Secure Field Update Systems

Innovation Workshop
"Today's security challenges of embedded systems and solutions"

Enrico Jörns – e.joerns@pengutronix.de

https://www.pengutronix.de

About Me

- Embedded software developer
- Co-maintainer of FOSS update framework RAUC
- At Pengutronix since 2014

- Embedded Linux consulting & support since 2001
- > 4500 patches in Linux kernel

OTA Updating – The Key To Security

- Fixing critical bugs (CVE‘s)
- Keeping software up to date

- Unattended updates

  → Fail Safe

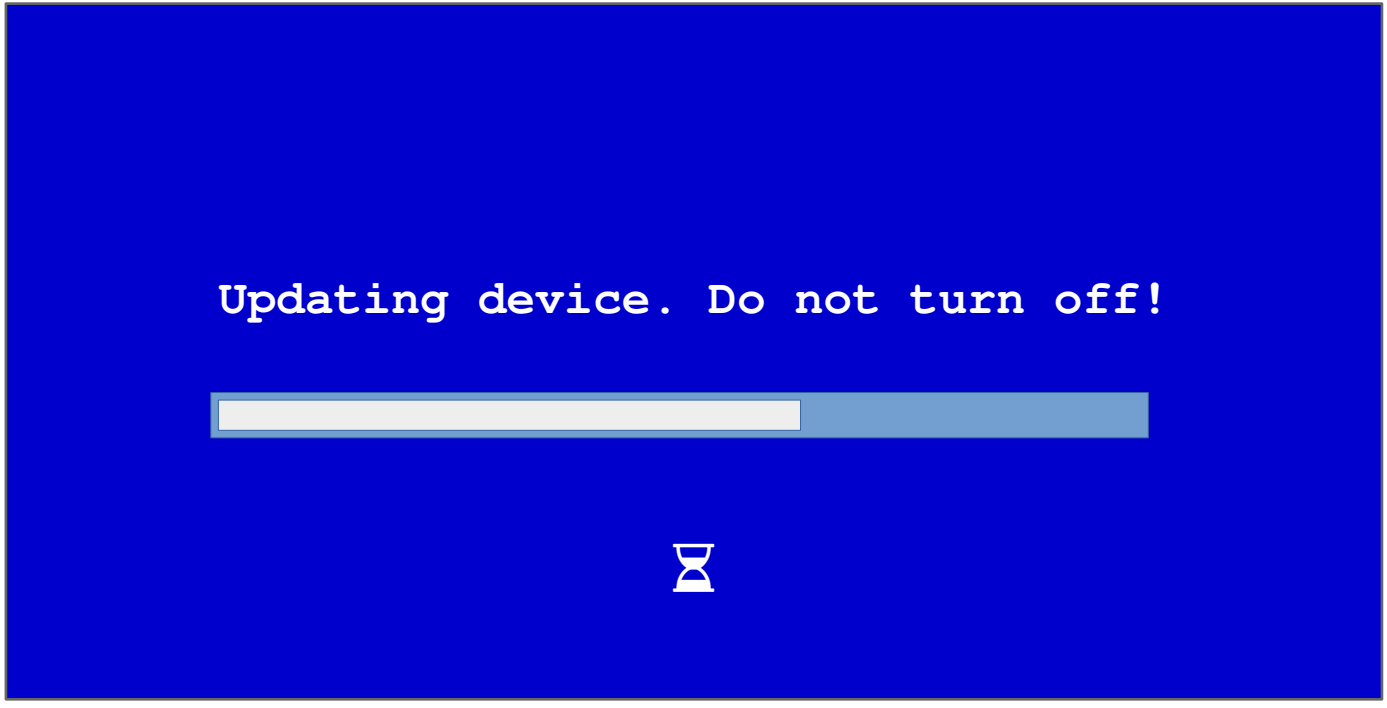
- Insecure environment

  → Secure the update process

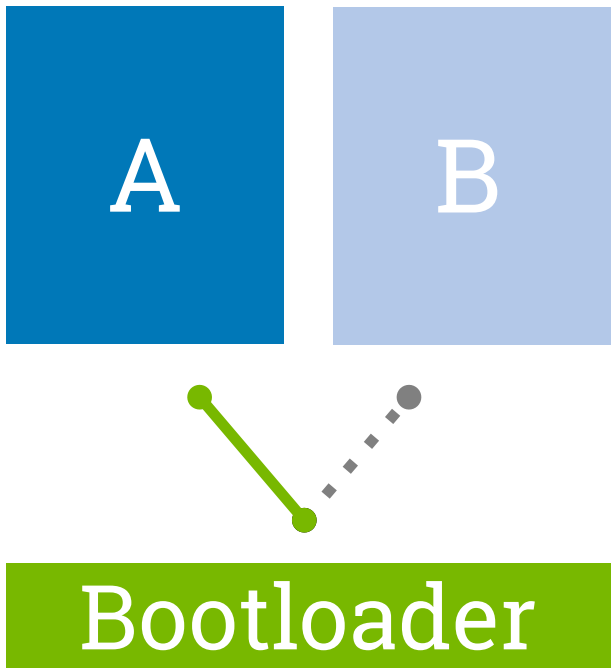
Image-Based Updating

- ✖ Package-based updates inappropriate
  - ✖ Require interactive administration
  - ✖ Conflicts
  - ✖ Untested combinations
  - ✖ Affected by file system corruption
- ✔ Full image-based system updates
  - ✔ Well-defined state (reproducibility!)
  - ✔ Well-tested state of application + software

