Owning the Stack:

Open-Source Device Management Without the SaaS Lock-In

Julien Vermillard - julien@clunkymachines.com

Julien - Independent IoT Architect & Developer

IoT architecture and platform design:

- Device-to-cloud architectures with a bias toward wireless and open-source
- CRA, Secure boot and end-to-end key management and OTA pipelines
- Embedded and Cloud software development (Java/Kotlin, Go, C, Rust, AWS..)

https://www.clunkymachines.com

https://www.linkedin.com/in/jvermillard/

>_ CRA

Lundi 13 avril 2026 / de 9h30 à 11h30 à Sopra Steria Montpellier

6 Rue de Pommessargues ZAC Eureka - Bat. Appollo, 34000 Montpellier

Tech&Product teams

Challengez votre architecture IoT face au Cyber Resilience Act en 90 min

Speaker

Julien Vermillard

+20xp IoT Architect

Clunky Machines

Organisateur

wireless logic france

Host

sopra steria

Convenience or Control?

IoT often starts with SaaS convenience:

- Instant dashboards, one-click OTA
- Fast, managed, outsourced

The trade-off

- Opaque data
- Constrained protocols
- Vendor-driven roadmap

The alternative

- Open standards: CoAP, LwM2M, MQTT
- Zephyr + MCUboot

The trade-off

- Dependency by default
- Or control by design and effort

Picking Protocols?

- TCP cost is negligible ⇒ **MQTT**
  Simple, widely deployed

- TCP is not fine (NB-IoT, Thread, …) ⇒ **CoAP**
  Lower overhead, better fit for constrained networks

- Interoperability matter? ⇒ **Lightweight M2M**
  Adds object model + device lifecycle
  Heavy if you control both ends

Device Management
Data operation modeling

Device-Management 101:

- Device sends **telemetry** (firmware version, sensor values, download %, ..)
- Server would like to **write** values to reconfigure the firmware/application
- Execute **operations**: reboot, FOTA, reset config, blink LED for identifications etc..
- File **download**: FOTA images, certificates, configurations

You probably can do a lot with telemetry + write for simple use cases

Example: CoAP

REST semantics with CBOR:

GET /config/device/name ⇒ “my Device”

POST /config/device/name ⇐ “My new Device”

POST /config/device ⇐ { “name”: “New new device”, “key”: “DEADBEEF”, “locked”: true }

Example: REST over MQTT

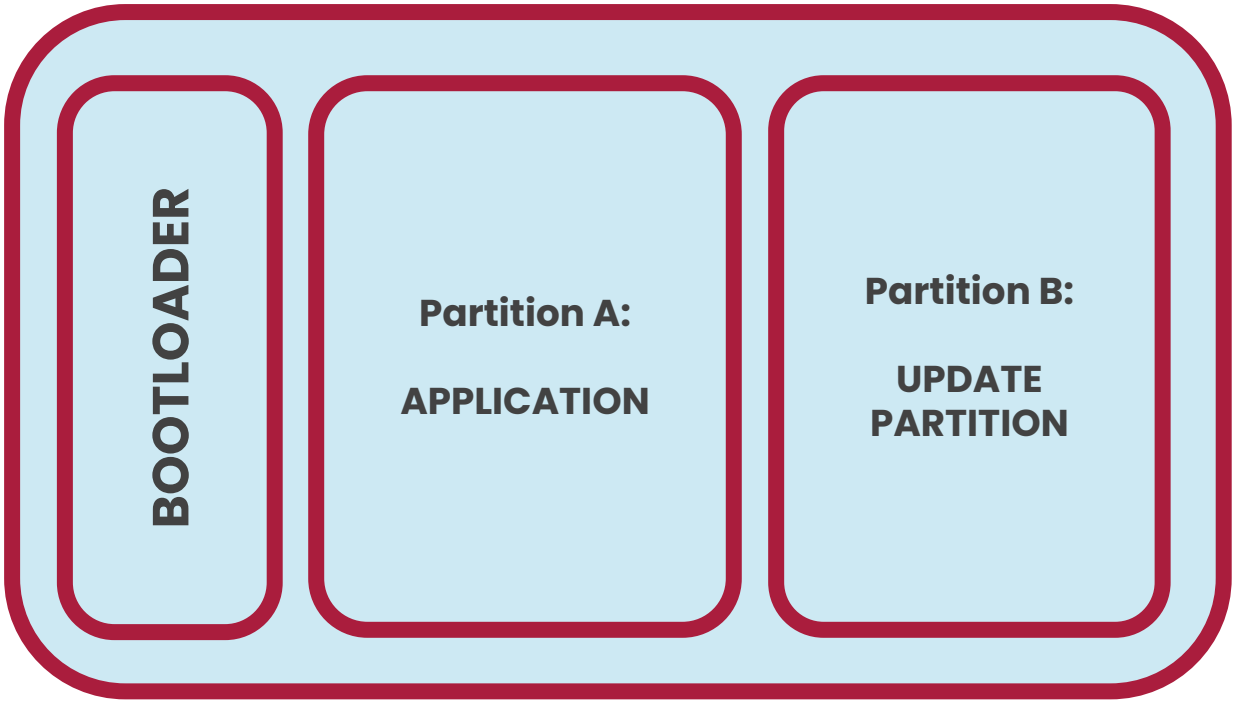
A topic for the device to send telemetry: **`device/{device ID}/data`**

