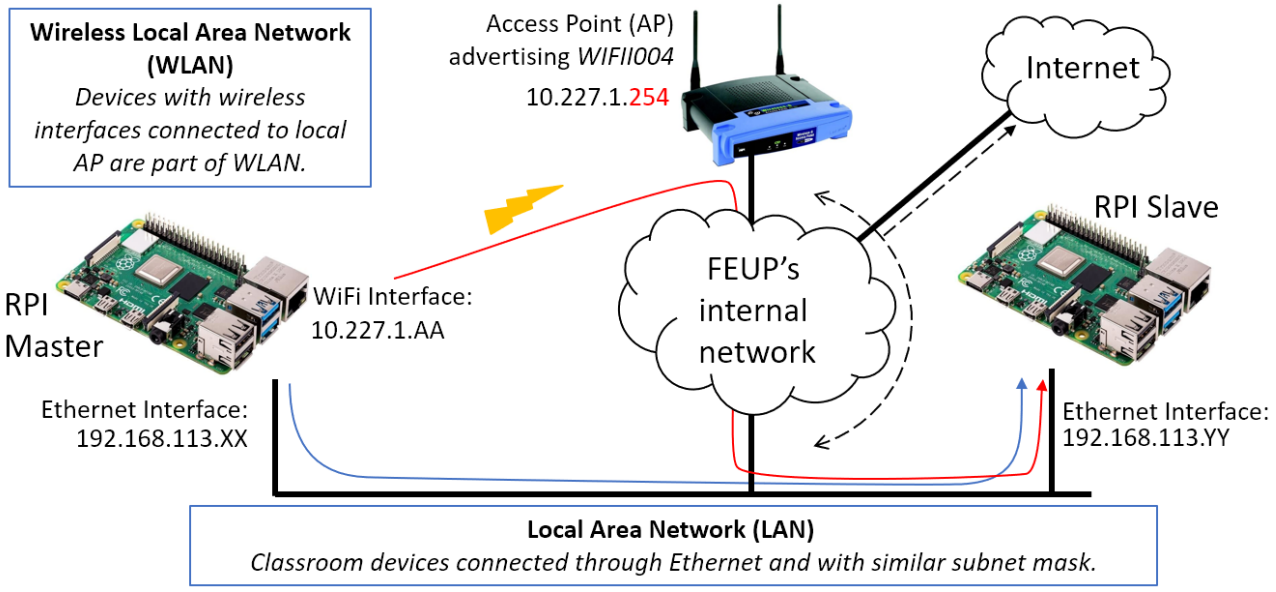
Measuring Response Times through Ethernet and WLAN

Pedro Santos, Luís Almeida, Mário Sousa, Paulo Portugal
ACI – 2020/2021

In this assignment, we will evaluate the Network Reaction Time of Modbus over the TCP/IP protocol stack over two physical media – **cable (Ethernet)** and **wireless** (defined by the **IEEE 802.11** standard, with commercial name ***WiFi***), while learning some fundamental concepts of IP networking and Ethernet and IEEE 802.11 connectivity.

The following setup will be used (refer to the figure below):

- **Room I004 (lab) Ethernet local area network (LAN)** – Ethernet network connecting all nodes in room I004.
- ***FEUPnet*** – network connecting the I004 LAN to other Ethernet LANs in FEUP, to *Eduroam*, and to the Internet.
- **Wireless local area network (WLAN)** ***WIFI004*** – a wireless network available in room I004. Despite the name, it is connected to *FEUPnet* and not directly to the Ethernet LAN of room I004.
- **RPI Master** – RPI executing the Modbus Master; during the assignment, it may be connected to the WLAN *WIFII004* and/or to room I004's Ethernet LAN.
- **RPI Slave** – RPI executing the Modbus Slave; remains connected to the Ethernet LAN of room I004 for the full duration of the assignment.



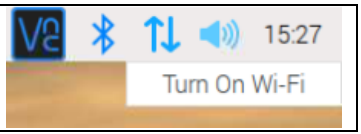
To observe the difference in time response, we will collect samples of the Network Reaction Time (TRR) when RPI Master is connected to the room's Ethernet LAN or to **WIFII004** WLAN. Differences will be caused by:

- All nodes in room I004 connected to the Ethernet network are in the same IP sub-network (192.168.113.0); the WLAN network we will be using (**WIFII004**) is connected to *FEUPnet*, and thus packet routes are longer (i.e., go through more machines).
- The media have different medium access mechanisms and propagation characteristics.

