Universidade do Porto

FEUP Faculdade de Engenharia

Integrated Master's Degree in Electrical and Computer Engineering

Telecommunications, Electronic and Computers

Networks and Services Communications

Lab5 - FAP Controller and AP

Mobile Communications Curricular Unit
Project Supervised by Eduardo Almeida and
Professor Manuel Alberto Pereira Ricardo

Anabela Machado Reigoto, up201405662
Baltasar de Vasconcelos Dias Aroso, up201404125
4MIEEC_T_RC

May, 2018

Contents

List of Figures

1 Introduction

This project belongs to the Curricular Unit of Mobile Communications and it was divided into three main parts: FAP Android App, FAP Management Protocol and FAP Controller + AP. This report is mainly about the last one, where we've needed to implement the FAP Controller in C and to configure the Access Point.

2 Configuration of the FAP Controller in C

The main purpose of this code implementation is to control the Access Point position according to the users position. So we receive some important informations about the user positions and the drone position by the Server that belongs to the second part of the project mentioned above.

2.1 Algorithm Explanation

This FAP Controller program is divided into three main parts:

- ***initializeFapController()***:
  This function verifies the correct initialization of the Management Protocol before continuing the program and then starts the alarm. This alarm is controlled by the *ALARMhandler* that calls the *updateFapCoordinates()* functions in every 10 seconds (value defined in "FAP_POSITION_UPDATE_PERIOD_SECONDS" in the beginning of the code).

- ***updateFapCoordinates()***:
  This function calls three functions from the Management Protocol to *getAllUsersGpsNedCoordinate()*, *getFapGpsNedCoordinates()* and then to find the ***averageUserCoords()*** (a FAP Controller function) to find the new best position for the Acess Point by keeping the 'z' coordinate unchanged and calculating the average of the 'x' and 'y' coordinates of all users, and finally *moveFapToGpsNedCoordinates()*, establishing that new average of 'x's and 'y's as the new FAP coordinates, in order to keep giving a strong Wi-Fi signal to the users.

- ***terminateFapController()***:
  This function verified the correct closure of the Management Protocol before continuing the program and then stops the alarm printing a message that indicates the end of the FAP Controller.

3 Configuration of the Access Point in the Raspberry Pi

In this section we will discuss how the configuration of the Access Point in the Raspberry Pi was made in order to give Wi-Fi to the users.

3.1 Configuration of the SD Card

- Insert the mini SD Card into the adapter and put it in the computer. Then we just need to configure the file "/etc/network/interface" to create a static IP address to the eth0 as showed in the figure 1.

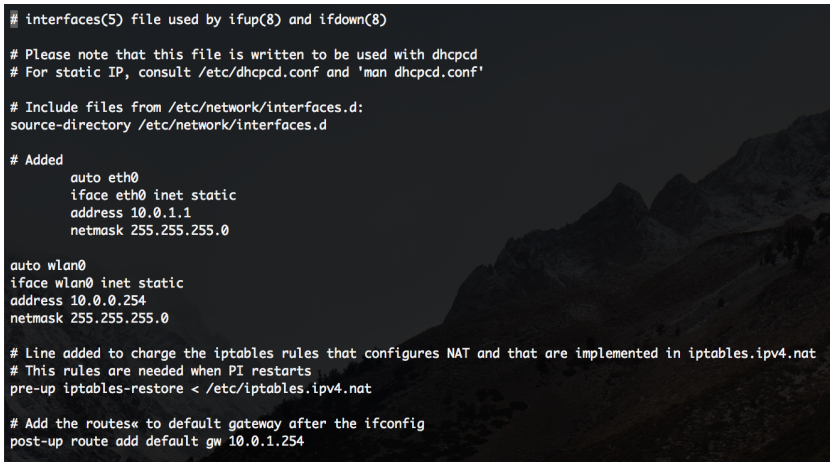

Figure 1: Configuration of the *interfaces* file

- After this configuration we insert the mini SD card into the raspberry pi. It is ready for the *ssh* connection with a computer that is needed for the visual interface of the raspberry pi's terminal and, therefore, for his configurations.
  So we needed to implement a *ssh* tunnel between the raspberry pi and the computer using an ethernet cable as explained in the next section.

3.2 Connection between Raspberry Pi and a Personal Computer

- First we set the cables correctly connected between the raspberry pi and the computer as shown in the figure 2.

Figure 2: Hardware cables display for the connection between the raspberry pi and the computer

- Then we configured the computer to be able to connect throught the ethernet cable.

