mcuboot: A Bootloader for IoT

David Brown
mcuboot

- Background
- Earlier work
- Zephyr port
- mcuboot
- Details
Background

- **IoT:**
  - Small, very low cost
  - 100 MHz, ½ MB Flash, ¼ MB RAM
  - Custom OSs common, Zephyr, Mnewt

- **Problems:**
  - Field upgrade
  - Security
Earlier Work

- Mynewt bootloader
- Solves part of upgrade problem
- Demo at last connect was with this (Carbon)
Zephyr Port

- Bootloader uses minimal OS features
  - Flash driver
  - Single thread
  - Crypto services
- Ported Mynewt bootloader to Zephyr
  - Minor changes to Zephyr (merged)
  - Able to demo with this
mcuboot

● Teams agreed to make bootloader separate project
● github.com/runtimeco/mcuboot
● Code builds for either Mynewt or Zephyr (other RTOSes considering it as well)
● It will boot any image with the required header (target OS agnostic)
● A little challenging as Mynewt and Zephyr have very different build environments
Flash Layout

<table>
<thead>
<tr>
<th>Bootloader</th>
<th>Slot 0</th>
<th>Slot 1</th>
<th>Scratch</th>
</tr>
</thead>
</table>

- Slot 0 is primary image, code always runs from here
- Slot 1 is upgrade
- If upgrade needed, bootloader swaps slots using scratch
- An image trailer tracks state of swapping and upgrades
## Slot Format

<table>
<thead>
<tr>
<th>Image Header</th>
<th>Application</th>
<th>TLVs</th>
<th>Image Trailer</th>
</tr>
</thead>
</table>

- Header contains size, and version information
- TLV has SHA256 hash, and signature (RSA or ECDSA)
- Trailer has “magic” and indicates state of slot (swap needed, in progress, etc).
Boot Process

- Bootloader checks both slots
- Normally boots slot 0, can verify signature before boot
- If new image is marked (in trailer) in slot 1, safely swaps the two slots
- After trying to boot the new image, will revert if it doesn’t mark itself as good
- OTA is mostly done by apps, only image swap and signature verification done by bootloader
Changes to Zephyr

- Apps must be linked to run at Slot 0 address
  - Set CONFIG_FLASH_BASE_ADDRESS
- Apps need to allow room for image header
  - Set CONFIG_TEXT_SECTION_OFFSET
  - Needed to be implemented for ARM/Cortex-M
- Needed to fix vector table init code to not assume table was at start of flash
  - Mynewt uses a RAM vector table
  - Zephyr’s vector table is in flash, so reg must be set
  - Has device-specific alignment requirements
Tooling

- Apps build as regular Zephyr apps, with a few config changes
- Bootloader is a regular Zephyr app with its own config
- For development, uses hardcoded public/private keys, deployment will need real (and protected) keys
- Assembling images currently done with a modified version of Mynewt’s ‘newt’ code (a Go application)
  - Needed to add image header and TLVs and possibly generate trailer
Useful Logos

Download Hi Res logos from here* to use on your slides

*http://link.linaro.org/logos
Some Images

- Download Hi Res photos from [here*](http://link.linaro.org/groupphotos) to use on your slides
Thank You

#BUD17

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