Low-cost LoRa Collar for cattle rustling applications

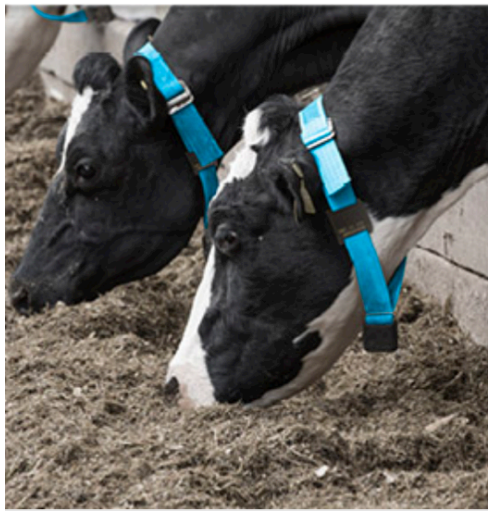
Prof. Congduc Pham
http://www.univ-pau.fr/~cpham
Université de Pau, France

Overview

- We will show how to build a simple low-cost collar for preventing cattle rustling in developing countries. The system consits of:
- An active LoRa end-device fixed to a collar
  - When powered on, sends every 10 minutes a beacon
  - Has built-in low-power management: can run about 1 year with 4 AA batteries
- A LoRa gateway
  - Will receive beacon messages and store them in a local MongoDB database
  - A local application can run to process received beacon info and detect whether an alarm should be raised or not
  - The gateway can send data to clouds if Internet connectivity is available
  - Remote applications can process beacon info with advanced statistical info processing

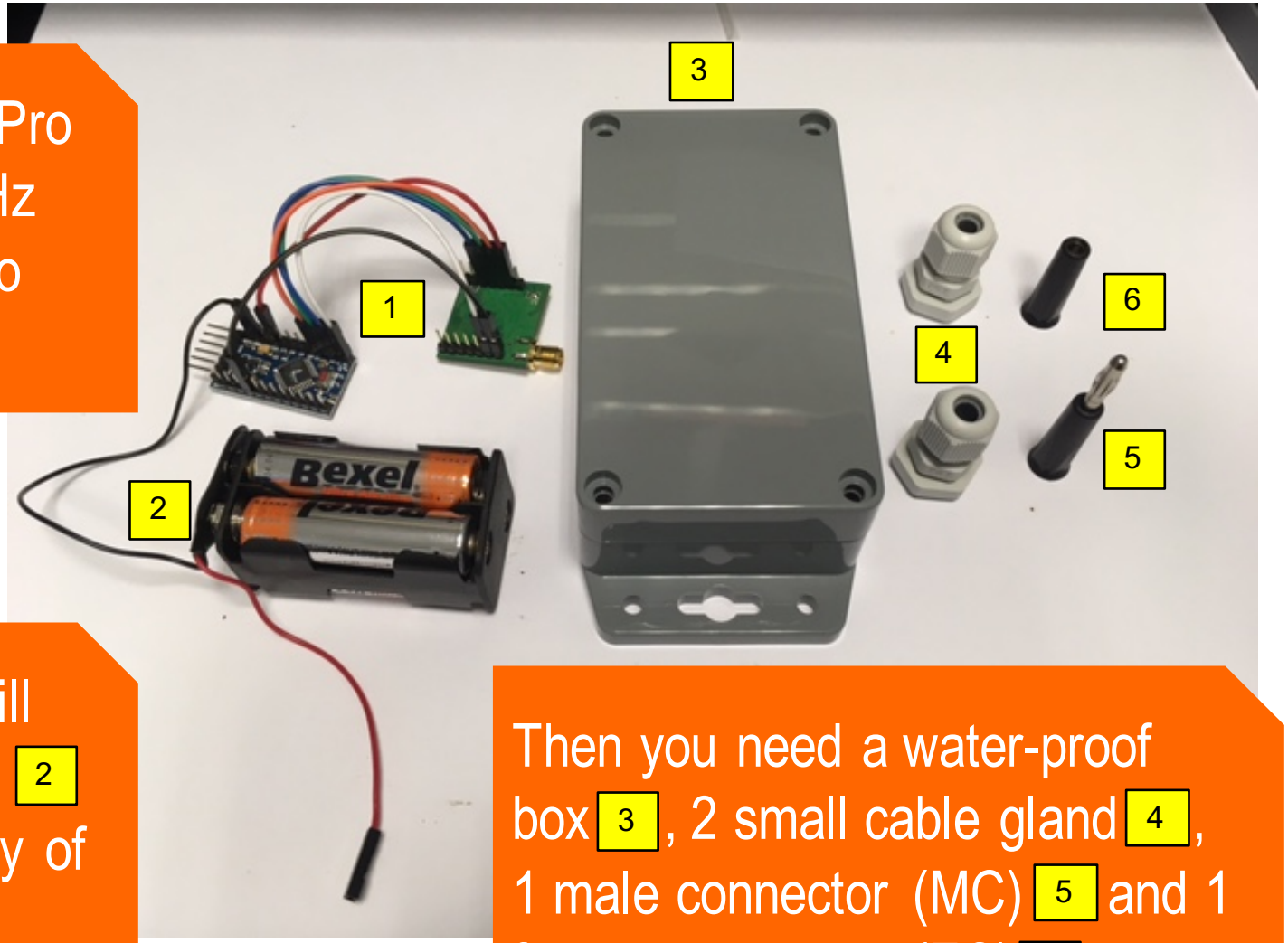
How it works?

- The collar will be fixed to the cow, around neck. Example picture from Afimilk Silent Herdsman for health monitoring
- In our case, reception of beacon means that the cattle is in range
- If out-of-range, disconnected or damaged device, an alarm can be raised
- To detect collar cutting, the power wire will also goes around the catlle's neck



BUILDING THE ACTIVE BEACONING SYSTEM

Use an Arduino Pro Mini 3.3v at 8MHz and a LoRa radio module 1



4 AA batteries will power the board 2 with an autonomy of several months

Then you need a water-proof box 3, 2 small cable gland 4, 1 male connector (MC) 5 and 1 female connector (FC) 6

