Wifi hacking with a 4 dollar microcontroller

Márk Szabó, Hacktivity 2016

Quick summary

We are going to use the ESP8266 microcontroller to perform the followings:

1. Set up a fake captive portal
2. Send beacon frames to seemingly spawn endless number of wifis
3. Send deauthentication frames to make the victim drop their connection

Schedule

20 minutes - me talking, 20 minutes - you working

Code & docs: https://github.com/markszabo/Hacktivity2016

Feel free to work on your own pace, ask questions and leave earlier

About me

Márk Szabó

Mechatronics engineering BSc, BME

Security & Privacy Master, EIT Digital:

- 1st year in Trento, Italy
- 2nd year in Budapest, ELTE, ‘Advanced cryptography’ specialization

The ESP8266

- 32-bit RISC CPU: Tensilica Xtensa LX106 (80 MHz)
- 64 KiB of instruction RAM, 96 KiB of data RAM
- External QSPI flash - 512 KiB to 4 MiB
- IEEE 802.11 b/g/n Wi-Fi
- Integrated TR switch, balun, LNA, power amplifier and matching network
- WEP or WPA/WPA2 authentication, or open networks
- 16 GPIO pins
- SPI, I²C,
- I²S interfaces with DMA (sharing pins with GPIO)
- UART on dedicated pins, plus a transmit-only UART can be enabled on GPIO2
- 1 10-bit ADC

Setting up the environment

Arduino IDE - arduino.cc

ESP8266 Arduino Core - https://github.com/esp8266/Arduino

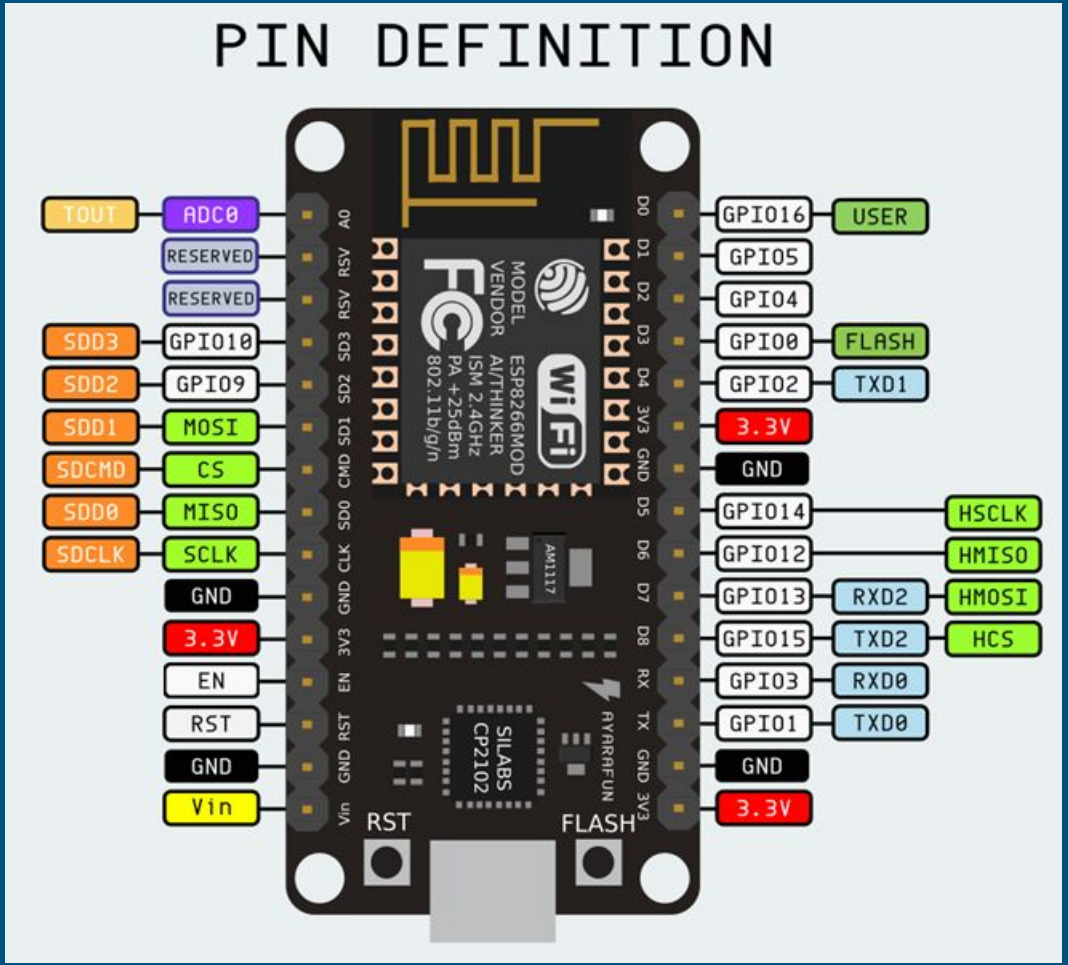
Arduino > Preferences > Additional board manager
http://arduino.esp8266.com/stable/package_esp8266com_index.json

