RAUC: (R)evolution of an Update Framework

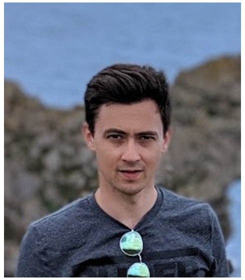
Embedded Linux Conference Europe 2022

Enrico Jörns – e.joerns@pengutronix.de

Pengutronix.

https://www.pengutronix.de

About Me & Pengutronix



- Embedded software developer
- RAUC co-maintainer
- Team Lead Integration at Pengutronix

Pengutronix.

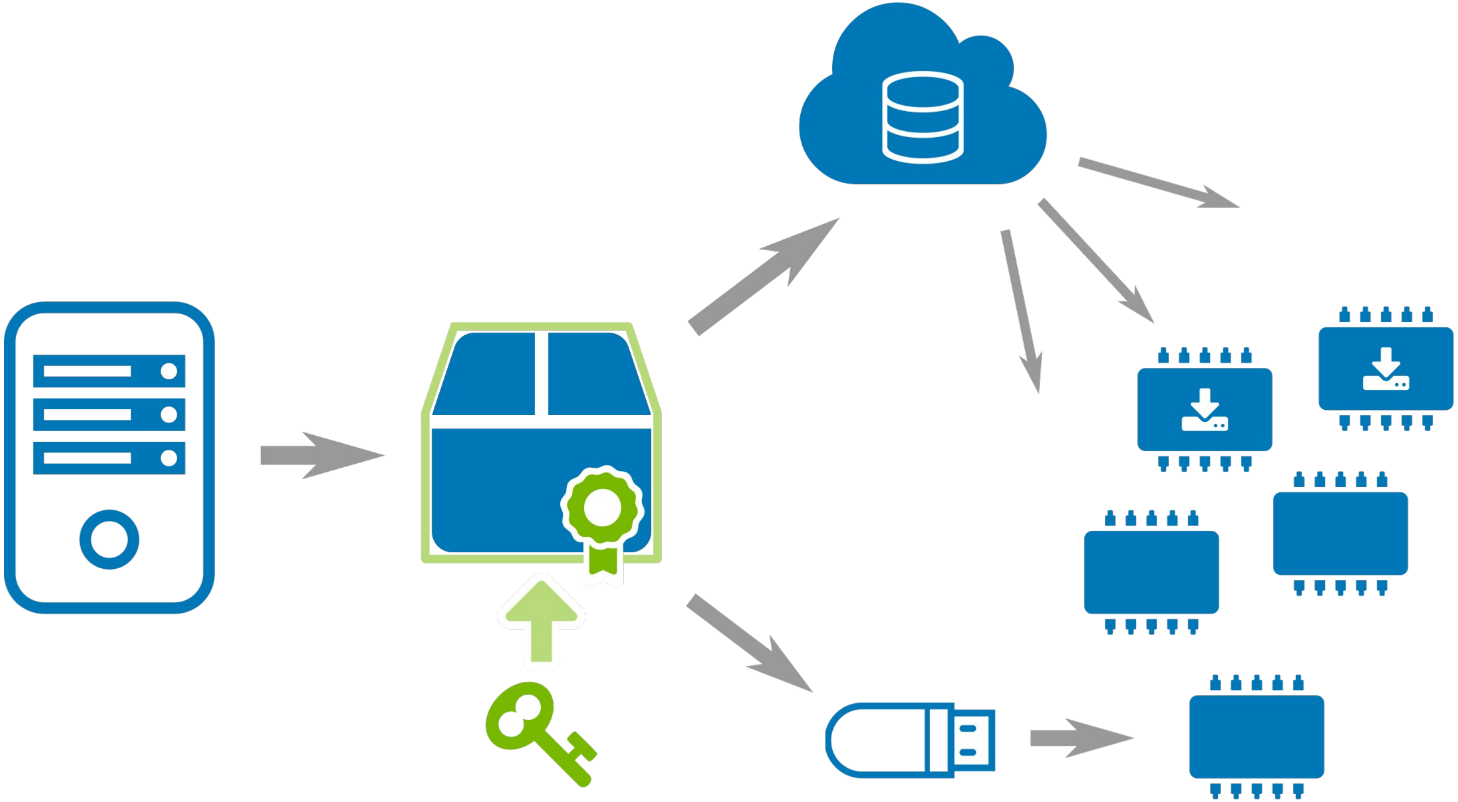
- Embedded Linux consulting & support since 2001
- ~ 6000 patches in Linux kernel

Structure

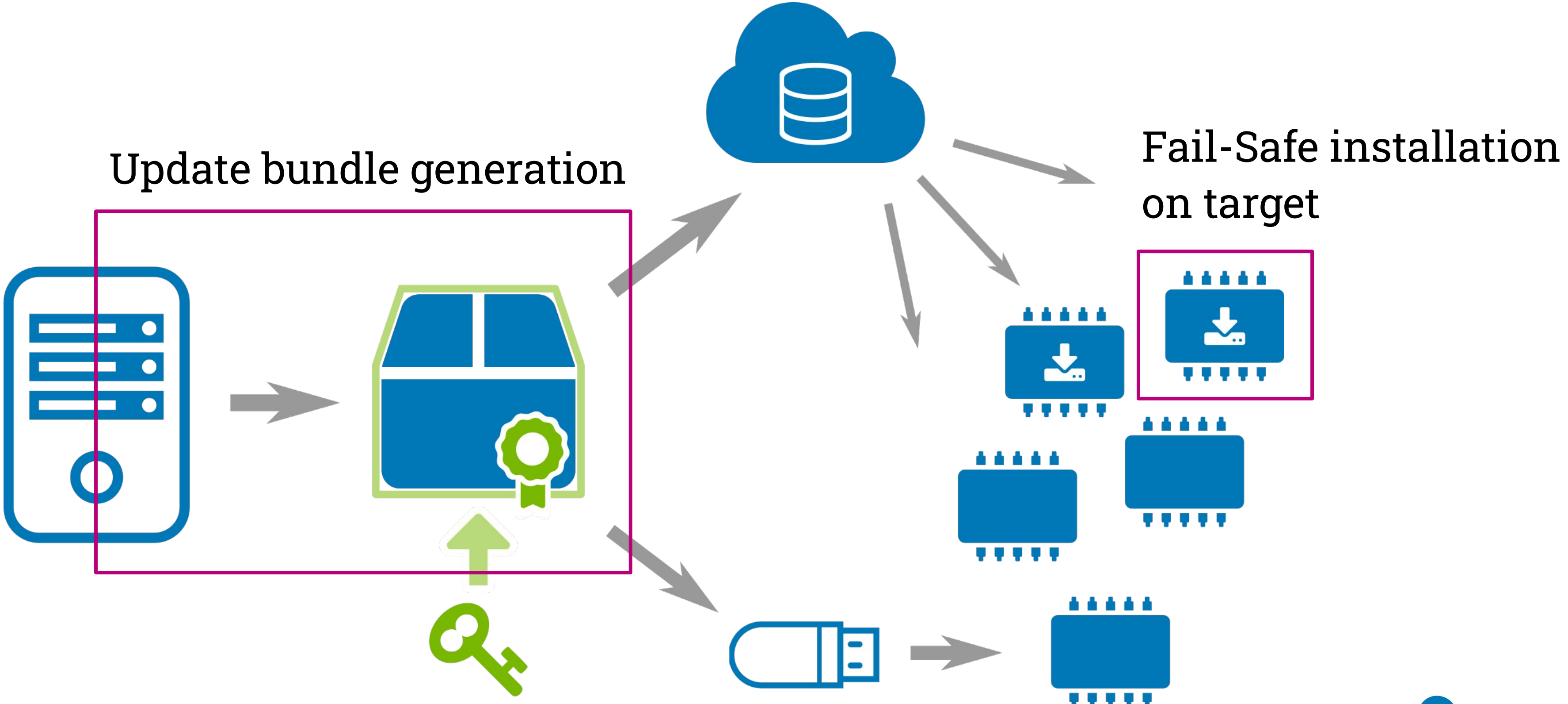
- Introduction + Overview
- Initial Bundle Format
- Verity Bundle Format
- HTTP(S) bundle streaming
- Adaptive Updates
- Encrypted Bundles
- Outlook & Community