Drill holes

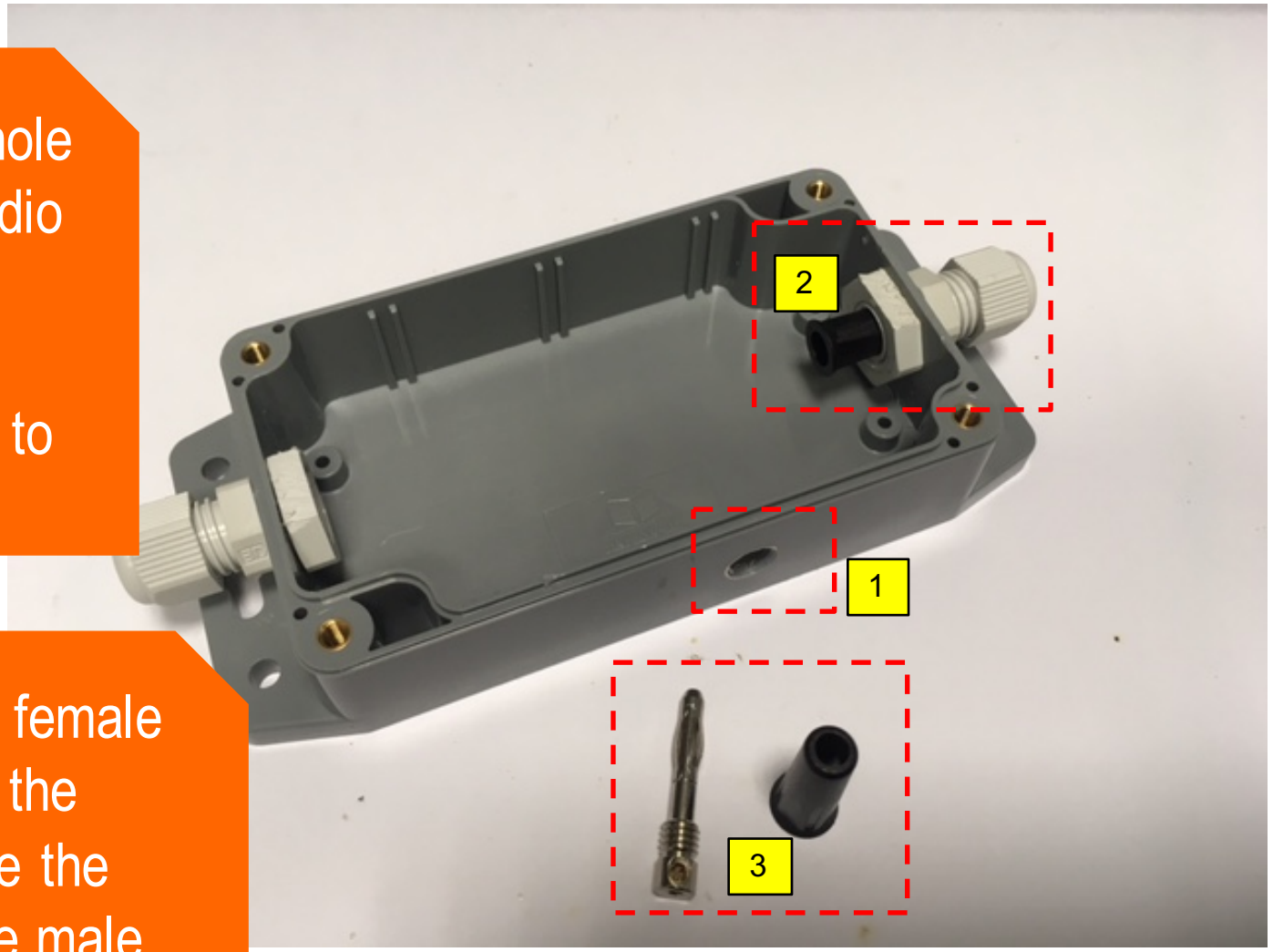
Drill 2 holes depending on the gland diameter (here 12mm, so hole of 13mm)

PLACE THE GLANDS

There is also 1 hole (8mm) for the radio module. 1

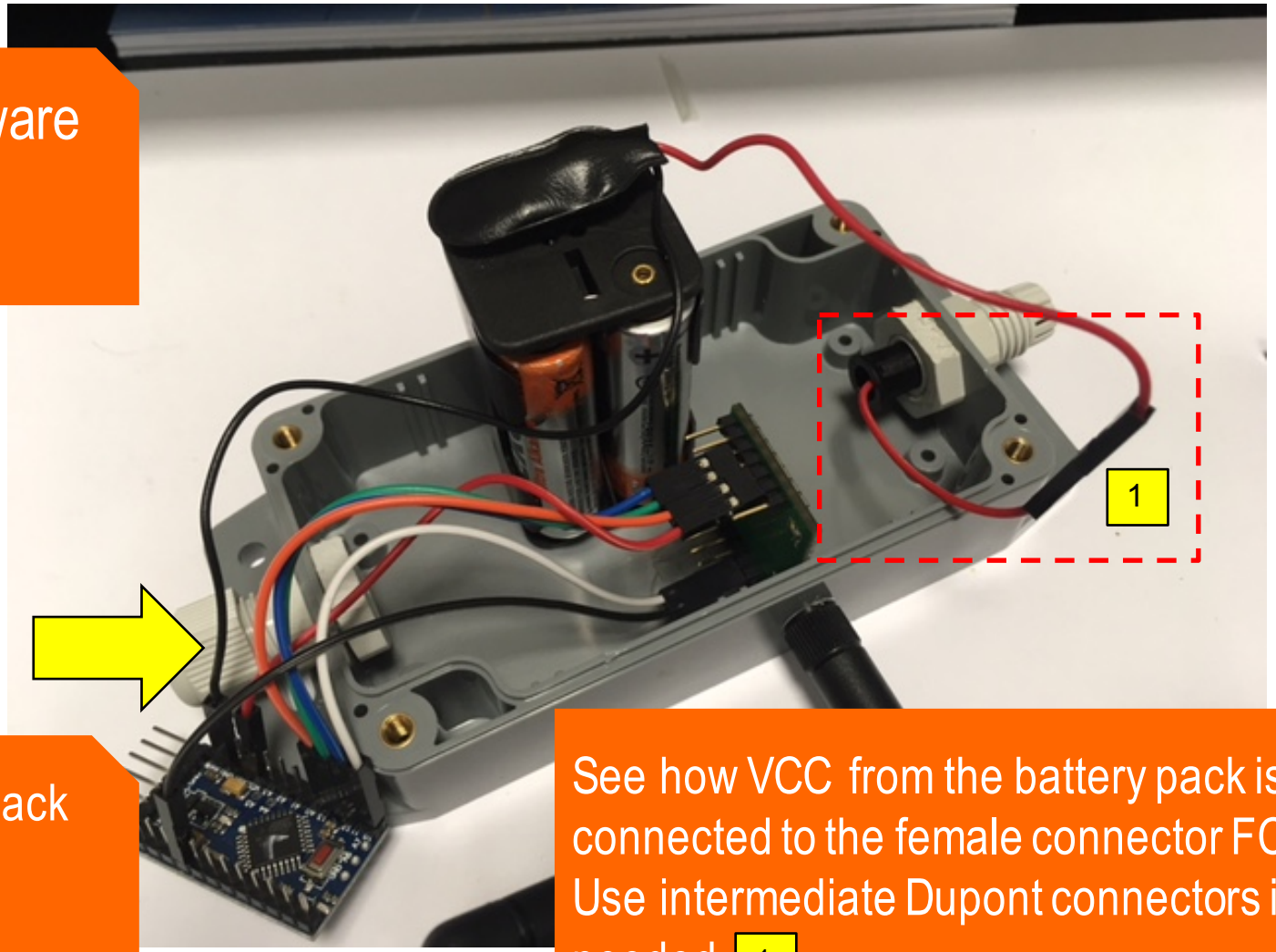
Screw the gland to the box

Fix (in force) the female connector FC to the gland 2, remove the plastic part of the male connector MC 3