Example payload: **`{“rssi”:-80, “temp”:30.5}`**

A topic for the device to receive tasks: **`device/{device ID}/tasks`**

`{“task”: “read”, “path”: “/config/device”}`

The test of ownership: FOTA



Zephyr: Sysbuild

- It's a higher-level build system on top of the regular application build
- In one invocation can build : bootloader + main app + secondary images
- To build: `west build --sysbuild`
- `west flash` Will flash all the different images built
- Sign the images for boot verification:

```
SB_CONFIG_BOOT_SIGNATURE_KEY_FILE="${APP_DIR}/keys/mcuboot_priv.pem"
```

MCUboot: Reliable Firmware Updates

- Cryptographic image validation (ECDSA / RSA signatures)
- Supports A/B (dual-slot) update strategy (XIP and overwrite)
- Integrates natively with Zephyr and sysbuild
- Test → Confirm → Revert, because you can’t test every real world configuration

MCUboot: Test → Confirm → Revert

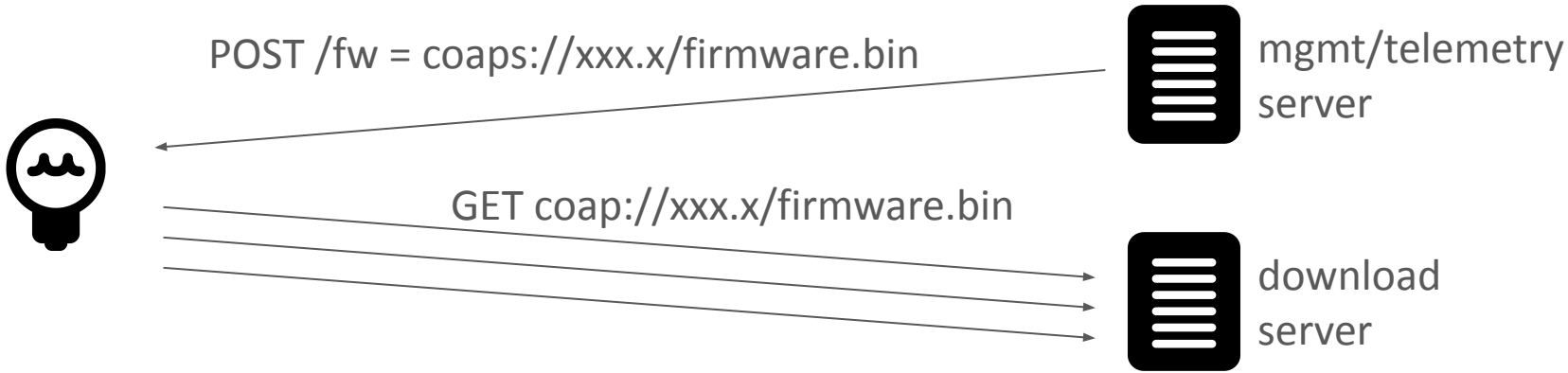
- New firmware written to secondary slot
- Image marked as “pending test”
- Bootloader selects new image on next reset
- System runs in probation mode (not yet permanent)
- Application must explicitly call confirm API
- If confirmed → image becomes permanent
- If not confirmed (crash/reset) → automatic revert

FOTA downloads: Pull and not Push

Device initiated FW download to maximise resiliency:

Control condition when to start (e.g.: battery status, not in user interaction)

