MCUboot Followup: The IoT Bootloader

David Brown
Introduction

- At BUD17 we gave an intro to MCUboot
- Since then, we’ve had a 1+ releases
- What’s been done, what have we learned
- The release process
Features

- Stable bootloader
- Two supported upgrade methods:
  - image swap
  - overwrite
- Digital signatures: RSA, ECDSA, (soon Ed25519)
- Modular design - Portable across OS’s
Project metadata

- Have a project website: http://mcuboot.com/
- JIRA for issue management: https://runtimeco.atlassian.net/
- Code hosted at Github: https://github.com/runtimeco/mcuboot
- Slack channel (see mcuboot.com for invite link)
- Much thanks to runtime.io for hosting
Releases

- **Version 0.9.0**
  - [https://github.com/runtimeco/mcuboot/releases/tag/v0.9.0](https://github.com/runtimeco/mcuboot/releases/tag/v0.9.0)
    - Most features intended for 1.0
    - Decided to release 0.9 to avoid stalling the process

- **Version 1.0.0**
  - [https://github.com/runtimeco/mcuboot/releases/tag/v1.0.0](https://github.com/runtimeco/mcuboot/releases/tag/v1.0.0)
    - Important image format change
    - Most other requests pushed to future versions
Portability

- Can be built as an app for Zephyr or Mynewt
- Support for booting applications on Zephyr, Mynewt or Riot
- Porting abstraction layer based on Mynewt Flash HAL
- Zephyr uses devicetree to describe partitions, this is used by MCUboot built with it
Mynewt has ‘newt’ tool to build and generate images, and sign as well

Wrote ‘imgtool.py’ for managing Zephyr (and other) images

- 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

On Zephyr side ‘samples/zephyr’ Makefile can:

- build MCUboot for a given platform, using Zephyr
- build two “hello world” programs, also on Zephyr
- sign each image with a distributed sample key
- flash the various configurations to show upgrades
The Simulator

- Despite desire, MCUboot is still complicated
- We were finding bugs that were hard to reproduce
- So, we wrote a simulator
- Compiles on a host machine along with the simulation
- Is able to test various configuration of images, upgrades and signatures
- Tests recovery of untimely upgrade interrupts, simulating power loss
- Many fixes have gone in because of this
- Run by Travis on every pull request given to github
The future

- Open issues: https://runtimeco.atlassian.net/issues/?filter=-5
- Support external flash, run from internal, upgrade/scratch in external
- RAM loading support, for non XIP devices
- Ed25519 signatures
- Build as Riot app
- Porting to additional OS’s
- Queries about bootloader from running app (for OTA upgrades)
- Two stage bootloader, first stage entirely in ROM
Thank You

#SFO17

BUD17 keynotes and videos on: connect.linaro.org
For further information: www.linaro.org