A/B Linux updates with RAUC and meta-rauc-community: now & in the future

Leon Anavi
Konsulko Group
leon.anavi@konsulko.com
leon@anavi.org
Yocto Project Summit 2021.11

Konsulko Group

Konsulko Group

- Services company specializing in Embedded Linux and Open Source Software
- Hardware/software build, design, development, and training services
- Based in San Jose, CA with an engineering presence worldwide
- http://konsulko.com/

Agenda

Konsulko Group

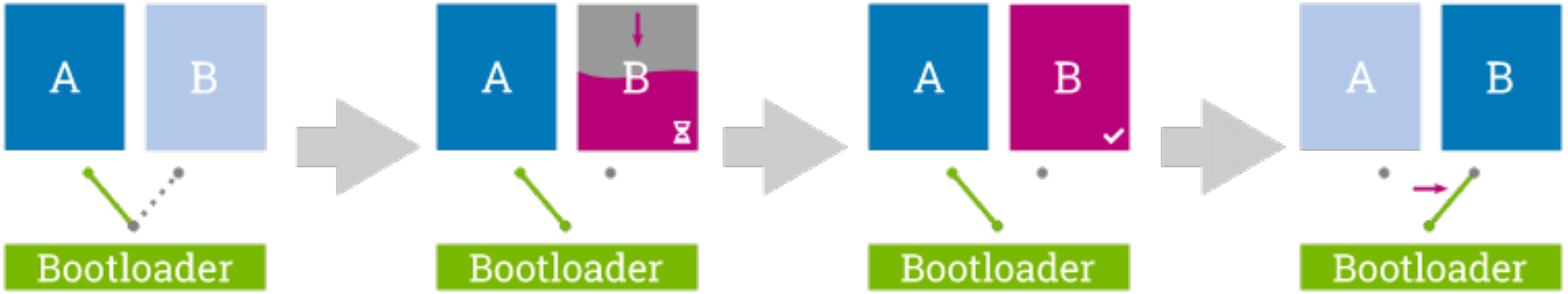
- What is RAUC?
- RAUC practical example with Raspberry Pi 4
- Eclipse hawkBit
- Conclusions
- Q&A

What is RAUC?

Konsulko Group

- A lightweight update client that runs on an Embedded Linux device and reliably controls the procedure of A/B updating the device with a new firmware revision
- Provides tool for the build system to create, inspect and modify update bundles
- Uses X.509 cryptography to sign update bundles
- Compatible with the Yocto Project and OpenEmbedded, PTXdist and Buildroot
- Started by Pengutronix in 2015, adopted by the community and the industry

How Does RAUC Work?



RAUC Licenses

Konsulko Group

- RAUC – LGPLv2.1
  https://github.com/rauc/rauc
- meta-rauc - MIT
  https://github.com/rauc/meta-rauc
- meta-rauc-community – MIT
  https://github.com/rauc/meta-rauc-community
- rauc-hawkbit – LGPLv2.1
  https://github.com/rauc/rauc-hawkbit
- rauc-hawkbit-updater – LGPLv2.1
  https://github.com/rauc/rauc-hawkbit-updater

Eclipse hawkBit

Konsulko Group

- Domain independent back-end framework for rolling out software updates to constrained edge devices as well as more powerful controllers and gateways connected to IP based networking infrastructure
- Written in Java
- Available in GitHub under EPL-1.0 License
- Compatible with **RAUC** and SWUpdate
- https://www.eclipse.org/hawkbit/

Are there any RAUC open source alternatives?

Konsulko Group

- Mender
- SWUpdate
- Swupd
- UpdateHub
- Balena
- Snap
- OSTree
- Aktualizr
- Aktualizr-lite
- QtOTA
- Torizon
- FullMetalUpdate
- Rpm-ostree (used in Project Atomic)

Common Embedded Linux Update Strategies

Konsulko Group

- A/B updates with dual redundant scheme (for example: RAUC)
- Delta updates
- Container-based updates
- Combined strategies