Tools > Board > Board Manager

Tools > Board > Generic ESP8266

Let’s start! - blink

Examples > Basics > Blink



Example 2 - wifi test

Examples > ESP8266 > WifiScan

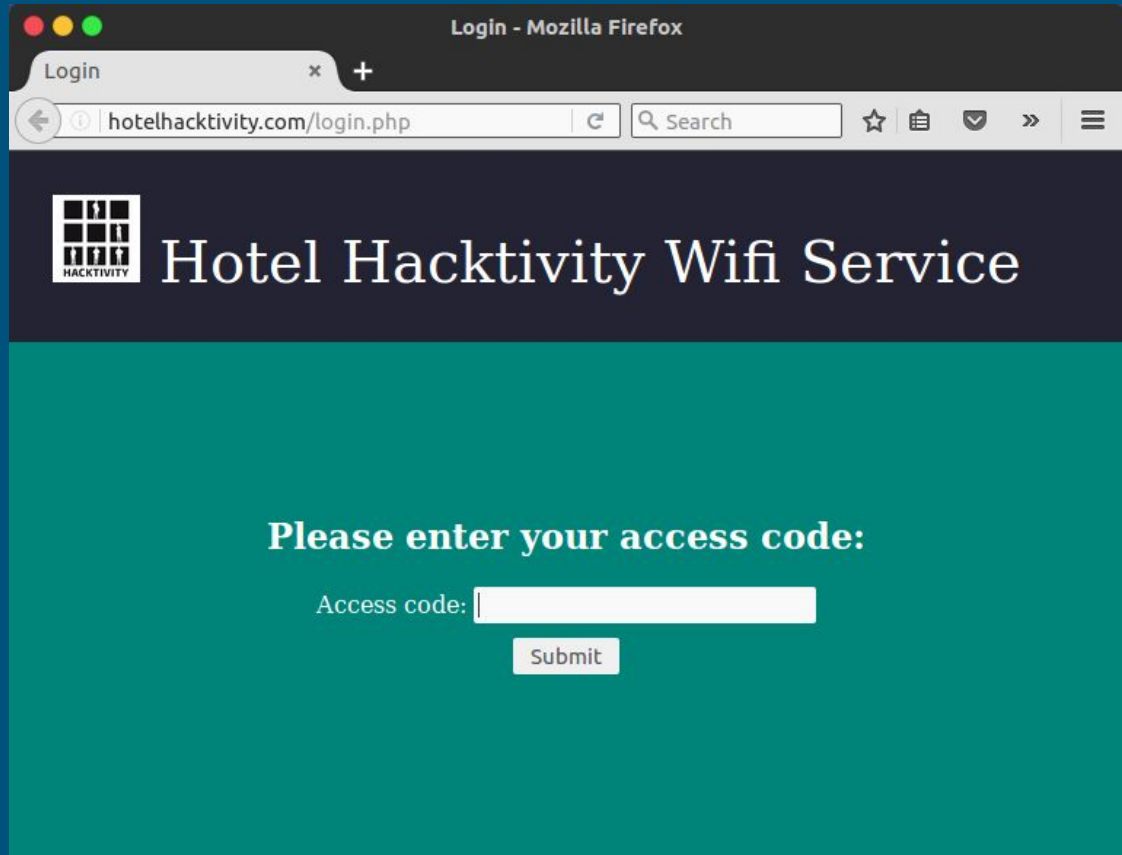
Upload

Tools > Serial Monitor

Baud: 115200

Fake captive portal - the scenario

All requests are redirected to http://hotelhacktivity.com/login.php

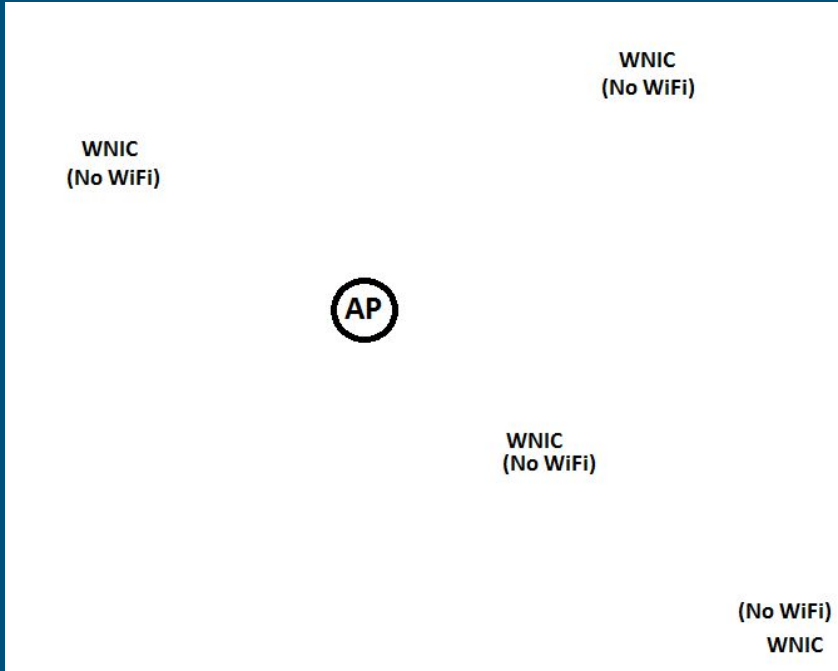


Fake captive portal - the code

- Examples > DNSServer > CaptivePortal
- Flash, observe, check the code
- Change Wifi name
- Add login.php
- Redirect to login
- Solve the image problem (cat HacktivityLogoSmall.jpg | base64)
- Store access code and display error
- Retrieve stored codes