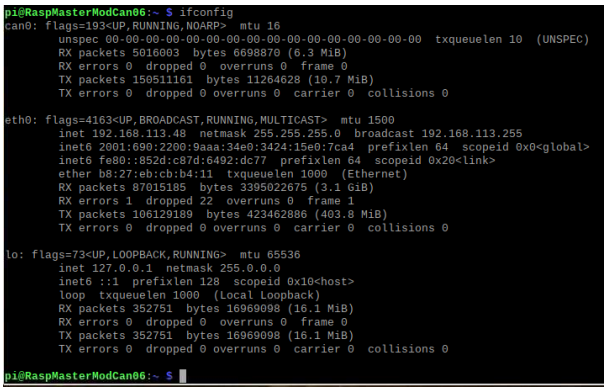
1. Network Interfaces and IP Layer Routing

1.1. Log in to the **RPi Master** via VNC. First of all, check that **WIFI is off** in the top right corner of the desktop.

IMPORTANT : Check that the WIFI connection is OFF



Open the **terminal** and run the command **`ifconfig`**. This command provides you networking/routing information about the network interface cards (NIC) available in the RPi. There should be at least three: WLAN (**`wlan0`**), Ethernet (**`eth0`**), and loopback (**`lo`**). You will notice that, unlike the Ethernet (**`eth0`**) interface, the WLAN interface does not have an IP address assigned. Register the results for later comparison (e.g., take a screenshot or take care not to close the terminal).



The main line to read carefully, under the **eth0** entry, is the line containing the IPv4 address information:

`inet 192.168.113.XX netmask 255.255.255.0 broadcast 192.168.113.255`

These fields correspond to the following:

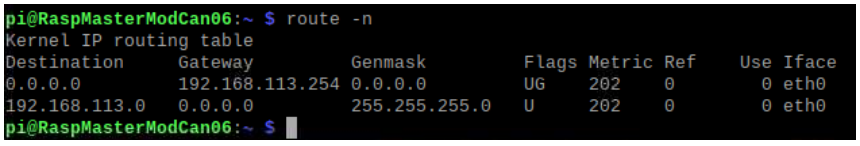
- **`inet`** – the IP address of the **eth0** network interface card
- **`netmask`** – **network mask:** a binary mask that identifies whether a target packet is to be transmitted to (or arrived from) the local network or a node outside the local network. This will be necessary for routing packets, as we will see next.

> Example: consider that you received/will send two packets with origin/destination IP addresses **192.168.113.23 and 192.168.111.25** and your subnet and network mask are **192.168.113.0** and **255.255.255.0**, respectively
>
> - **`192.168.113.23 AND 255.255.255.0 == 192.168.113.0`**
> ⇨ **`Packet to/from node in local network`**
> - **`192.168.111.25 AND 255.255.255.0 != 192.168.113.0`**
> ⇨ **`Not a packet to/from node in local network`**

- **`broadcast`** – the destination IP address to use in packets meant to be broadcast in the local network

Check if the same line is found under the **`wlan0`** entry.

1.2. Enter the command **route -n**, that provides you with the routing information used by the OS to route packets produced by applications or received from the network.



We find the following columns:

- **`Destination`**: IP address of destination node or network. The IP address **0.0.0.0** identifies the default case, i.e., the gateway to use if none of the remaining rules applies. Also, recall that IP addresses ending in **0** (e.g., **192.168.113.0**) identify networks and not specific nodes.
- **`Gateway`**: IP address of corresponding gateway to reach that node or network.
- **`Genmask`**: similar to **netmask**. Through the matching operation between the destination IP address of the packet of interest and the destination IP address in the table entries (up to the bits defined by **`genmask`**, as described above), the TCP/IP stack identifies which route to use.