Fail-Safe Updating



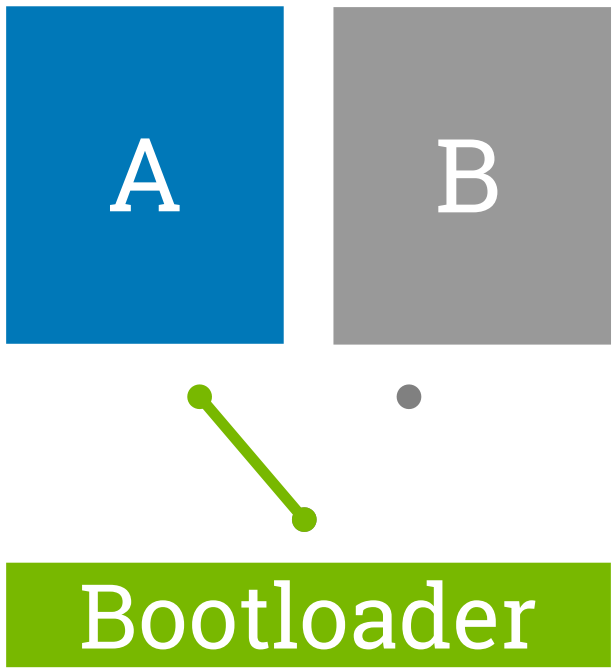
Fail-Safe Updating



A: Active (running) system

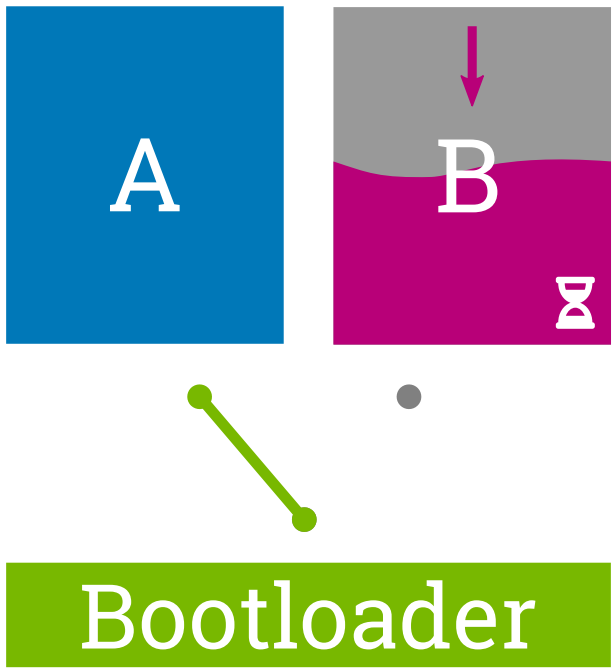
B: Non-running system

Fail-Safe Updating



- Deactivate partition to update

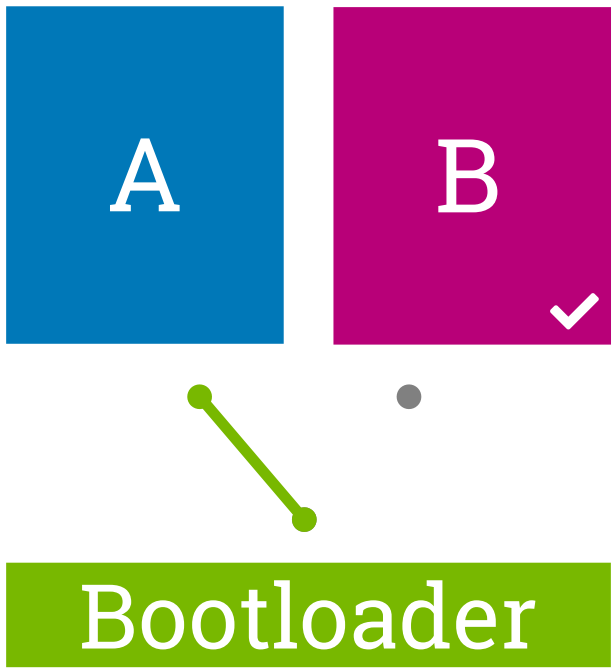
Fail-Safe Updating



- Deactivate partition to update
- Write update(s) to disk

  Critical Operation!

