Investigacion Proyecto Final: Sistema de Control de Temperatura y Humedad con ESP32, MQTT, Telegraf, InfluxDB y Grafana

Alumno:
William Leyton
COHORTE 2022

Contenido

1. Introducción

El presente informe presenta la investigación de proyecto final de Internet de las cosas (IoT) que se centra en la medición y visualización de la humedad y temperatura ambiente utilizando un microcontrolador ESP32 con el sensor DHT11 ademas de una pantalla lcd que nos mostrara los datos de temperatura, humedad y estado del relay. La solución IoT se desarrolla en un entorno virtualizado utilizando un servidor virtual con el sistema operativo Linux Ubuntu.

El objetivo principal de este proyecto es diseñar una solución completa y funcional que permita capturar los datos del sensor, transmitirlos a través del protocolo MQTT (Message Queuing Telemetry Transport), almacenarlos en una base de datos de series temporales y finalmente visualizarlos en tiempo real mediante una plataforma de visualización interactiva como Grafana.

El informe se estructura de manera que se describen en detalle los componentes clave de la solución, incluyendo la configuración del broker MQTT (Mosquitto), la utilización de Telegraf como agente de recopilación de datos, la instalación y configuración de InfluxDB como base de datos de series temporales, y finalmente, la configuración de Grafana como herramienta de visualización.

Además, se aborda la virtualización del servidor Ubuntu en un entorno Windows utilizando VMware Workstation, lo que permite tener un entorno de desarrollo y pruebas aislado y flexible.

2. Funcionamiento del Broker MQTT (Mosquitto)

2.1. ¿Qué es un Broker MQTT?

Un **Broker MQTT (Message Queuing Telemetry Transport)** es un intermediario de mensajería que facilita la comunicación entre los dispositivos en una red IoT (Internet de las cosas) mediante el protocolo MQTT. MQTT es un protocolo ligero y eficiente que se utiliza ampliamente en aplicaciones IoT debido a su capacidad para enviar mensajes de manera rápida y confiable, incluso en entornos con ancho de banda limitado y conexiones inestables.

El funcionamiento básico de un Broker MQTT se basa en el concepto de publicación y suscripción de mensajes. Los dispositivos que deseen enviar datos publican mensajes en un tema específico en el Broker, y otros dispositivos interesados en esos datos se suscriben a ese tema para recibir los mensajes.

Este se encarga de gestionar la entrega de los mensajes a los dispositivos suscritos, garantizando que todos los interesados reciban los datos relevantes de manera oportuna y eficiente. Además, el Broker puede autenticar y autorizar a los clientes para garantizar la seguridad de la comunicación en la red.

En este proyecto final elegi Mosquitto para facilitar la comunicación entre el microcontrolador ESP32 y la aplicación Grafana que visualiza los datos de humedad y temperatura. Mosquitto permite que el ESP8266 publique los datos del sensor DHT11 en un tema específico y que Grafana se suscriba a ese tema para recibir los datos y mostrarlos en tiempo real.

2.2. Publicación y Suscripción de Mensajes

El funcionamiento básico de **MQTT** se basa en el mecanismo de publicación y suscripción de mensajes. Los dispositivos IoT pueden publicar mensajes en tópicos específicos y otros dispositivos interesados pueden suscribirse a esos tópicos para recibir los mensajes. En esta sección, se explica

cómo el microcontrolador ESP32 publica los datos del sensor DHT11 y cómo Grafana se suscribe para recibir y visualizar los datos en tiempo real.

- **Publicación de Mensajes desde el ESP8266:**

  El ESP32, como cliente MQTT, puede publicar mensajes en tópics específicos del Broker. Para ello, es necesario utilizar una biblioteca MQTT en el código del ESP32 para establecer la conexión con el Broker y enviar los datos del sensor DHT11.

  El código del ESP32 debe incluir las siguientes acciones:

- **Conexión al Broker MQTT:**

  El ESP32 debe establecer una conexión con el Broker Mosquitto. Para ello, se debe especificar la dirección IP y el puerto del Broker, y proporcionar las credenciales de autenticación en caso de haber configurado la autenticación previamente.

- **Lectura de los Datos del Sensor DHT11:**

  El ESP32 debe leer los datos de humedad y temperatura del sensor DHT11 a intervalos regulares.

- **Publicación de los Datos en Tópics Específicos:**

  Una vez que se obtienen los datos del sensor, el ESP32 debe publicarlos en tópicos específicos del Broker MQTT. Por ejemplo, se puede publicar la humedad en el tópico "sensor/humidity" y la temperatura en el tópico "sensor/temperature_celsius" o “sensor/temperature_fahrenheit”.

Suscripción y Visualización de Mensajes en Grafana:

Grafana, como cliente MQTT, se suscribe a los tópics relevantes para recibir los mensajes publicados por el ESP32. Para ello, Grafana debe configurarse para conectarse a la base de datos y suscribirse a los tópics que contienen los datos del sensor.

