Test sending data between CubeSat

To verify the transfer time:

- Send data, the weight of which is previously known between the Raspberry Pi of two CubeSat.
- Measure the time required for this transmission.
- Then divide the weight of the transmitted data by the time, which will give me the actual throughput.

To carry out this test, I set up a communication between the Raspberry Pi and the CubeSat.
This communication will be done with the Wi-Fi protocol, because in my case the data transmission is between objects that are not physically connected. This allows me to establish a wireless link between the devices and the access point.
I specify that wifi is a communication protocol that is usually used for a wireless internet connection, but it can also be used to send information without going through the internet protocol.
And more precisely I use the IEEE 802.11n standard, more commonly known as Wi-Fi 4.

The test steps are as follows:

- Step 1: Setting up the AP/STA mode

I chose the AP/STA mode (it is a client access point) because its use has several advantages in a CubeSat network; namely:

- It simplifies the configuration of the network. Indeed, CubeSat can create their own Wi-Fi access point with this mode, which allows them to connect to each other without the need for additional equipment.
- Then, it allows to share the Internet connection between the CubeSats. This can be useful for applications such as data dissemination or telemetry.

1) RaspAP Installation

RaspAP software manages a DHCP service that allows clients who connect to the WiFi access point to obtain an IP configuration.

To set up this access point, I had to install the RaspAP software on the map using the following command: "curl -sL https://install.raspap.com | bash".

2) Connection to the Access Point:

Once the installation was complete, I could see that the access point is displayed on my computer.

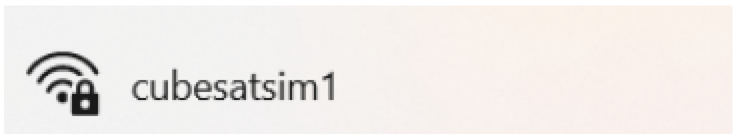


Fig. 1 - Display of the availability of the cubesatsim1 Wi-Fi access point on the computer

I want to clarify that from my computer, I do not connect to the Internet in this context. I simply check if it is possible to connect to the Raspberry Pi Zero from my computer. Because with the screen of my computer it makes the configuration more visible and therefore easier to check that there is a Wi-Fi access point available.
It also allows me to check the operation of the access point.
And once this Wi-Fi connection is established, other Raspberry Pi can connect to this access point.

In summary, my computer connects, thanks to Wi-Fi protocol, to an access point. In my case, my Raspberry Pi Zero can communicate in wifi so the computer can detect it and normally connect to the Raspberry PI Zero.

However, when I tried to connect my computer to the access point “cubesatsim1” with the CubeSats login password, the network and internet settings informed me that there was a connection but no internet access.

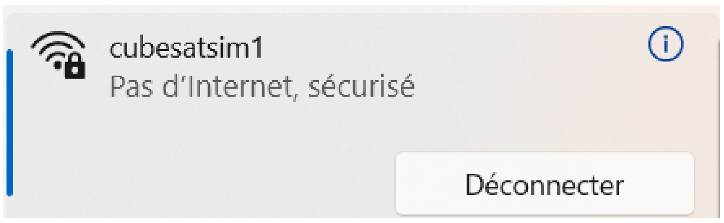

cubesatsim1
Pas d’Internet, sécurisé

Fig. 2 - Display of the unavailability of the cubesatsim1 wifi access point on the computer

3) Approach for WiFi client verification and attempt to restore connection:

- To check that the wifi client was activated, I went to the RaspAP application using the link “http://CubeSatsim1.local.” Then, I entered the login and password necessary to identify myself.
- I checked if the Wi-Fi client was showing an active connection, but it wasn’t.
  BridgedEnable = 0
  WifiManaged = wlan0

4) Fix Internet Access Problem:
1ère tentative :

- Always with RaspAP application connected to «http://CubeSatsim1.local»

- I changed the settings in the configuration file with the command: sudo nano/etc/raspap/hostapd.ini
- I changed the LogEnable and WifiAPEnable parameters as follows: LogEnable = 1

  WifiAPEnable = 1

Unfortunately, these changes did not resolve the connection issue.

2nd attempt:
I searched the web, and one site reported that Internet access was no longer working by default after installing the RaspAp software.
To make the connection, here's what I did:

- I modified the file "sysctl.conf" by uncoupling the linee : #net.ipv4.ip_forward=1
- I added the following line in the file "rc.local" before the part "exit 0" : iptables -t nat -A POSTROUTING -j MASQUERADE

Unfortunately, this did not solve the connection problem.

3rd attempt:
I finally realized that the problem could come from my computer.
So I tested the connection on another interface, and it worked.
It was the firewall of my computer that prevented the Wi-Fi connection to the cubesatsim1 access point.

