universidade de aveiro
theoria poiesis praxis

Departamento de Electrónica, Telecomunicações e Informática
Mestrado em Eng. de Computadores e Telemática
Ano 2024/2025

Redes e Sistemas Autónomos

Autonomous Networks and Systems

Practical Guide 2 – Ad-hoc Networks

Objectives

- Set up B.A.T.M.A.N. ad-hoc network
- Validate the multi-hop ad-hoc network with simple tests
- Observe the status of the network
- Try new topologies of the ad-hoc network to force different paths
- Analyse metrics of the network
- Compare with 802.11s

Duration

2 weeks

1st week

Introduction

The Better Approach to Mobile Ad-hoc Networking (B.A.T.M.A.N.) is a protocol for multi-hop mobile **ad hoc networks**. The official documentation is available here.

In this practical guide, we will learn how to deploy a Wi-Fi ad-hoc network with support for multi-hop. Supporting more than one hop means the network will behave like a mesh network, where the nodes can reach other nodes that are directly or indirectly connected, if they are part of the same batman group and Wi-Fi channel.

Environment

The setup we will be mounting for the experience uses Raspberry Pi (RPi) 5 4GB, and there is **one available per student**:

Raspberry Pi 5 4GB Dev Kit

Each RPi is identified with a sticker, example "NAP 7**06**". For this example, the **id** of the device is **6**.

Every RPi is already flashed with the official Raspbian and it has already some pre-configuration (the batman binary is installed).

In terms of networking and authentication, the board is configured with the following:

- Hostname: raspberrypi-7**id**
- Ethernet interface: 192.168.3.**id** (it also accepts DHCP)
- SSH login
  - Username: nap
  - Password: openlab

The internal WiFi chipset of the RPi will be used for the batman setup. The important thing about this chipset is that it allows the configuration of the Wi-Fi interface in **IBSS mode** which is required for the configuration of ad-hoc networks.

Form groups of 4 people to perform this guide.

1. Connecting your PC to the RPi board

1.1. Configure **your PC** with an IP in the **same LAN** of the RPi Ethernet interface (192.168.3.0/24) and connect an Ethernet cable between your PC and RPi.

Example: 192.168.3.1/24 (**no** gateway needed)

1.2. Use **ssh** to connect to the RPi, using the IP and authentication described in the previous page. In Windows use Putty or ssh through WSL. In Linux use ssh directly.

2. Configure the Wi-Fi interface in preparation for the batman network.

2.1. Run the following commands, so the wpa_supplicant service doesn't interfere with the configuration we will do for batman:

```
sudo service wpa_supplicant stop
sudo systemctl mask wpa_supplicant.service
sudo killall wpa_supplicant
```

2.2. Change the Wi-Fi interface mode to IBSS

(check the Wi-Fi interface name (`ip a`), and use it to **replace** with the right name in the following commands)

```
sudo ip link set wlanX down
```

```
sudo iw wlanX set type ibss
```

```
sudo ifconfig wlanX mtu 1500
```

2.3. Select the right Wi-Fi channel. Every group with 4 boards will create a separated ad-hoc network. Select channels among the following list [3, 5, 7, 9, 11]

```
sudo iwconfig wlanX channel Y
```

```
sudo ip link set wlanX up
```

