Architecture:
  - www.arm.com/psa
- Threat-based analysis method:
  - www.cypress.com/psoc6security

From www.beningo.com under

- Blog > CEC – Getting Started with Secure Software