Can resume on disconnection or reboot






DEMO



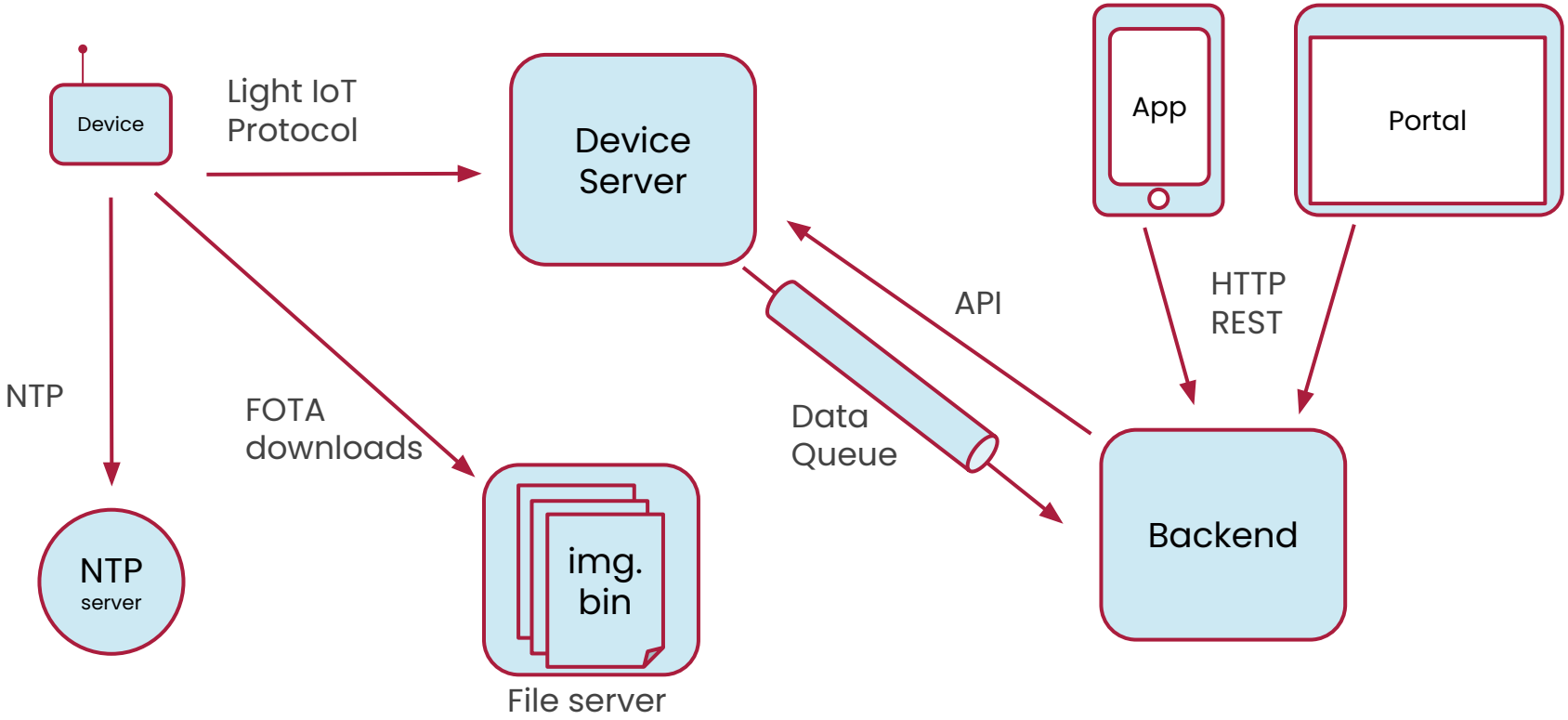
Clunky Machines



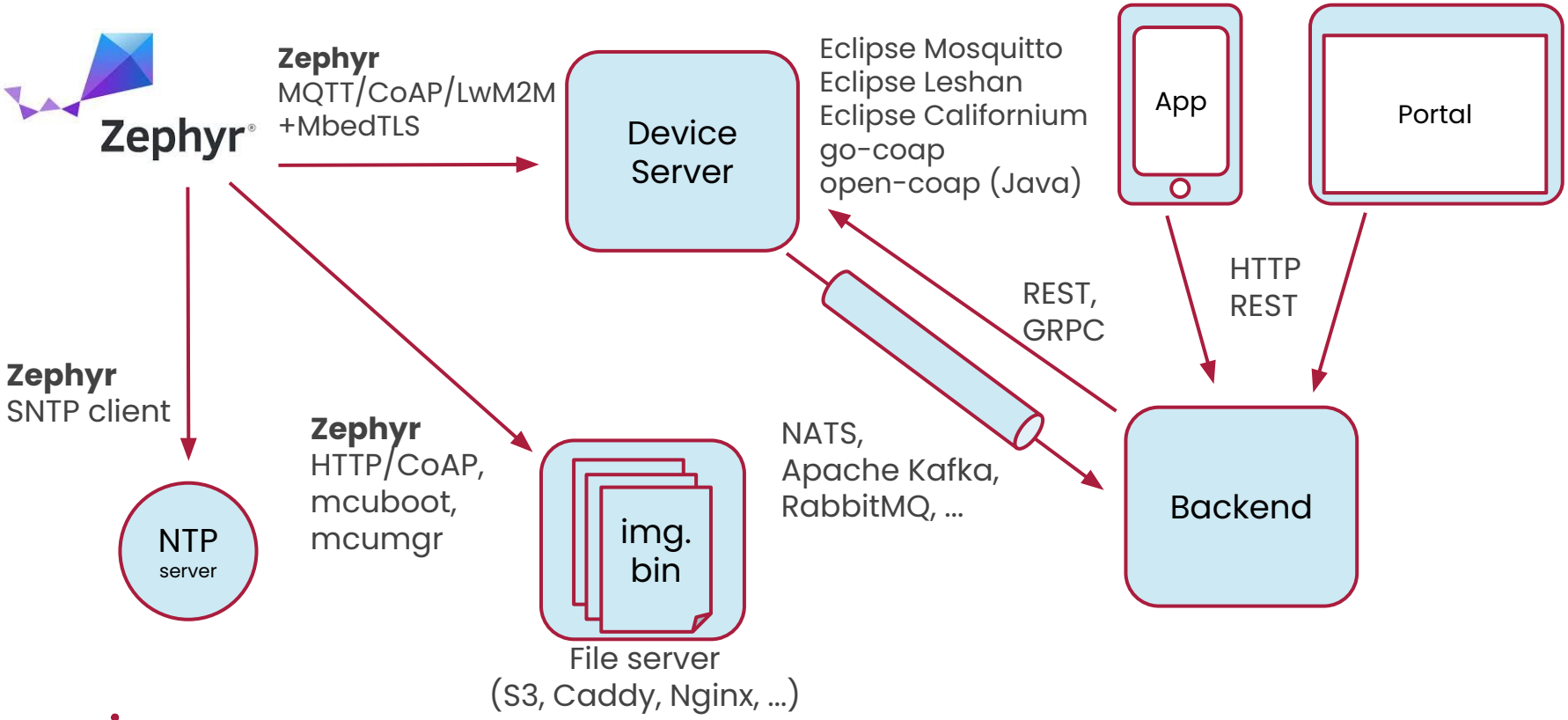

Reference End-To-End Architectures



Clunky Machines



Reference Architecture



Reference Architecture




Beyond the Demo



Clunky Machines

Scaling & Production considerations

- Stateful to stateless service behind load balancer
- Reconnection storms and rate limiting
- OTA campaign orchestration (phased rollout, cohort targeting)
- Device identity lifecycle (provisioning, rotation, revocation)
- Observability at fleet scale (metrics, tracing, failure analytics)

On-Prem, Data Sovereignty, Compliance

- Deployment flexibility: on-prem, private cloud, sovereign cloud
- Full control over data location and retention policies
- No forced multi-tenant SaaS exposure
- Compatible with regulated environments (energy, industrial, public sector)
- Control is a double-edged sword: compliance responsibility shifts to you

Trade-Offs - A Strategic Choice

It has never been easier to DIY with Zephyr + open-source building blocks

- More control, more responsibility, more effort
- Fewer long-term constraints, easier migrations
- Security and compliance become internal obligations

When to own the stack?

- Long product lifecycle (10~20 years)
- Regulated or sovereign environments (On-prem, Gov/Private Cloud)
- Strategic need for protocol and data control (very low power)

Thanks!

Own the stack, and help improve it!

https://linkedin.com/in/jvermillard

https://clunkymachines.com

Julien Vermillard - julien@clunkymachines.com