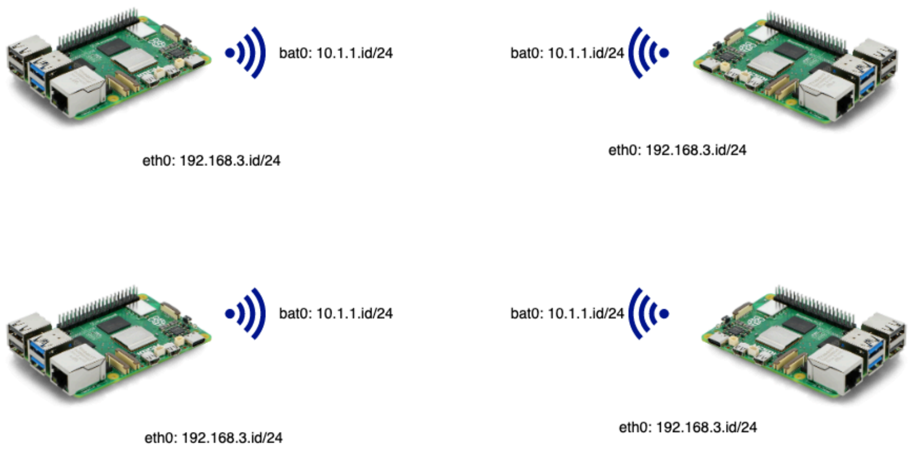
2.4. Confirm if the Wi-Fi interface is in the correct mode **(ibss)**, with the following command:

(NOTE: ignore the frequency value. It will be set afterwards!)

```
sudo iw dev
```

3. First batman network configuration

3.1. The network of every group will look like the following figure:



3.2. Find out (**search online**) what is the frequency (**<u>f0</u>**), **<u>in MHz</u>**, for the Wi-Fi **<u>channel</u>** selected in the configuration of exercise **2.3**.

3.3. Then, invent a **<u>network name</u>** for your batman network.

3.4. On your RPi, run the following command, according to the frequency and the chosen ad-hoc network name:

```
sudo iw wlanX ibss join network_name f0
```

3.5. Finalize the batman configuration with the following:

```
sudo modprobe batman-adv
```

```
sudo batctl if add wlanX
```

```
sudo ip link set up dev wlanX
```

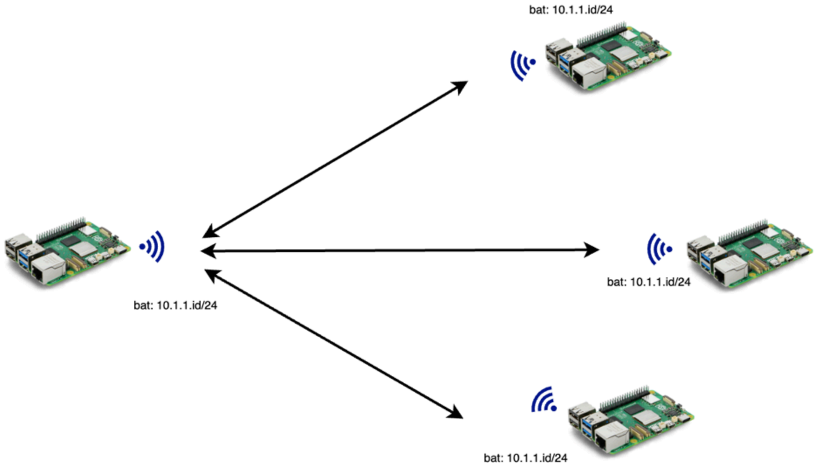
```
sudo ip link set up dev bat0
```

```
sudo ifconfig bat0 10.1.1.id/24
```

(Configure the IP for the batman LAN, according to the **id** of your board)

4. Testing the batman network for the first time

When the boards are placed together and close enough, each board connects to the other in a direct way, as seen in the following figure from the perspective of one of the RPis:



4.1. Together with the colleagues from the other groups, test the wireless connection to their RPis, using the ping command. Validate that you can reach every node of the ad-hoc network.

`sudo batctl ping 10.1.1.X` (layer 2 test)

```
ping 10.1.1.X
```

4.2. Use the following to check the neighbors table. The command `watch` is to keep observing in real time:

```
sudo watch -n 0.1 batctl n
```

Example of output:

```
Every 0.1s: batctl n
raspberrypi-706: Thu Mar 16 19:52:03 2023[B.A.T.M.A.N. adv
2021.3…
IF             Neighbor             last-seen
   wlan0     24:ec:99:b4:0f:a6     0.444s
   wlan0     24:ec:99:8b:e7:08     0.128s
```

5. Analysis of the batman topology and conditions

5.1. Utilizing the ssh connection to each RPi, save the **MAC address** of the wireless interface (wlan0) and map it with the id of the board, so you can identify the devices.

