Rapport de LO53

Simon Magnin-Feysot (GI04 - LEIM)

Chapitre 1

Access point

Access points need to be turn into the monitor mode in order to sniff every packet available around it. We can do it using the web interface of the access point or we write a script shell (mon.sh) to do it in a terminal through ssh.

The access point application was programmed in C. The project impose this language but anyway, the AP system is so minimalist the C is the best choice. We used the base code given on moodle. This code already get wifi packets and analyse them in order to get the MAC address and the RSSI. We use also the given file `rssi_list.h` given and we write the corresponding `rssi_list.c`. We made some modifications, We don't store the rssi in mWatt but in db, the conversion is done by the server. We assume, the server is here to do the most bigger part of the computation, and the AP and mobile just send minimal informations.

To comminucate with the server, we just use TCP socket and not HTTP (wich requires the additional libcurl, and doesn't provide anything essential for our program), so the use of TCP protocol was minimalist but fits perfectly to our goals. But we needed to define a little bit more than just the use of TCP. How they communicate ?

— AP is launched and wait fort TCP connection from the server
— Server opens a connection and send a MAC address to the AP
— AP gets the MAC, looking for the MAC in the linked list and return the corresponding RSSI
— The server get the RSSI and close the connection
— AP waits for the next TCP connection

We just have 2 threads running, one for the pcap function analysing every packet in the air, and the second (the main thread) which is handling the TCP connection to communicate with the server. This is also the main thread which is comparing the asking MAC address with all the addresses stored in the linked list. If the asking MAC address is not found, the AP will return 4000 to the server, because 4000 dBm is an impossible value to get with this APs.

The AP has a MIPS processor, which is different than our laptop processor we need to cross compile with the given toolchain. The main problem was for debugging we did'nt have GDB, the only solution was to print some traces.