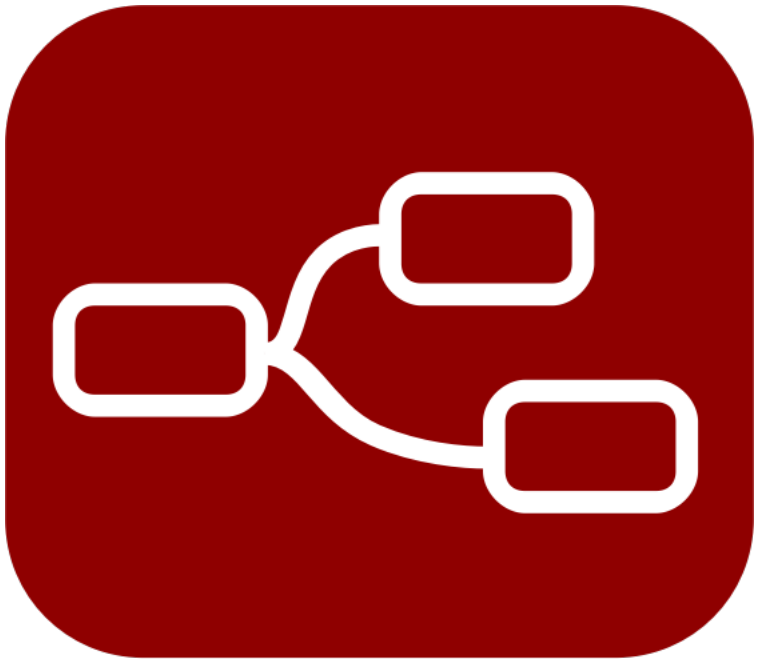


Sistema de alarma IoT con Node-RED

Matías Cabrera

Departamento de Ciencia y Tecnología, Universidad Nacional de Quilmes

El objetivo del proyecto es crear un sistema de alarma IoT (Internet of things) usando NodeRED, que nos pueda avisar cuando una puerta, ventana o porton de nuestra casa se abra.

La idea es que un dispositivo, a través del protocolo mqtt, detecte la apertura del portón y envíe la informacíon al servidor para que este decida cómo notificar la distancia sensada.

1 Setup

1.1 Características de los equipos utilizados

- Notebook: Positivo BGH FX1000
  - Procesador Intel(R) Core(TM) M-5Y10c CPU @ 0.80GHz
  - RAM 4GB
  - Disco SSD 128
- Placa de desarrollo: NodeMCU v3 Lolin
  - Procesador ESP8266 @ 80MHz (3.3V) (ESP-12E)
  - 4MB de memoria FLASH (32 MBit)
  - WiFi 802.11 b/g/n
- Sensor ultrasónico: HC-SR04
  - Rango de medición: 2cm a 400cm
  - Apertura del pulso ultrasónico: 15

1.2 Sistema operativo y software utilizado y/o necesario

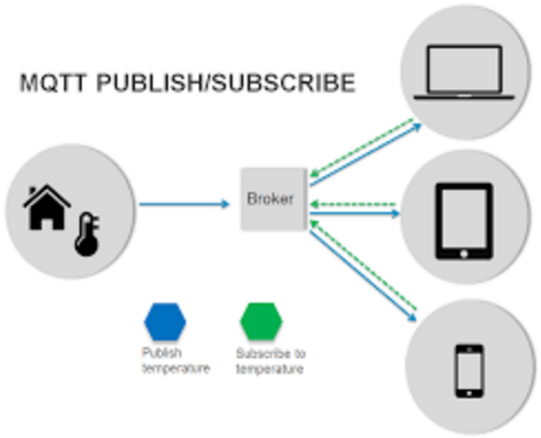
- Ubuntu 18.04.3 LTS
- NodeJs
- Node-RED
- Mosquitto
- Arduino IDE

2 Introducción

2.1 Qué es IoT?

”Internet of things” es un concepto que podría definirse como la agrupación y la interconexión de dispositivos a travs de una red (bien sea privada o Internet), donde todos ellos podrían ser visibles e interaccionar. Respecto a los dispositivos podrían ser cualquiera, desde sensores y dispositivos mecánicos hasta objetos cotidianos. El objetivo es una interacción de máquina a máquina, o lo que se conoce como una interacción M2M (machine to machine).