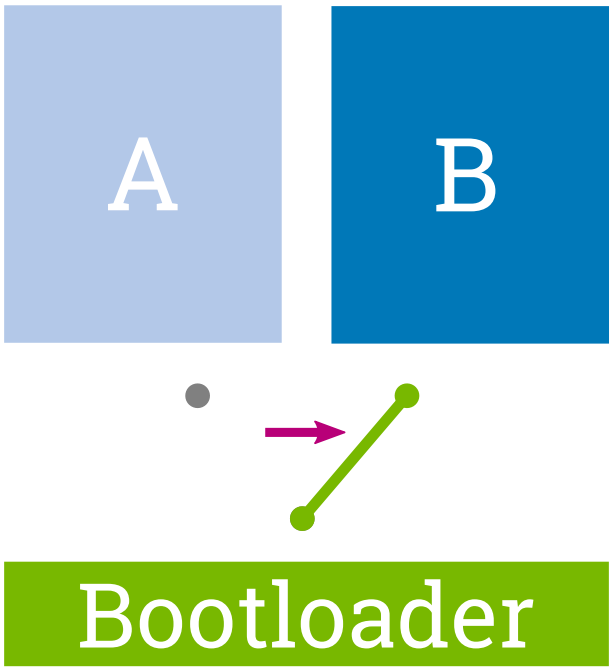
Fail-Safe Updating



- Deactivate partition to update
- Write update(s) to disk

  Critical Operation!
- Update fully completed + verified, etc.

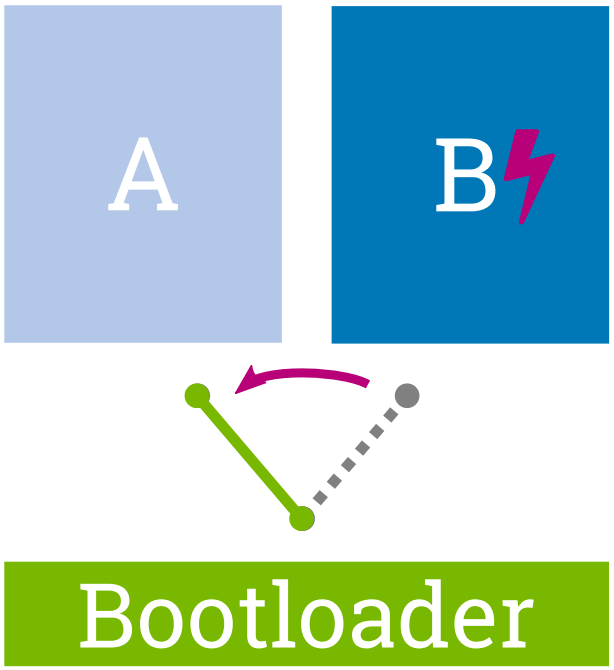
Fail-Safe Updating



- Deactivate partition to update
- Write update(s) to disk

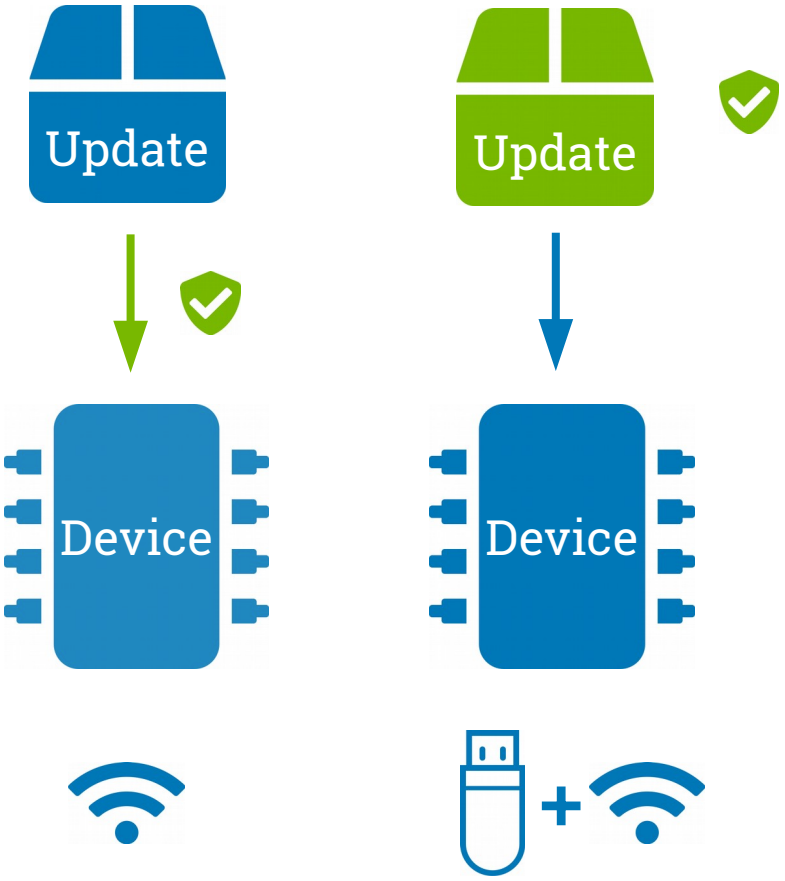
  Critical Operation!
- Update fully completed + verified, etc.
- Activate updated slot