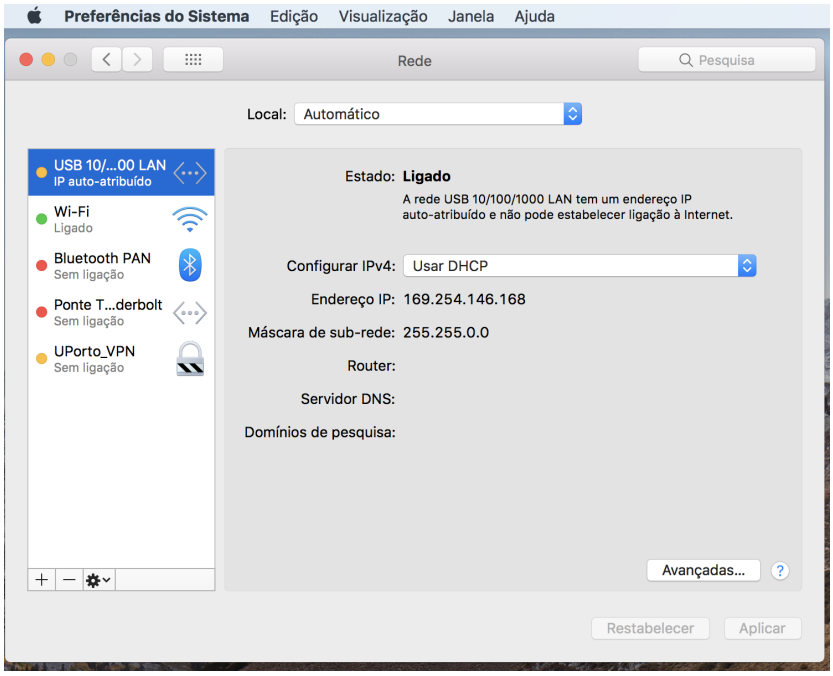

Figure 3: Macintosh network preferences configurations in the OS Sierra

- Then we tested the connection by the Ethernet cable and the correspondent *ssh* connection throught the port 22 as we can see in the figure 4.

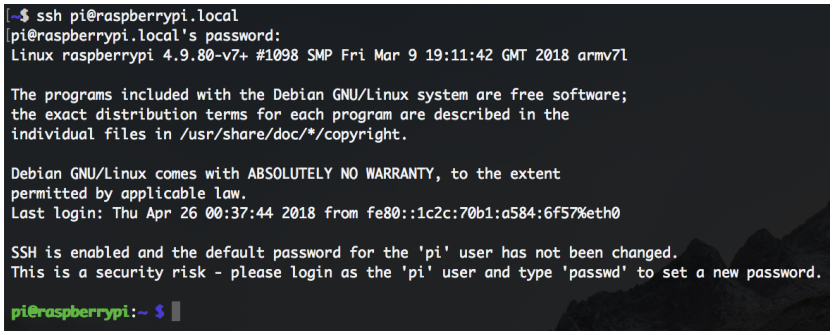

Figure 4: Connectivity between the raspberry pi and the macintosh

Now that we are already "inside" the raspberry pi, we can configure the SD card that will contain the files needed for this project.

3.3 Configuration of the hostapd

- Create the file "hostapd.conf" in the directory "/etc/hostapd" and add the configurations displayed in the figure 5.

```
# This two commands are the default interface and driver for an hostapd configuration
interface=wlan0
#driver=n180211

# This is the name of the wireless network that appears to the users
ssid=USER-AWARE-FAP

# Module bitfield (ORed bitfield of modules that will be logged; -1 = all
# modules):
# bit 0 (1) = IEEE 802.11
# bit 1 (2) = IEEE 802.1X
# bit 2 (4) = RADIUS
# bit 3 (8) = WPA
# bit 4 (16) = driver interface
# bit 5 (32) = IAPP
# bit 6 (64) = MLME
#
# Levels (minimum value for logged events):
#  0 = verbose debugging
#  1 = debugging
#  2 = informational messages
#  3 = notification
#  4 = warning
#
### So as we will use some of the modules we select -1 to logged it all in both output methods
logger_syslog=-1
logger_syslog_level=2
logger_stdout=-1
logger_stdout_level=2

# g = IEEE 802.11g (2.4GHz), if we've wanted to use 5GHz this command should be equal to "a" from IEEE 802.11a
hw_mode=g
```

```
# Country code (ISO/IEC 3166-1). Used to set regulatory domain.
country_code=PT

# Channel number (IEEE 802.11)
# If CONFIG_ACS build option is enabled, the channel can be selected
# automatically at run time by setting channel=acs_survey or channel=0, both of
# which will enable the ACS (Automatic Channel Selection) survey based algorithm.
channel=10

# Maximum number of stations allowed in station table. New stations will be
# rejected after the station table is full. IEEE 802.11 has a limit of 2007
# different association IDs, so this number should not be larger than that.
# (default: 2007) but in this project we will se it to 10 users
max_num_sta=10

# Open system authentication should be used with IEEE 802.1X.
# Bit fields of allowed authentication algorithms:
# bit 0 = Open System Authentication
# bit 1 = Shared Key Authentication (requires WEP)
### As we use WPA and not WEP we just set the bit 0
auth_algs=1

#ignore_broadcast_ssid=0

# For WPA-PSK, either
# wpa_psk or wpa_passphrase must be set and wpa_key_mgmt must include WPA-PSK.
# This field is a bit field that can be used to enable WPA (IEEE 802.11i/D3.0)
# and/or WPA2 (full IEEE 802.11i/RSN):
# bit0 = WPA
# bit1 = IEEE 802.11i/RSN (WPA2) (dot11RSNAEnabled)
wpa=2
wpa_passphrase=UserAwareFAP
wpa_key_mgmt=WPA-PSK
```