5.2. Now run the command to check the **originators table**. This command lists all other nodes in the network and **registers which direction to send the packets** (marked with *). Identify who is the nexthop. In the current topology, all should be direct.

```
sudo watch -n 0.1 batctl o
```

Example of output:

```
Every 0.1s: batctl o              jetson-14: Thu Mar 16 19:53:05
2023[B.A.T.M.A.N. adv 2021.3…
Originator       last-seen (#/255) Nexthop      [outgoingIF]
  24:ec:99:8b:e7:08 0.696s ( 12) 24:ec:99:b4:0f:a6 [ wlan0]
* 24:ec:99:8b:e7:08 0.696s ( 43) 24:ec:99:8b:e7:08 [ wlan0]
  24:ec:99:b4:0f:a6 0.756s ( 10) 24:ec:99:8b:e7:08 [ wlan0]
* 24:ec:99:b4:0f:a6 0.756s ( 78) 24:ec:99:b4:0f:a6 [ wlan0]
```

The column #/255 is the sequence number.

2nd week

In this week's guide we will perform a **comparison** between **two approaches** used to **determine routes** in wireless ad-hoc networks. In the previous week we already explored a **proactive approach** with the B.A.T.M.A.N routing protocol. For this week we will still estabilish a mesh network with B.A.T.M.A.N, and **at the same time** we will establish a **secondary mesh network** based on the **802.11s** standard for mesh Wi-Fi networks. This standard uses by default a **reactive routing** protocol called **Hybrid Wireless Mesh Protocol** (HWMP), that is **based** on the **AODV** routing protocol, HWMP allows other modes, it can be configured to be proactive, or hybrid between the two.

6. Automatic configuration of batman

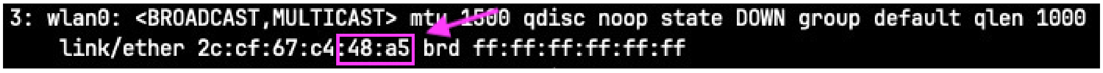
6.1. In order to make the setup of the B.A.T.M.A.N network easier, we have already created a Bash script (setup_batman.sh) in the home directory of the board. Now you only need to run the script:

```
./setup_batman.sh wlan0 10.1.1.id/24 36 5180 rsatestSSID
```

For this week's class we will be using the 5GHz range of the internal Wi-Fi card, RPi 5 only supports some channels, along with corresponding frequencies.You can check all available channels and frequencies with the command `iw list` . Here are some you can use today:

Channel	Frequency (MHz)
36	5180
40	5200
44	5220
48	5240
149	5745

6.2. Make sure to take note of the Wi-Fi interface MAC address by running the command `ip a` , identifying the **wlan0** interface and saving the last **4 digits** of the hardware address. Example output:



```
3: wlan0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 2c:cf:67:c4:48:a5 brd ff:ff:ff:ff:ff:ff
```

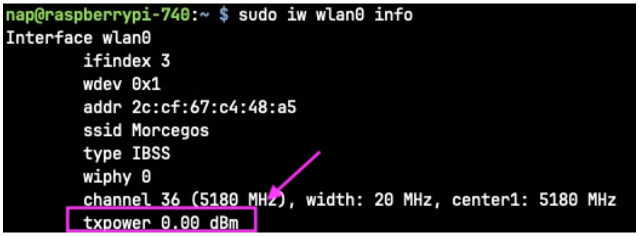
6.3. Change the power transmission on the physical interface. This will allow the creation of more scenarios with more ease (less distance apart to create hops). Use **0dBm**, as seen in the following command example, with the following command:

```
sudo iw dev wlan0 set txpower fixed 0
```

Verify if it has been set properly with the command:

```
sudo iw wlan0 info
```

Output example:

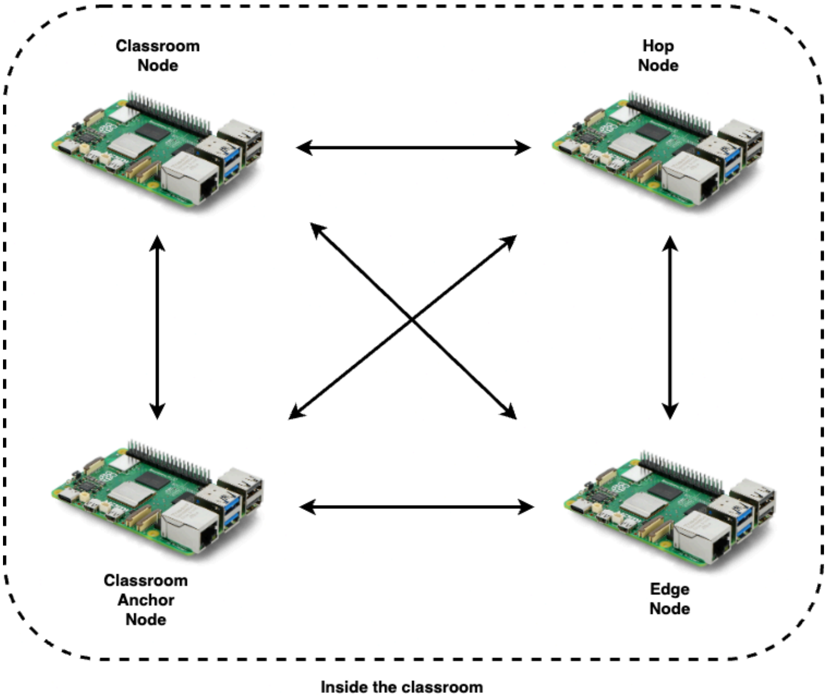


```
nap@raspberrypi-740:~ $ sudo iw wlan0 info
Interface wlan0
        ifindex 3
        wdev 0x1
        addr 2c:cf:67:c4:48:a5
        ssid Morcegos
        type IBSS
        wiphy 0
        channel 36 (5180 MHz), width: 20 MHz, center1: 5180 MHz
        txpower 0.00 dBm
```

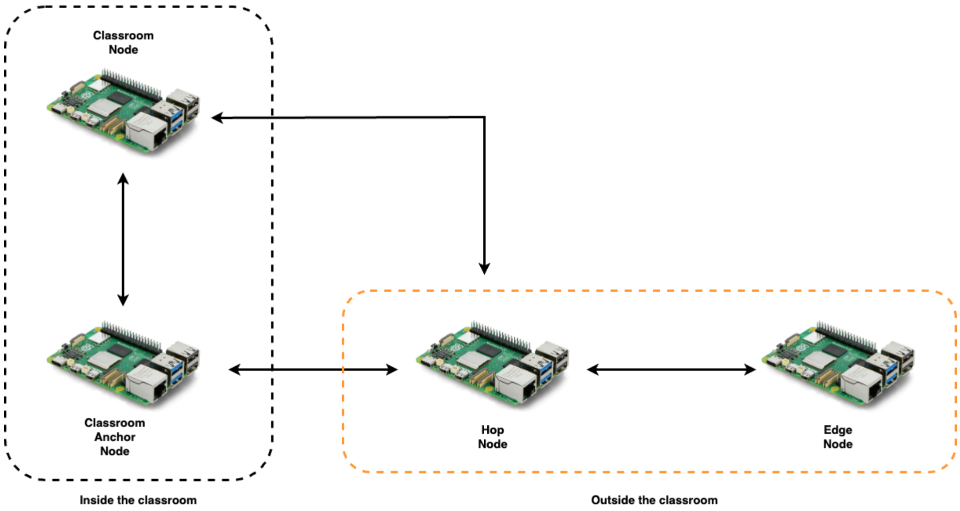
7. Evaluating B.A.T.M.A.N performance

7.1. Using the following topologies, we will now evaluate the performance of the mesh network in those scenarios. Establish **two ssh** connections to your board, open a new terminal window in your computer and connect to the board. You will need two windows: one to **monitor the neighbours table** and another to perform the commands needed to evaluate the network. Choose the members to fulfil the roles and maintain them for all scenarios. For each of the following scenarios perform exercise 7.2:

- 1 – All nodes have a **direct connection to each other**.