Fail-Safe Updating: Fallback



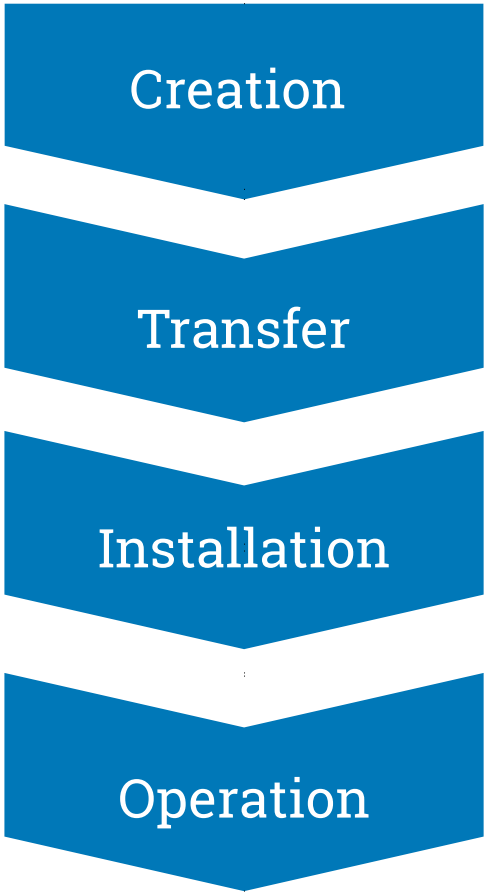
- Boot Failure / Runtime crash
- Fall back to other slot
- Availability
- Potential downgrade!

Updating – Authentication



- Prevent unauthorized access!
- Signing the update artifact
- Verification on target
- Use well-proven & open source crypto (OpenSSL)

What Could Go Wrong?



- Compromised key
- Misconduct of employees

- Man-in-the-Middle (OTA or USB stick)
- Replay / Freshness

- Manipulated file system / protocols
- Downgrades
- Manipulated system time
- Interruptions
- Wrong target system

- Persistent manipulation of update system (keys / code)

Robust Tools

- Error detection and handling
- Modularization
- Standard crypto algorithms und libraries
- Verification, data integrity
- No project-specific hand-crafted solution

→ **Open Source Update Frameworks!**

mender.io

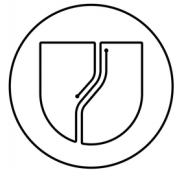
- Open Source Client + Server
- A+B update / update modules
- Trusted connection (TLS)
- Update streaming
- „Hosted Mender“
- .mender files → tar archives + json file hierarchy

Code: Go
License: Apache-2.0
Since: 2015.12
Contributors: 25

Yocto: meta-mender

https://github.com/mendersoftware/mender

swupdate



- Kconfig
- RSA / CMS for signing (semi-automated)
- Symmetric encryption
- Lua scripting
- sw-description file, using the libconfig syntax or JSON
- Update-defined installation

Code: C
License: GPL-2.0
Since: 2014.07
Contributors: 61

Yocto: meta-swupdate

https://github.com/sbabic/swupdate

RAUC

- Built-in signing (mandatory)
- X.509 (CMS), PKCS#11
- Flexible and extensible
- Application-controlled update process
- Binary delta update streaming (casync)