2.2 Qué es MQTT?



Es un protocolo de comunicación M2M (machine-to-machine) de tipo message queue. Se basa en una conexión abierta que se ”reutiliza” en cada comunicación, a diferencia de una petición HTTP donde cada transmisión se realiza a través de conexión. El funcionamiento del MQTT es un servicio de mensajería push con patrón publicador/suscriptor (pub-sub) donde los clientes se conectan a un servidor (broker) quien se encarga de distribuir los mensajes a quienes corresponda. Para filtrar los mensajes que son enviados a cada cliente los mensajes se clasifican en tópicos organizados jerárquicamente. Un cliente puede publicar un mensaje en un

determinado tópico. Otros clientes pueden suscribirse a este tópico, y el broker les hará llegar los mensajes.

2.3 Qué es Node-RED?

Node-RED es una herramienta de programación visual hecha en JavaScript desarrollada originalmente por IBM, con licencia Apache 1.0. Es un editor de flujo basado en el navegador donde se puede agregar o eliminar nodos y conectarlos entre sí con el fin de hacer que se comuniquen entre ellos. Tiene como principal ventaja que nos permite conectar fácilmente los dispositivos de hardware, APIs y servicios en línea sin conocer en profundidad la tecnología de cada uno.

3 Instalación

3.1 NodeJS

- `sudo apt install nodejs`
- `sudo apt install npm`

3.2 Mosquitto

- `sudo apt install mosquitto`

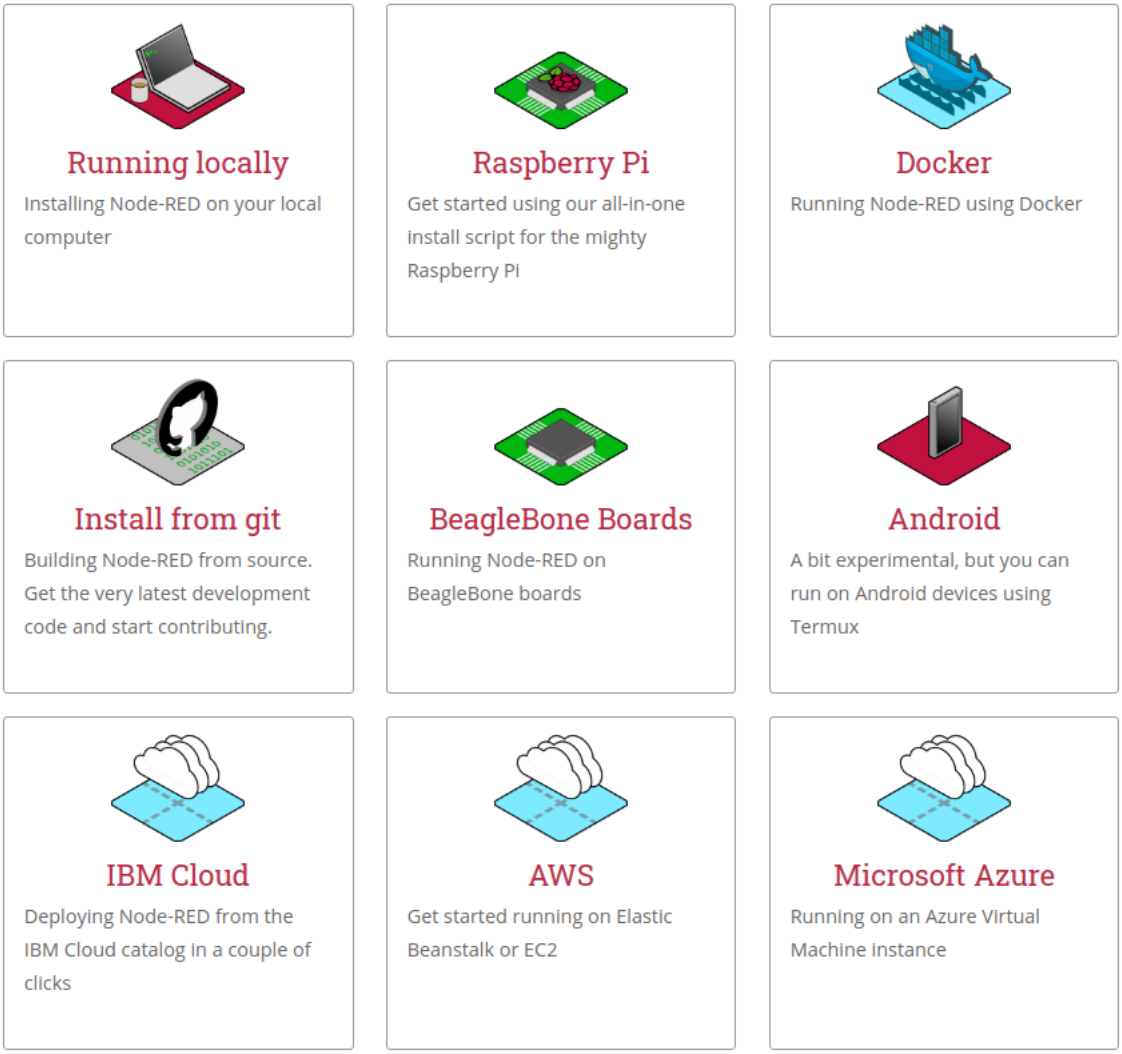
3.3 Arduino IDE

- `sudo apt install arduino`

3.4 Node-RED

Instalación local:

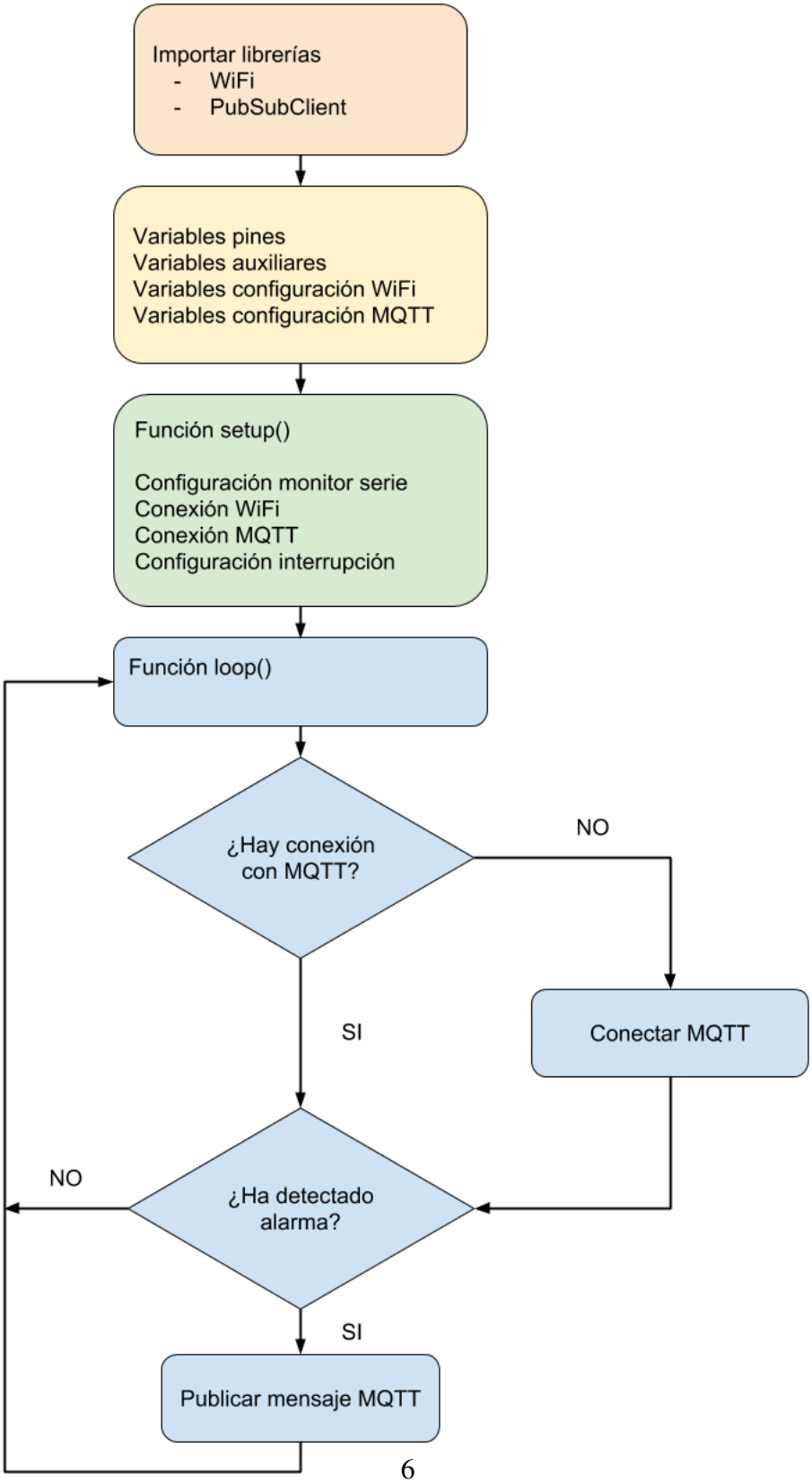
- `sudo npm install -g --unsafe-perm node-red`
- `docker run -it -p 1880:1880 --name mynodered nodered/node-red`
- `sudo snap install node-red`