Overview

(OTA) Field Updates



RAUC – Scope


Update bundle generation
Fail-Safe installation
on target

RAUC

- An Embedded Linux update framework
  - Written in C (with glib, OpenSSL, curl, ...)
  - LGPL-2.1 License
  - Hosted on GitHub: https://github.com/rauc/rauc

- Fail-Safe (image-based) atomic (A/B) updates
- Cryptographic signing + verification of updates

https://rauc.readthedocs.io/

RAUC – Configuration Basics

```
[system]
compatible=Test System
bootloader=u-boot

[slot.rootfs.0]
device=/dev/mmcblk0p1
...

[slot.rootfs.1]
device=/dev/mmcblk0p2
...
```

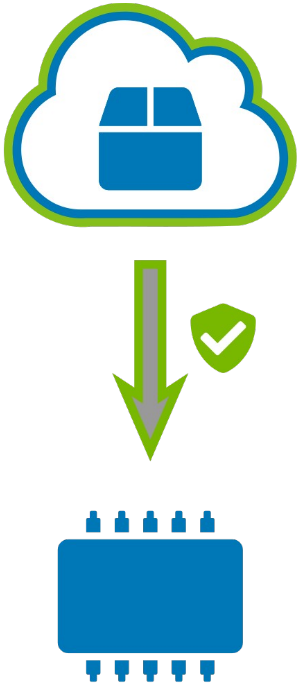
System configuration → on target

```
[update]
compatible=Test System
version=2022.09

[bundle]
format=verity

[image.rootfs]
image=rootfs.ext4
```

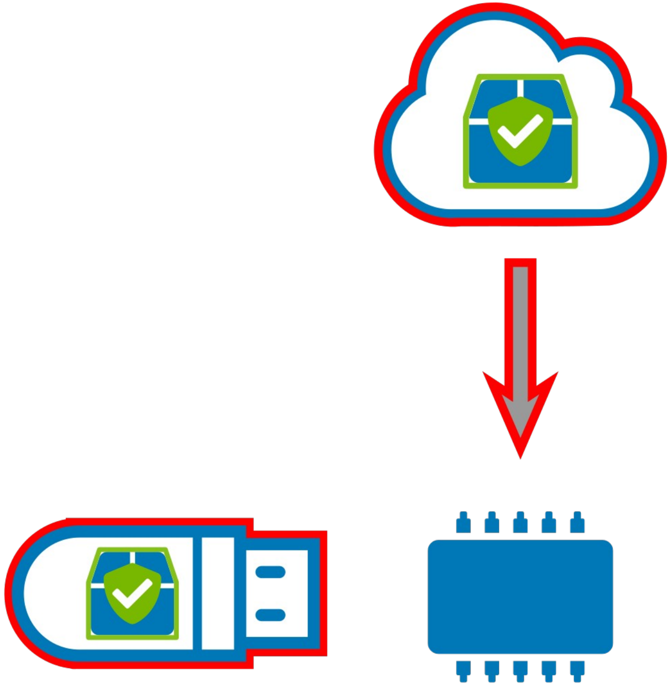
Update manifest → in bundle

Initial Bundle Format

Authenticated Artifacts

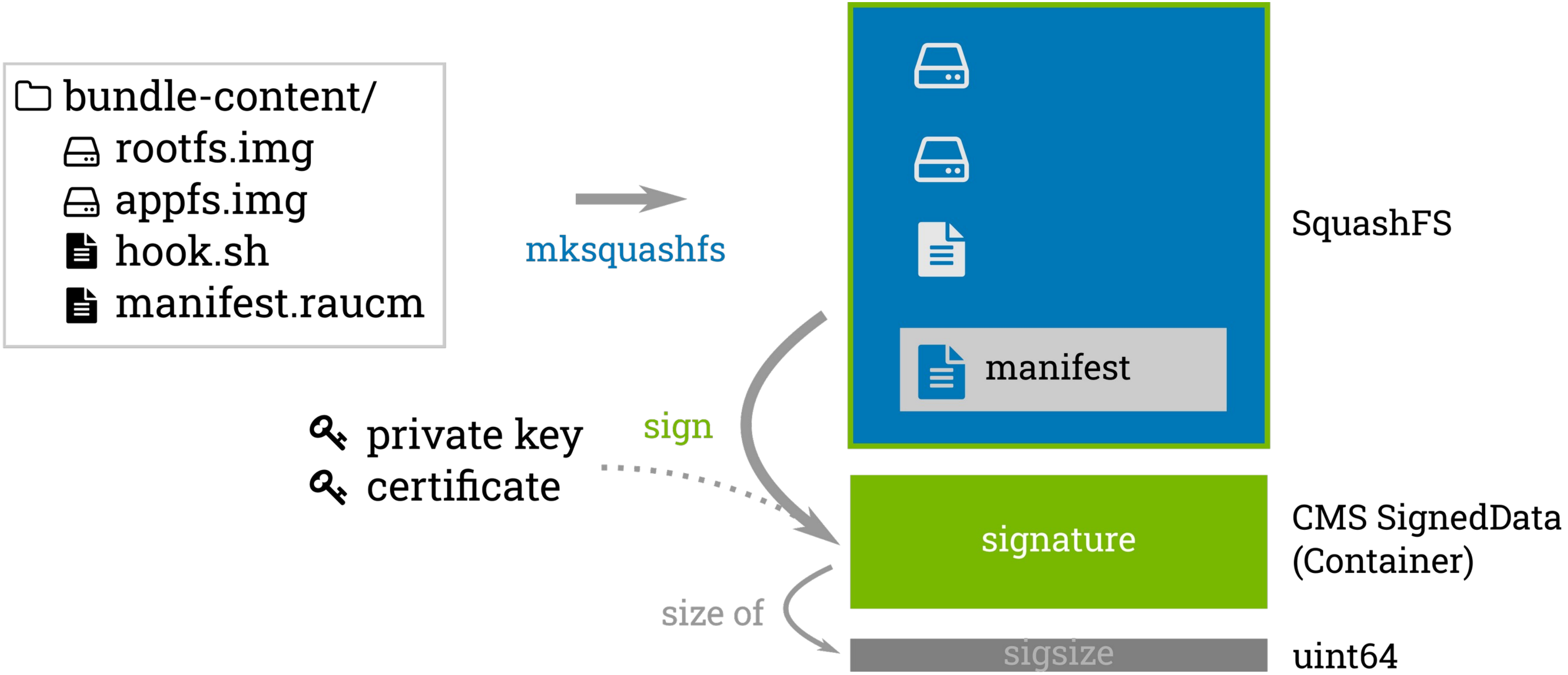


authenticated channel



authenticated artifact

Initial Bundle Format – Generation



🗀 bundle-content/
- rootfs.img
- appfs.img
- hook.sh
- manifest.raucm

mksquashfs

manifest

SquashFS

private key

certificate

sign

signature

CMS SignedData (Container)