Code: C (with glib)
License: LGPL-2.1
Since: 2015.04
Contributors: 41

Yocto: meta-rauc

https://github.com/rauc/rauc

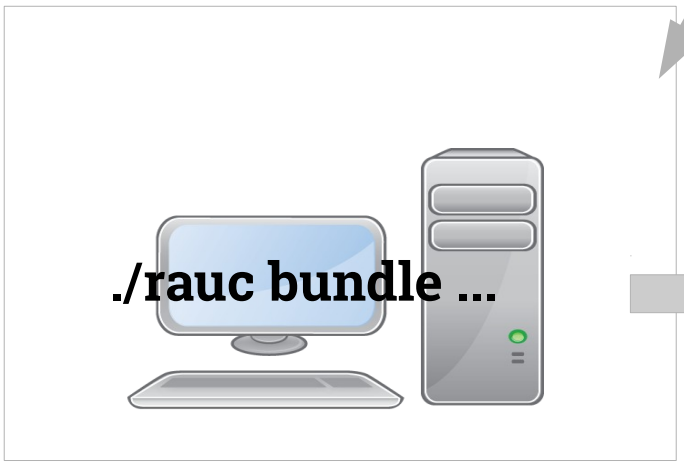
Robust Auto-Update Controller

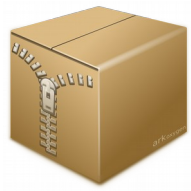
RAUC

source code

compile

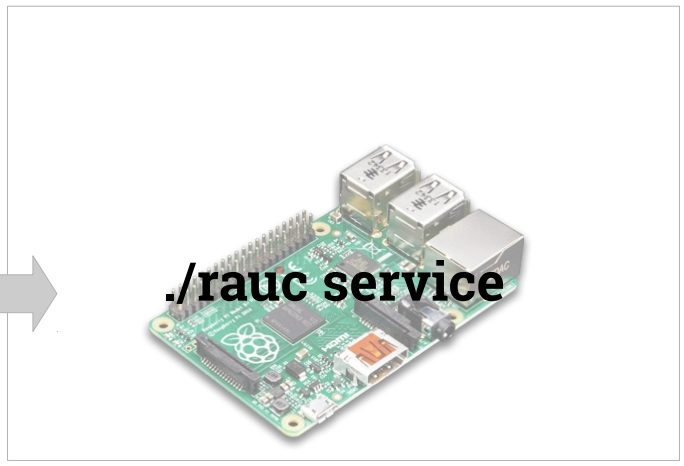
host tool





Bundle

target service (+ CLI)

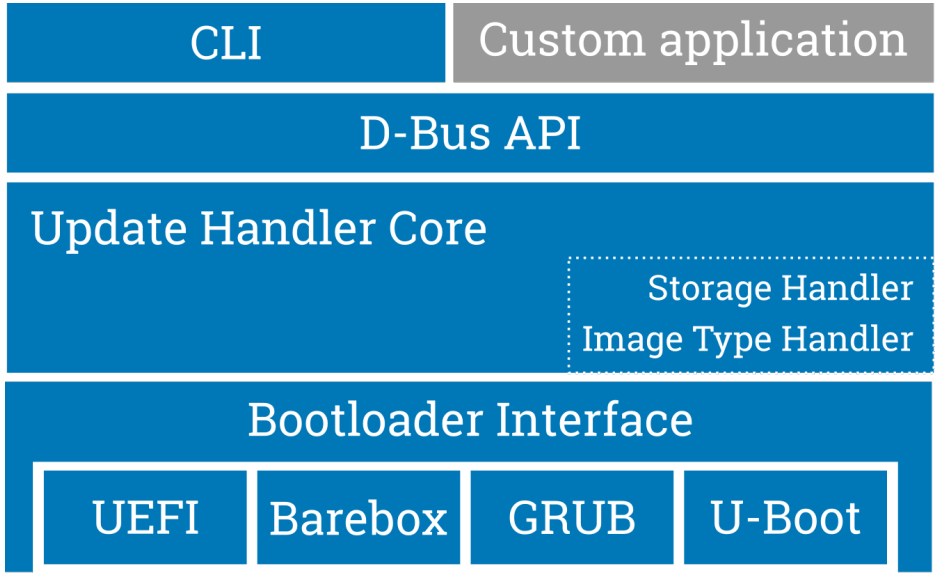


→ Creating / inspecting updates

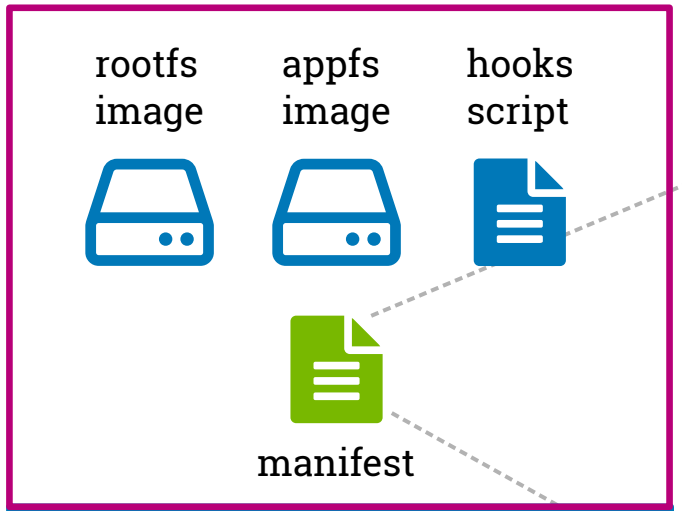
→ Installation of updates

RAUC – Update Service

- Focus on being update core
- Configurable for different use cases
- Extensible with hooks + handlers
- FS image + tar support



RAUC – Update Bundle Format



squashfs

RAUC manifest

```
[update]
compatible=MyProduct2000
version=2019.06-2
build=20190627134503

[image.rootfs]
filename=image.ext4
size=419430400
sha256=b14c1457dc1046...

[image.appfs]
filename=app.img
size=219430400
sha256=ecf4c031d01cb9...
hooks=post-install
```

