




# Mobile Beehive Scale 2024

Cellular IoT with CoAP/DTLS 1.2 CID in the “wild”  
May 2024, Achim Kraus



# Speaker

- Followup of “DTLS 1.2 CID Meets Zephyr, 2023” (therefore +1 year)
- Achim Kraus
  - 25+1 years experience in software development and testing (Java/C),
  - 15+1 years experience with distributed device-systems (IP, RS485, ArcNet),
  - 7+1 years experience with IoT, CoAP/DTLS/LWM2M.
  - Committer
    - Eclipse/Californium
    - Eclipse/tinydtls
  - IETF co author of RFC 9146, DTLS 1.2 CID

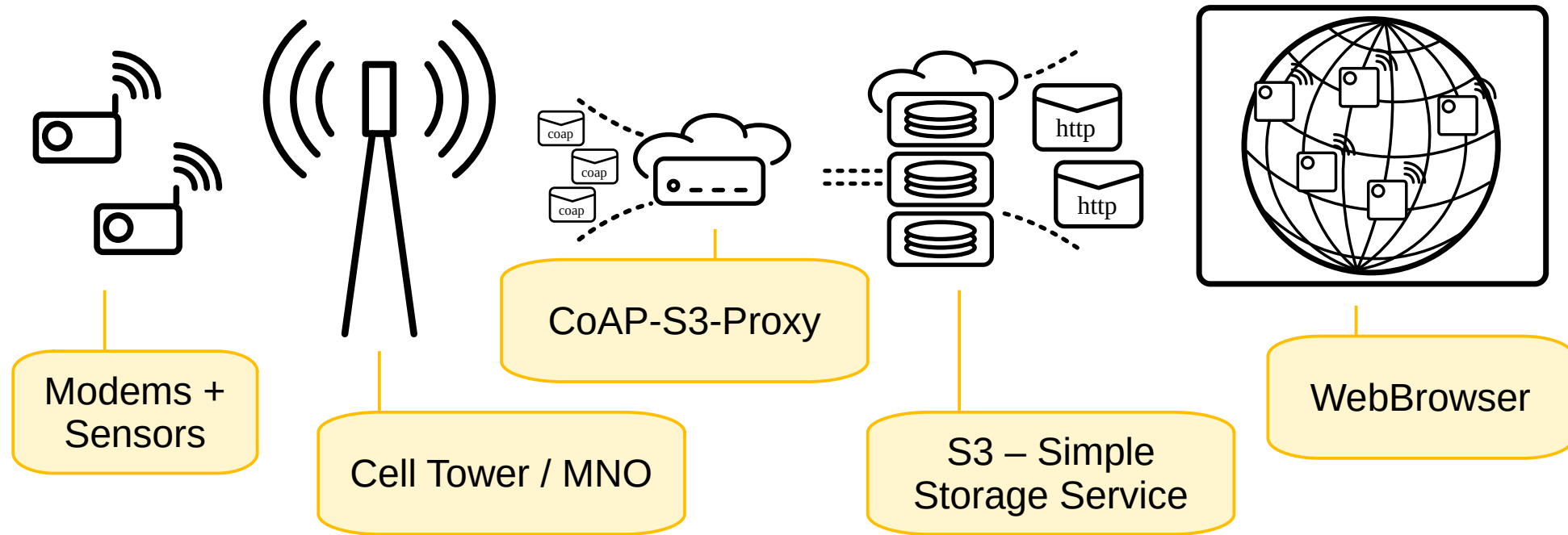
# Contents

- Short technical overview of the system
- Deployments in the “wild”
  - Mobile Bee-Hive-Scale
  - Battery Vampire
  - Wine Refrigerator Shepherd
- Q&A

# CoAP-DTLS 1.2 CID

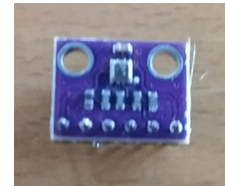
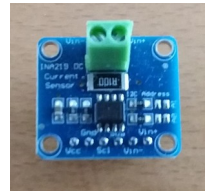
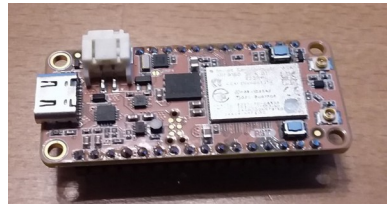
- CoAP, RFC 7252  
http like REST over UDP. Efficient and reliable.  
Great match for radio-messages based communication.
- DTLS 1.2 CID, RFC 6347, RFC 9146  
TLS like encryption over UDP. CID (Connection ID) supporting quiet-phases unlocks the new power saving functions for radio messages based communication as PSM and RAI.
- Together a IETF standardized way to communicate efficient, reliable and secure over radio messages. Of course, it works on wires as well.