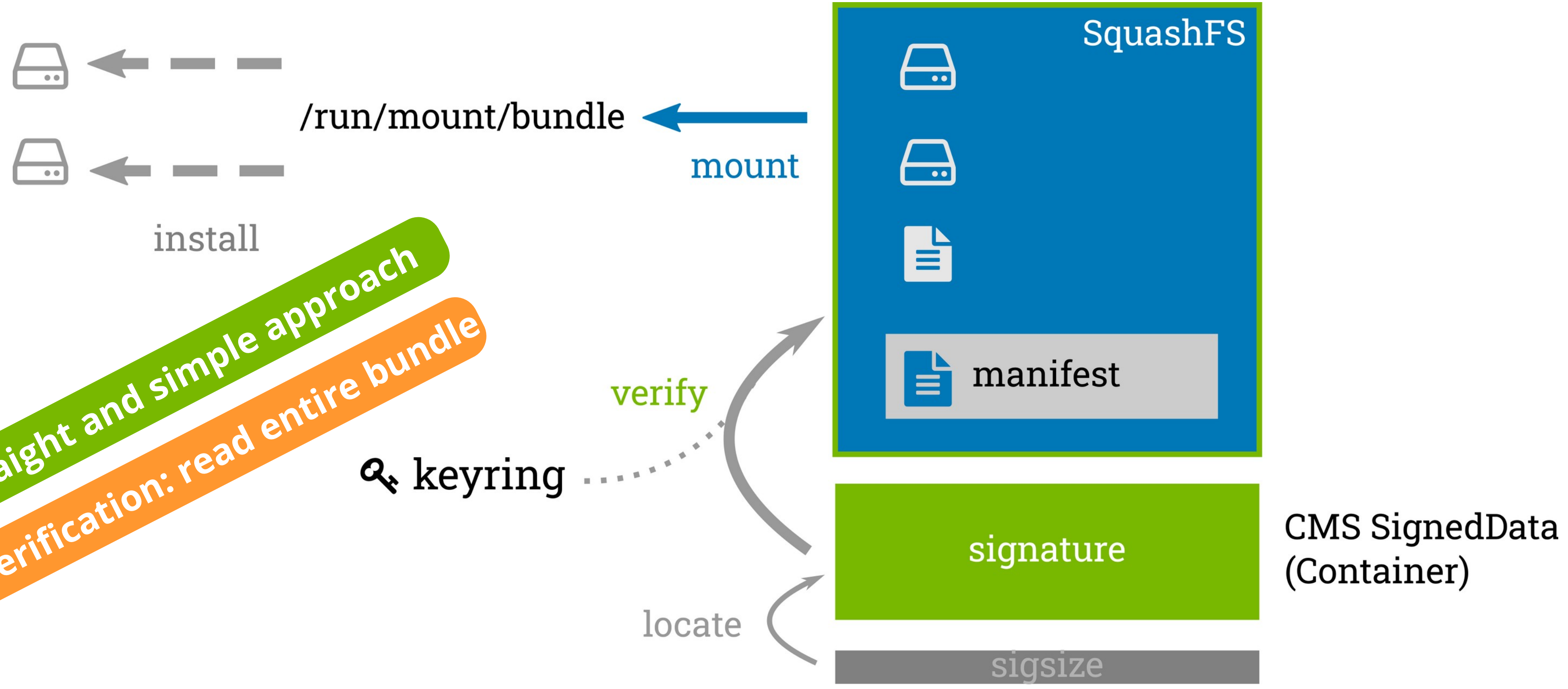
size of

sigsize

uint64

```
rauc bundle --key=… --cert=… bundle-content/ update.raucb
```

Initial Bundle Format – Verification

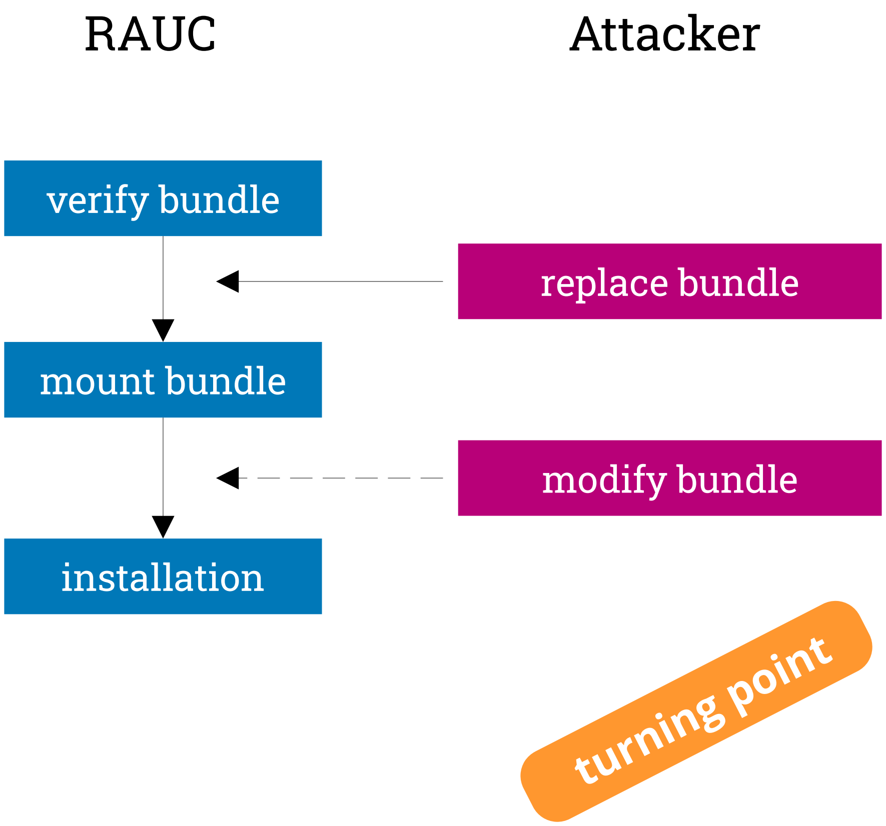


✓ straight and simple approach

✗ verification: read entire bundle

```
rauc install --keyring=… update.raucb
```

CVE-2020-25860

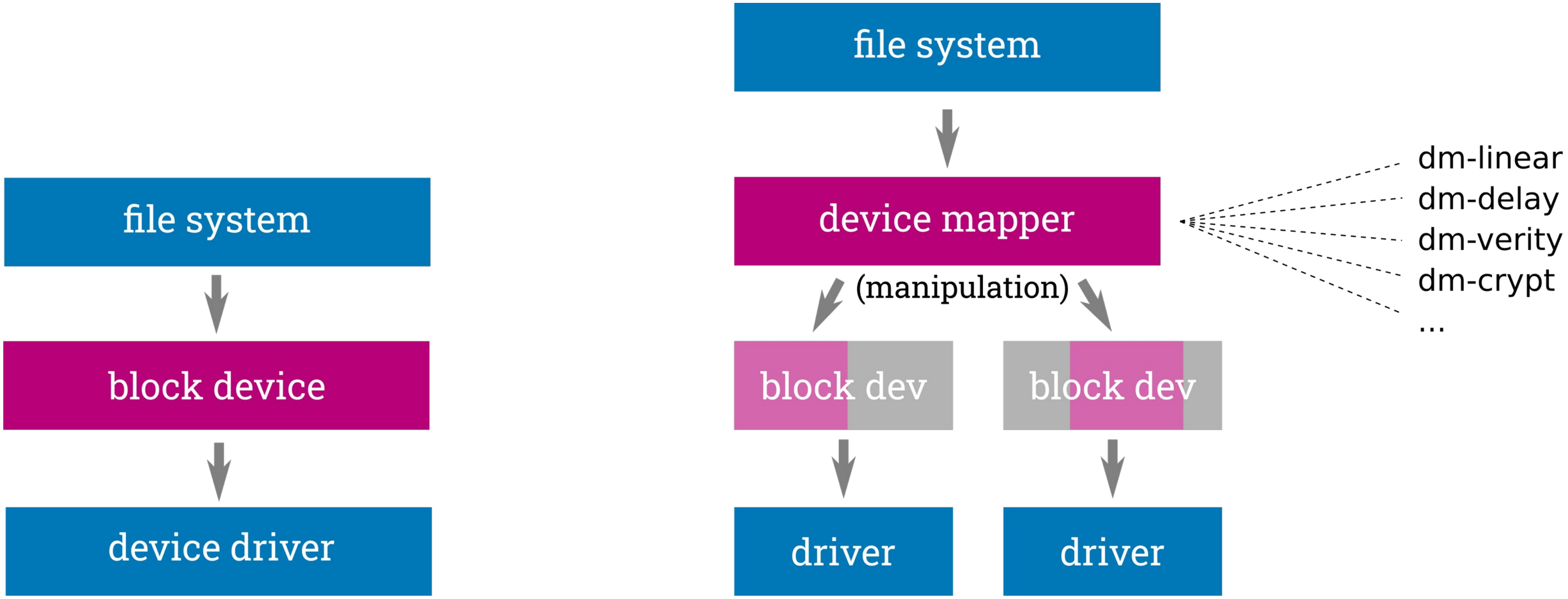


- TOCTOU vulnerability (CWE 367)
- Disclosure Date: 12/21/2020
- Fixed in RAUC 1.5
- Mitigations
  - Do not close fd
  - Ensure exclusive access
- Need for a better bundle format → ‘verity‘