Combined Update Strategies

Konsulko Group

- Container technology has changed the way application developers interact with the cloud and some of the good practices are nowadays applied to the development workflow for embedded devices and IoT
- Containers make applications faster to deploy, easier to update and more secure through isolation
- Yocto/OE layer meta-virtualization provides support for building Xen, KVM, Libvirt, docker and associated packages necessary for constructing OE-based virtualized solutions
- There are use cases on powerful embedded devices where **containers** are combined with **RAUC** A/B updates of the base Linux distribution built with Yocto/OE

Yocto/OpenEmbedded Layers for RAUC

Konsulko Group

- **meta-rauc**

  Layer for RAUC, the embedded Linux update framework

- **meta-rauc-community**

  Layer with examples for integration of RAUC, the embedded Linux A/B update framework

meta-rauc

Konsulko Group

- Yocto/OpenEmbedded meta layer for RAUC
- Supports releases Gatesgarth, Dunfell, Zeus, Warrior, Thud, Sumo, Morty, Pyro and Krogoth
- Available under MIT license in GitHub: https://github.com/rauc/meta-rauc
- 33 contributors, the RAUC co-maintainer Enrico Jörns from Pengutronix is the leading contributor

meta-rauc-community

Konsulko Group

Yocto/OpenEmbedded meta layer with demo examples for integration of RAUC, the embedded Linux A/B update framework:

- **meta-rauc-raspberrypi**: for Raspberry Pi
- **meta-rauc-qemux86**: for QEMU (qemux86-64)
- **meta-rauc-sunxi**: for Allwinner sunxi SoCs
- **meta-rauc-tegra**: for NVIDIA Jetson platforms, based on L4T

https://github.com/rauc/meta-rauc-community

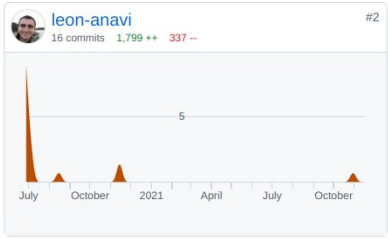
Contributors

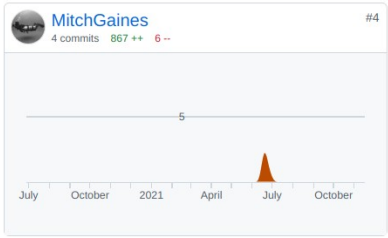
- **meta-rauc-community** started in 2020
- Moved to RAUC GitHub organization in 2021
- 5 contributors

Jun 28, 2020 – Nov 18, 2021

Contributions: Commits

Contributions to master, excluding merge commits and bot accounts





Notable Contributions

Konsulko Group

Support for Sunxi Boards #17

Merged **leon-anavi** merged 8 commits into `rauc:master` from `MitchGaines:sunxi` on Jun 22

Conversation 2 | Commits 8 | Checks 0 | Files changed 19

MitchGaines commented on Jun 22 — Contributor

Add Layer for qemux86 #25

Merged **leon-anavi** merged 16 commits into `rauc:master` from `ejoerns:topic/qemux86` on Sep 17

Conversation 1 | Commits 16 | Checks 1 | Files changed 17

ejoerns commented on Sep 17 — Member

meta-rauc-tegra: NVIDIA Jetson TX2 #27

Merged **leon-anavi** merged 1 commit into `rauc:master` from `leon-anavi:master-tegra` 10 days ago

Conversation 1 | Commits 1 | Checks 1 | Files changed 15

leon-anavi commented 14 days ago — Member

RAUC Integration Steps

Konsulko Group

- Select an appropriate bootloader
- Enable **SquashFS** in the Linux kernel configurations
- **ext4** root file system (RAUC does not have an ext2 / ext3 file type)
- Create specific partitions that matches the RAUC slots
- Configure Bootloader environment and create a script to switch RAUC slots
- Create a certificate and a keyring to RAUC's **system.conf**

RAUC Example with Raspberry Pi 4

Konsulko Group

- Download Poky, meta-openembedded and meta-raspberrypi:

```
git clone -b dunfell git://git.yoctoproject.org/poky poky-rpi-rauc
cd poky-rpi-rauc
git clone -b dunfell git://git.openembedded.org/meta-openembedded
git clone -b dunfell git://git.yoctoproject.org/meta-raspberrypi
```

