universidade de aveiro
theoria poiesis praxis

DEPARTAMENTO DE ELECTRÓNICA, TELECOMUNICAÇÕES E INFORMÁTICA
MESTRADO EM ENG. DE COMPUTADORES E TELEMÁTICA
ANO 2021/2022

REDES E SISTEMAS AUTÓNOMOS

AUTONOMOUS NETWORKS AND SYSTEMS

PRACTICAL GUIDE 2 – MESH NETWORKS

Objectives

- Set up a mesh network based on LoRa.
- Understand the different roles that each node can have.
- Observe the IPv6 addresses.
- Analyse metrics and log to understand the different events, connecting the smartphone App with BLE.
- Configure the mesh network in different topologies, configuring the role and priority of each node.

Duration

2 weeks

1. Download and install Visual Studio Code, NodeJS and Pymakr plugin

1.1. Download and install VSCode (up to v1.65.2, v1.66 might not work):
https://code.visualstudio.com/Download

1.2. Download and install NodeJS:
https://nodejs.org/en/

1.3. Add the following plugin to VSCode:
https://marketplace.visualstudio.com/items?itemName=pycom.Pymakr

1.4. Connect LoPy4 to your PC using a USB to micro-USB cable. Launch VSCode and it will automatically connect to the board. You should see the MicroPython CLI:
`>>>`

2. LoPy4 introduction

2.1. The LoPy4 is MicroPython-programmable quadruple bearer board. It works with LoRa, Sigfox, WiFi and Bluetooth, making it an exceptional enterprise-grade IoT platform. It is programmable with MicroPython and the Pymakr plugin for fast IoT application development, easy programming in-field and extra resilience with network failover.

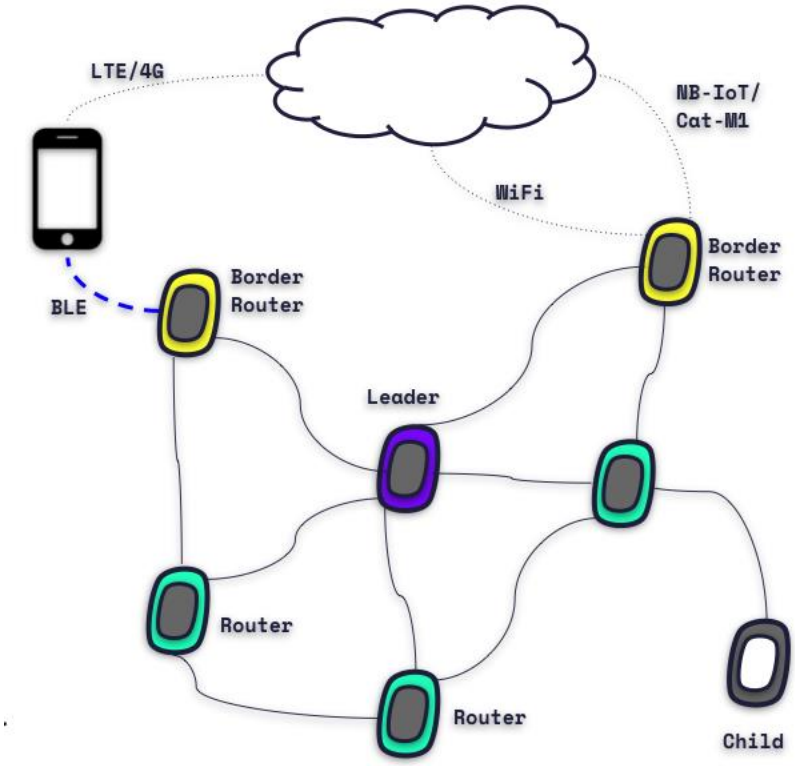
2.2. The LoPy4 boards support Pybytes cloud management. However, due to restrictions in the UA firewall, Pybytes is not reachable by the boards in the campus network. For this reason, the LoPy4 boards which will be used in this practical guide have been prepared in advance with the latest release candidate firmware (1.20.2.rc11) which includes the most recent Pymesh release.

2.3. In VSCode, firstly, create a new folder for the MicroPython workspace. Download the project files from the board in the following way:
Launch the Command Palette (CTRL+SHIFT+P) and search for **Pymakr > Download project.**

2.4. If you are prompted by the question if you would like to download all the files, choose "Yes". You'll see several files:
- **boot.py** This file is empty. This is the first script that runs on the module when it turns on. In this project it will be empty most probably.
- **main.py** This script runs directly after boot.py and should contain the main code you wish to run on your device. This file already contains some code – connection to the WiFi network, a callback to produce the LED blinking when the boards are sending messages and some lines related to Pymesh.
- **pybytes_config.json** This config file has the configuration needed for Pybytes (not used in this guide).
- **pymesh_config.json** This config file has the configuration for Pymesh – key of the mesh network, MAC, Bluetooth name, LoRa settings, etc.
- **pymesh.py** This script doesn't run automatically. It has commands useful for the Pymesh experiment.