Beacon frames

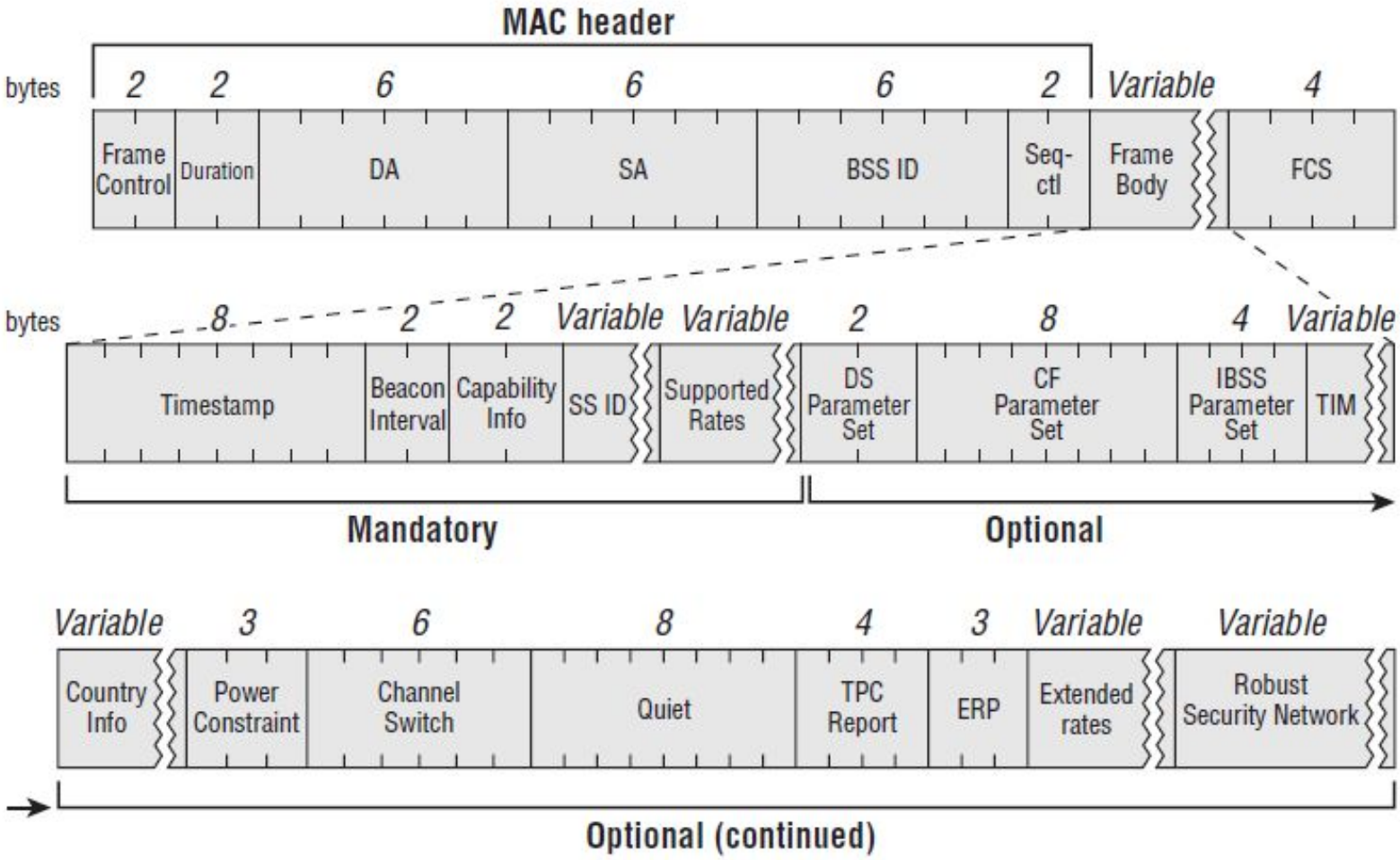
“Beacon frame is one of the management frames in IEEE 802.11 based WLANs. It contains all the information about the network. Beacon frames are transmitted periodically to announce the presence of a wireless LAN.” (Wikipedia)



Beacon frames - basic code

From https://github.com/kripthor/WiFiBeaconJam

FIGURE 4.5 Beacon frame structure



```
uint8_t packet[128] = {
0x80, 0x00, //frame control
0x00, 0x00, //duration
/*4*/ 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, //DA - destination address, broadcast in this case
/*10*/ 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, //SA - source address, will be overwritten later
 /*16*/ 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, //BSSID - same as SA in this case, will be overwritten later
 /*22*/ 0xc0, 0x6c, //Seq-ctl
//Frame body starts here
/*24*/ 0x83, 0x51, 0xf7, 0x8f, 0x0f, 0x00, 0x00, 0x00, //timestamp
 /*32*/ 0x64, 0x00, //beacon interval
 /*34*/ 0x01, 0x04, //capability info
 /* SSID */
0x00, //ID meaning SSID
0x06, //length
0x72, 0x72, 0x72, 0x72, 0x72, 0x72, //SSID name
0x01, //ID meaning Supported rates
0x08, //length
0x82, 0x84, 0x8b, 0x96, 0x24, 0x30, 0x48, 0x6c, //Supported rates
0x03, //ID meaning channel (?)
0x01, //length
0x04 //will be overwritten later with the actual channel
};
```

Deauthentication frames

The function wifi_send_pkt_freedom()

- Was added in sdk 1.3 (but missing from the header files)
- Was added to the header files in sdk 1.5 but limited to beacon frames (frames must start with 0x80, 0x00)

Solution:

Use sdk 1.3 and add it to the header files manually

It’s your turn!

Code & docs: https://github.com/markszabo/Hacktivity2016

Feel free to work on your own pace, ask questions and leave earlier