- Download RAUC related layers:

```
git clone -b dunfell https://github.com/rauc/meta-rauc.git
git clone -b dunfell https://github.com/leon-anavi/meta-rauc-community.git
```

- Initialize the build environment:

```
source oe-init-build-env
```

RAUC Example with Raspberry Pi 4

Konsulko Group

- Add layers:

```
bitbake-layers add-layer ../meta-openembedded/meta-oe/
bitbake-layers add-layer ../meta-openembedded/meta-python/
bitbake-layers add-layer ../meta-openembedded/meta-networking/
bitbake-layers add-layer ../meta-openembedded/meta-multimedia/
bitbake-layers add-layer ../meta-raspberrypi/
bitbake-layers add-layer ../meta-rauc
bitbake-layers add-layer ../meta-rauc-community/meta-rauc-raspberrypi/
```

RAUC Example with Raspberry Pi 4

Konsulko Group

- Add to local.conf:

```
MACHINE = "raspberrypi4"
DISTRO_FEATURES_append = " systemd"
VIRTUAL-RUNTIME_init_manager = "systemd"
DISTRO_FEATURES_BACKFILL_CONSIDERED = "sysvinit"
VIRTUAL-RUNTIME_initscripts = ""
IMAGE_INSTALL_append = " rauc"
IMAGE_FSTYPES="tar.bz2 ext4 wic.bz2 wic.bmap"
SDIMG_ROOTFS_TYPE="ext4"
ENABLE_UART = "1"
RPI_USE_U_BOOT = "1"
PREFERRED_PROVIDER_virtual/bootloader = "u-boot"
WKS_FILE = "sdimage-dual-raspberrypi.wks"
```

RAUC Example with Raspberry Pi 4

Konsulko Group

- Build a minimal bootable image:

```
bitbake core-image-minimal
```

- Flash the image to a microSD card and boot it on Raspberry Pi 4:

```
sudo umount /dev/sdX*
bzcat tmp/deploy/images/raspberrypi4/core-image-minimal-raspberrypi4.wic.bz2 | sudo dd of=/dev/sdX
sync
```

- Attach USB to UART debug cable to Raspberry Pi 4, plug Ethernet cable and the microSD card. Turn on Raspberry Pi 4. Verify that the system boots successfully.

RAUC Update Bundle

Konsulko Group

- Add to conf/local.conf:

```
IMAGE_INSTALL_append = " nano"
```

- Build a RAUC bundle:

```
bitbake update-bundle
```

Manual RAUC Update of Raspberry Pi 4

Konsulko Group

- On the build system:

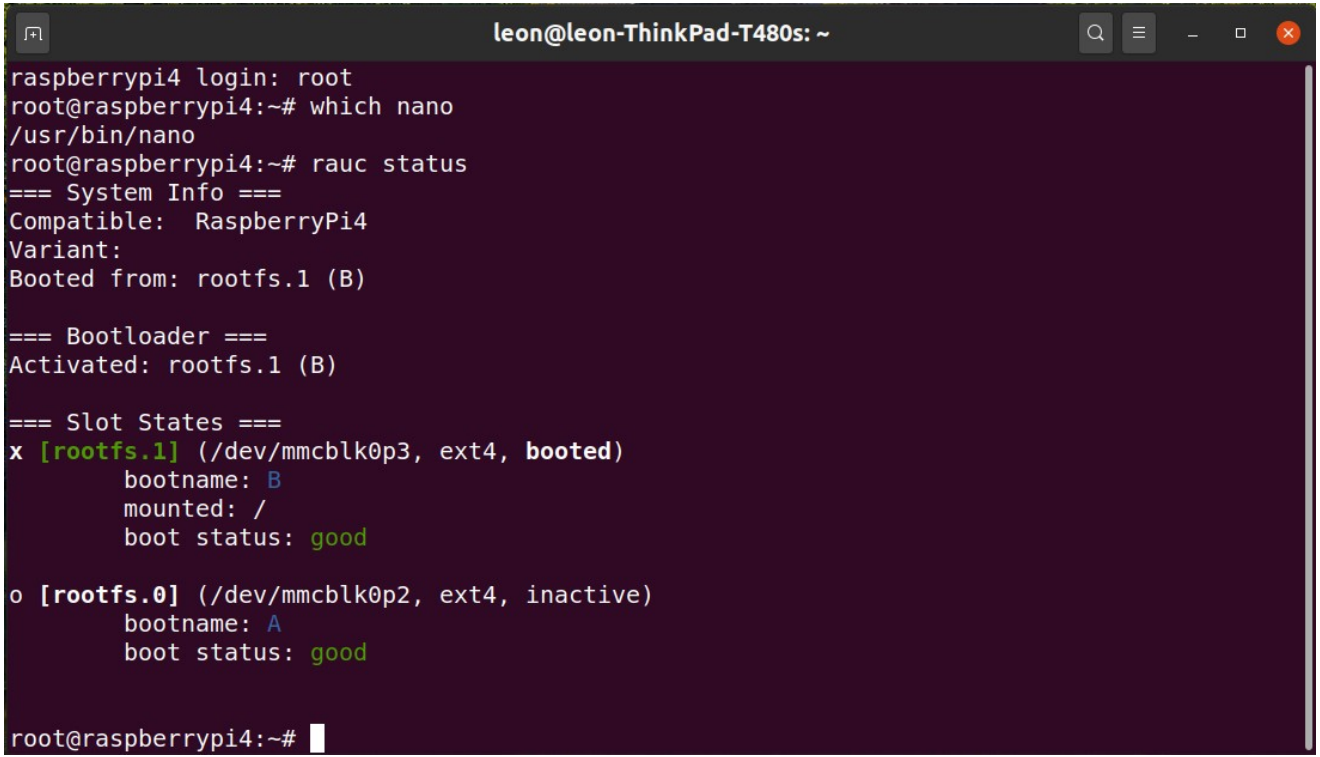
```
cd tmp/deploy/images/raspberrypi4/
python3 -m http.server
```

- On the embedded device, in this case Raspberry Pi 4:

```
wget http://example.com:8000/update-bundle-raspberrypi4.raucb -P /tmp
rauc install /tmp/update-bundle-raspberrypi4.raucb
reboot
```

Check RAUC Status After Update

Konsulko Group



U-Boot Environment for RAUC

Konsulko Group

RAUC relies on the following U-Boot environment variables:

- BOOT_ORDER - a space-separated list of boot targets in the order they should be tried
- BOOT_<bootname>_LEFT - contains the number of remaining boot attempts to perform for the respective slot

- For details:

  https://rauc.readthedocs.io/en/latest/integration.html#set-up-u-boot-environment-for-rauc

boot.cmd.in for RAUC & Raspberry Pi

Konsulko Group

```
fdt addr ${fdt_addr} && fdt get value bootargs /chosen bootargs
test -n "${BOOT_ORDER}" || setenv BOOT_ORDER "A B"
test -n "${BOOT_A_LEFT}" || setenv BOOT_A_LEFT 3
test -n "${BOOT_B_LEFT}" || setenv BOOT_B_LEFT 3
test -n "${BOOT_DEV}" || setenv BOOT_DEV "mmc 0:1"
setenv bootpart
setenv raucslot
for BOOT_SLOT in "${BOOT_ORDER}"; do
  if test "x${bootpart}" != "x"; then
    # skip remaining slots
  elif test "x${BOOT_SLOT}" = "xA"; then
    if test ${BOOT_A_LEFT} -gt 0; then
      setexpr BOOT_A_LEFT ${BOOT_A_LEFT} - 1
      echo "Found valid RAUC slot A"
      setenv bootpart "/dev/mmcblk0p2"
      setenv raucslot "A"
      setenv BOOT_DEV "mmc 0:2"
    fi
  elif test "x${BOOT_SLOT}" = "xB"; then
    if test ${BOOT_B_LEFT} -gt 0; then
      setexpr BOOT_B_LEFT ${BOOT_B_LEFT} - 1
      echo "Found valid RAUC slot B"
      setenv bootpart "/dev/mmcblk0p3"
      setenv raucslot "B"
      setenv BOOT_DEV "mmc 0:3"
    fi
  fi
done
```

```
if test -n "${bootpart}"; then
  setenv bootargs "${bootargs} root=${bootpart} rauc.slot=${raucslot}"
  saveenv
else
  echo "No valid RAUC slot found. Resetting tries to 3"
  setenv BOOT_A_LEFT 3
  setenv BOOT_B_LEFT 3
  saveenv
  reset
fi
fatload mmc 0:1 ${kernel_addr_r} @@KERNEL_IMAGETYPE@@
if test ! -e mmc 0:1 uboot.env; then saveenv; fi;
@@KERNEL_BOOTCMD@@ ${kernel_addr_r} - ${fdt_addr}
```