3. Pymesh introduction

3.1. Pymesh is the LoRa full-mesh network technology. A Mesh network acts like a net, this means that any node within the network can connect with any other node.



3.2. A LoPy4 can have any of the Pymesh Node Role: Leader, Router, Child or Border Router.
We can identify each role a LoPy4 has at a determined moment looking to the LED:
Magenta - LEADER
Green - ROUTER
White - CHILD
Red - Searching / Detached from any Pymesh
Cyan - SINGLE LEADER (no other Router in the same Pymesh)

Or running `print(pymesh.status_str())` in the CLI:
PYMESH_ROLE_DISABLED = 0
PYMESH_ROLE_DETACHED = 1
PYMESH_ROLE_CHILD = 2
PYMESH_ROLE_ROUTER = 3
PYMESH_ROLE_LEADER = 4

3.3. A mesh network must have at least one border router. The border router connects to the internet using BLE or Wifi. If there are more than one border router, you can set different priority levels. In the default configuration for this practical guide, the **main.py** and **pymesh_config.json** are the same for every board – all are set as border routers with same priority.

4. First mesh network

4.1. Organize groups of 3 or 4 students (each board per student connected to VSCode) to set up a mesh network. The boards are pre-configured in the following way:
- MAC address corresponds to the tag number (1 to 21)
- Network key of board numbers 1 to 4: **0d914ad1-83a22b8f-8fe4bb7e-ab36c5cc**
- Network key of board numbers 5 to 8: **16a1148f-252feb12-981d048f-e75ef9ec**
- Network key of board numbers 9 to 12: **58e8e54c-713899f3-18a49e80-d6909d5a**
- Network key of board numbers 13 to 16: **f38d8d83-163708ad-cafd91fb-c3f940b8**
- Network key of board numbers 17 to 21: **cdaa3e76-e84116b1-3c0d50c9-052e2e11**

4.2. Place the boards close to each other. Because the boards don't have a LoRa antenna, the range is expected to be within 1 meter. Wait until the mesh network is formed. You may need to reboot some of the boards for them to detect the neighbours and change to a correct role. Ensure they don't have the LED with cyan color (single leader), as it will indicate it is "alone".

4.3. Right click in the **pymesh.py** file and click "Pymakr > Run current file" to start the Pymesh CLI. This CLI is dedicated to the Pymesh feature only.

4.4. Run "h" to see the available commands. Run "mml" to list the devices already connected to the mesh network – you should see the MAC addresses of all the devices. If they are not listed, repeat the procedure in **4.2.**

4.5. Send messages between nodes to test the network. In the Pymesh CLI, run "s", insert the MAC address of the other node, type a message, and insert the number of repetitions. Observe the receiving node with the LED blinking when it receives the message. In the receiving node, run "rm" to check the content of the last received message.

4.6. Find the IPv6 addresses of each node. Try to explain it.

5. Android app for Pymesh metrics

5.1. From your smartphone, download the Android app from:
https://github.com/pycom/pycom-libraries/tree/master/pymesh/mobile_app

5.2. Install the App using sideloading (if your smartphone allows):
https://www.xda-developers.com/how-to-sideload-install-android-app-apk/

5.3. Download lib.zip from elearning and upload to the LoPy4. Include in main.py `import ble_rpc`

5.4. Change in the "pymesh_config.json" **ble_api** to "true".

5.5. Connect to the LoPy4 and analyse the metrics:
https://docs.pycom.io/pymesh/lib-ble-rpc/

6. Extend the mesh network to more nodes

6.1. Extend the network of each group to add more nodes (of the other groups). To do this, change the network key in the **pymesh_config.json** in the new nodes to match the same key of the current nodes.

6.2. Together with your colleagues think of an architecture like what is shown in **3.1**. Change the role of each node to build the envisioned architecture. Confirm the status of the network with the LED colours, and commands of the pymesh CLI. You can try to increase the LoRa transmission power to ease the network build up with better coverage (e.g. 20dBm).

```
>>> pymesh.cli_start()
>h
List of available commands
br - enable/disable or display the current Border Router functionality
brs - send packet for Mesh-external, to BR, if any
buf - display buffer info
config - print config file contents
debug - set debug level
gps - get/set location coordinates
h - help, list of commands
ip - display current IPv6 unicast addresses
mac - set or display the current LoRa MAC address
mml - display the Mesh Mac List (MAC of all nodes inside this Mesh), also inquires Leader
mp - display the Mesh Pairs (Pairs of all nodes connections), also inquires Leader
ot - sends command to openthread internal CLI
pause - suspend Pymesh
resume - resume Pymesh
rm - verifies if any message was received
rst - reset NOW, including NVM Pymesh IPv6
s - send message
self - display all info about current node
sleep - deep-sleep
stop - close this CLI
tx_pow - set LoRa TX power in dBm (2-20)
ws - verifies if message sent was acknowledged
```