ENDIAN TECHNOLOGIES

Meaningful connection.

Update my Board!

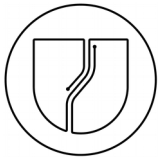
Integrate an open-source software update solution on your board.

Session Overview

- Background
- Integration
  - Mender
  - libostree (aktualizr)
  - SWUpdate
  - resin.io

Background

- Board Support Package development
  - Bootloader, Linux kernel, user-space (Yocto/OE-core)
  - **Software update solution**
- Homegrown
- Open-source alternatives


Background

- Talks on software update
  - “Comparison of Linux Software Update Technologies” by Matt Porter (video, slides)
  - “Embedded Systems Software Update for IoT: The Current State of Play” by Chris Simmonds (video, slides)
  - “”Software Updates for Connected Devices: Key Considerations” by Eystein Stenberg (video, slides)

Background

Embedded Linux and beyond

https://mkrak.org

Mender

“Mender is an end-to-end open source updater for connected devices and IoT”

- https://docs.mender.io/
- Apache 2.0
- Golang
- Symmetric A/B image update

Mender (Requirements)

- U-boot
  - CONFIG_BOOTCOUNT_ENV/LIMIT
  - Persistent storage of U-boot environment (EMMC/FLASH)
  - fw_setenv/fw_getenv tools
- Two partitions for root-filesystem
- One partition for persistent storage
- eMMC/SD or UBI volumes

Mender (Yocto/OE-core)

- meta-mender
  - meta-mender-core, meta-mender-raspberrypi....
  - “INHERIT += "mender-full"”
  - Fully automatic U-boot patching
    - rocko, recent U-boot, eMMC/SD
- Output = “.mender” and “.sdimg” images

Mender (U-boot)

- Two patches (board independent)
  - Variables and scripts/commands
  - mender_setup, mender_try_recovery etc...
- One board specific patch
  - Integration of above commands (mender_*)
  - BOOTCOUNT_ENV/LIMIT
  - BOOTENV options (ENV_IS_IN_MMC/NAND/FLASH)

Mender (Yocto/OE-core)

```
meta-mender$ tree meta-mender-beaglebone/
meta-mender-beaglebone/
├── conf
│   └── layer.conf
└── recipes-bsp
    └── u-boot
        ├── patches
        │   └── 0001-BBB-Use-Mender-boot-code-for-selecting-boot-device-a.patch
        ├── u-boot_%.bbappend
        ├── u-boot-beaglebone.inc
        └── u-boot-fw-utils_%.bbappend

4 directories, 5 files
```

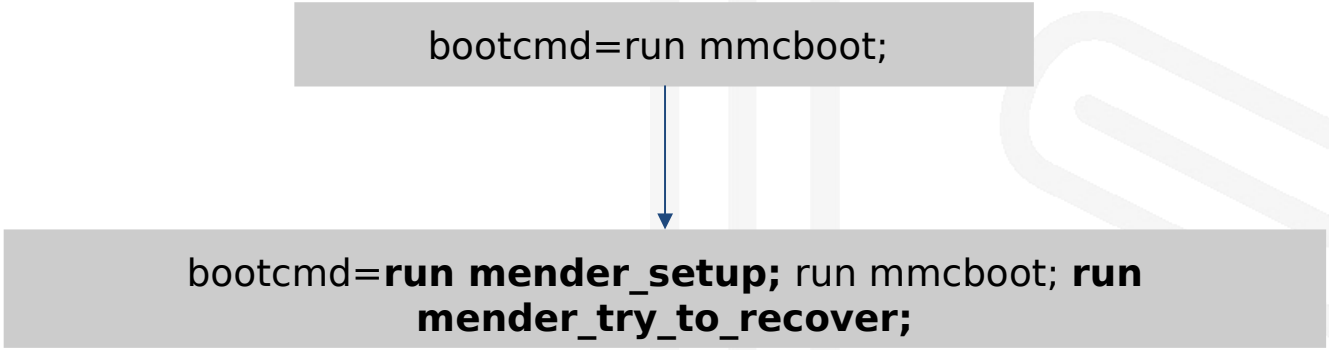
Mender (Yocto/OE-core)