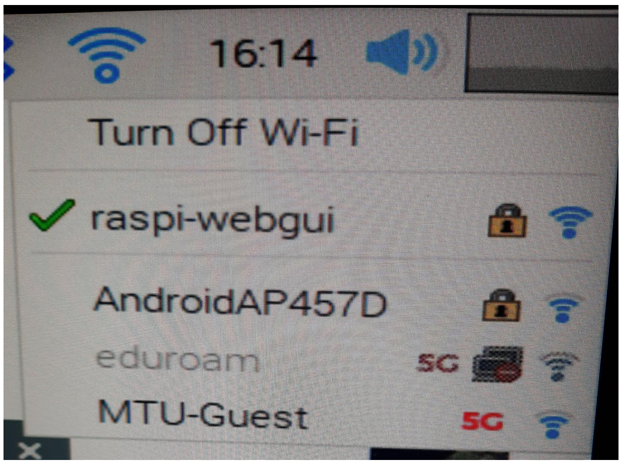


Fig. 3 - Display of network connection success on another interface

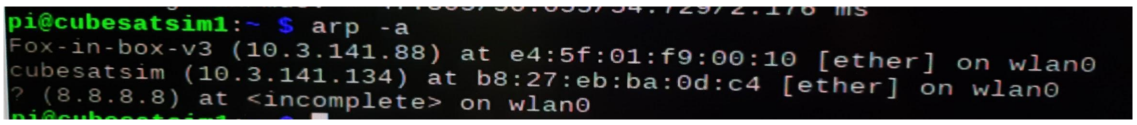
- Step 2: Set up a wifi connection on a second Raspberry Pi zero

So I connected another Raspberry Pi Zero to the access point to build a network.

Process I followed to set up a Wi-Fi connection between the 2 Raspberry Pi Zero:

- I entered in the command terminal that appears:
  “sudo nano /etc/wpa_supplicant/wpa_supplicant.conf” pour accéder au fichier wpa_supplicant.conf.
- I added to the file the wireless network settings that were created with the access point:
  ```
  network= {
                  ssid="raspi-webgui"
                  psk="ChangeMe"
  }
  ```
- Then I restarted the wpa_supplicant service with the command:
  sudo systemctl restart wpa_supplicant

To check that the connection is established, I looked at which interface is connected on my access point. And I can see that the Raspberry Pi Zero which is called «cubesatsim» is well connected on the access point. I specify not to confuse the two Raspberry Pi Zero that the access point is called “cubesatsim1”.

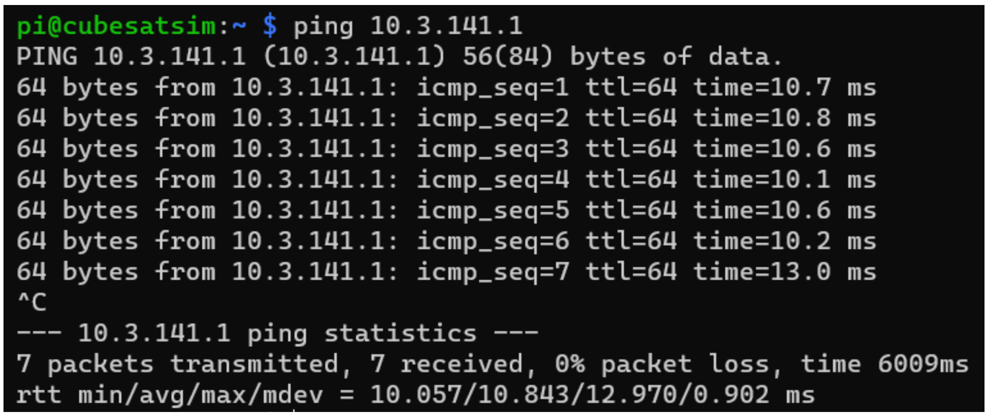


```
pi@cubesatsim1:~ $ arp -a
Fox-in-box-v3 (10.3.141.88) at e4:5f:01:f9:00:10 [ether] on wlan0
cubesatsim (10.3.141.134) at b8:27:eb:ba:0d:c4 [ether] on wlan0
? (8.8.8.8) at <incomplete> on wlan0
```

Fig. 4 - Screenshot of the control terminal that lists all the interfaces connected to the access point

- Step 3: Test of data transfer between two Raspberry Pi Zero in network

Test 1: Data transfer from Raspberry Pi Zero to the access point:
I launch the “ping” command from the Raspberry Pi Zero named "cubesatsim" to the access point named “cubesatsim1”.