4 Instructivo paso a paso

4.1 Sistema de alarma con NodeMCU

La lógica es sencilla apartando las características que debe tener con respecto al protocolo MQTT. El programa detecta si la apertura del portón es mayor a 15cm, y de ser así publica un mensaje en el tópico /casa/patio/porton.


Importar librerías
- WiFi
- PubSubClient
Variables pines
Variables auxiliares
Variables configuración WiFi
Variables configuración MQTT
Función setup()
Configuración monitor serie
Conexión WiFi
Conexión MQTT
Configuración interrupción
Función loop()
¿Hay conexión con MQTT?
NO
SI
Conectar MQTT
¿Ha detectado alarma?
NO
SI
Publicar mensaje MQTT

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>

const char* ssid = "red";
const char* password = "clave";
const char* mqtt_server = "broker";

const int trig = 2; //input sensor D4
const int echo = 0; //output sensor D3

WiFiClient espClient;
PubSubClient client(espClient);
long lastMsg = 0;
char msg[100];
int value = 0;

void setup() {

  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
  pinMode(BUILTIN_LED, OUTPUT);
  Serial.begin(115200);

  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
  if (!client.connected()) {
    reconnect();
  }

}

void loop() {

  client.loop();

  const int distancia = measureDistance();

  if (distancia > 15) {
    ++value;
    snprintf (msg, 100, "Porton abierto %d cm", distancia);
```

```
    Serial.print("Mensaje publicado: ");
    Serial.println(msg);
    client.publish("casa/patio/porton", msg);
  }

  delay(5000);