SquashFS & Loopback Device Support

Konsulko Group

- To install RAUC bundles the kernel used on the embedded device must support both loop block devices and the SquashFS file system
- For Raspberry Pi in linux-raspberrypi_%.bbappend:

```
do_configure_append() {
        kernel_configure_variable SQUASHFS y
        kernel_configure_variable BLK_DEV_LOOP y
        kernel_configure_variable SQUASHFS_FILE_CACHE y
        kernel_configure_variable SQUASHFS_DECOMP_SINGLE y
        kernel_configure_variable SQUASHFS_ZLIB y
        kernel_configure_variable SQUASHFS_FRAGMENT_CACHE_SIZE 3
}
CMDLINE_remove = "root=/dev/mmcblk0p2"
```

Generate RAUC Certificate

Konsulko Group

Use script openssl-ca.sh from meta-rauc to to create a certificate and a key:

- The target RAUC package must use the generated keyring file
- RAUC bundle recipe must use the generated key and certificate

- For details:
  https://github.com/rauc/meta-rauc/blob/master/scripts/README

RAUC Bundle Generator update-bundle.bb

Konsulko Group

```
DESCRIPTION = "RAUC bundle generator"

inherit bundle

RAUC_BUNDLE_COMPATIBLE = "RaspberryPi4"
RAUC_BUNDLE_VERSION = "v20200703"
RAUC_BUNDLE_DESCRIPTION = "RAUC Demo Bundle"
RAUC_BUNDLE_SLOTS = "rootfs"
RAUC_SLOT_rootfs = "core-image-minimal"
RAUC_SLOT_rootfs[fstype] = "ext4"

RAUC_KEY_FILE = "${THISDIR}/files/development-1.key.pem"
RAUC_CERT_FILE = "${THISDIR}/files/development-1.cert.pem"
```

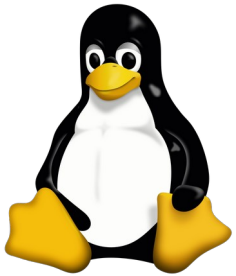
Conclusions

Konsulko Group

- RAUC is a secure, reliable, free and open source framework for A/B software updates of embedded Linux devices
- **meta-rauc-community** is the Yocto/OpenEmbedded layer providing RAUC example integration on popular embedded devices
- As of the moment **meta-rauc-community** provides examples for Raspberry Pi, QEMU x86-64, Allwinner (SunXi) and NVIDIA Jetson Tegra TX2
- More contributors wanted to extend the examples on more embedded devices

Thank You!

Konsulko Group



RAUC

yocto PROJECT · openembedded

Useful links

- Software Updates with RAUC, the Yocto Project and OpenEmbedded, Leon Anavi Yocto Project Summit 2020
  https://pretalx.com/yocto-project-summit-2020/talk/JJYPH3/
- Getting Started with RAUC on Raspberry Pi, an article at konsulko.com
  https://www.konsulko.com/getting-started-with-rauc-on-raspberry-pi-2/
- Behind the Scenes of an Update Framework: RAUC, Enrico Jörns, ELCE 2019
  https://www.youtube.com/watch?v=ZkumnNsWczM
- Embedded Recipes 2019 - Remote update adventures with RAUC, Yocto and Barebox
  https://www.youtube.com/watch?v=hS3Fjf7fuHM
- Secure and Safe Updates for Your Embedded Device, Enrico Jörns, FOSDEM 2017
  https://archive.fosdem.org/2017/schedule/event/secure_safe_embedded_updates/