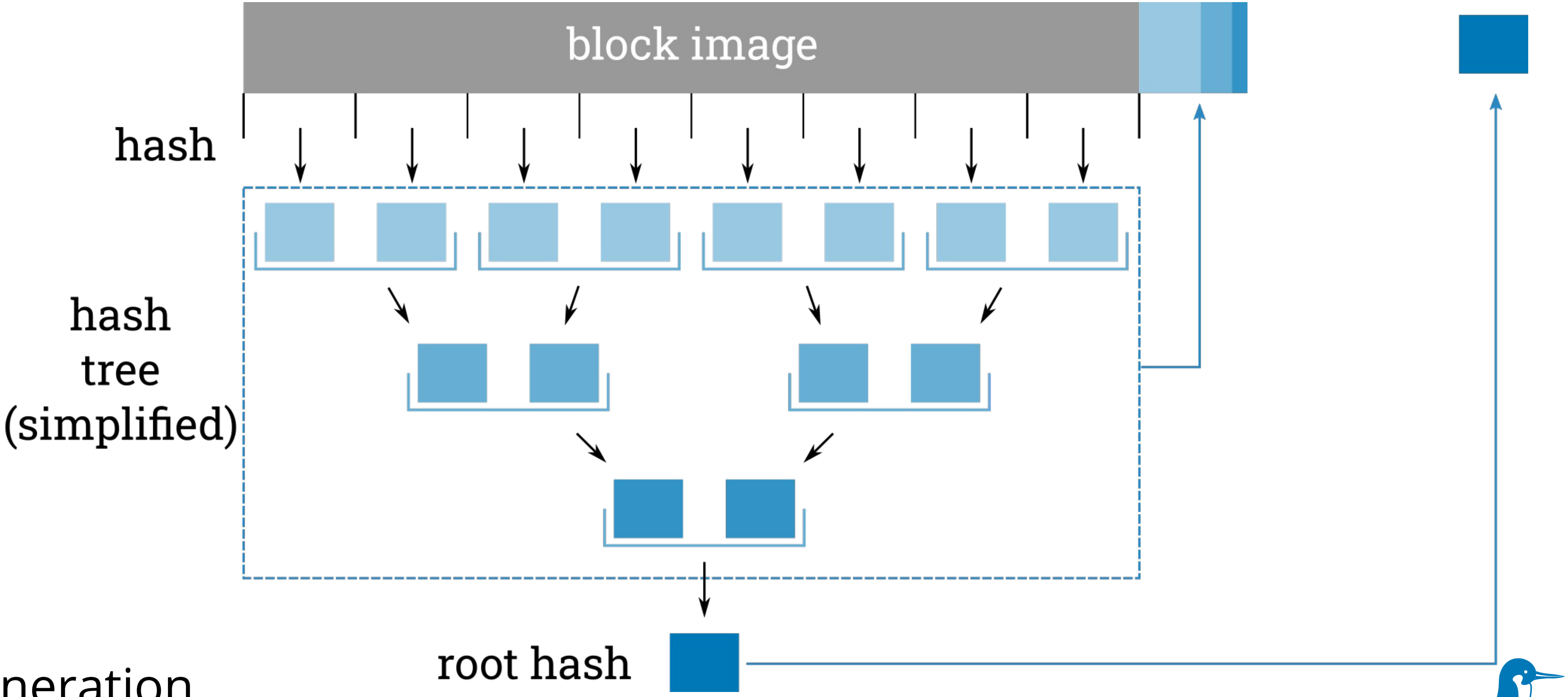
https://github.com/rauc/rauc/security/advisories/GHSA-cgf3-h62j-w9vv