# Overview - Building-Blocks



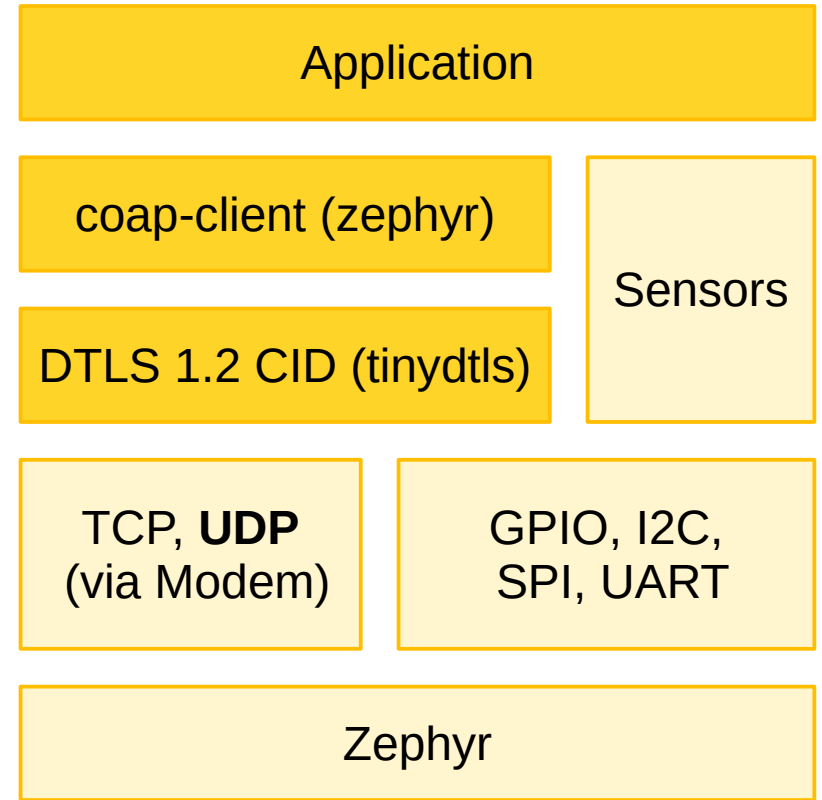
# Modem

- HW: currently nRF9160 based device are supported, see <https://github.com/boaks/zephyr-coaps-client?tab=readme-ov-file#supported-devices>
- OS: Zephyr (+ Nordic Connect SDK)
- Sensors: growing selection, see <https://github.com/zephyrproject-rtos/zephyr/tree/main/drivers/sensor>



# Modem

- OS: Zephyr (+ Nordic Connect SDK)
- SW: zephyr coap-client + tinydtls (DTLS 1.2 CID, open source)
- All together: customizable, highly efficient, reliable and secure open-source demo,
  - e.g. Thingy: 91 ½ year, 1 message exchange every hour from internal battery (1350mAh).
  - <https://github.com/boaks/zephyr-coaps-client>



# Cloud-CoAP

- HW: cloud-VM or own computing center.  
Starting at 2 cores and 2 GB RAM (Footprint Linux + Java)  
(4 cores + 16 GB RAM handles about 1.000.000 concurrent devices with 50.000 messages per second)
- OS: Linux
- Software: Eclipse/Californium
  - java CoAP+DTLS 1.2 CID stack (open-source)  
Reliable, efficient, and e2e encrypted device communication  
<https://github.com/eclipse-californium/californium>
- Java API for receiving and sending CoAP messages.