PUT HARDWARE IN PLACE

Put all the hardware parts in the box



GND from battery pack is connected to the board (see arrow)

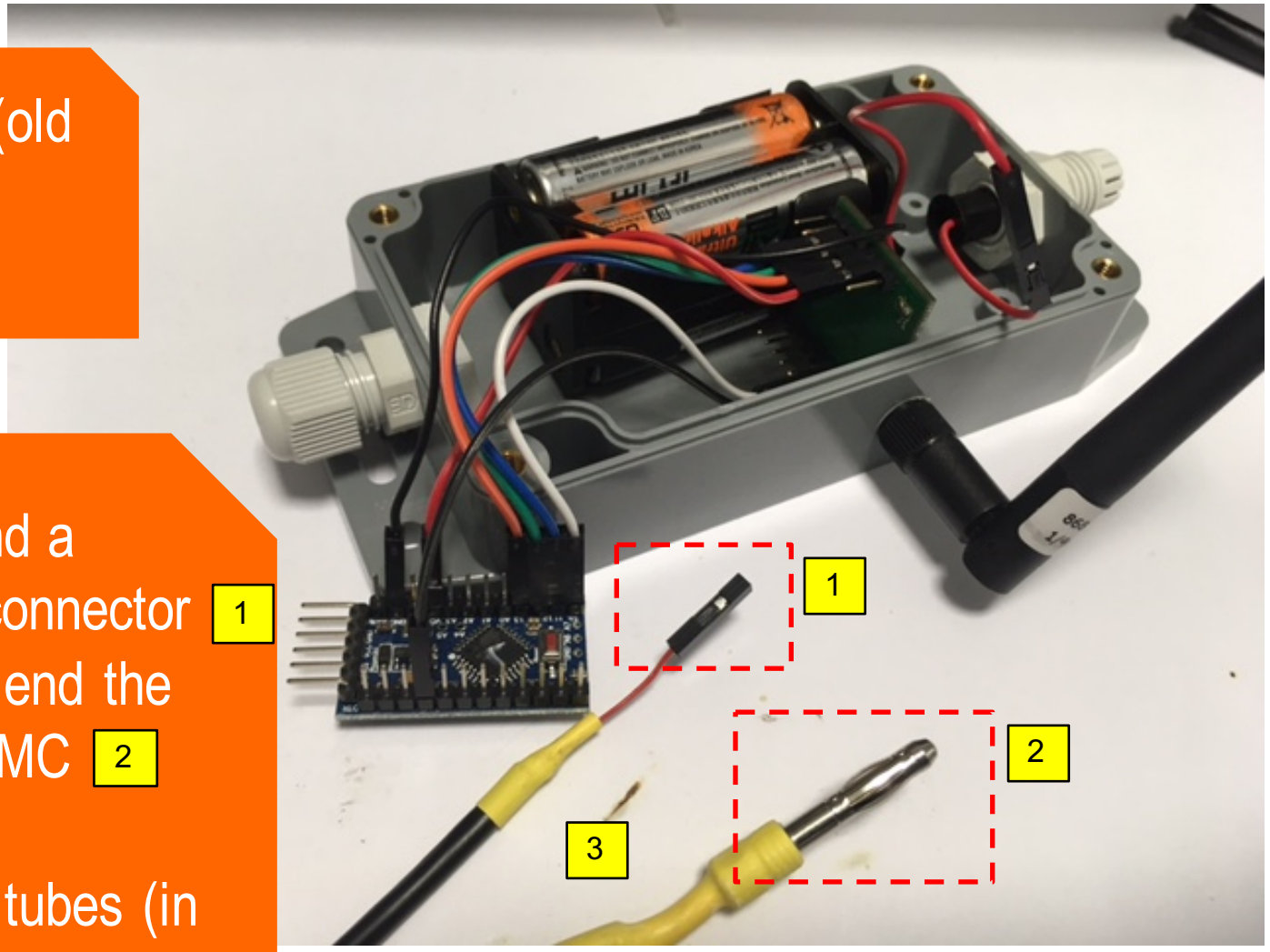
See how VCC from the battery pack is connected to the female connector FC. Use intermediate Dupont connectors if needed. 1

Get a long wire

Take a 1m wire (old USB wire for instance)