A bundle defines the system's entire target state

RAUC – System Configuration

```
[system]
compatible=MyProduct2000
bootloader=barebox

[keyring]
path=/etc/rauc/keyring.pem

[slot.rootfs.0]
device=/dev/sda0
type=ext4
bootname=system0

[slot.rootfs.1]
device=/dev/sda1
type=ext4
bootname=system1

[...]
```

Target device: RAUC system configuration

```
[update]
compatible=MyProduct2000
version=2019.06-202
build=20190627134503

[image.rootfs]
filename=rootfs.ext4
size=419430400
sha256=b14c1457dc1046...

[image.appfs]
filename=appfs.ext4
size=219430400
sha256=ecf4c031d01cb9...
hooks=post-install
```

Update Bundle: manifest

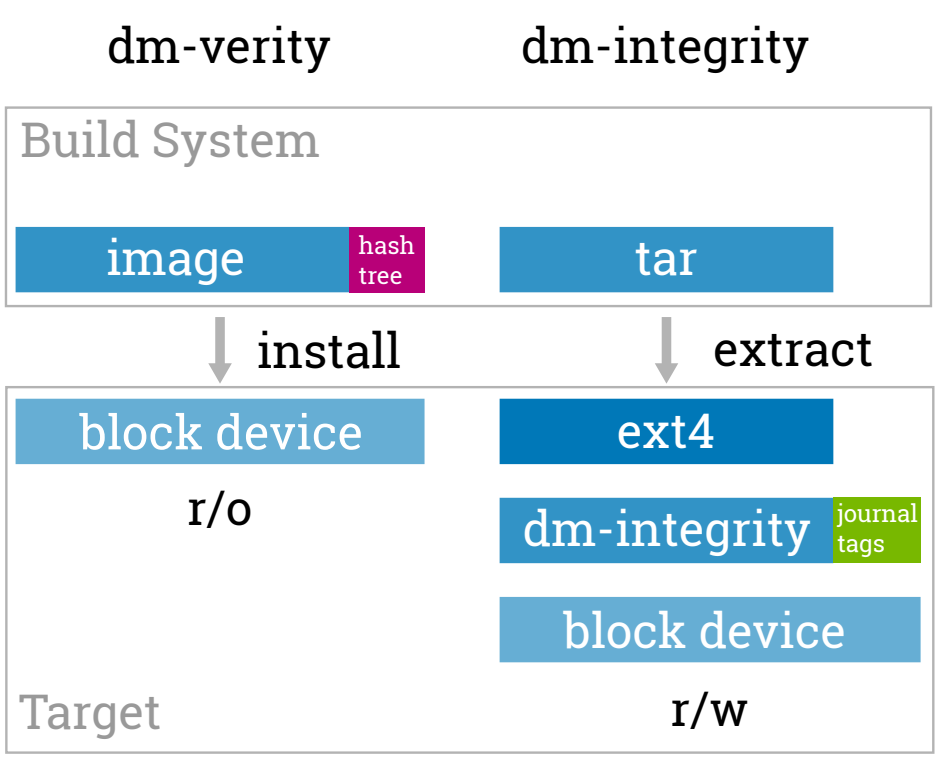
RAUC – Integration

- Supported in all common embedded Linux build systems

- Command line interface for testing / basic usage
- Controllable by custom applications (via D-Bus)

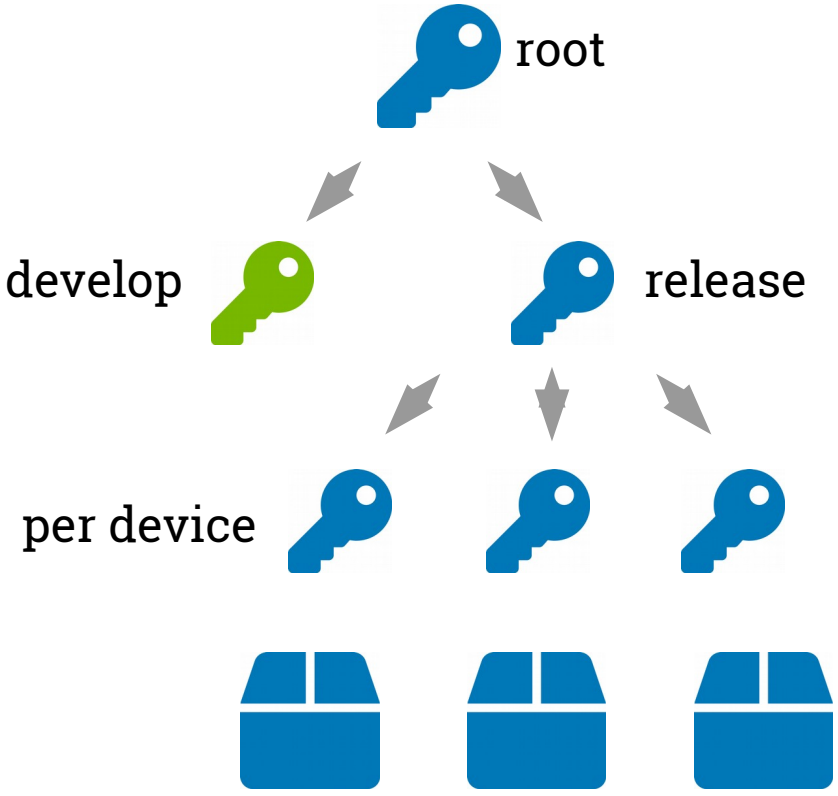
- Hawkbit deployment server
  - rauc-hawkbit-client (python, C)