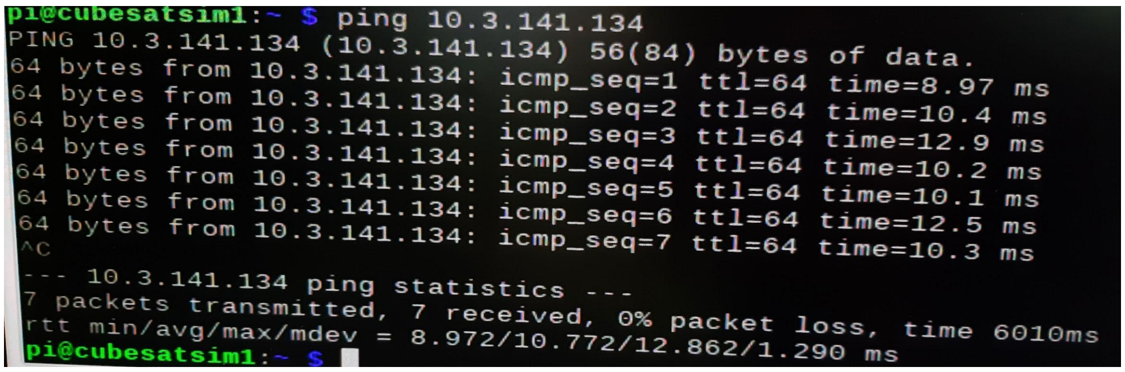
```
pi@cubesatsim:~ $ ping 10.3.141.1
PING 10.3.141.1 (10.3.141.1) 56(84) bytes of data.
64 bytes from 10.3.141.1: icmp_seq=1 ttl=64 time=10.7 ms
64 bytes from 10.3.141.1: icmp_seq=2 ttl=64 time=10.8 ms
64 bytes from 10.3.141.1: icmp_seq=3 ttl=64 time=10.6 ms
64 bytes from 10.3.141.1: icmp_seq=4 ttl=64 time=10.1 ms
64 bytes from 10.3.141.1: icmp_seq=5 ttl=64 time=10.6 ms
64 bytes from 10.3.141.1: icmp_seq=6 ttl=64 time=10.2 ms
64 bytes from 10.3.141.1: icmp_seq=7 ttl=64 time=13.0 ms
^C
--- 10.3.141.1 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6009ms
rtt min/avg/max/mdev = 10.057/10.843/12.970/0.902 ms
```

Fig. 5 - Screenshot of the control terminal confirming the data transfer between the Raspberry Pi Zero and the access point

I see on the screen the confirmation of the transmission of the data packets and their reception.

Test 2: Data transfer from the access point to the Raspberry Pi Zero:

Here I test the data transmission in the other direction. To do this, I launch the “ping” command from the access point named “cubesatsim1” to the Raspberry Pi Zero named "cubesatsim”.



```
pi@cubesatsim1:~ $ ping 10.3.141.134
PING 10.3.141.134 (10.3.141.134) 56(84) bytes of data.
64 bytes from 10.3.141.134: icmp_seq=1 ttl=64 time=8.97 ms
64 bytes from 10.3.141.134: icmp_seq=2 ttl=64 time=10.4 ms
64 bytes from 10.3.141.134: icmp_seq=3 ttl=64 time=12.9 ms
64 bytes from 10.3.141.134: icmp_seq=4 ttl=64 time=10.2 ms
64 bytes from 10.3.141.134: icmp_seq=5 ttl=64 time=10.1 ms
64 bytes from 10.3.141.134: icmp_seq=6 ttl=64 time=12.5 ms
64 bytes from 10.3.141.134: icmp_seq=7 ttl=64 time=10.3 ms
^C
--- 10.3.141.134 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6010ms
rtt min/avg/max/mdev = 8.972/10.772/12.862/1.290 ms
pi@cubesatsim1:~ $
```

Fig. 6 - Screenshot of the control terminal confirming the data transfer in the opposite direction, from the access point to the Raspberry Pi Zero

And I see on the screen the confirmation of the transmission of the data packets and their reception.

Test 3: Transfer of a 23 Mbit packet between two Raspberry Pi Zero:

The two previous tests confirmed that I could transfer data between two Raspberry Pi Zero.

I now want to check if it is possible to send a 23 Mbits packet (=2875000 bytes) between two Raspberry Pi zero.
However, I found that the transmission of a data packet was limited in weight.
Thus, the weight of 23 Mbits of the package I had considered for my calculations exceeds the sending capacity of a Raspberry Pi Zero.
This complicates my research, because the calculations I have developed do not correspond to the transmission capabilities of CubeSats.

This limitation can be due to various factors, such as network bandwidth or the hardware capabilities of the Raspberry Pi Zero.

The raspberry Pi can transmit in one go 65536 bytes per milliseconds.
Therefore, to overcome this hazard I must transmit my 23 Mbit packet (2,875,000 bytes) in several pieces of 65,536 bytes.

I will try this approach if the time allotted to the internship is sufficient.
If I don’t have the time, this approach can be exploited by the team that will succeed me.