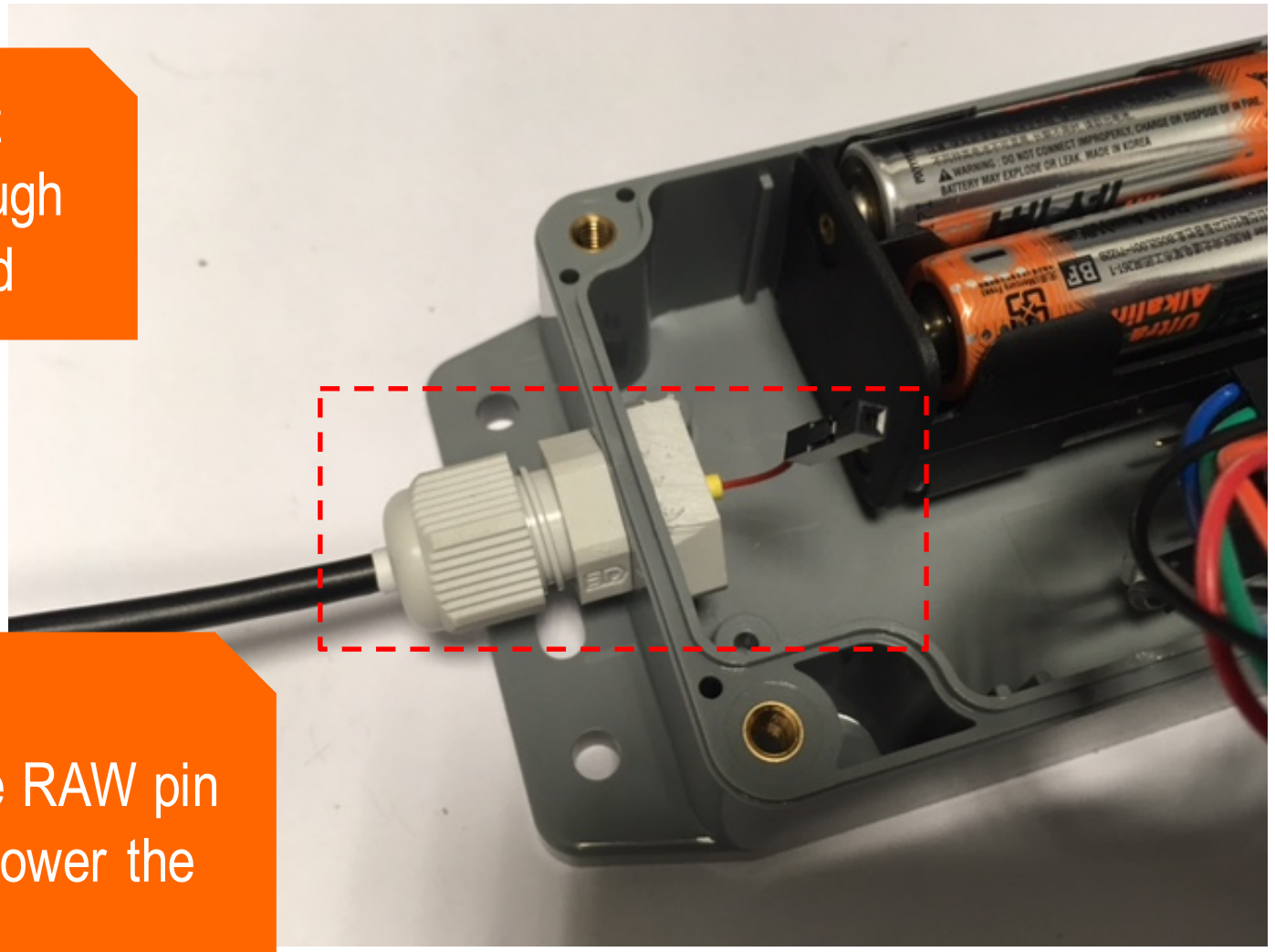
Solder at one end a female Dupont connector 1 and at the other end the male connector MC 2

Use heat-shrink tubes (in yellow) 3



Connecting the power wire (1)

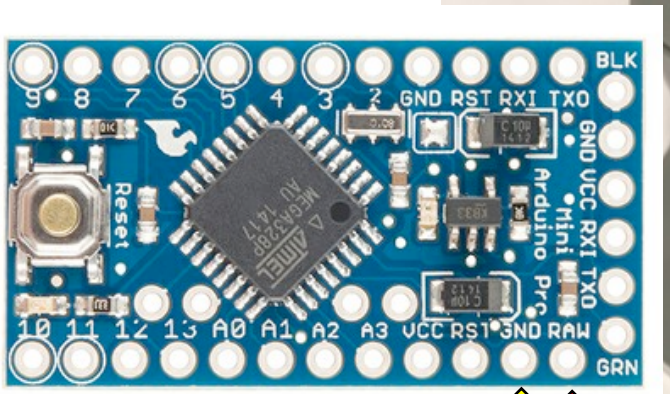
Pass the Dupont female end through the second gland

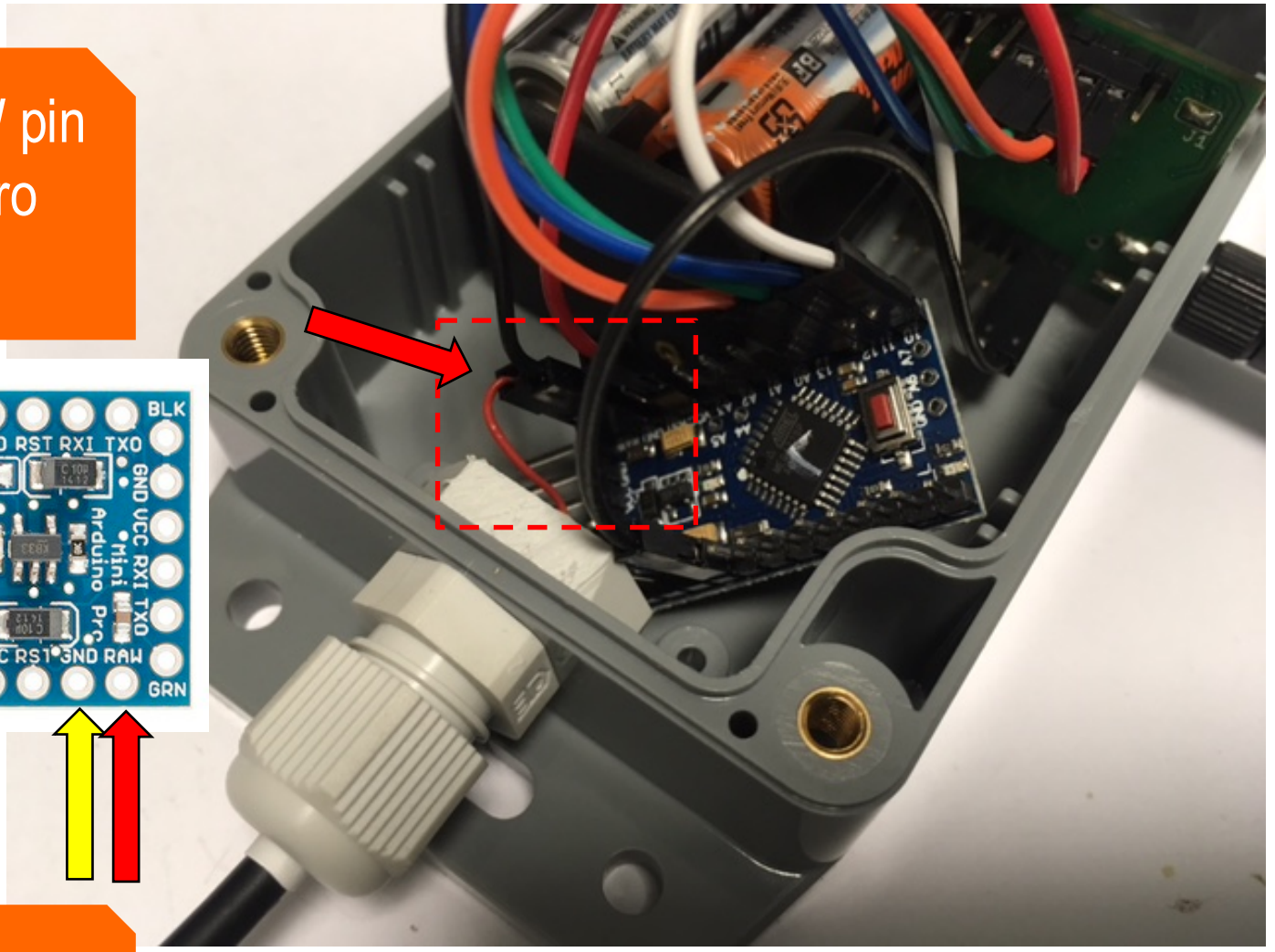


This end will be connected to the RAW pin of the board to power the board

Connecting the power wire (2)