Custom Application

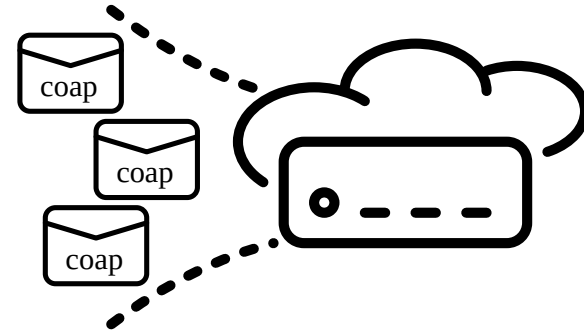
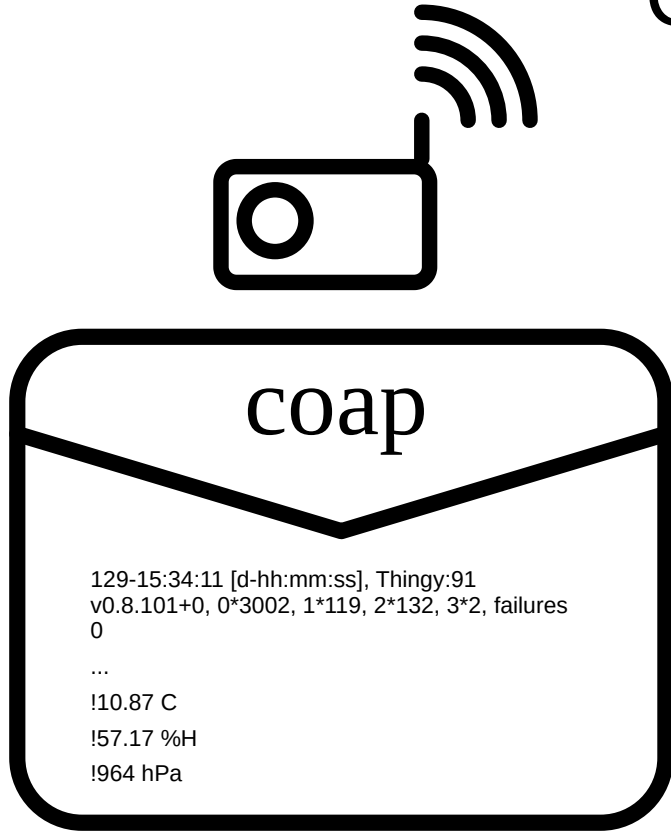
Californium

Java Runtime

Linux



# Cloud-CoAP



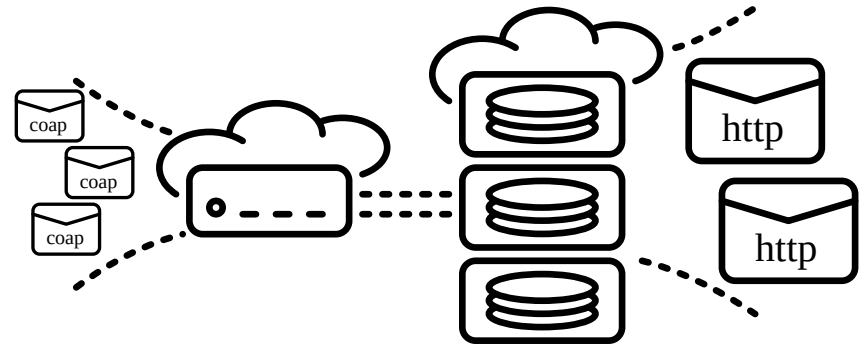
```
public void handlePOST(final CoapExchange exchange) {  
    exchange.getRequestPayload();  
    ...  
}
```

# Demo Custom Application: CoAP-S3-Proxy

- The S3-proxy is one application on top of Cloud-CoAP as bridge into common cloud computing.
- S3 (Simple Storage Service), very common, available on different clouds. Keeps data received from devices and provides configuration for devices.

Also used for FOTA

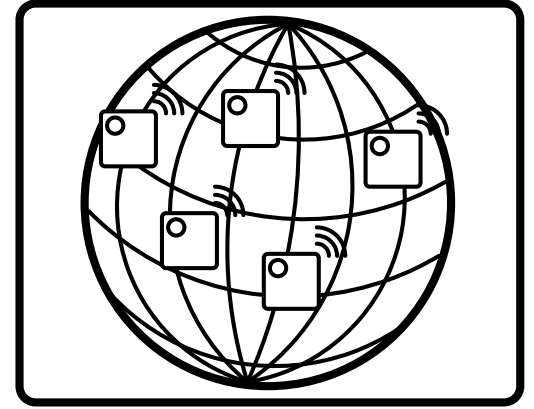
- Converts CoAP requests into S3 request.
- [https://github.com/boaks/californium/tree/add\\_s3\\_proxy/demo-apps/cf-s3-proxy-server](https://github.com/boaks/californium/tree/add_s3_proxy/demo-apps/cf-s3-proxy-server)



