MCUboot: A Secure Bootloader For Microcontroller-class Devices

Aditi Hilbert <aditi@runtime.io>
Szymon Janc <szymon.janc@codecoup.pl>

September 25, 2017

CODECOUP

runtime

IoT challenges

- **Small, memory-constrained, low-cost**
    - Example hardware: 100 MHz, 256 KB of flash, 32 KB of RAM!
- **Security**
    - Millions of connected devices present large attack surface
    - Devices deployed in unprotected areas
- **Scale and Variety**
    - Customers want long-term flexibility in choice of HW and OS without vendor lock-in
    - Customers need consistent and easy management across the fleet

MCUboot: Features

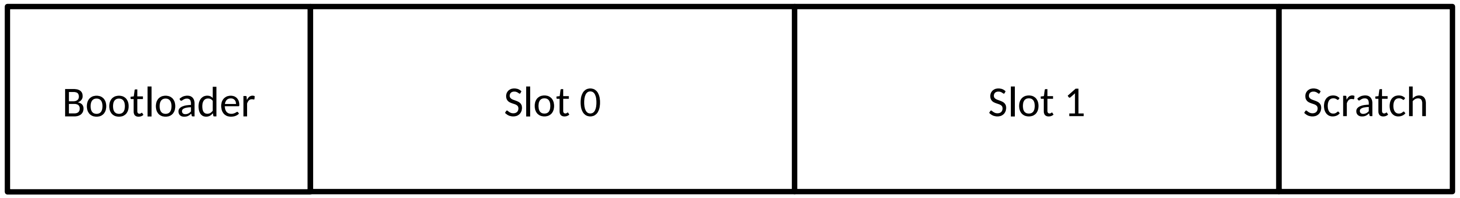
Goal: Provide the foundation for secure upgrade

- **Image Verification**
  - Digital signatures supported: RSA, ECDSA, (soon Ed25519)
- **Two supported upgrade methods:**
  - Image swap
  - Overwrite
- **Modular design:**
  - Portable across Operating Systems
    - Currently supports Apache Mynewt, Zephyr OS, Riot OS
  - Simple porting layer provided by the OS
    - Uses minimal OS features: flash driver, single thread, crypto services

Version 1.0 just released!