Figure 5: Configuration of the *hostapd* file

- Restart the hostapd service using:

```
pi@raspberrypi:~ $ sudo service hostapd restart
```

- Test that configurations using the following command and let it be running while observing the devices that connect to the SSID "USER-AWARE-FAP":

```
pi@raspberrypi:~ $ sudo hostapd /etc/hostapd/hostapd.conf
```

3.4 Configuration of the DHCP

- Recreate the file "dhcpd.conf" in the directory "/etc/dhcp" and add the configurations displayed in the figure 6.

```
# If this DHCP server is the official DHCP server for the local
# network, the authoritative directive should be uncommented.

authoritative;

default-lease-time 600;
max-lease-time 7200;

subnet 10.0.0.0 netmask 255.255.255.0 {
        range 10.0.0.1 10.0.0.10;
        option broadcast-address 10.0.0.255;
        option routers 10.0.0.254;
        option domain-name "local";
        option domain-name-servers 10.0.1.254;
}
```

Figure 6: Configuration of the *dhcpd* file

- Add the DNS server (*nameserver 10.0.1.254*) in the file "/etc/resolv.conf".
- Test that configurations using the following command:

```
pi@raspberrypi:~ $ sudo dhcpd -t -cf /etc/dhcpd/dhcpd.conf
```

- Uncomment the interface command and equals it to the interface already configurated (wlan0) in the file "/etc/default/isc-dhcp-server" as shown in the figure 7.

```
# Defaults for isc-dhcp-server (sourced by /etc/init.d/isc-dhcp-server)

# Path to dhcpd's config file (default: /etc/dhcp/dhcpd.conf).
#DHCPDv4_CONF=/etc/dhcp/dhcpd.conf
#DHCPDv6_CONF=/etc/dhcp/dhcpd6.conf

# Path to dhcpd's PID file (default: /var/run/dhcpd.pid).
#DHCPDv4_PID=/var/run/dhcpd.pid
#DHCPDv6_PID=/var/run/dhcpd6.pid

# Additional options to start dhcpd with.
#       Don't use options -cf or -pf here; use DHCPD_CONF/ DHCPD_PID instead
#OPTIONS=""

# On what interfaces should the DHCP server (dhcpd) serve DHCP requests?
#       Separate multiple interfaces with spaces, e.g. "eth0 eth1".
INTERFACESv4="wlan0"
#INTERFACESv6=""
```

Figure 7: Configuration of the *isc-dhcp-server* file

- Restart the isc-dhcp-server service using:

```
pi@raspberrypi:~ $ sudo service isc-dhcp-server restart
```

3.5 Configuration of the NAT and the Default Gateway

- Uncomment the following line in the "/etc/sysctl.conf" file: *net.ipv4.ip_forward=1.*
- Activate the forwarding of the IP packets with this command:

```
pi@raspberrypi:~$ sudo sh -c "echo 1 > /proc/sys/net/ipv4/ip_forward"
```

- Turn the Raspberry Pi into a router using this iptable rules.

```
pi@raspberrypi:~ $ sudo iptables -t nat -A POSTROUTING -o eth0 -j
    MASQUERADE
pi@raspberrypi:~ $ sudo iptables -A FORWARD -i eth0 -o wlan0 -m state
     --state RELATED,ESTABLISHED -j ACCEPT
pi@raspberrypi:~ $ sudo iptables -A FORWARD -i wlan0 -o eth0 -j
    ACCEPT
```

- Create the file "/etc/iptables.ipv4.nat" and add this configurations on it.

```
pi@raspberrypi:~ $ sudo sh -c "iptables-save > /etc/iptables.ipv4.nat
```

- Add *"pre-up iptables-restore < /etc/iptables.ipv4.nat"* in the end of the "/etc/network/interfaces" to charge the configurations above explained as soon as the Raspberry Pi restarts. (already shown in the figure 1).

4 References

- Dexter Industries. (2018). Connect to your Raspberry Pi from a Mac. [online] Available at: https://www.dexterindustries.com/getting-started/using-the-pi/connect-to-your-raspberry-pi-from-a-mac/ [Accessed 23 April 2018].
- W1.fi. (2018). [online] Available at: https://w1.fi/cgit/hostap/plain/hostapd/hostapd.conf [Accessed 23 April 2018].
- Pplware. (2018). Como transformar o Raspberry Pi num Access Point Wireless - Pplware. [online] Available at: https://pplware.sapo.pt/linux/como-transformar-o-raspberry-pi-num-access-point-wireless/ [Accessed 23 April 2018].
- Raspberrypi.org. (2018). Lesson 3: Dynamic Host Configuration Protocol (DHCP) - Lesson Plan - Networking Lessons | Raspberry Pi Learning Resources. [online] Available at: https://www.raspberrypi.org/learning/networking-lessons/lesson-3/plan/ [Accessed 23 April 2018].
- Braindisconnect.com. (2018). Linux DHCP Server - Braindisconnect. [online] Available at: http://www.braindisconnect.com/wiki/index.php?title=Linux_DHCP_Server [Accessed 23 April 2018].