Connect to RAW pin of the Arduino Pro Mini

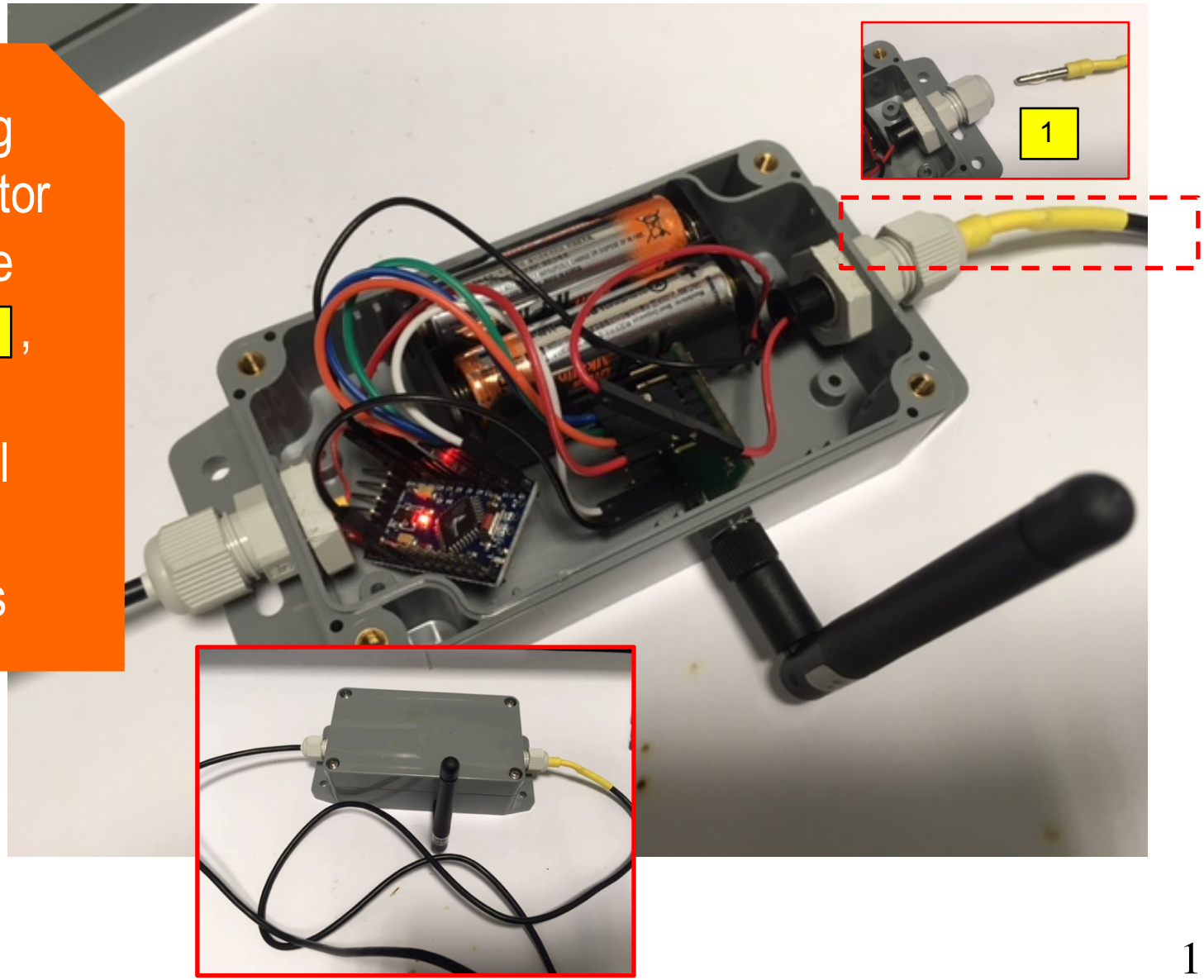




GND was already connected at previous step

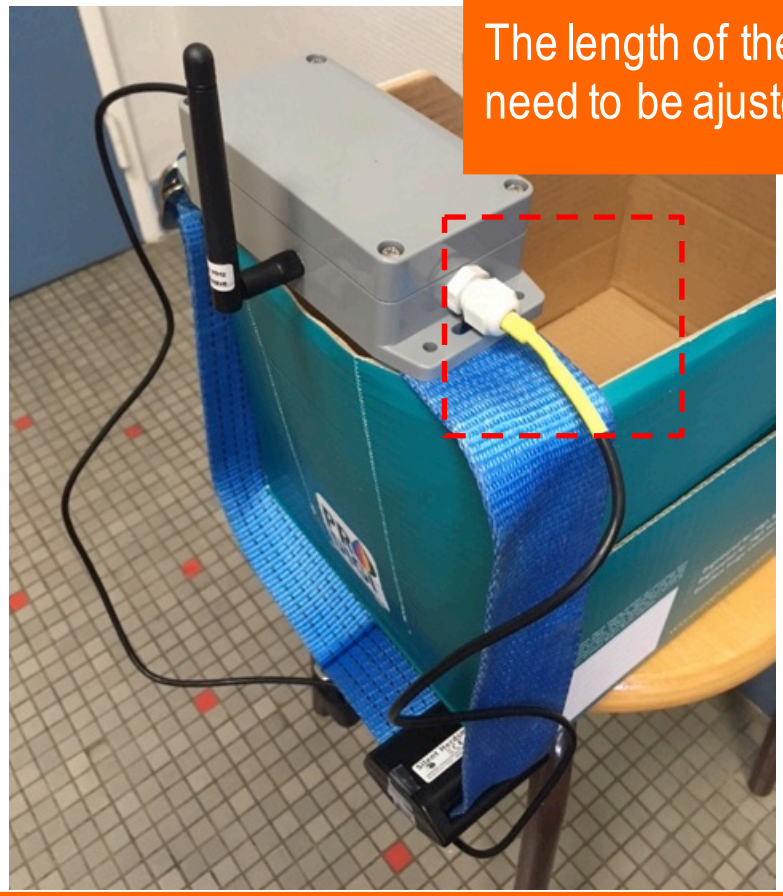
ACTIVATING THE BEACON SYSTEM

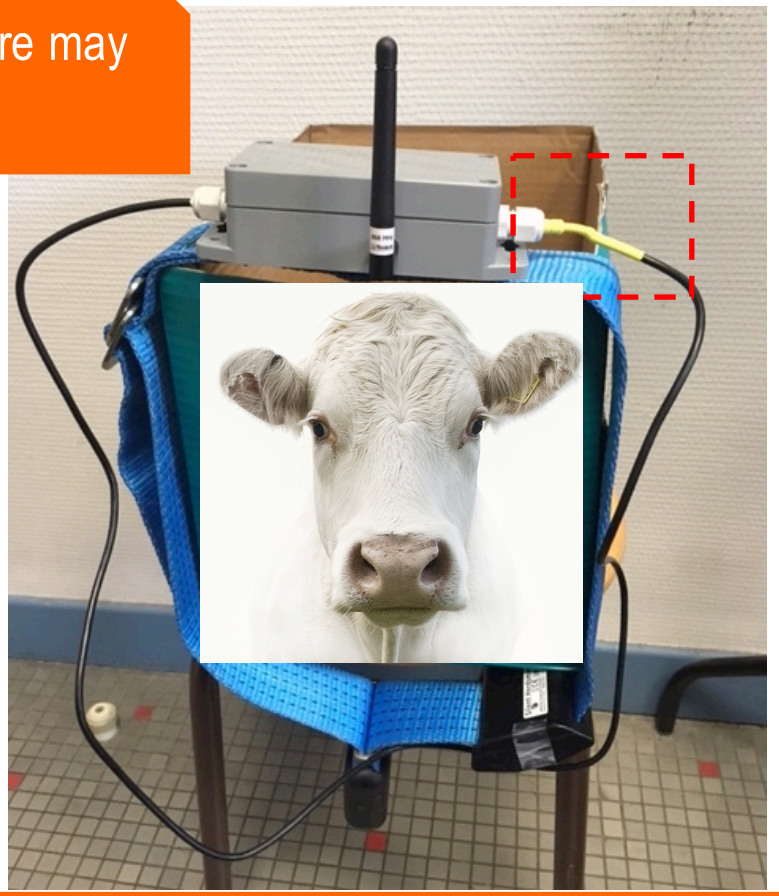
When connecting the male connector MC to the female connector FC 1, the board will be powered and will start sending periodic beacons



FROM BEACON SYSTEM TO BEACON COLLAR

The length of the wire may need to be ajusted

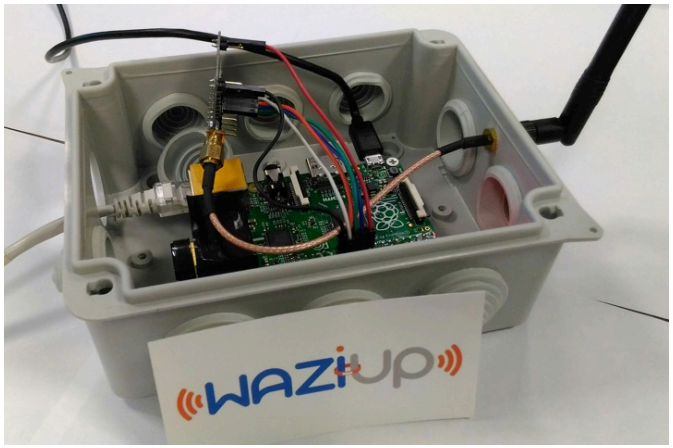




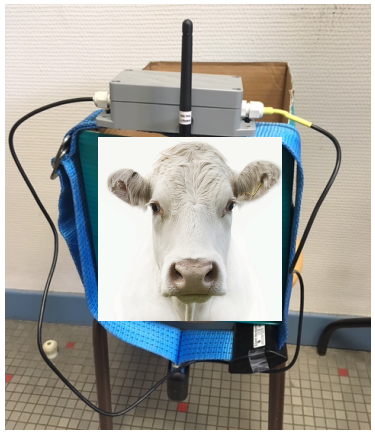
Afimilk collar courtesy of I. Andonovic from University of Strathclyde

Use a robust belt for the collar, fix the box on the belt. Activate the beacon collar by connecting MC to FC by passing the long wire around the cattle's neck. Any attempt to remove or cut the collar will disconnect the beacon system.

DEFAULT CONFIGURATION



\!SRC/14/BC/0



The default configuration in the Arduino_LoRa_Simple_Beacon_collar example is:

Use LoRa mode 1
Send beacons to the gateway every 10 minutes
Node short address is 14
BC (beacon counter) starts at 0, increases by 1 at each beacon, returns to 0 after 65536 beacons