Verity Bundle Format

Background: Kernel Device Mapper

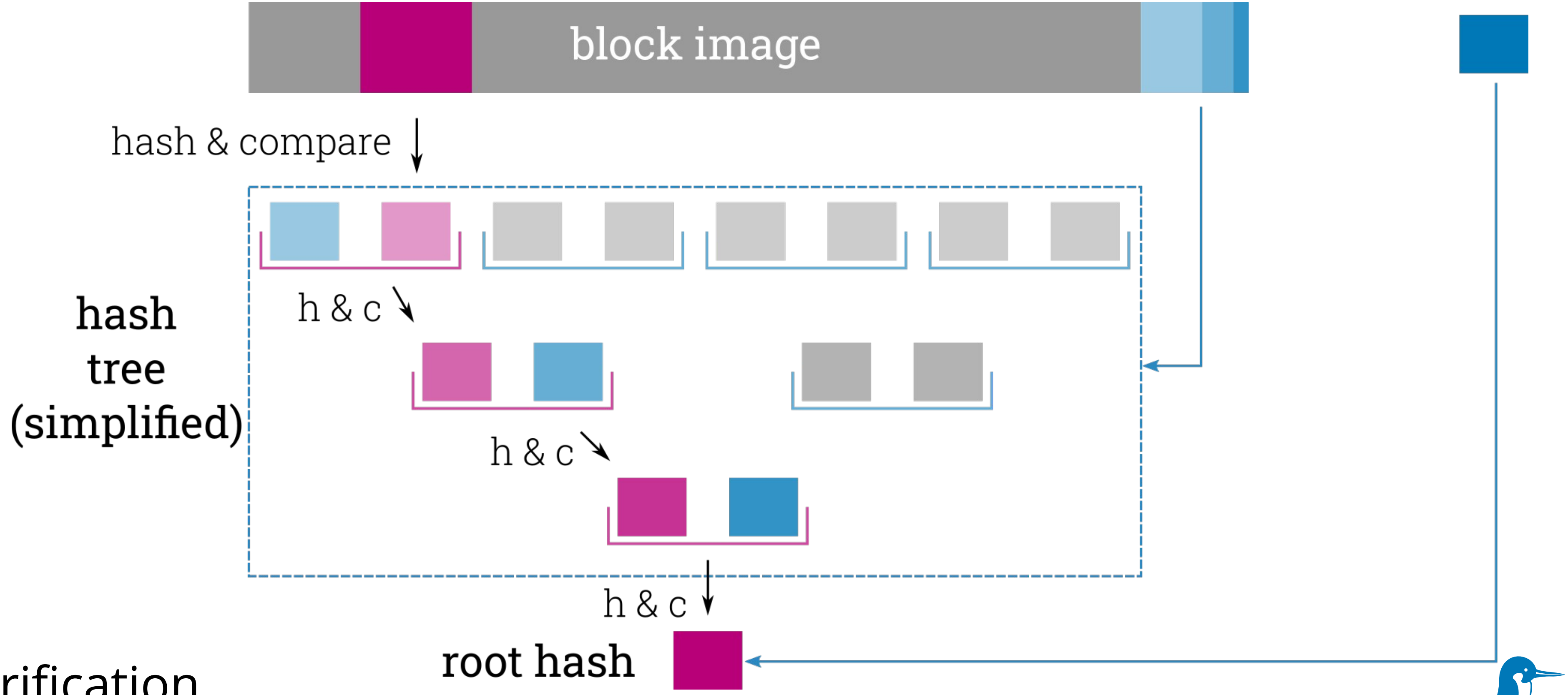


Kernel Device Mapper – dm-verity



Generation

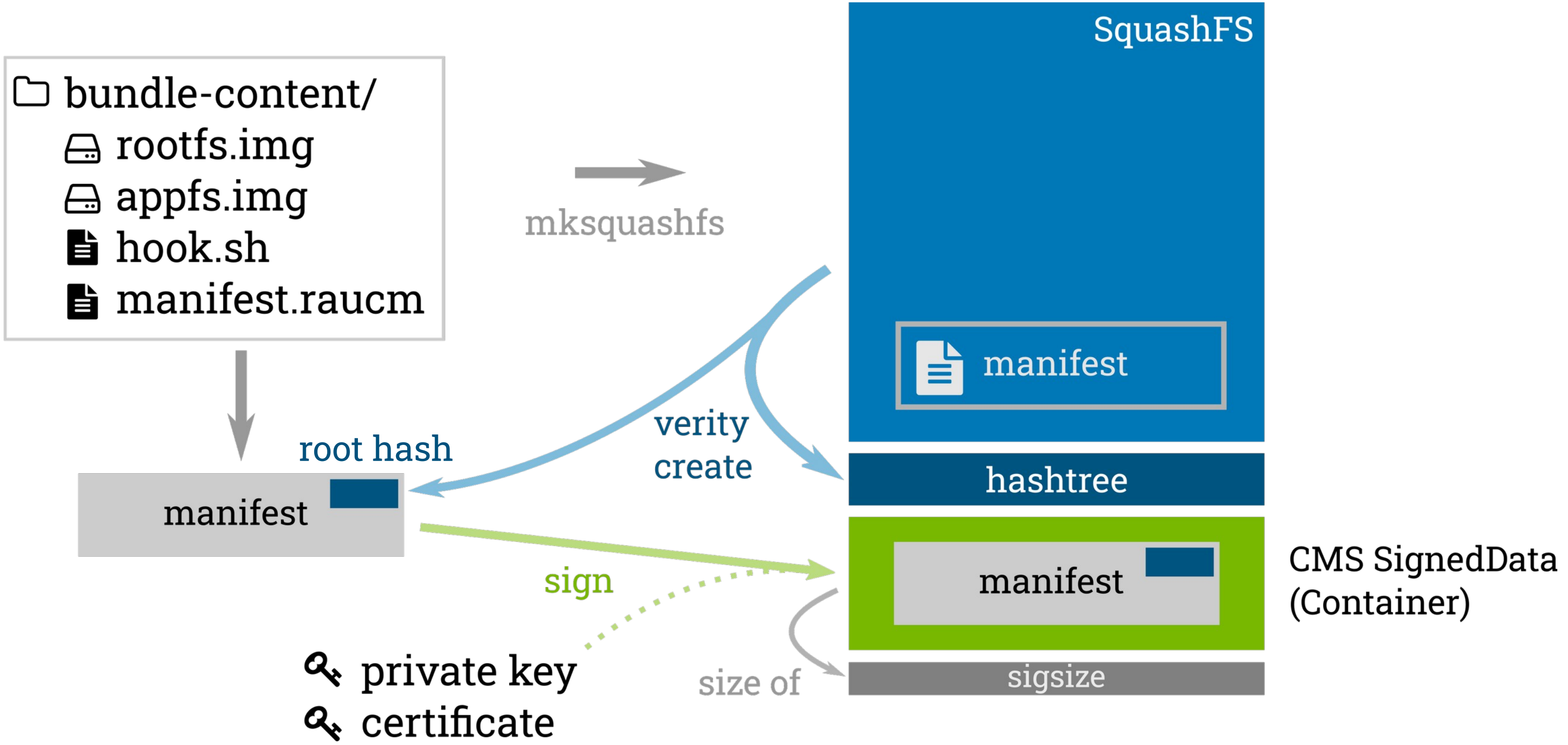
Kernel Device Mapper – dm-verity



Verification

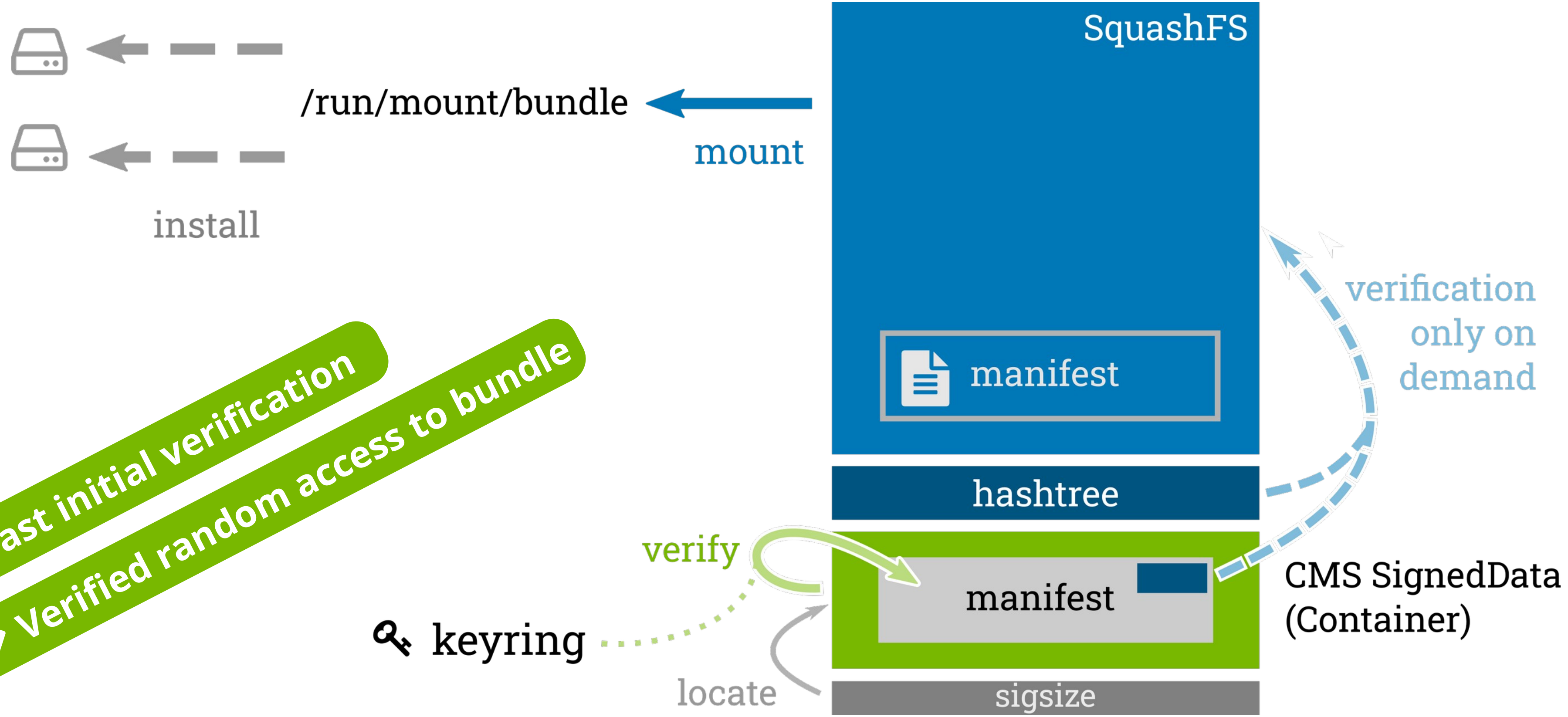
New verity Bundle Format – Generation

```
[bundle]
format=verity
```



```
rauc bundle --key=… --cert=… bundle-content/ update.raucb
```