- U-boot fork

require recipes-bsp/u-boot/u-boot-fw-utils-mender.inc

require recipes-bsp/u-boot/u-boot-mender.inc

Mender (U-boot)



Mender (U-boot)

loadimage=load **mmc 0:1** ${loadaddr} ${bootdir}/${bootfile}

loadimage=load **${mender_uboot_root}** ${loadaddr} ${bootdir}/${bootfile}

libostree

“git for operating system binaries”

- https://ostree.readthedocs.io
- C & GPLv2
- Image updates
  - Binary deltas

libostree (Requirements)

- Operates on top the Unix filesystem layer
  - hard-links
- Never boot to physical rootfs
  - initramfs chroot to “deployment”
- /usr is immutable
- Persistent state in /var
- Complex

libostree (Yocto/OE-core)

- meta-updater
- Aktualizr
  - SOTA client
- “INHERIT += "sota"”
- initramfs image (init.sh)
- Physical sysroot
- Deployment sysroot
  - Each build will be “committed” and made deployable

libostree (meta-updater)

- Integration point
  - Load U-boot env from uENV.txt

```
bootcmd_otenv=ext2load mmc 0:2 $loadaddr /boot/loader/uEnv.txt; env import
-t $loadaddr $filesize
bootcmd_args=setenv bootargs "$bootargs $bootargs_fdt
ostree_root=/dev/mmcblk0p2 root=/dev/ram0 rw rootwait rootdelay=2
ramdisk_size=8192"
bootcmd_load=ext2load mmc 0:2 $kernel_addr_r "/boot"$kernel_image; ext2load
mmc 0:2 $ramdisk_addr_r "/boot"$ramdisk_image
bootcmd_run=bootm $kernel_addr_r $ramdisk_addr_r $fdt_addr_r
bootcmd=run bootcmd_dtb; run bootcmd_otenv; run bootcmd_args; run
bootcmd_load; run bootcmd_run
```

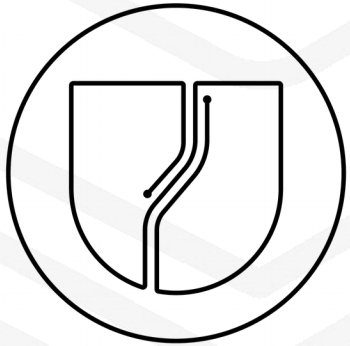
libostree (meta-updater)

- /etc
  - Each deployment carries a copy
  - 3-way merge with local copy
- /var
  - Writable area
  - Empty
- /usr
  - immutable

SWUpdate

“SWUpdate is a Linux Update agent with the goal to provide an efficient and safe way to update an embedded system”

- http://sbabic.github.io/swupdate/
- C & GPLv2
- Framework
- NOR / NAND, UBI volumes, SD / eMMC



SWUpdate (Yocto/OE-core)

- meta-swupdate
  - client support, recovery OS image
  - swupdate.bbclass (.swu)
- meta-swupdate-boards
  - reference implementations (BBB, RPi3, WandBoard)
  - Symmetric (sw-description)

SWUpdate (sw-description)

- Description files
- libconfig syntax
- Handlers
  - Script, u-boot
  - swupdate.bbclass (.swu)

SWUpdate (sw-description)

```
software =
{
        version = "0.1.0";
        beaglebone = {
                hardware-compatibility: [ "1.0"];
                stable : {
                        copy1 : {
                                < ... >
                        };
                        copy2 : {
                                < ... >
                        };
                };
        }
}
```

SWUpdate (sw-description)

```
copy1 : {
        images: (
                < ... >
        );
        scripts: (
                < ... >
        );
        uboot: (
                < ... >
        );
};
```