Receiving beacons

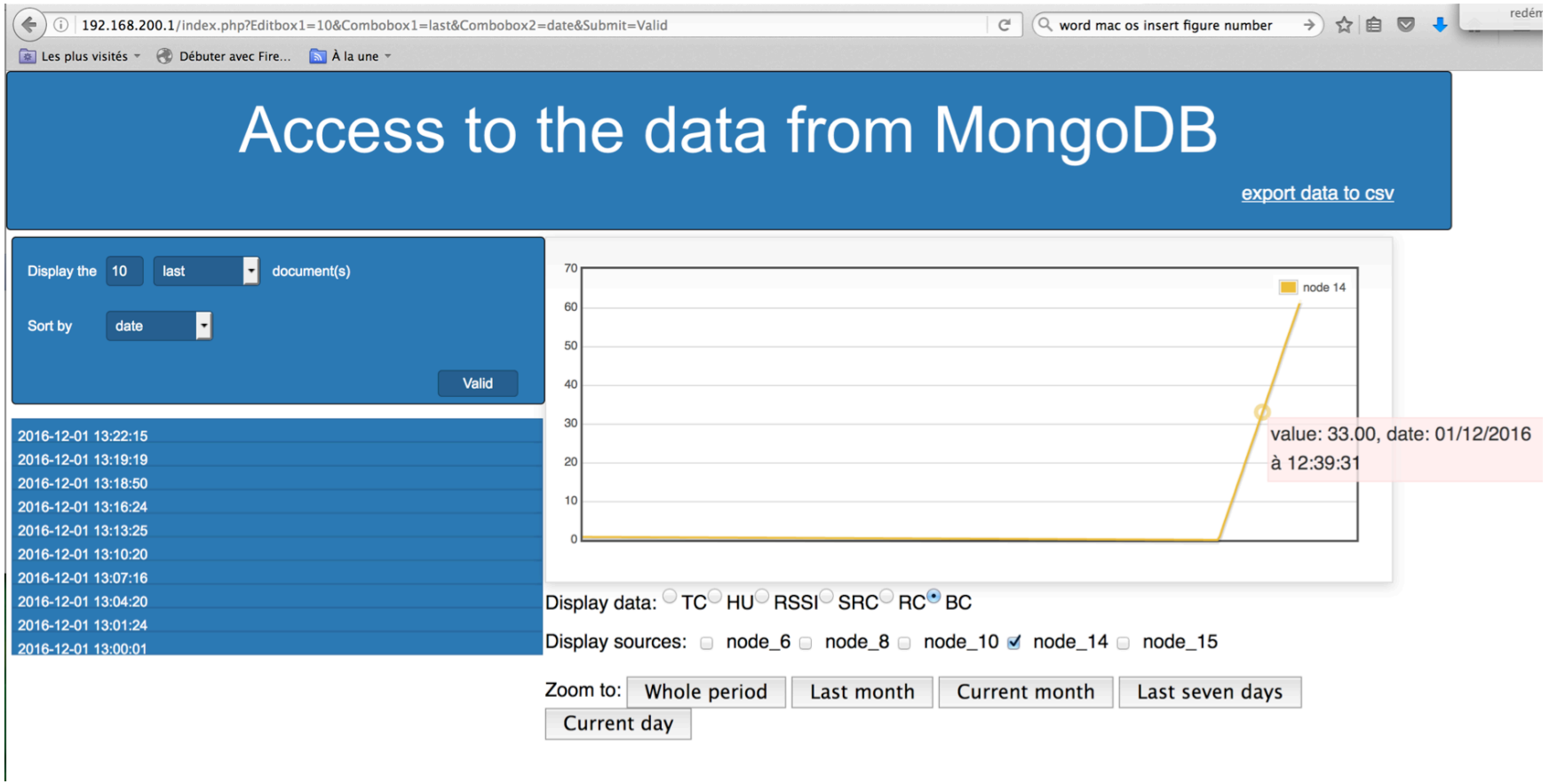
```
--- rxlora. dst=1 type=0x12 src=14 seq=14 len=18 SNR=7 RSSIpkt=-53 BW=125 CR=4/5 SF=12
2016-11-18T12:39:42.566460
rcv ctrl pkt info (^p): 1,18,14,14,18,7,-53
splitted in:  [1, 18, 14, 14, 18, 7, -53]
(dst=1 type=0x12(DATA WAPPKEY) src=14 seq=14 len=18 SNR=7 RSSI=-53)
rcv ctrl radio info (^r): 125,5,12
splitted in:  [125, 5, 12]
(BW=125 CR=5 SF=12)
rcv timestamp (^t): 2016-11-18T12:39:42.565

got first framing byte
--> got data prefix
--> DATA with_appkey: read app key sequence
app key is  0x05 0x06 0x07 0x08
in app key list
valid app key: accept data
SRC/14/BC/14
number of enabled clouds is 1
--> cloud[0]
uploading with python CloudMongoDB.py
MongoDB with max months to store is 2
MongoDB: removing obsolete entries
MongoDB: deleting data older than 2 month(s)...
MongoDB: 0 documents deleted
MongoDB: saving the document in the collection...
MongoDB: saving done
--> cloud end
```

Here, we store in the local MongoDB database

CONNECT TO THE GATEWAY

- Use the WiFi interface
  - SSID is for instance WAZIUP_PI_GW_24EBD4F300
  - Password is loragateway
  - Gateway address is 192.168.200.1

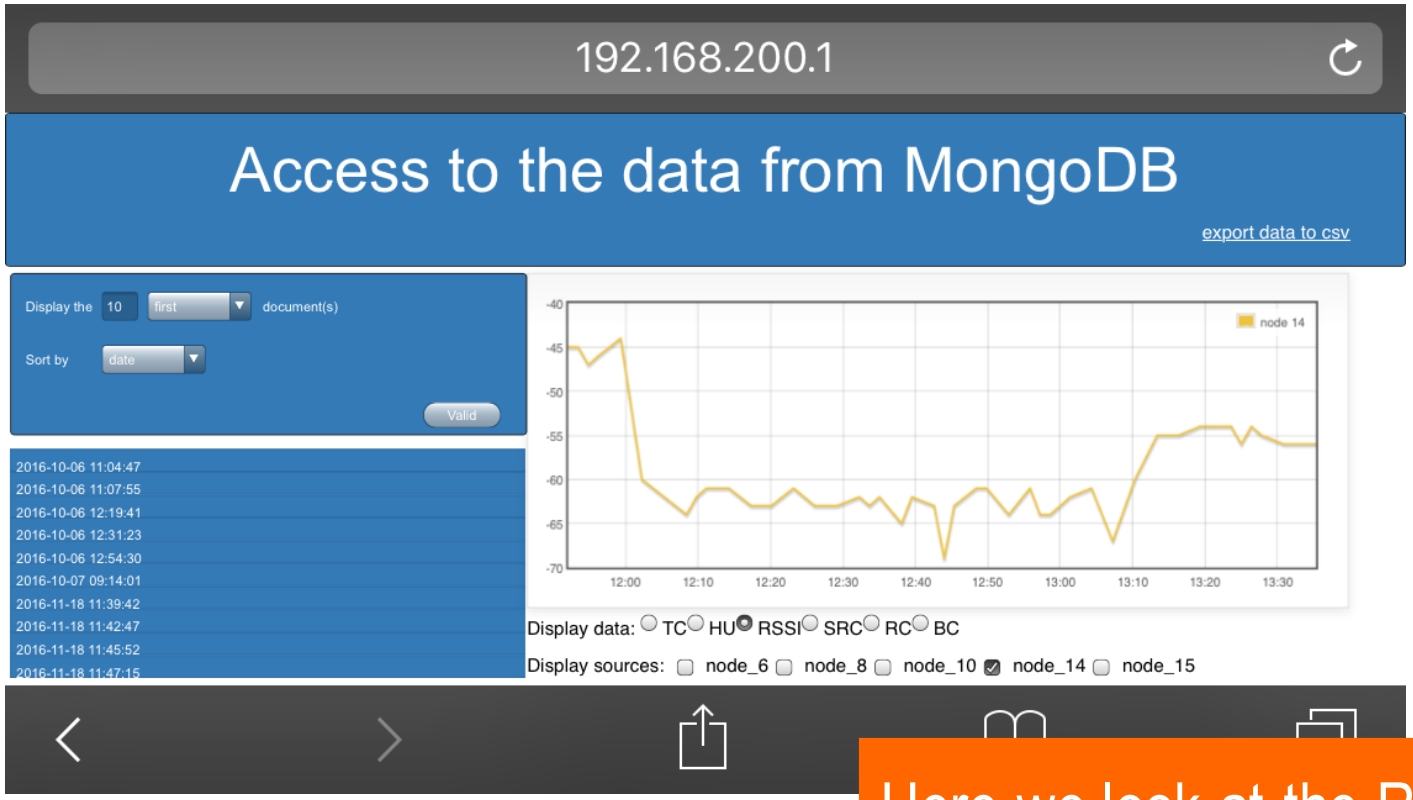
VIEW DATA FROM THE LOCAL WEB SERVER



You can check Beacon Counter to see for gaps (packet losses) and also you can check the time between 2 beacons. If BC comes back to 0, means that the beacon system has been reset (disconnected and reconnected?) which is likely an alarm!