Revisiting the previous example:

- **`192.168.113.23 AND 255.255.255.0 == 192.168.113.0`**
  - ⇨ **`Packet to/from node in local network`**
  - ⇨ **`Request MAC address of target node (192.168.113.23)`**

- **`192.168.111.25 AND 255.255.255.0 != 192.168.113.0`**
  - ⇨ **`Not a packet to/from node in local network`**
  - ⇨ **`Request MAC address of local gateway`**
    `(Q:does the destination IP address in the IP header change?)`

- **`Flags`**: **U**: route is up; **G**: use gateway.
- **`Metric`**: '*distance*' in number of hops; used to indicate preferential routes.
- **`Iface`**: associated network interface (Ethernet, Wireless LAN, etc.)

Register the results for later comparison (i.e., don't close the terminal, or take a screenshot).

1.3. Ping the **RPI Slave** to its IP address in the Ethernet LAN. Open the **terminal** and input the command **`ping 192.168.113.YY`** . Check for a successful ping; take note of the duration of the various attempts for future comparison with the WLAN case.

1.4. Trace the route that packets followed between the **RPI Master** and the **RPI Slave**: **`traceroute 192.168.113.YY`** . This command lets you know all the machines that your packets went through before reaching the **RPI Slave**. Take note of how many hops the packets went through, for later comparison with the WLAN case.

1.4. Run also **`traceroute www.google.com`** . Interpret the results.

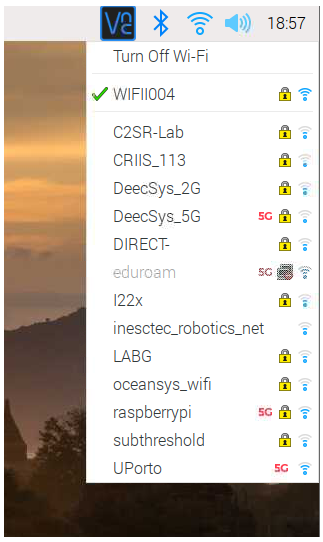
2. Connecting to the WLAN

2.1. Enter the command **`iwconfig`**, that provides you with connection and link information specific to the WLAN interface. Take note for future reference (e.g., a print screen or don't close the . Pay close attention to the fields:

`Retry short long limit:7`

If the value is not 7, insert the commands of Point 4.3 of this guide.

2.2. Click on the *Network Manager* icon on the top right corner of the desktop and select the **WIFII004** network. The password is also **WIFII004**. Now you are also connected to a Wireless Local Area Network (WLAN).



2.3. Re-run the three commands below. Make sure a few seconds have passed since the connection to the **WIFII004** network was established. Compare the results with what you previously observed.

- `ifconfig`
- `iwconfig`
- `route -n`

2.4. Following the last command, you will notice that your **device is connected to two networks** – Ethernet and WLAN – and correspondingly there are two gateways. In normal conditions only one network/gateway is available, either Ethernet or WLAN; however, due to our particular setup, in which we are accessing the **RPI Master** through VNC over the Ethernet connection, we need to keep both WLAN and Ethernet connections active. Due to the conflict of having two active networks/gateways, we will add a new route to the IP addressing table to force all traffic to the **RPI Slave** to be routed through the WLAN interface.

Run the following command, where the **`WLAN Gateway IP is 10.227.1.1`**:

`sudo ip route add <Slave Ethernet IP>/32 via <WLAN Gateway IP> dev wlan0`

Re-run the command **`route -n`** . Identify the new line and interpret it considering the explanation in previous points.