# CoAP-S3-Proxy

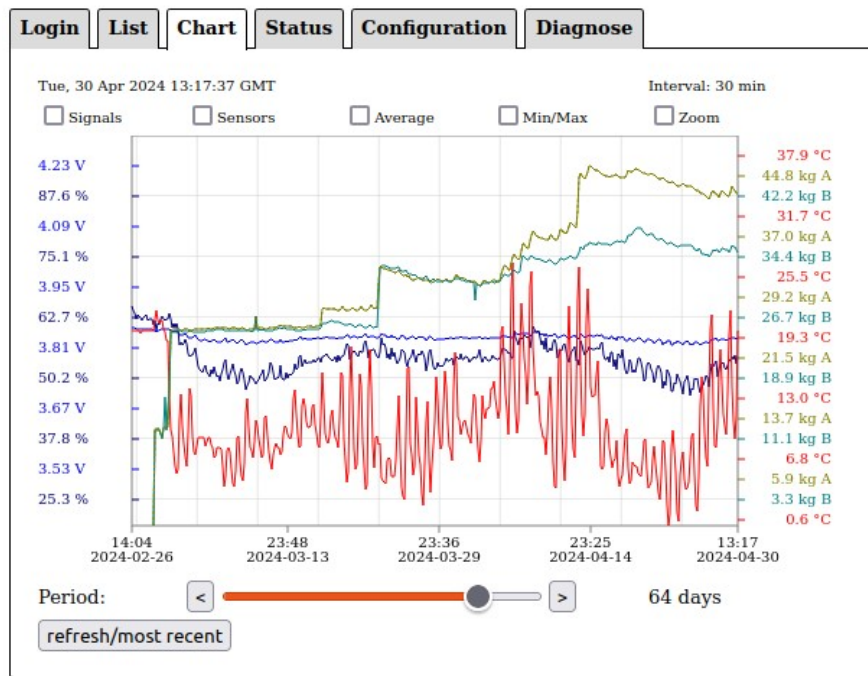
- S3 reduces the maximum number of messages per second (depends on S3-provider, 300-3000 messages per second and more).
- 1.000.000 devices, 1 message per hour => less than 300 messages per second.
- FOTA requires multiple CoAP request (e.g. 400 for 400KB) but 1 S3 request, there you benefit from Californium's performance.
- Permission system for S3 is sometimes limited. Intended for backend systems, not users.
- The CoAP-S3-Proxy mitigates that by supporting multiple S3 buckets using different API-keys.
- Installation scripts for ExoScale, AWS, DigitalOcean. Or manual installation step-by-step.

# Web-Browser App



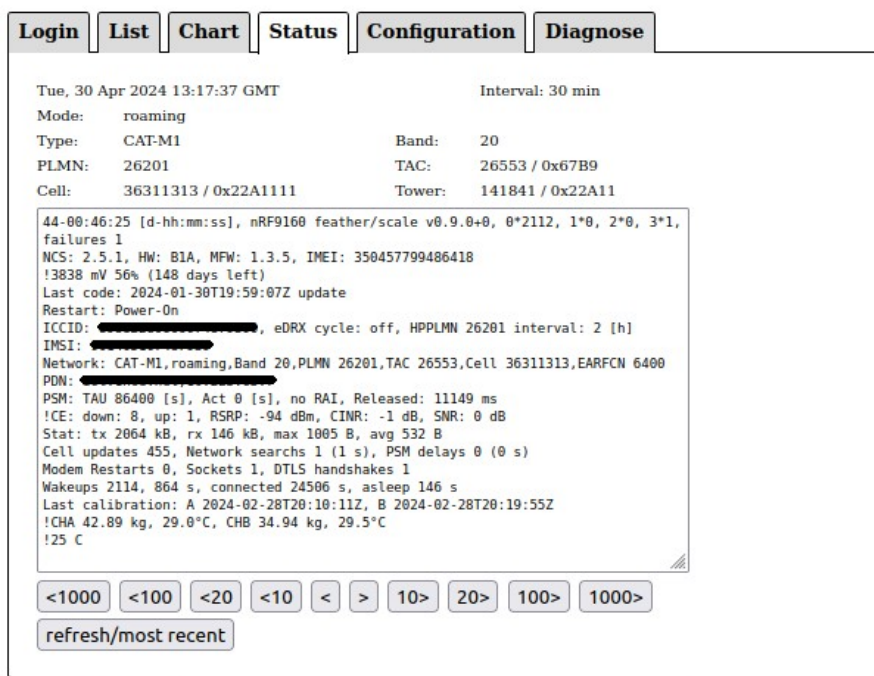
- HW: Smart Phone, Tablet, PC
- OS: Linux, Windows, iOS, Android
- Software: Java Script executed in Web-Browser
- Login via HTTPs service, possibly co-located at the CoAP-S3-Proxy. Return S3 credentials for login credentials.
- Direct S3 HTTPs request to fetch or write stored device data
- Simple charts based on device data.