Una vez que Grafana se ha suscrito a los tópics, recibirá los mensajes con los datos de humedad y temperatura en tiempo real. Luego, Grafana puede procesar y visualizar los datos en forma de gráficos, paneles, o cualquier otro formato personalizado que se desee.

3. Telegraf: Agente de Recopilación de Datos

3.1. Descripción de Telegraf

Telegraf es una herramienta de código abierto desarrollada por InfluxData que actúa como agente de recopilación de datos para InfluxDB. Su función principal es recolectar datos de diferentes fuentes y enviarlos a la base de datos de series temporales InfluxDB para su almacenamiento y análisis.

Tambien es altamente configurable y extremadamente flexible, lo que permite a los usuarios seleccionar las fuentes de datos que deseen monitorear y enviar. Ofrece una amplia gama de plugins de entrada y salida que facilitan la integración con diversas fuentes de datos y sistemas de monitoreo.

Algunos de los plugins de entrada disponibles en Telegraf incluyen:

- **Entrada de MQTT:** Permite a Telegraf suscribirse a tópicos MQTT y recibir datos publicados por dispositivos MQTT, como el ESP32 con el sensor DHT11.

- **Entrada de SNMP:** Permite la consulta y recolección de datos de dispositivos SNMP habilitados, como enrutadores y switches.
- **Entrada de archivos de registro:** Permite leer y analizar datos de archivos de registro generados por aplicaciones y sistemas.
- **Entrada de sensores de hardware:** Permite la lectura de datos de sensores de hardware, como temperaturas y voltajes, directamente desde el sistema operativo del servidor.

3.2. Configuración de Entradas y Salidas de Telegraf

Para utilizar Telegraf en el proyecto, primero se debe instalar y configurar adecuadamente el agente en el servidor virtual Ubuntu.

Se debe abrir una terminal y poner los siguientes comandos

sudo apt update

sudo apt install telegraf

Una vez instalado, se debe configurar las entradas de Telegraf para recopilar los datos del broker MQTT. Esto se logra editando el archivo de configuración de Telegraf, que se encuentra en "/etc/telegraf/telegraf.conf".

En el archivo de configuración, se agrega el plugin de entrada MQTT con los detalles de conexión al broker, como la dirección IP, puerto, nombre de usuario y contraseña (si se configuró la autenticación en el broker). Además, se especifican los tópics a los que Telegraf debe suscribirse para recibir los datos publicados por el ESP32.

Luego de configurar las entradas, se deben configurar las salidas de Telegraf para enviarlos a la base de datos InfluxDB. En el mismo archivo de configuración de Telegraf, se agrega el plugin de salida de InfluxDB con los detalles de conexión, como la dirección IP, puerto y nombre de la base de datos InfluxDB.

Después de realizar los cambios en la configuración, se debe reiniciar el servicio de Telegraf para que los cambios surtan efecto:

sudo systemctl restart telegraf

Con la configuración adecuada de las entradas y salidas de Telegraf, el agente está listo para recopilar los datos del broker MQTT y enviarlos a la base de datos InfluxDB para su almacenamiento y análisis. De esta manera, Telegraf juega un papel fundamental en la integración y recopilación eficiente de datos en la solución IoT, facilitando la visualización y monitoreo de los mismos en tiempo real a través de Grafana.

4. InfluxDB: Base de Datos

4.1. Conceptos de InfluxDB

InfluxDB es una base de datos de series temporales de código abierto diseñada para almacenar, consultar y visualizar datos que varían con el tiempo. Está especialmente diseñada para manejar grandes volúmenes de datos que se generan continuamente, como datos de sensores, registros de eventos y métricas de rendimiento.

Algunos de los conceptos clave de InfluxDB son:

- **Series Temporales:** Los datos se almacenan en series temporales, que consisten en puntos de datos con una marca de tiempo y un valor. Cada serie temporal se identifica mediante una combinación única de "medida" y "etiquetas". La "medida" representa el tipo de datos (por ejemplo, temperatura), y las "etiquetas" son pares clave-valor que se utilizan para indexar y organizar los datos.
- **Puntos de Datos**: Cada punto de datos en una serie temporal contiene una marca de tiempo que indica cuándo se generó el dato y un valor numérico o de texto que representa la medición o evento en sí.
- **Bases de Datos:** Se organizan las series temporales en bases de datos separadas. Cada base de datos puede contener múltiples series temporales relacionadas con un tema específico.
- **Retención de Datos**: Permite definir políticas de retención que determinan cuánto tiempo se conservan los datos en la base de datos antes de ser eliminados. Esto es útil para mantener un control sobre el tamaño de la base de datos y gestionar el almacenamiento a largo plazo de datos históricos.