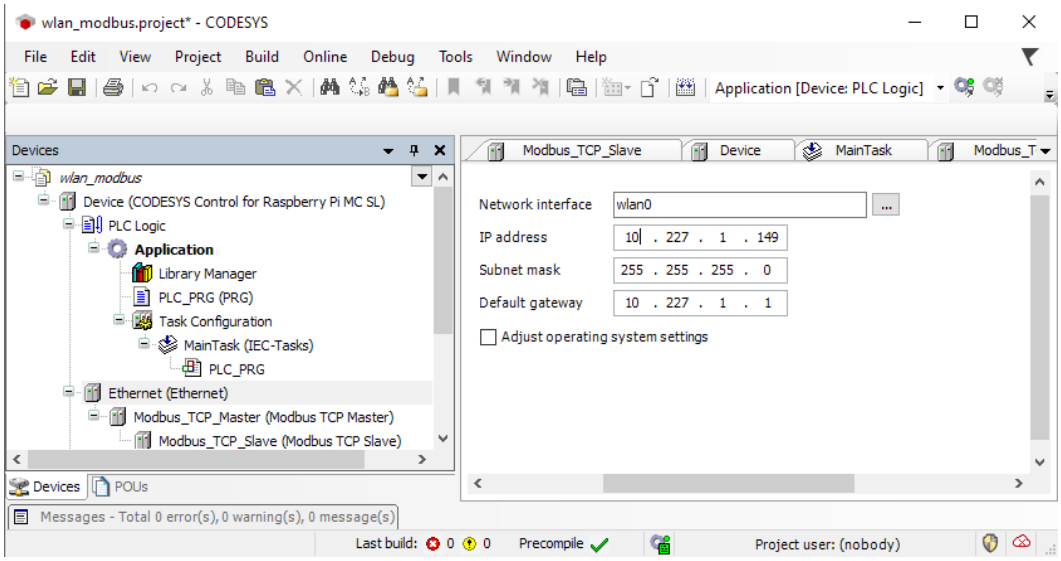
2.5. Re-run the **`ping 192.168.113.YY`** and **`traceroute 192.168.113.YY`** commands. To understand the results, remember that the **WIFII004** WLAN access point (AP) is not connected directly to the cabled Ethernet LAN of the laboratory, as in the case of the Ethernet interface.

3. Modbus Network Reaction Time (TRR) over WLAN and Ethernet

We will now compare the Network Reaction Time of Modbus traffic between the **RPI Master** and the **RPI Slave**.

3.1. Open *Codesys*. Setup the Modbus communication between the **RPI Master** and **RPI Slave**, following the instructions of script *"Introduction to Codesys and Modbus/TCP configuration"* (from Assignment 2). In summary:

1. Create a project targeting a Raspberry Pi. Add an Ethernet interface, and the corresponding Modbus Master and Slave.
2. Select the network configuration of the WLAN interface (see image below; click on the "…" button for quick configuration). As for the Modbus slave, set its IP address to be that of the Ethernet LAN network.



3. Create two channels:
   - *Read Input Registers (Function Code 4)*, with an offset of 0, and length 1.
   - *Write Multiple Registers (Function Code 16)*, with an offset of 0, and length 1.
4. Create a simple program that copies whatever it is present at the inputs to the outputs (e.g., **input:=output**). Set the Main Task as *Freewheeling*.

3.2. Set **TCR=5ms** Open Wireshark. Select the WLAN interface and start capture. Confirm that the traffic is now taking place over the WLAN interface. Stop the capture.

3.3. Carry out TRR over Modbus measurements for **TCR=5ms**, **TCR=20ms**, **TCR=40ms**, and **TCR=60ms** by activating the Arduino program over *CuteCom*. Take 100 samples and plot the corresponding boxplots. Compare these with the boxplots you produced when using Ethernet.

If you wish to re-do the Ethernet measurements, you just need to:
1. In the Desktop's shortcut to the *Network Manager*, disconnect from the wireless network;
2. In *Codesys*, select the Modbus Master to use the network configuration the Ethernet interface.

When connecting back to the **WIFII004** wireless network, do not forget to add back the route:

```
sudo ip route add <Slave Ethernet IP>/32 via <WLAN Gateway IP> dev wlan0
```

3.4. Process the results. What are the main differences you observe between the WLAN and the Ethernet Modbus TRR measurements?

4. Impact of MAC-level Retransmissions in Modbus TRR

While still connected to the WLAN, let us change some of the MAC-level retransmissions and observe its impact in the network reaction time. By default, the WLAN network interface card (NIC) will attempt to re-transmit a packet up to 7 times. We will observe what happens when we lower this limit to 1.

4.1 To this end, execute the following commands:

```
sudo iwconfig wlan0 retry short 1
sudo iwconfig wlan0 retry long 1
```

The commands are self-explanatory. **`short`** and **`long`** stand for two classes of packet size.

4.2 Re-run the Modbus experiments. What do you observe? Consider also whether you are performing the measurement at a time of high wireless traffic (weekday) or low wireless traffic (weekend).

4.3 Revert the operation with:

```
sudo iwconfig wlan0 retry short 7
sudo iwconfig wlan0 retry long 7
```

THE END