# Web-Browser App



Load: 1569 b, 144 ms

Version 0.16.1, 18. April 2024



Load: 1569 b, 144 ms

Version 0.16.1, 18. April 2024

# Web-Browser App

- Permission system of S3 provides a basic access protection, but no fine-grained role-based grants. If S3 offers read/write grants, that used on bucket base for “user” and “admin” accounts.
- Intention of the system is to provide UDP experience with low initial invest. Not a “out-of-the-box” end-user-system.
- More sophisticated UI or backends are out of scope and requires additional project based development.

# Mobile BeeHiveScale - Rational



- Bees are collecting pollen as food reserve.
- In good times the bees collecting more pollen than they eat.
- Spring and early summer are frequently good times.
- It depends a lot on the vegetation and weather, how much pollen the bees are able to collect.
- Comparing sites, e.g. orchards, and moving the bee-hives to good sites helps to optimize the honey production.

# Mobile BeeHiveScale - Rational

- On bad times it's vice versa, the bees eat more than they collect.
- Usually at the end of a blossom period or in cold weather periods.
- Especially at the end of a blossom period it helps to optimize the honey quantity, if you know the timepoint, when more is eaten. It's about 500g a day per hive, if you're too late exchanging the honey with sugar-water.
- Therefore a mobile beehivescale helps to keep more honey.





# BeeHiveScale

- No new idea, beehivescales are build since years
- Some open source / maker variants are available, e.g. <https://beelogger.de/> or <https://www.hiveeyes.org/>
- Also commercial offerings are available
- The Mobile-BeeHiveScale present here is neither the cheapest, nor the most precise one. But it offers excellent cellular connectivity. It works well from 3x AA battery cells, sending every hour for 1 year. Of course, only if a cellular network is available, not every valley is covered.

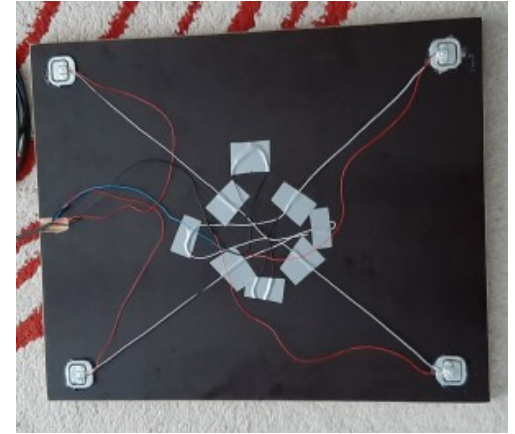
<https://github.com/boaks/mobilebeehivescale>

# Mobile BeeHiveScale

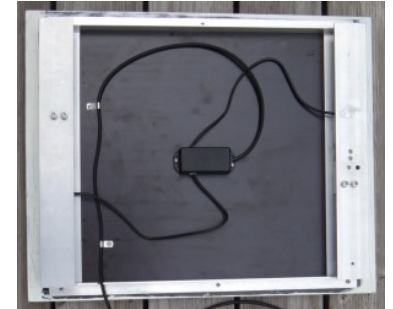
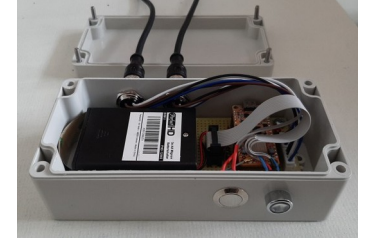
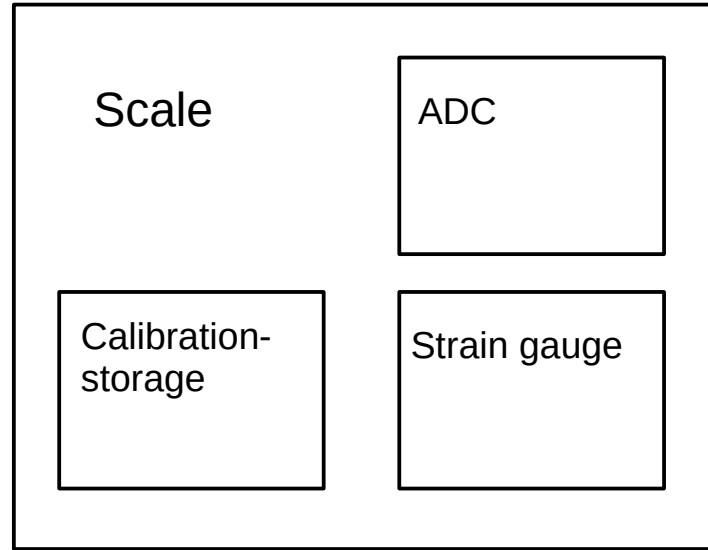
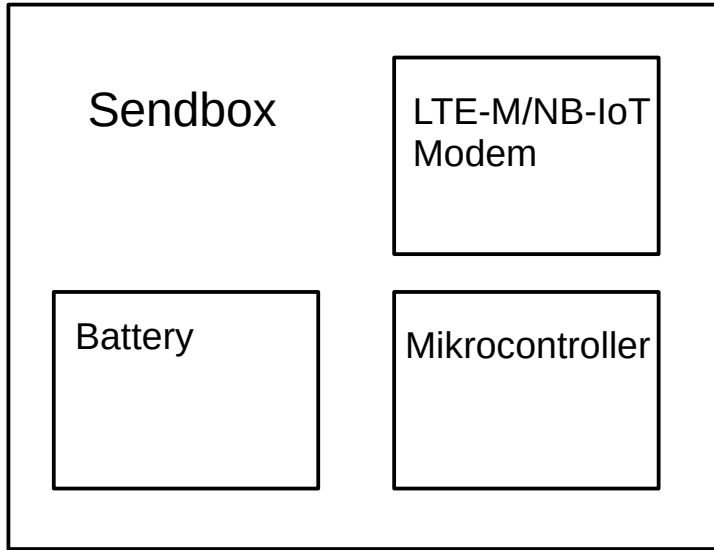
- V1.0 started January 2023
- First approach with simple load cells for “bath room scales”
- One year operating one sandbox, two scales