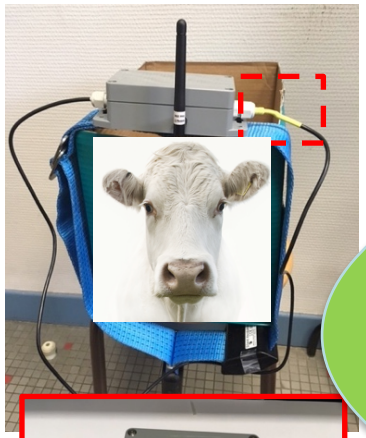
Visualize it on your smartphone!

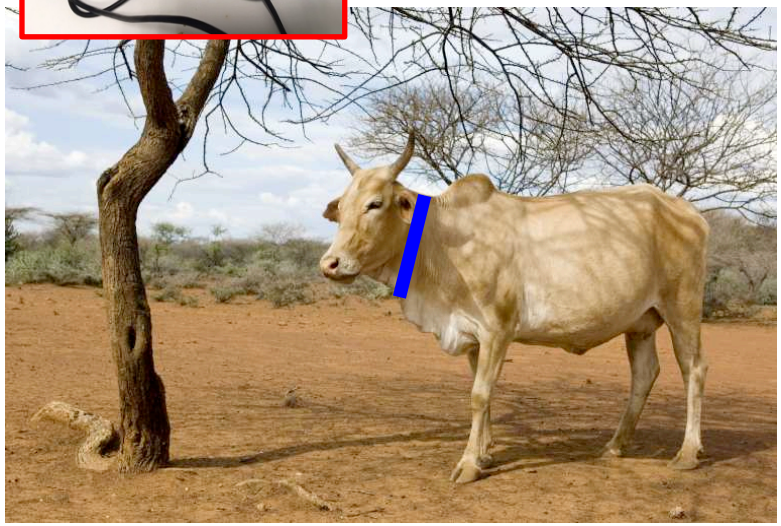
- Don't forget to join the WAZIUP_PI_GW_xxxxxxxxxx WiFi

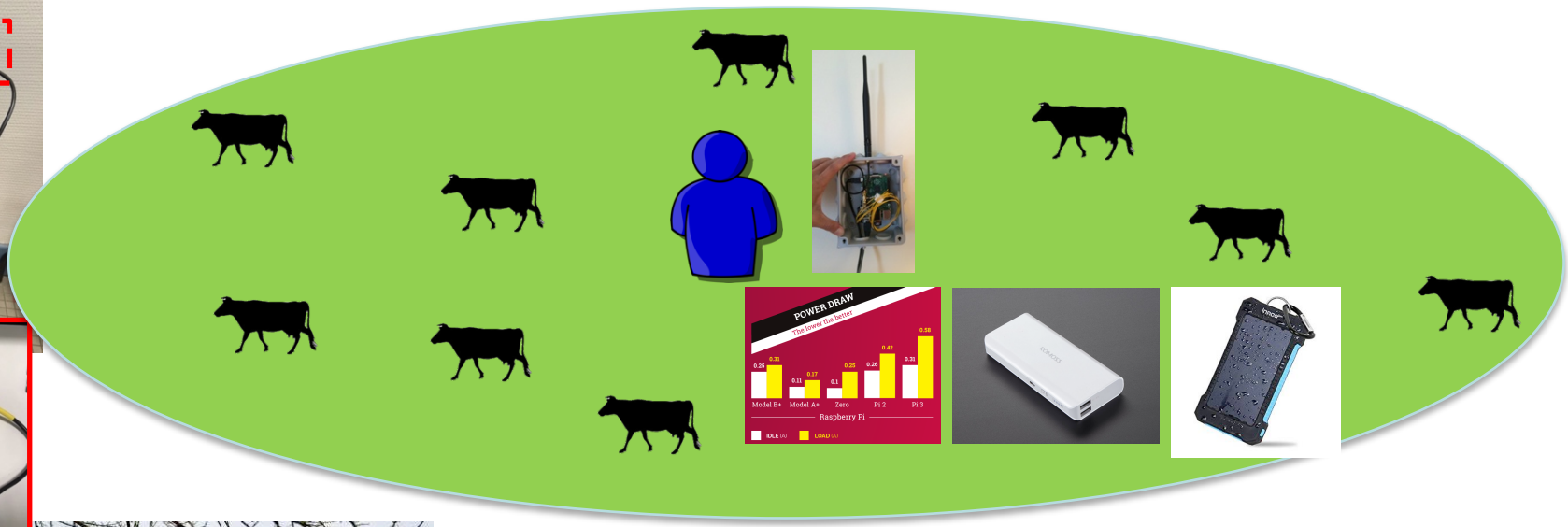


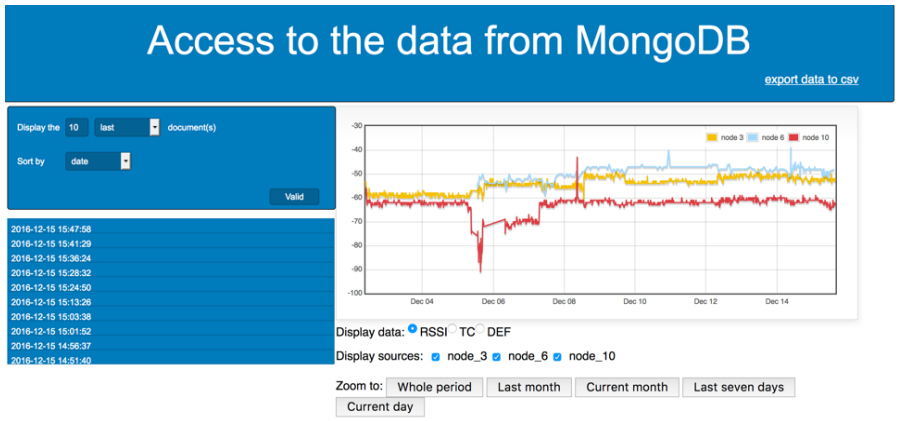
Here we look at the RSSI field

Use an autonomous gateway









WHAT'S NEXT?

- Use beacon information to detect abnormal situation
- When a beacon is received, the following information can be collected
  - Beacon number
  - RSSI of the packet
- Perform field tests for correlation between LoRa BW & SF parameters and range
- Set BW & SF for maximum usage range and use statistical methods to reduce false alarms
- Use machine learning, fingerprinting?
  - Can look at RSSI analysis studies for geolocalization purposes
- Beacon packet can be encrypted to avoid malicious spoofing
- Investigate whether accelerometer data can improve the system