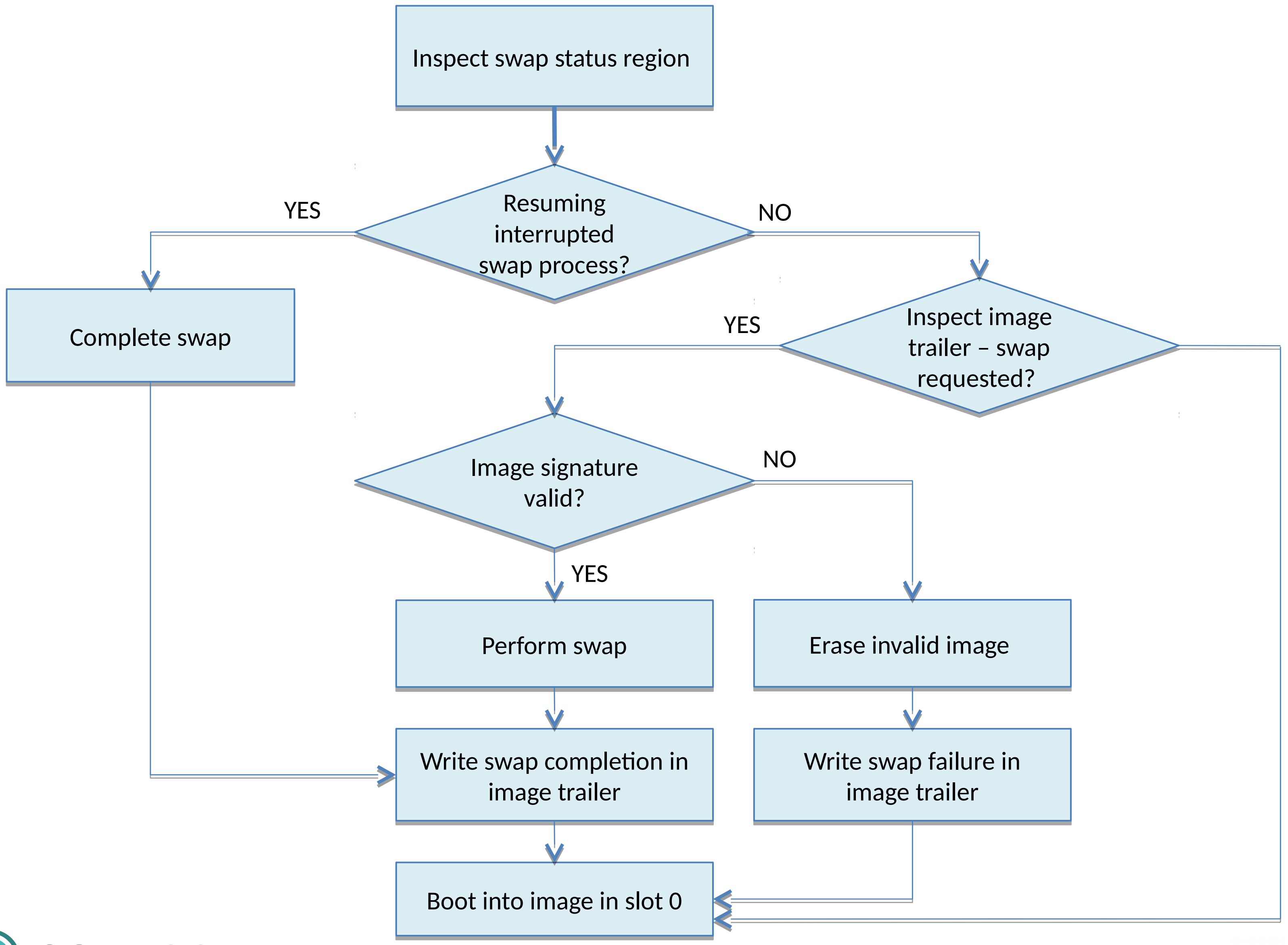
CODECOUP runtime

Flash Layout



- Slot 0: Primary image, code always runs from here
- Slot 1: New image for upgrade
- During upgrade, MCUboot swaps slots using scratch
- Image trailer indicates state of swap and upgrade
- Image header contains image size and version information

Boot Operation


Inspect swap status region
Resuming interrupted swap process?
YES
NO
Complete swap
Inspect image trailer – swap requested?
YES
Image signature valid?
NO
YES
Perform swap
Erase invalid image
Write swap completion in image trailer
Write swap failure in image trailer
Boot into image in slot 0

CODECOUP

runtime

Tools

- Newt tool (in Go) from Apache Mynewt
  - Build images
  - Sign images
  - Load
  - Run and debug images

- Imgtool.py from Linaro
  - keygen: Generate private/public keypairs to use for signing
  - getpub: Extract a public key as C source to be included in bootloader
  - sign: Add a signature to an image

- Simulator
  - Bootloaders are tricky!
  - Compiles on a host machine along with the simulation
  - Tests various configuration of images, upgrades and signatures
  - Tests recovery of untimely upgrade interrupts, simulating power loss
  - Run by Travis on every pull request given to github

Roadmap

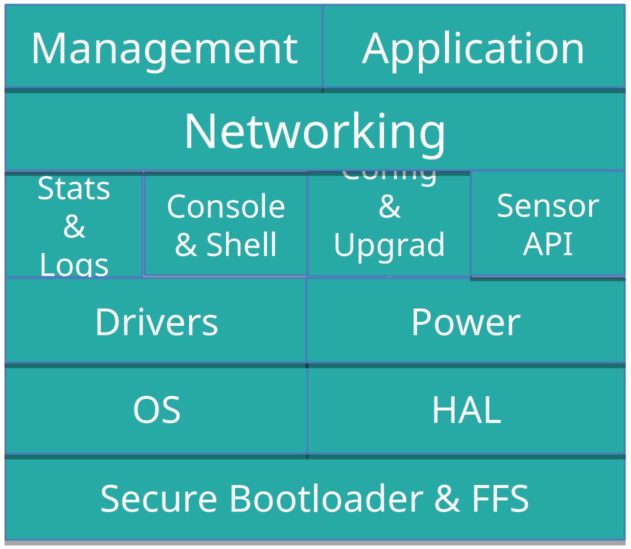
- Support for multiple flash devices
- More efficient crypto libraries, additional signature algorithms
- More error detection
- Key invalidation and revocation
- Abstraction layer to leverage HW-based security (e.g. accelerator, secure OTP)
- Additional tools for testing and debugging
- Porting to additional OS
- Testing with lots of HW!

MCUboot: Project Details

- **Has evolved out of the Apache Mynewt bootloader**
- https://github.com/runtimeco/mcuboot
- **Mailing list:** dev-mcuboot@lists.runtime.co
- **Slack:**

  https://join.slack.com/t/mcuboot/shared_invite/MjE2NDcwMTQ2MTYyLTE1MDA4MTIzNTAtYzgyZTU0NjFkMg
- **Version 1.0 just released!**

Origins of MCUboot: Apache Mynewt



https://mynewt.apache.org/

Any module can be decoupled and used by other Operating Systems!

- MCU agnostic: ARM Cortex-M*, AVR, MIPS, RISC-V
- Pre-emptive, multi-threaded, power optimized RTOS
- Open networking stacks including BLE host & controller
- Secure Bootloader and Image Upgrade
- Flash file systems, console, sensor framework & more
- Build & Package Management – Newt Tool
- Open Management Interfaces (e.g., OIC 1.1 / IoTivity)

Thank You!

CODECOUP

runtime