}
```

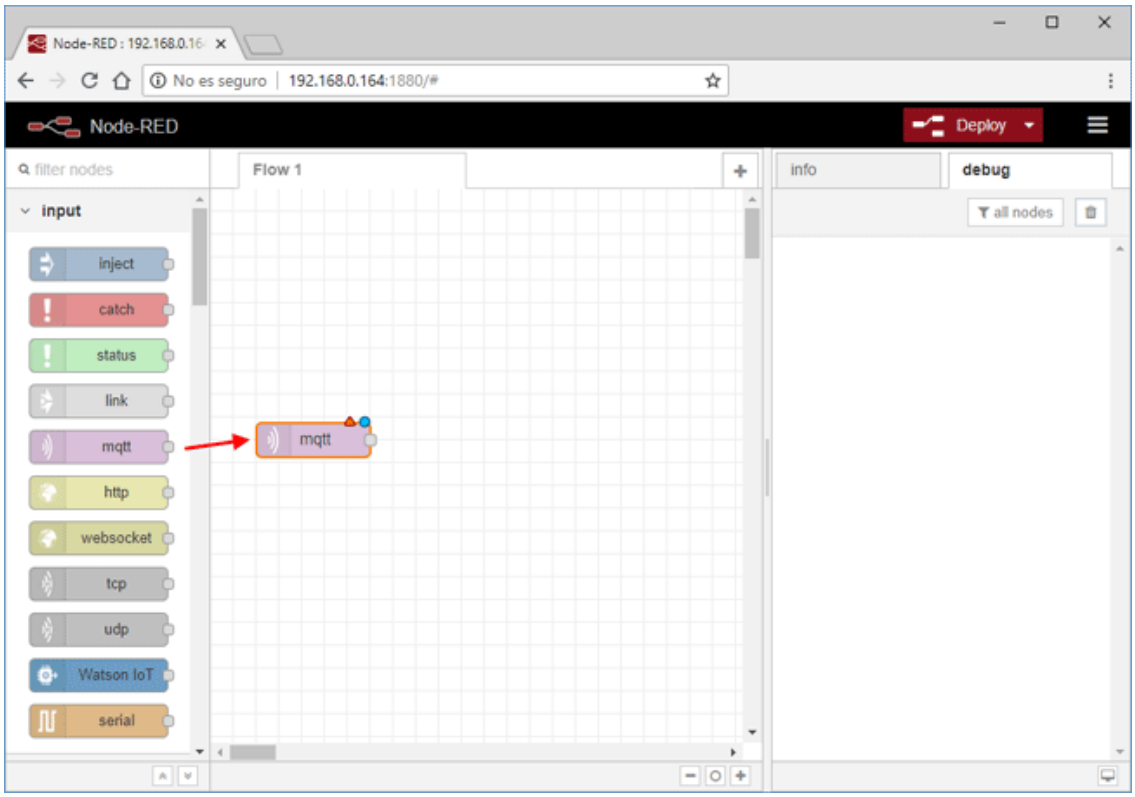
A través de Arduino IDE grabamos el programa en la NodeMCU y ya tenemos nuestro dispositivo conectado a nuestra red IoT para que pueda publicar en el tópico casa/patio/porton.

4.2 Integrar dispositivo con Node-RED

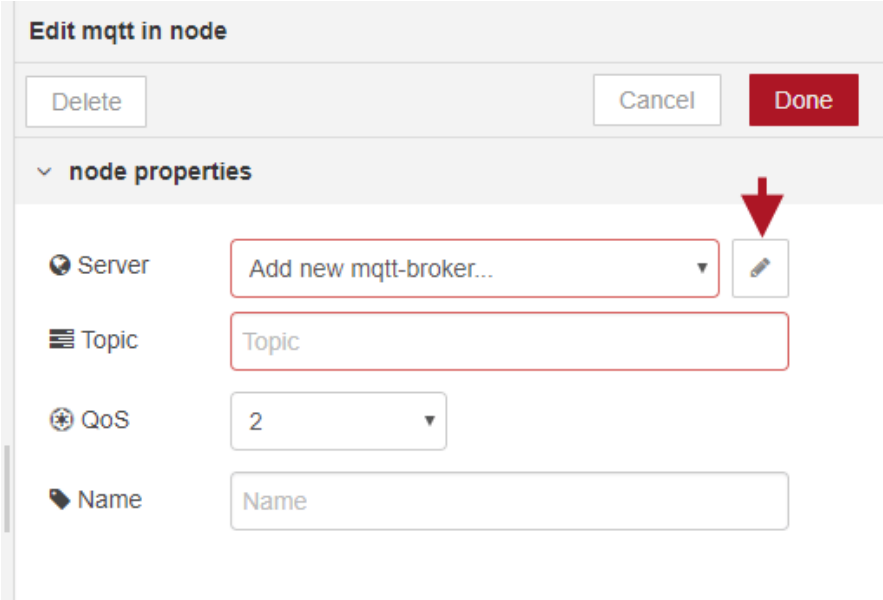
Ejecutar Node-RED e ingresar al navegador a localhost:1880

- `sudo node-red`

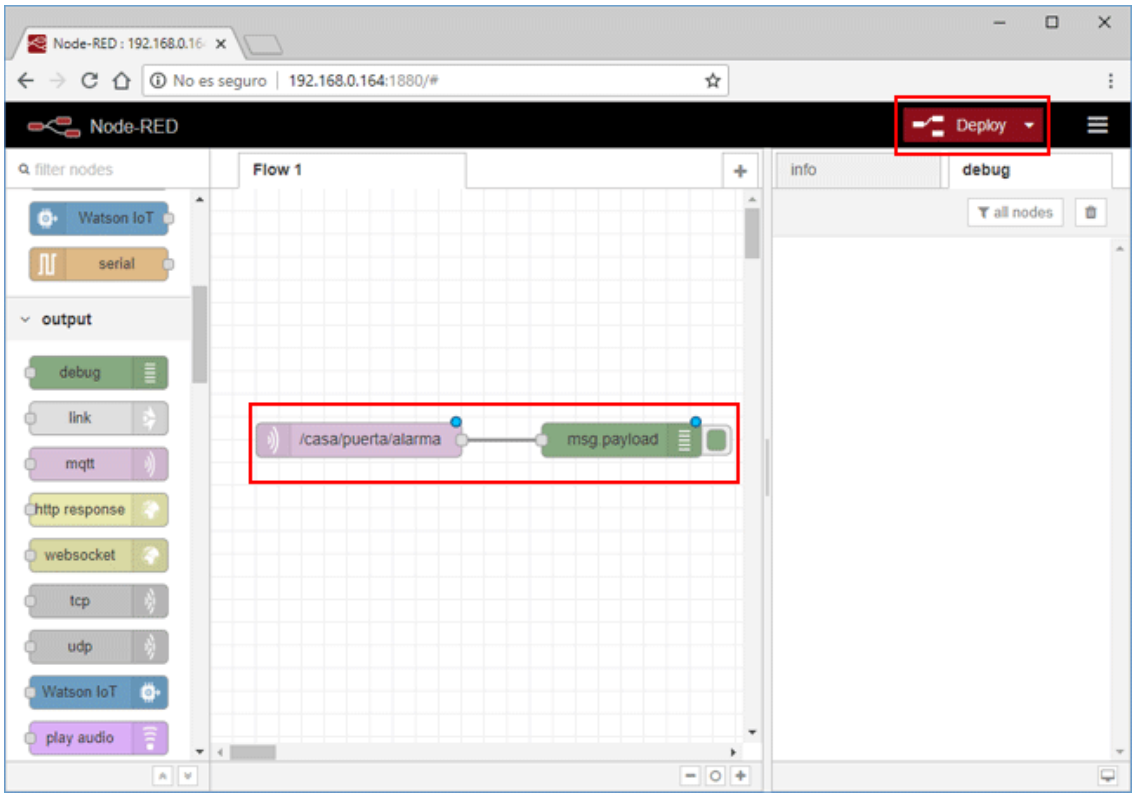
Arrastrar el nodo MQTT input



Setear el broker y el tópico al que el nodo está suscripto



Arrastrar un nodo output y conectarlo al nodo MQTT input



Una vez conectados y configurados los nodos, el broker ya se encuentra listo para recibir mensajes y saber distribuirlos.

5 Problemas

El principal problema con el que me encontré fue la inestabilidad de Node-RED. Programar el mismo flujo muchas veces tenía distintos resultados.

La solución parcial que encontré fue reiniciar Node-RED y el servicio de Mosquitto y/o borrar los nodos y volver a configurarlos.