New `verity` Bundle Format – Verification

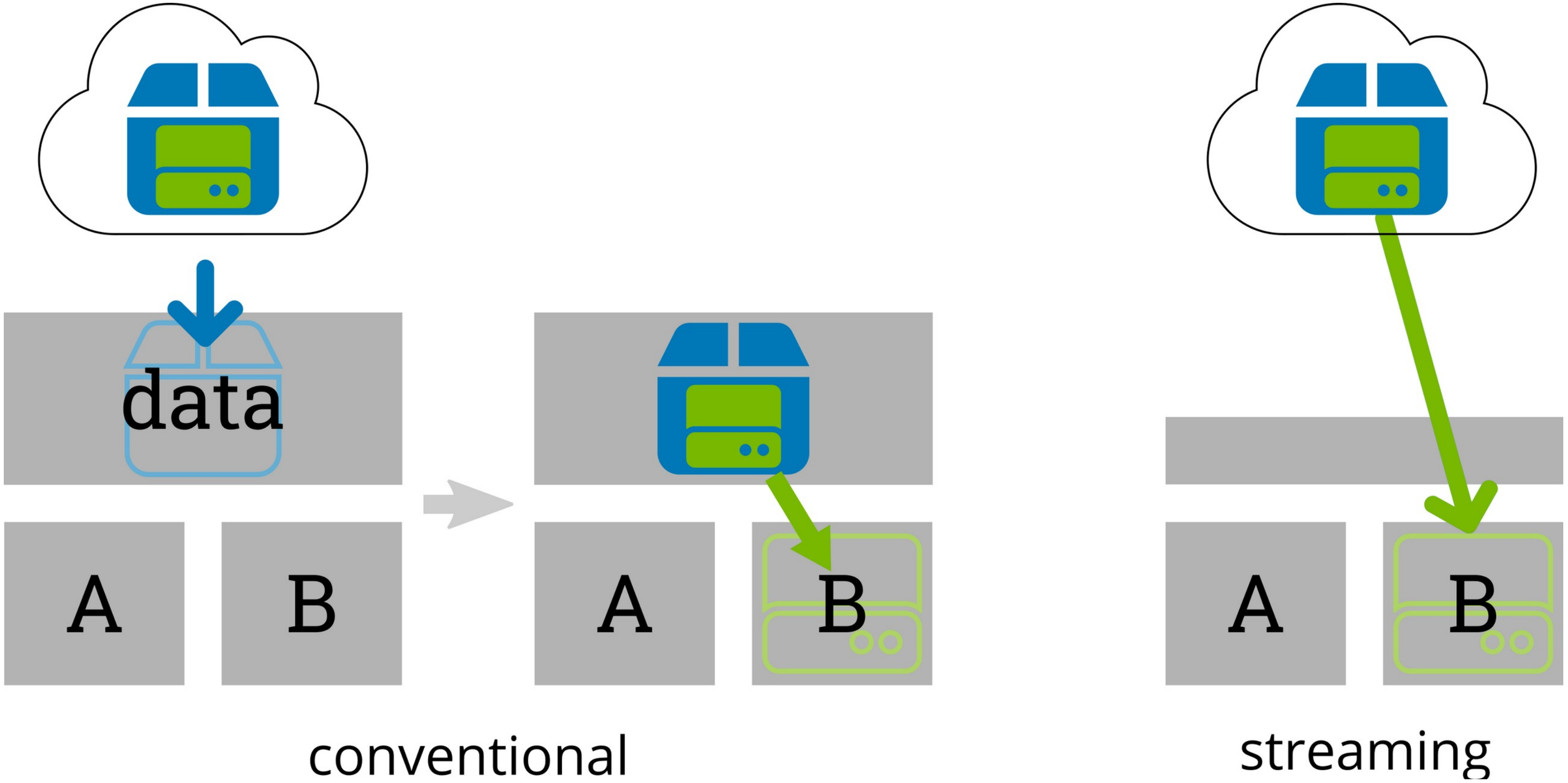


- ✓ Fast initial verification
- ✓ Verified random access to bundle

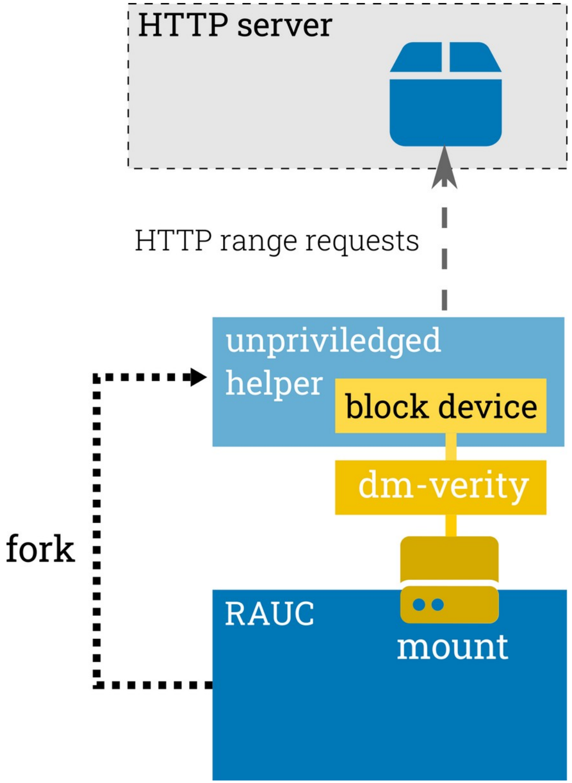
```
rauc install --keyring=… update.raucb
```

Update Bundle Streaming

Bundle Download vs. Bundle Streaming



HTTP(S) Streaming Support



- Unprivileged helper process forked
- Translates block device access to HTTP range requests
- Verified bundle mounted

✓ Verified random access to remote bundle

```
rauc install http://example.com/encrypted.raucb
```

HTTP(S) Streaming Support

- Supports (by libcurl):
  - HTTP versions 1.1 and 2
  - Basic Authentication (user:password@...)

```
rauc install http://user:password@example.com/bundle.raucb
```

  - HTTPS (optionally client certificates)

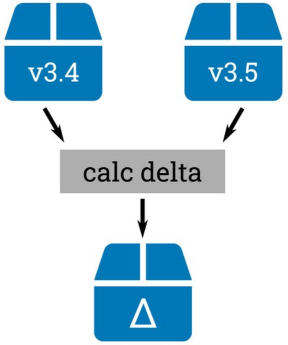
```
rauc install --tls-cert/key=<PEMFILE|PKCS11-URL> https://example.com/bundle.raucb
```

  - custom HTTP headers (e.g. for bearer tokens)

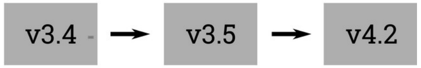
```
rauc install --http-header='HEADER: VALUE' https://example.com/bundle.raucb
```

Saving Download Bandwidth

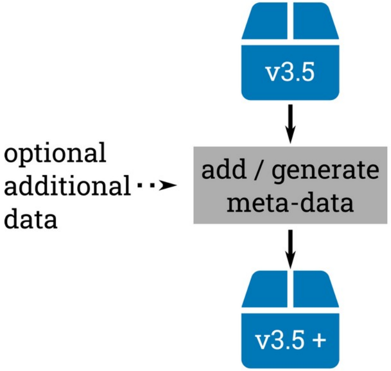
Delta Updates vs. RAUC Adaptive Updates



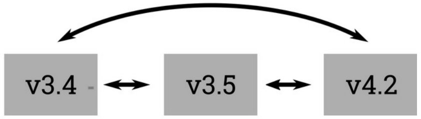
delta update



incremental only



adaptive update



adaptive / differential

- Optimal / minimal delta
- Complexity during generation
- In-field versions must be known (or server-side logic)

- Original bundle + meta-data for optimized updates
- adaptive selection of one or multiple (supported) methods

RAUC Adaptive Updates

```
[update]
compatible=Test System

[bundle]
format=verity

[rootfs.image]
adaptive=block-hash-index;delta-image
filename=rootfs.ext4

[appfs.image]
adaptive=tree-rsync-checksum
filename=app.tar.gz
```

Bundle Manifest

✓ Adaptive selection of optimization

RAUC 1.8

Supports: `block-hash-index`

→ rootfs: block-hash-index

→ appfs: conventional update

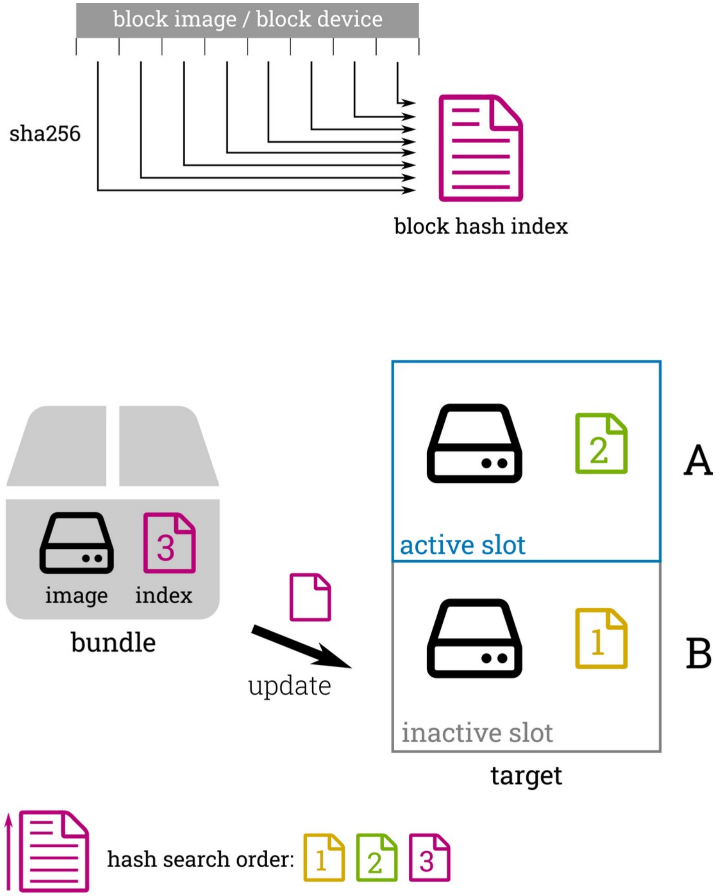
RAUC 1.9?