- 2 – One of the nodes is **only reachable with one hop through another node**.



7.2. Fill the following table, with tests for the two scenarios, and interpret the results:

Scenario	Source (MAC)	Next hop (MAC)	Destination (MAC)	**tp**	**rtt**
1					
2					

tp = throughput (Mbps)

Measure with the command:

```
sudo batctl tp <Edge_Node_MAC_address>
```

rtt = round trip time (ms)

Measure with the ping command from the **Classroom Anchor Node** to the **Edge Node**, look carefully at the first ***time*** returned by the command (take a note of it).

8. Setup of 802.11s Mesh network

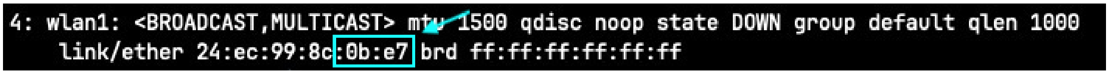
8.1. Similar to the setup needed to create the B.A.T.M.A.N network, we have already provided a Bash script (setup_s.sh) in the home directory of the board You can see this script with the command `cat setup_s.sh` , in order to see the configuration of a 802.11s network. By now you should've already noticed the black USB Wi-Fi pen attached to your board, we will use it to deploy the 802.11s network in the **2.4GHz** range. To deploy the network, you only need to run the script**:**

```
./setup_s.sh wlan1 10.1.2.id/24 11 rsatestMeshID
```

Remember to use the **wlan1** interface, which corresponds to the interface name of the Wi-Fi pen. Now the name you choose for your network is only an ID used in the standard, it no longer appears as a conventional Wi-Fi network.

After the interface is configured DON'T PING EACH OTHER!!

8.2. Make sure to take note of the Wi-Fi interface MAC address by running the command `ip a`, identifying the **wlan1** interface and saving the last **4 digits** of the hardware address. Example output:

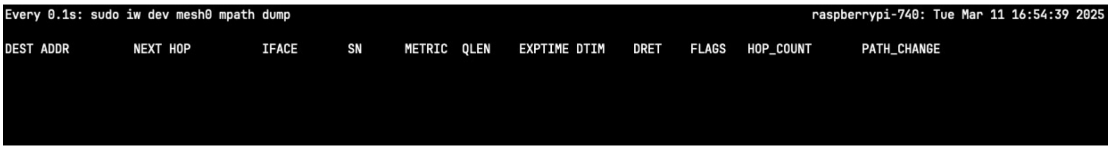


```
4: wlan1: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 24:ec:99:8c:0b:e7 brd ff:ff:ff:ff:ff:ff
```

8.3. In order to check your neighbour’s status you can run the following command:

```
sudo watch -n 0.1 sudo iw dev mesh0 mpath dump
```

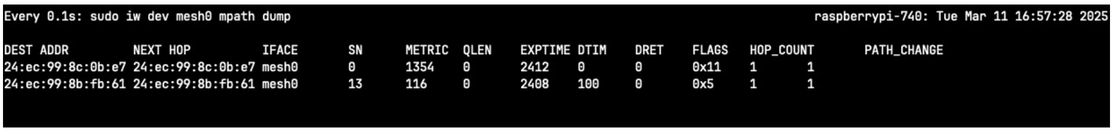
Example output (**yours should remain empty**):



```
Every 0.1s: sudo iw dev mesh0 mpath dump                                                    raspberrypi-740: Tue Mar 11 16:54:39 2025

DEST ADDR         NEXT HOP          IFACE       SN      METRIC  QLEN    EXPTIME DTIM    DRET    FLAGS   HOP_COUNT       PATH_CHANGE
```