Updates and Verified Boot



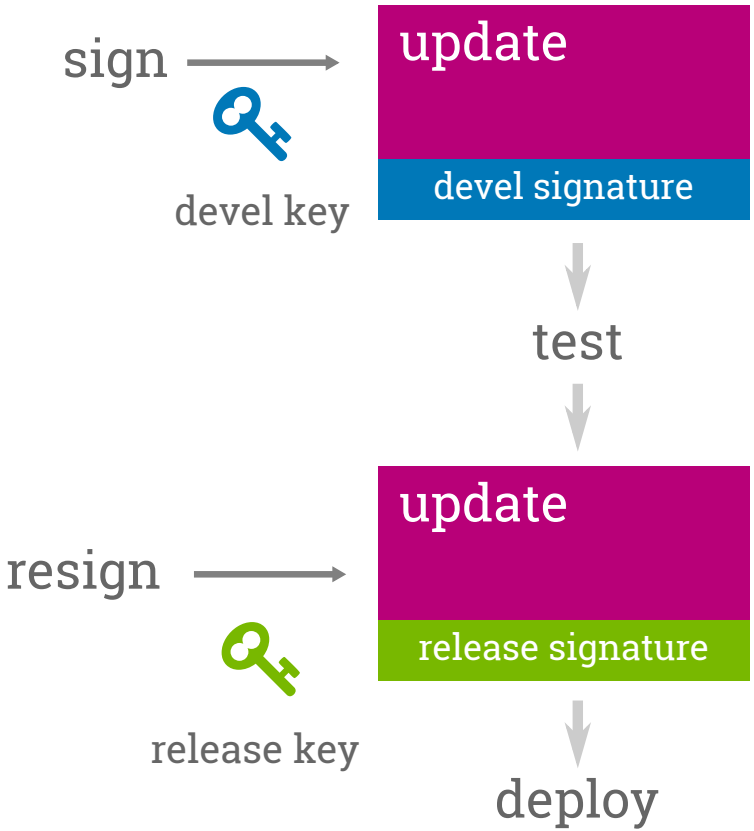
- Image-based updates:
  - Signatures for verified boot remain valid
- File system-based updates:
  - Possible only with dm-integrity/dm-crypt (AEAD)