Supports: `block-hash-index`,
`delta-image`,
`tree-rsync-checksum`

→ rootfs: `delta-image`

→ appfs: `tree-rsync-checksum`

Adaptive: block-hash-index Updates



- Chunk & hash → index list
- Update: Transfer (block) hash index file
- Get hash index of target slots
- Walk through hash index list
    - Copy chunk from inactive or active slot
    - Read from remote bundle only if not found locally

Adaptive: tree-rsync-checksum (Outlook)

Future version

- Bundle generation:
  - Convert file system tar to directory tree
  - Generate checksums for files (stored in xattrs or separate file)
- Bundle installation
  - (skip mkfs)
  - `rsync --delete --copy-dest=<active-slot> <bundle>/rootfs.tree <inactive-slot>`

Adaptive: delta-image (Outlook)

Future version

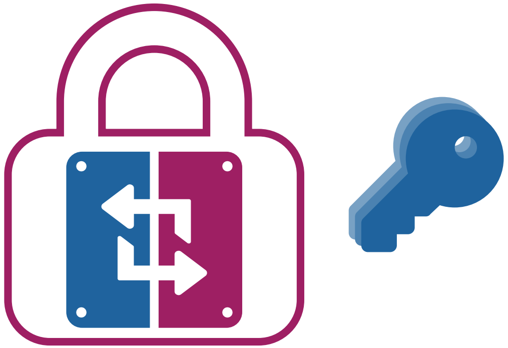
- Generate conventional binary image deltas
- Place additional to normal images in bundle

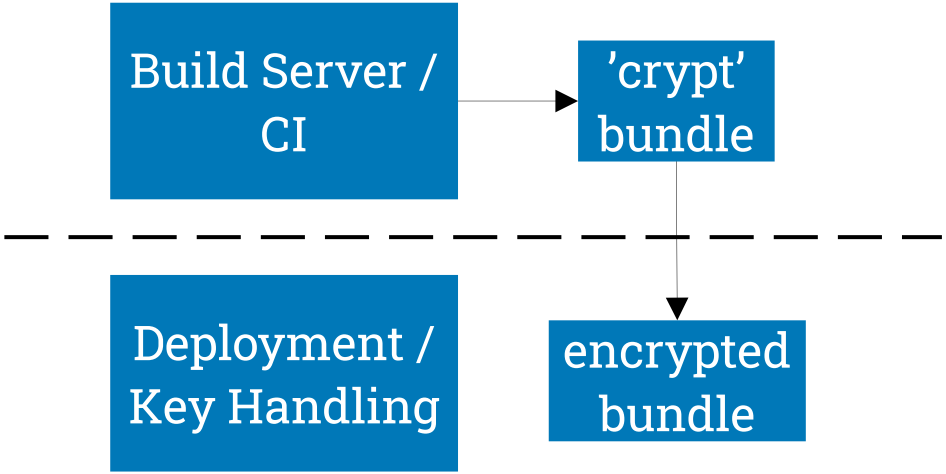
→ apply delta image if available, otherwise full image

Bundle Encryption

Bundle Encryption

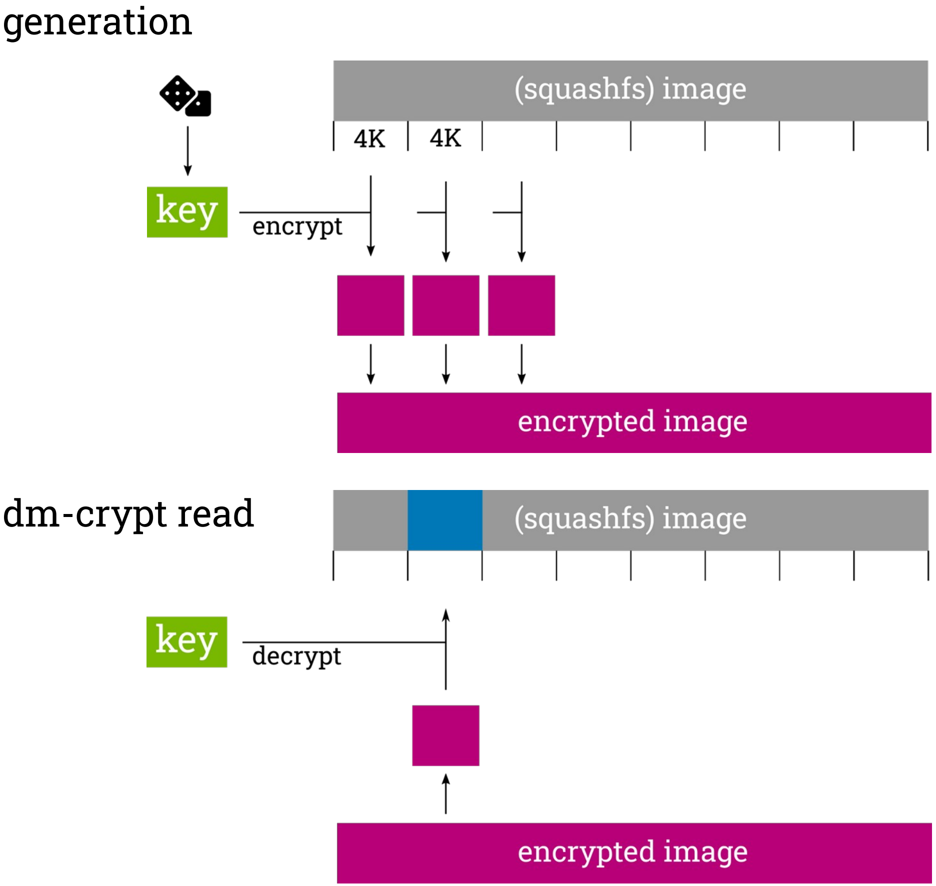
- Hide sensitive data
- Hide application IP from third-party





→ two-step process

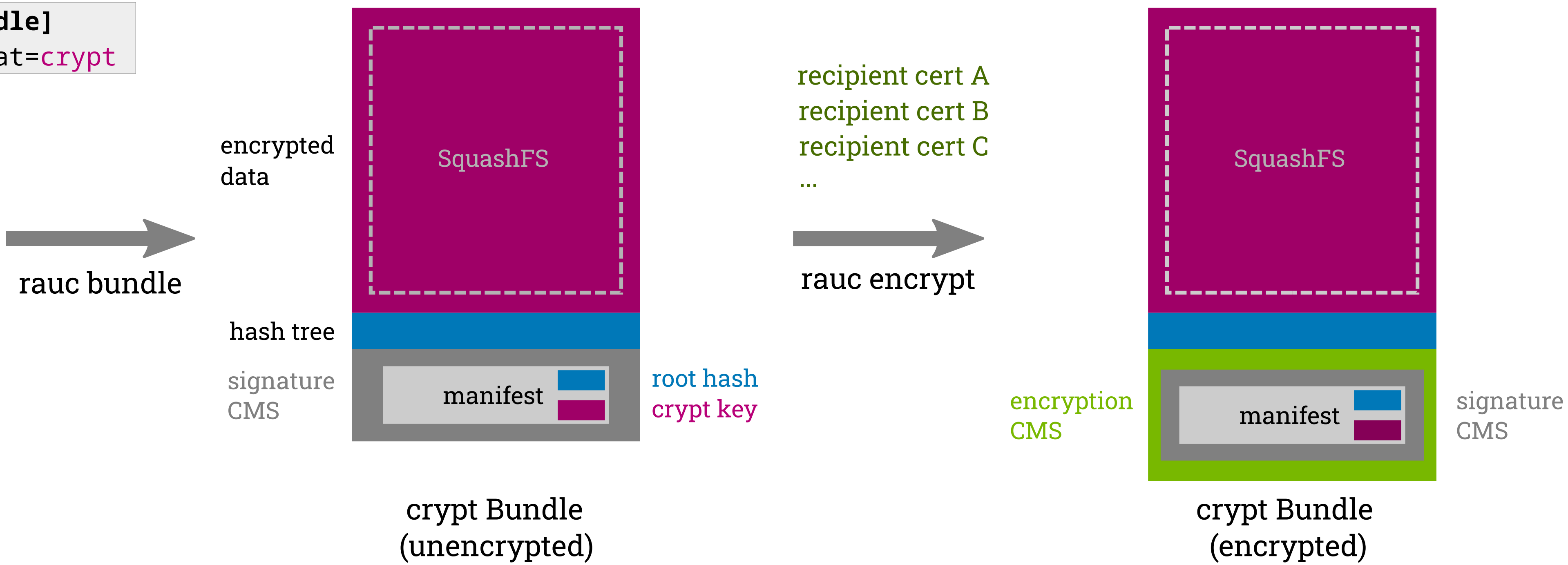
dm-crypt – Block Device Decryption



- We just add another layer...
- Device mapper: dm-crypt (Symmetric with AES-256)
- Transparent encryption / decryption

Encrypted Bundle – Generation

```
[bundle]
format=crypt
```



rauc bundle

encrypted data

SquashFS

hash tree

signature CMS

manifest

root hash

crypt key

crypt Bundle (unencrypted)

recipient cert A
recipient cert B
recipient cert C
…

rauc encrypt

SquashFS

encryption CMS

manifest

signature CMS

crypt Bundle (encrypted)