Communication works reliable, but the measurements of that simple scale depends too much on the temperature, especially on sunshine.

➔ V2.0, redesign solution, started September 2023



# Mobile BeeHiveScale v2.0 – Building Blocks

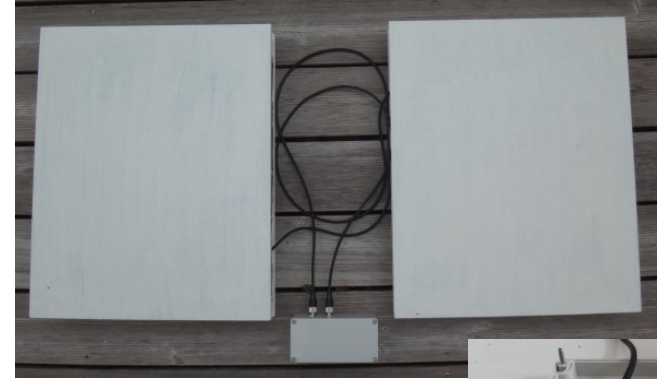


# Mobile BeeHiveScale v2.0

- Move ADC (converts analog value into digital value) from sendbox to scale, short analog wires are less temperature affected
- Move calibration-storage from sendbox to scale (EEPROM), easier to exchange scales and sendboxes on moves and to calibrate the scales
- Use parallel beam load cell, these load cells comes with temperature compensation, which reduces the dependency on that.
- Test run for 6 months in my office and garden now 1 month under the beehive.
- 2 sendboxes, 4 scales. More (2+4) on the way.