Authentication – X.509 PKI

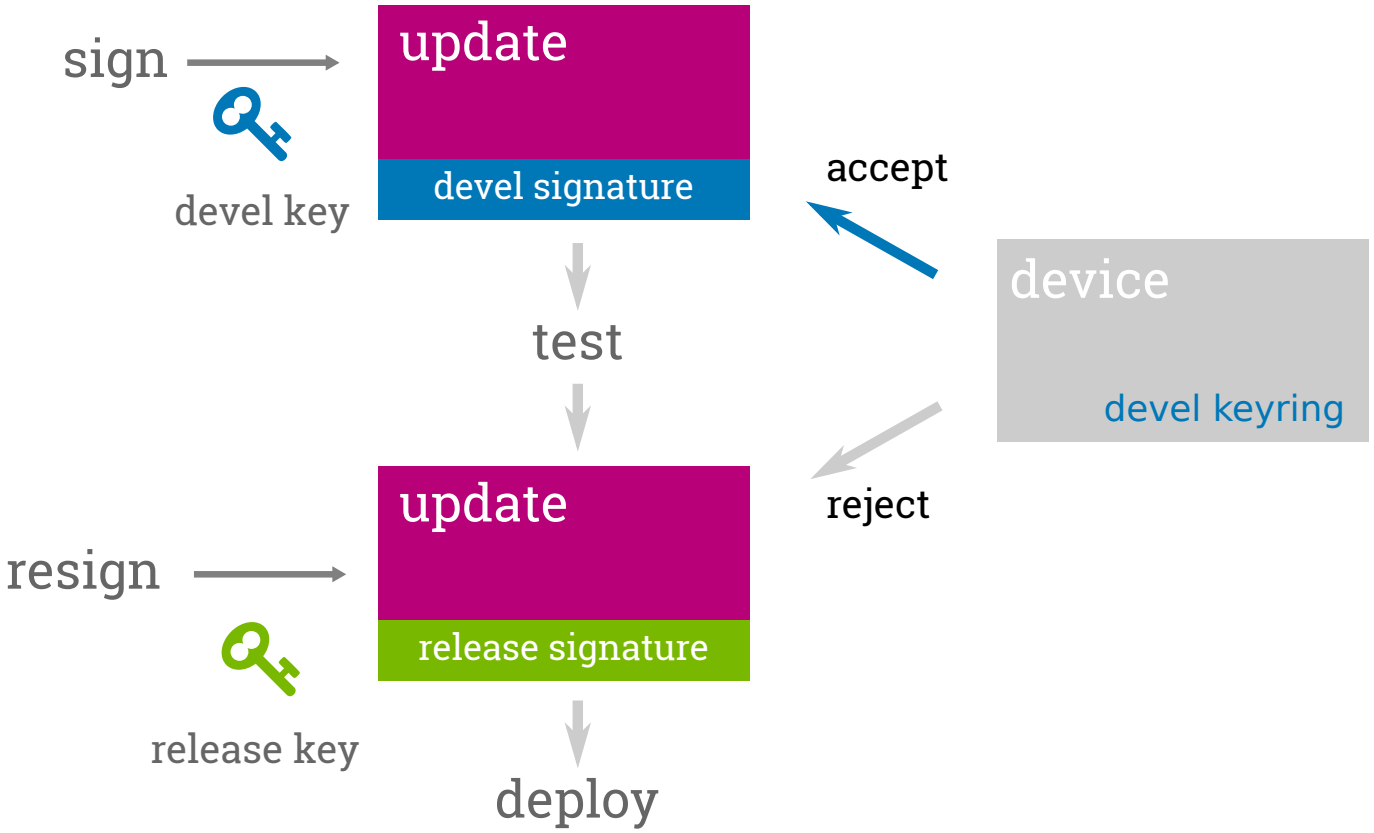


- Self-signed (for testing!)
- Development vs. release key
- Per-device keys
- Replace and revoke keys

Re-Signing

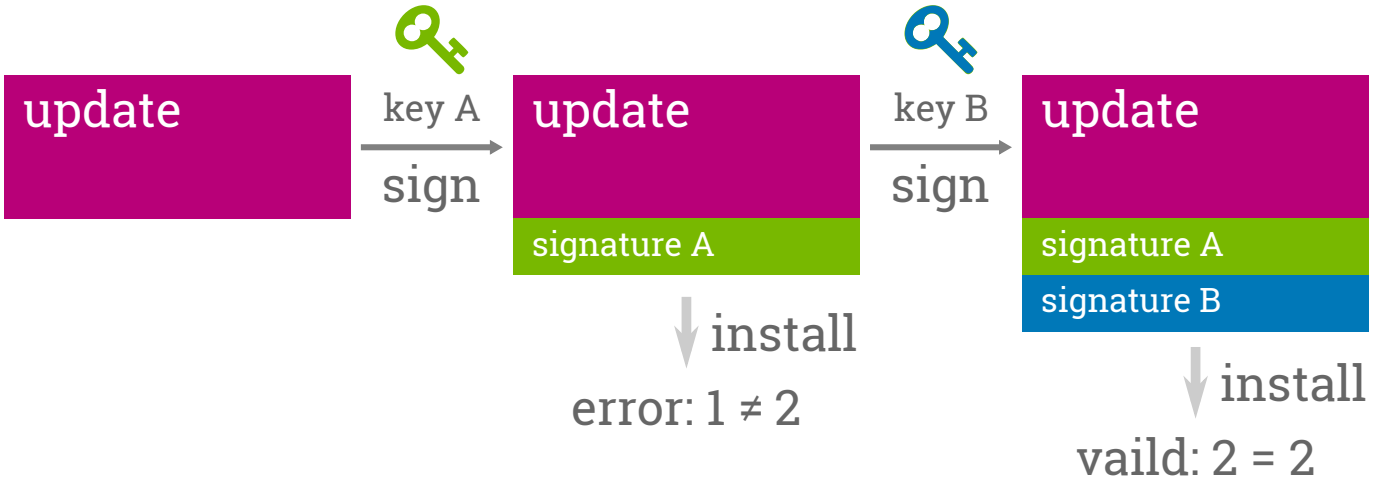

sign
update
devel signature
devel key
test
update
resign
release signature
release key
deploy

Re-Signing (with RAUC)



Threshold-Signatures

- N of M: of M keys at least N signatures required



→ No one can release an update on his own

  - The impact of a compromised key is limited

Encrypted Updates

- Where is the key located?
  - As clear-text in the system
    - One key per product variant
    - One key per device
  - Secured by a TPM/TEE
    - Also interesting for VPN keys

RAUC – Security Features

- ✔ Standard signature methods: X.509/CMS/PKCS#11/OpenSSL
- ✔ Certificate revocation / -expiry
- ✔ Intermediate Certificates
- ✔ Re-signing
- 🔧 Threshold signatures
- 🔧 Signed Timestamps
- 🔧 Encryption

Thank you!

Questions?

e.joerns@pengutronix.de

Links

- https://github.com/rauc/rauc
- https://mender.io/
- https://github.com/sbabic/swupdate
- https://rauc.readthedocs.io/en/latest/