Filled example:



```
Every 0.1s: sudo iw dev mesh0 mpath dump                                                    raspberrypi-740: Tue Mar 11 16:57:28 2025

DEST ADDR         NEXT HOP          IFACE       SN      METRIC  QLEN    EXPTIME DTIM    DRET    FLAGS   HOP_COUNT       PATH_CHANGE
24:ec:99:8c:0b:e7 24:ec:99:8c:0b:e7 mesh0       0       1354    0       2412    0       0       0x11    1       1
24:ec:99:8b:fb:61 24:ec:99:8b:fb:61 mesh0       13      116     0       2408    100     0       0x5     1       1
```

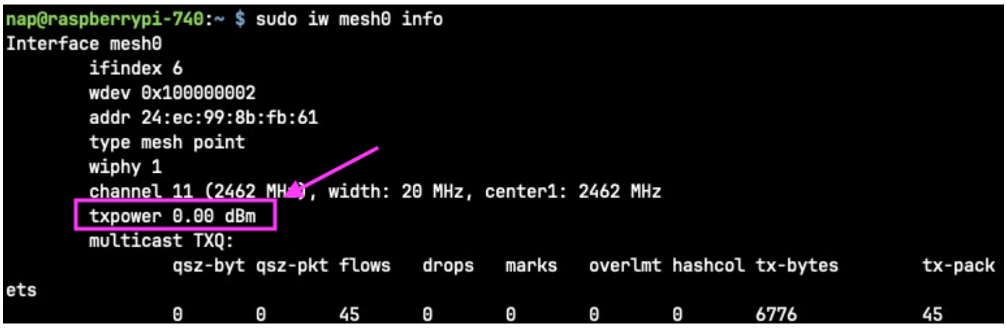
8.4. Change the power transmission on the physical interface. This will allow the creation of more scenarios with more ease (less distance apart to create hops). Use **0dBm**, as seen in the following command example, with the following command:

```
sudo iw dev mesh0 set txpower fixed 0
```

Verify if it has been set properly with the command:

```
sudo iw mesh0 info
```

Output example:



```
nap@raspberrypi-740:~ $ sudo iw mesh0 info
Interface mesh0
        ifindex 6
        wdev 0x100000002
        addr 24:ec:99:8b:fb:61
        type mesh point
        wiphy 1
        channel 11 (2462 MHz), width: 20 MHz, center1: 2462 MHz
        txpower 0.00 dBm
        multicast TXQ:
                qsz-byt qsz-pkt flows   drops   marks   overlmt hashcol tx-bytes        tx-pack
ets
                0       0       45      0       0       0       0       6776            45
```

9. Evaluating 802.11s mesh performance

9.1. Using the same topologies as in 8.1, we will now evaluate the performance of the 802.11s mesh network in those scenarios. Establish **two ssh** connections to your board, open a new terminal window in your computer and connect to the board. You will need two windows: one to **monitor the neighbours table** and another to **perform the commands** needed to evaluate the network. **Perform first in each scenario the rtt measurement**.

9.2. Fill the following table, with tests for the two scenarios, and interpret the results:

tp = throughput (Mbps)

Measure with steps below:

- In the **Classroom Anchor Node** perform the following command to enable the an iperf3 server to perform the throughput measurement:

  ```
  iperf3 -s -p 7575
  ```

- In the **Edge Node** perform measurement with the following iperf3 client command:

  ```
  iperf3 -c <Anchor Node IP> -p 7575 -f M
  ```

rtt = round trip time (ms)

Measure with the ping command from the **Classroom Anchor Node** to the **Edge Node**, look carefully at the first ***time*** returned by the command (take a note of it).

Scenario	Source(MAC)	Next hop (MAC)	Destination (MAC)	tp	rtt
1					
2					