SWUpdate (sw-description)

```
copy1 : {
        images: (
                {
                        filename = "core-image-full-cmdline-beaglebone.ext4.gz";
                        device = "/dev/mmcblk1p2";
                        type = "raw";
                        compressed = true;
                }
        );
        < ... >
        < ... >
};
```

SWUpdate (sw-description)

```
copy1 : {
        < ... >
        scripts: (
                {
                        filename = "emmcsetup.lua";
                        type = "lua";
                }
        );
        < ... >
};
```

SWUpdate (sw-description)

```
copy1 : {
        < ... >
        < ... >
        uboot: (
                {
                        name = "boot_targets";
                        value = "legacy_mmc1 mmc1 nand0 pxe dhcp";
                },
                {
                        name = "bootcmd_legacy_mmc1";
                        value = "setenv mmcdev 1;setenv bootpart 1:2; run mmcboot";
                }
        );
};
```

SWUpdate

- “Remote Software Updates for IoT Devices with Eclipse hawkBit” - Diego Rondini, Kynetics

resin.io

“Resin.io brings the benefits of Linux containers to the IoT. Develop iteratively, deploy safely, and manage at scale.”

- https://docs.resin.io/introduction/
- ResinOS
- Container Deltas (apps)
- Symmetric A/B image (ResinOS)
- Proprietary “console”

resin.io (requirements)

- U-boot, Grub
  - Update hooks in user-space
    - /mnt/boot/resinOS_uEnv.txt
    - /mnt/boot/grub.cfg
- ResinOS
  - Yocto based distribution
  - Container Deltas (apps)
- Dual rootfs parts + three “persistent” parts
- Only eMMC/SD support

resin.io (Yocto/OE-core)

- meta-resin
  - npm to setup the Yocto environment?!
- custom board layer (resin-<board family>)
  - CoffeeScript configuration files?!
- key take-away
  - “inherit resin-u-boot”
  - “inherit kernel-resin”
  - update hook

resin.io (u-boot)

- /mnt/boot/resinOS_uEnv.txt
- Three patches (board independent)
  - Variables and scripts/commands
  - resin_set_kernel_root, etc...
- One board specific patch
  - Integration of above commands (resin_*)
  - CONFIG_PARTITION_UUIDS
  - CONFIG_CMD_PART

resin.io (u-boot)

```
@@ -177,7 +181,7 @@
        "mmcbootpart=" __stringify(CONFIG_SYS_MMC_IMG_LOAD_PART) "\0" \
        "mmcrootpart=2\0" \
        "mmcargs=setenv bootargs console=${console},${baudrate} " \
-               "root=/dev/mmcblk${mmcblk}p${mmcrootpart} rootwait rw " \
+               "${resin_kernel_root} rootwait rw " \
                "${cma_size}\0" \
        "loadbootenv=" \
                "load mmc ${mmcdev}:${mmcbootpart} ${loadaddr} ${bootdir}/${bootenv}\0" \
```

resin.io (u-boot)

```
@@ -222,6 +226,10 @@
#else
#define BOOT_ENV_SETTINGS MMC_BOOT_ENV_SETTINGS
#define CONFIG_BOOTCOMMAND \
+       "setenv resin_kernel_load_addr ${loadaddr};" \
+       "run resin_set_kernel_root;" \
+       "setenv mmcdev ${resin_dev_index};" \
+       "setenv mmcbootpart ${resin_boot_part};" \
        "run ramsize_check; " \
        "mmc dev ${mmcdev};" \
        "mmc dev ${mmcdev}; if mmc rescan; then " \
```

Honorary Mentions

- RAUC
  - meta-rauc
- swupd
  - meta-swupd

Summary

- Proven solutions
- Seamless integration with Yocto
- No reason to go “homegrown”!
- Collaboration

Questions?

?

Thank you!

mirza@mkrak.org
mirza.krak@endian.se