# Mobile BeeHiveScale v2.0

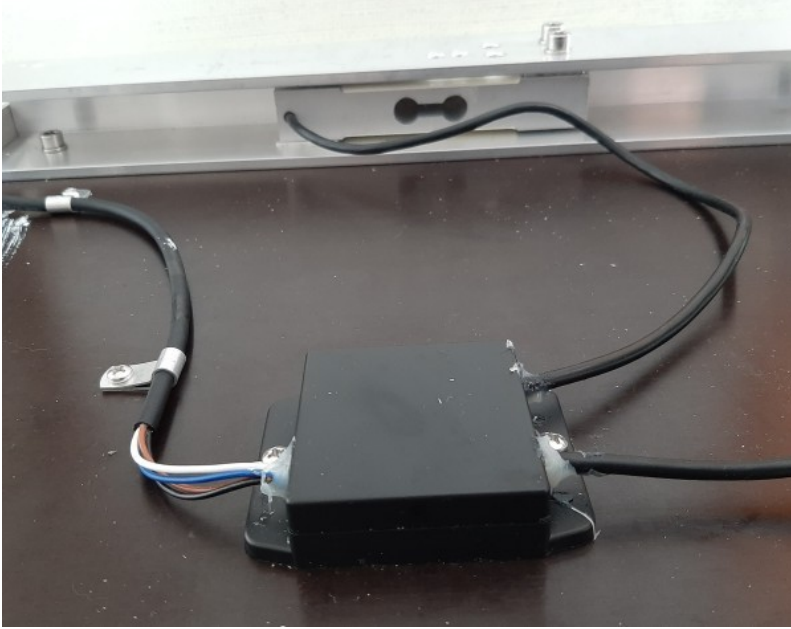


Scales

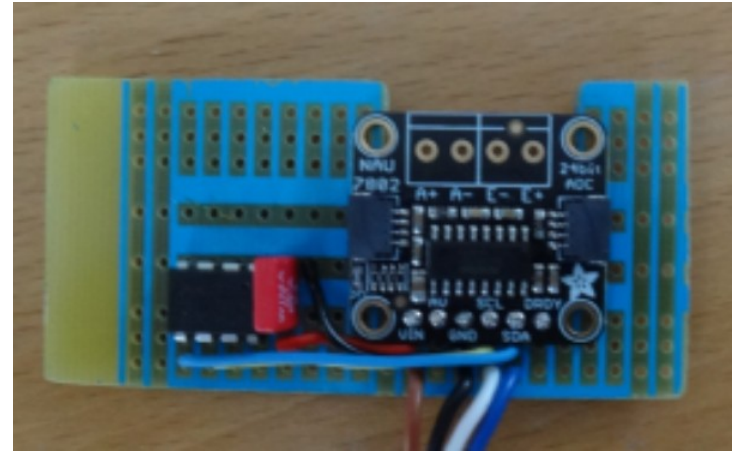


Sandbox

# Mobile BeeHiveScale v2.0



Parallel beam load cell + ADC box



ADC board with EEPROM

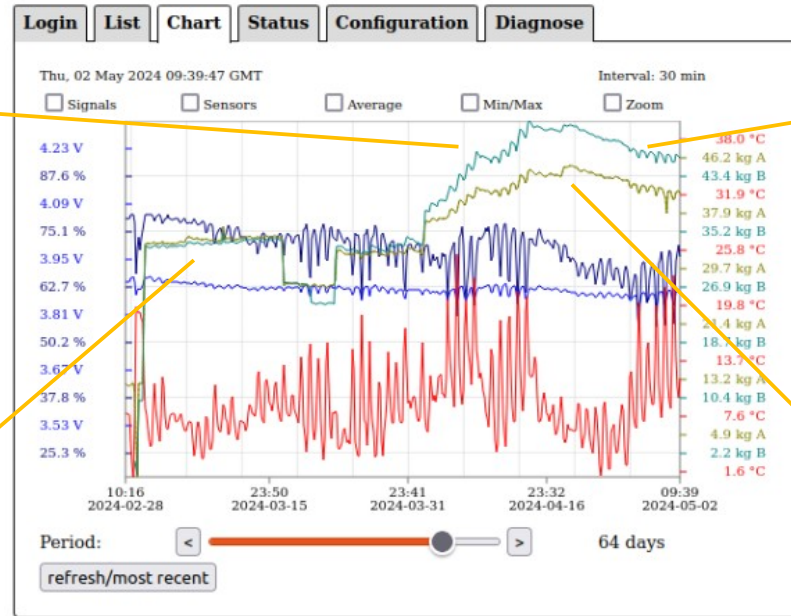


# Mobile BeeHiveScale v2.0

With bees  
Larger ripples,  
bees on the fly  
pollen

Test without bees  
Small ripples

## Device BeeHiveScale-45

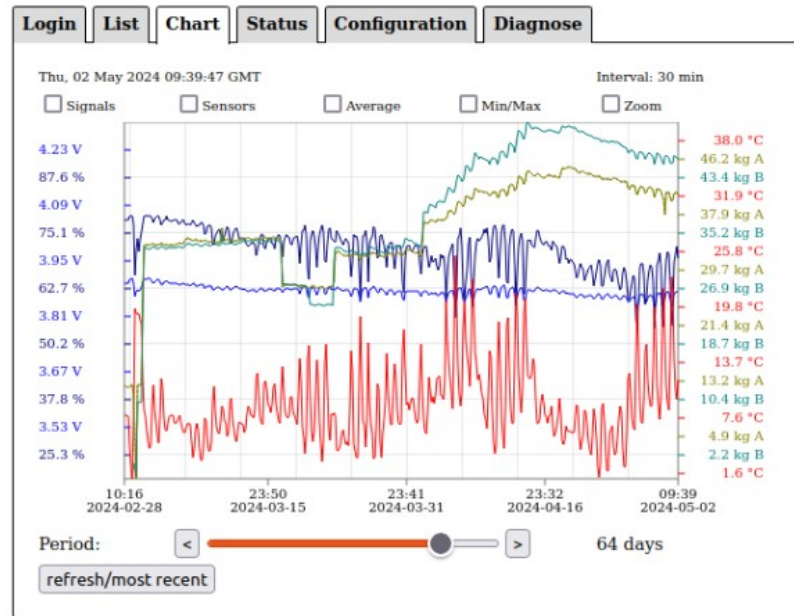


Good weather  
Larger ripples,  
bees on the fly  
No pollen

Bad weather  
Small ripples,  
bees stay in hive

# Mobile BeeHiveScale v2.0

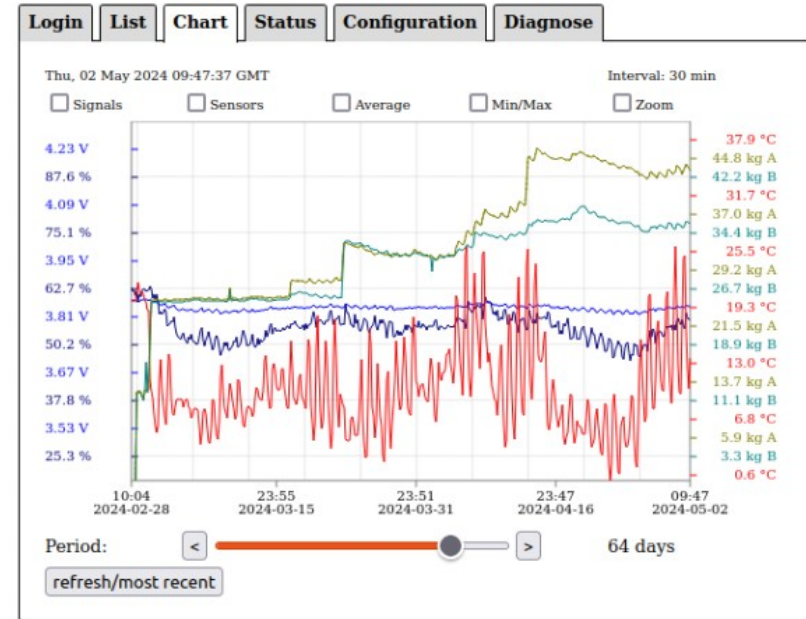
## Device BeeHiveScale-45



Load: 5 kb, 400 ms

Version 0.16.1, 18. April 2024

## Device BeeHiveScale-18



Load: 5 kb, 320 ms

Version 0.16.1, 18. April 2024



# Battery Vampire

- German classic sports cars tends to have a pretty high battery discharge when not operated.
- The battery vampire measures the car-battery voltage and presents that as chart.
- The user sees, when it's time to recharge the battery or drive the car.
- Even with permanent preservation charging, monitoring shows, that this isn't interrupted
- Two cars monitored for a couple of months.



# Battery Vampire

## Device Car-Battery-1



Garage with  
power supply

Garage without  
power supply

# Wine Refrigerator Shepherd

- Thingy:91 monitors the temperature of a wine refrigerator
- Works “out-of-the-box”
- Shows:
  - refrigerator works well
  - Cellular network searches are draining the battery a lot in that case (basement)



Device cali.352656101079724



Load: 440 kb, 1750 ms

Version 0.16.1, 18. April 2024

# Cellular IoT Experience in the “Wild”

## Sum-up

- Running a cellular device from battery over longer time works.
- Use-case considered here is metering or monitoring. Smaller messages (couple of 100 bytes) are exchanged low frequently (e.g. every couple of hours).
- The energy consumption may be split in 3 domains:
  - sending
  - quiescent current
  - network searches

# Cellular IoT Experience in the “Wild”

## Sum-up

- For sending the message with a couple of 100 bytes, the protocol and the interval is relevant. The amount of data gets in my experience only relevant, if some kilo bytes are reached.
- For the quiescent current the hardware design and the battery self discharge need to be considered. Using an efficient protocol the experience is, 1 message exchange takes the similar energy than 1-2h quiet phase. Therefore be careful: exchanging a message every day instead of every hour, doesn't make the device run 10x times longer, it may stop after 3x times!
- Network searches are hard to predict. Low radio signals or moving devices need to consider an reasonable energy buffer for that.