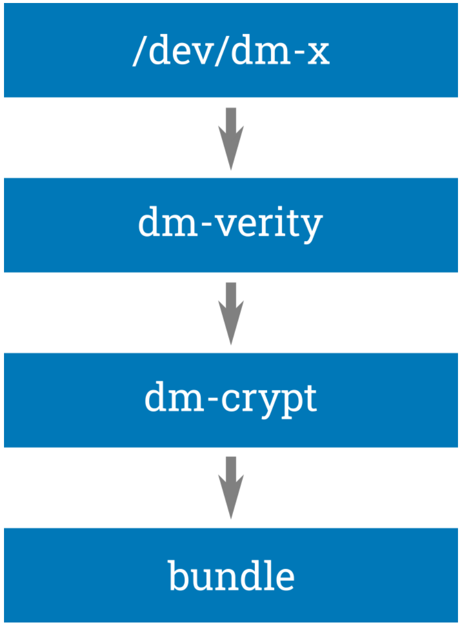
```
rauc bundle … bundle-content/ crypt.raucb
```

```
rauc encrypt --to=…  --to=…  crypt.raucb encrypted.raucb
```

Bundle Decryption (Installation)

```
[system]
compatible=Test System
bootloader=barebox

[encryption]
key=crypt-key.pem
cert=crypt-cert.pem

[slot.rootfs.0]
...
```

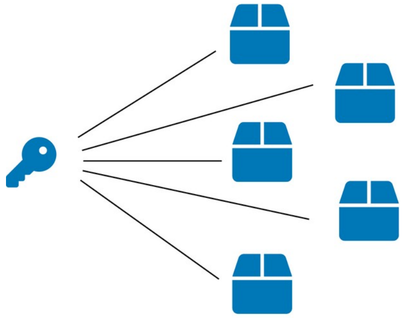
Or PKCS#11 URI



✓ Compatible with streaming!

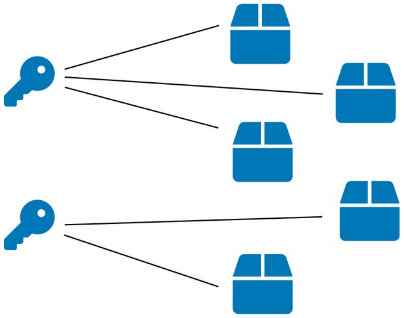
```
rauc install https://example.com/encrypted.raucb
```

Supported Encryption Use Cases



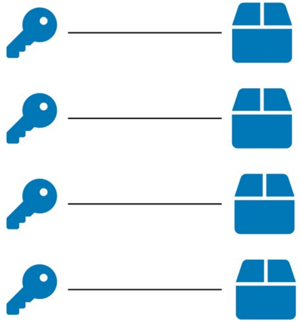
shared key

- Single key for all
- No per-device revocation
- All compromised at once



group key

- Multiple groups
- Less impact of compromised key



per-device key

- Protected key

  (TPM, HSM, TEE)
- Individual revocation

Outlook & Community

Custom Meta-Data in Manifest

```
[update]
compatible=My Product Name
description=Verbose Text
version=v1.9.2-r0
build=20220911223717

…

[meta.pengutronix]
mac=de:ad:be:ef:01
location=Dublin
class=edge

[meta.device]
key=value
```

- Standard bundle information not always sufficient
- Vendor-defined `meta.*` sections
- No built-in interpretation
- Forwarded / exposed via
  - D-Bus API
  - rauc info

Installation History / Event Logging

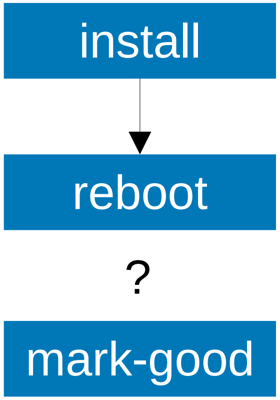
- So far: status file written

```
[slot.rootfs.1]
bundle.compatible=Test System
bundle.version=2022.09
status=ok
sha256=efbcb10…
size=104611840
installed.timestamp=2022-09-12T23:42:36Z
installed.count=3
activated.timestamp=2022-09-12T23:42:36Z
activated.count=3
```

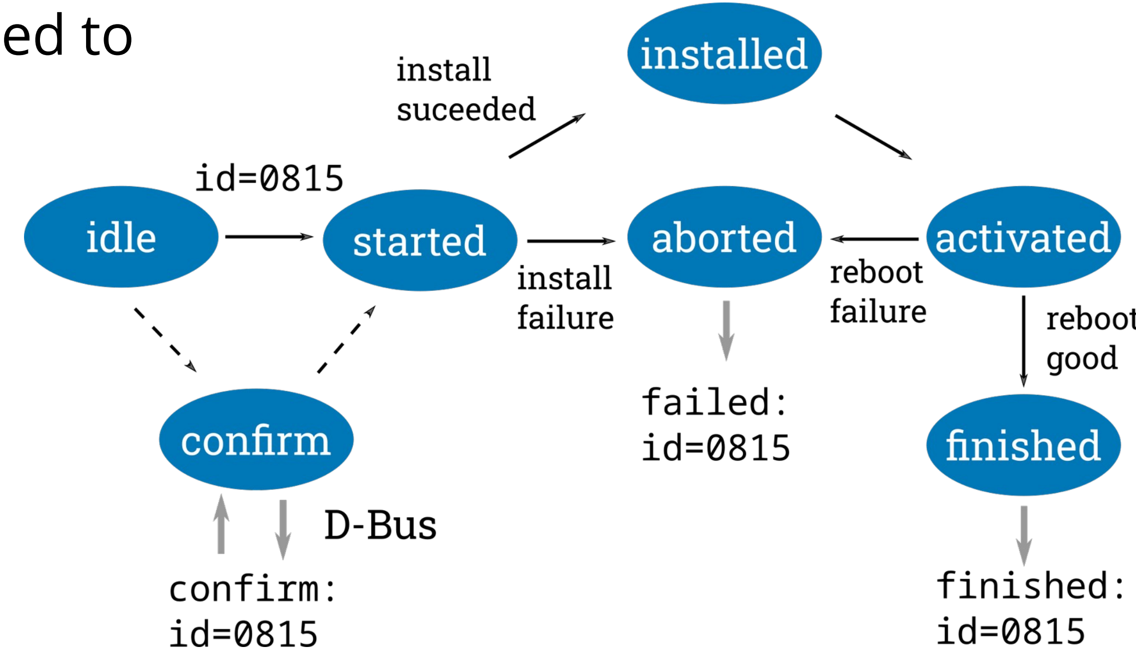
- Plan: Have configurable event logging
- History of all installations

Life Cycle Handling

- Current Scope: individual installation
  - Confirmation not tied to installation



- Solution: transaction IDs, tracking of update life-cycle

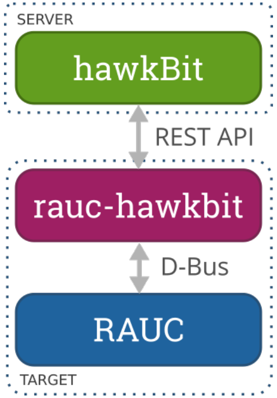


Feature Wishlist

- Multiple signers, M-of-N signatures (supported by OpenSSL)
- Application / Container Updates (https://github.com/rauc/rauc/issues/969)
- Streaming upload from Browser
- Simple Deployment Server

Ecosystem: rauc-hawkbit-updater

- Eclipse hawkBit: Open Source back-end framework for software rollouts
- RAUC adapter in C started by Prevas (2018)
- Moved to RAUC Org (2020)



→ Refactoring, Fixing, Cleanup

→ Initial release 1.0 (2021)

→ current release 1.2 (2022)

https://github.com/rauc/rauc-hawkbit-updater

Community: meta-rauc-community

- Bitbake layer collections for example integrations
- Maintained by Leon Anavi
- Supported boards:
  - qemux86-64
  - raspberrypi
  - Sunxi
  - Tegra

✓ Good starting point!

✓ Contributions welcome!

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

Community: RAUC-related Projects / Products

- Valve Steam Deck
  - RAUC + desync (casync variant in Go)
  - Patches mainlined by Collabora

- Home Assistant Operating System
  - Buildroot updated with RAUC

Home Assistant

- Oniro
  - Eclipse project for distributed systems

oniro

Thank You!

Questions?

Join the discussion and get help on: #rauc IRC/Matrix channel

Pengutronix.