5. Grafana: Herramienta de Visualización

5.1. Introducción a Grafana

Grafana es una popular herramienta de código abierto utilizada para la visualización y análisis de datos en tiempo real. Se integra con diversas bases de datos de series temporales, como InfluxDB, y permite crear paneles y gráficos interactivos para mostrar datos de manera clara y comprensible.

Es ampliamente utilizado en aplicaciones de monitoreo, IoT y análisis de datos debido a su capacidad para integrarse con diversas fuentes de datos y su amplia gama de opciones de personalización.

Las principales características de Grafana incluyen:

- **Soporte para múltiples fuentes de datos:** Grafana admite una variedad de bases de datos y sistemas de almacenamiento, lo que permite la integración con bases de datos de series temporales como InfluxDB, bases de datos SQL, Elasticsearch y más.
- **Creación de paneles personalizados:** Los usuarios pueden diseñar paneles de control personalizados con múltiples gráficos, tablas, medidores y otros componentes visuales para mostrar diferentes métricas y datos.
- **Alertas y notificaciones:** Grafana permite establecer reglas de alerta basadas en umbrales de datos, y puede enviar notificaciones por correo electrónico, Slack u otras plataformas cuando se activan las alertas.
- **Exploración y análisis de datos:** Grafana proporciona herramientas de consulta y filtrado que permiten a los usuarios explorar y analizar datos de forma interactiva.

6. Integración de los 4 Servicios

La integración de los 4 servicios (Broker MQTT, Telegraf, InfluxDB y Grafana) se realiza de la siguiente manera:

1. **Broker MQTT (Mosquitto):** Este broker actúa como un intermediario de mensajes entre los dispositivos IoT y los servicios de backend. Los dispositivos, como el ESP8266 con el Sensor DHT11, publican datos en tópicos específicos y los suscriptores, como Telegraf, se conectan a esos tópicos

para recibir los datos. En esta configuración, el Mosquitto se ejecuta en el servidor virtual y permite la comunicación bidireccional entre los dispositivos y los servicios.

2. **Telegraf:** se encarga de recopilar datos del Broker MQTT y enviarlos a InfluxDB. En este caso, Telegraf se configura para suscribirse a los tópicos específicos en el Broker MQTT, donde el ESP32 publica los datos del Sensor DHT11. Cuando se publican nuevos datos, Telegraf los recibe y los almacena en la base de datos de InfluxDB para su posterior análisis y visualización.

3. **InfluxDB:** es una base de datos de series temporales que almacena los datos recopilados por Telegraf. Cuando Telegraf envía datos a InfluxDB, estos se organizan en series temporales, lo que permite un almacenamiento eficiente y un acceso rápido a los datos. InfluxDB se encarga de almacenar los datos históricos de temperatura y humedad recopilados del Sensor DHT11, lo que proporciona una solución escalable para la gestión de grandes volúmenes de datos en tiempo real.

4. **Grafana:** se utiliza para visualizar los datos almacenados en InfluxDB en forma de paneles y gráficos interactivos. Grafana se conecta a InfluxDB como fuente de datos y utiliza consultas para recuperar los datos y mostrarlos en los gráficos. Los paneles de Grafana pueden personalizarse para mostrar información relevante, como gráficos de temperatura y humedad en tiempo real, tendencias históricas y alertas basadas en umbrales de datos.

El flujo de datos entre los 4 servicios es el siguiente:

- El ESP32 con el Sensor DHT11 publica datos de temperatura y humedad en un tópico específico en el Broker MQTT.
- Telegraf, configurado para suscribirse a ese tópico, recibe los datos del Broker MQTT y los envía a InfluxDB.
- InfluxDB almacena los datos en series temporales para su posterior análisis.
- Grafana se conecta a InfluxDB como fuente de datos y muestra los datos en tiempo real y las tendencias históricas en paneles y gráficos interactivos.

Esta integración permite una solución completa y escalable para la recopilación, almacenamiento, análisis y visualización de datos en un proyecto IoT basado en el ESP32 y el Sensor DHT11. Con esta configuración, es posible monitorear y analizar las condiciones ambientales de manera efectiva, proporcionando información valiosa para la toma de decisiones y la optimización de procesos en diversas aplicaciones.

7. Virtualización con Ubuntu Server en Windows

La virtualización con Ubuntu Server en Windows desempeña un papel fundamental en la solución IoT propuesta, ya que permite ejecutar un servidor virtualizado con el sistema operativo Ubuntu dentro de un entorno de Windows. A continuación, se explica el funcionamiento y el rol de la virtualización en la solución IoT:

Funcionamiento:

La virtualización se realiza utilizando software de virtualización, como VMware Workstation, que permite crear y administrar máquinas virtuales en un sistema operativo anfitrión, en este caso, Windows. Al crear una máquina virtual, se puede asignar una cantidad específica de recursos de hardware, como CPU, memoria RAM y espacio en disco, para su uso exclusivo por parte de la máquina virtual.

El sistema operativo Ubuntu Server se instala dentro de la máquina virtual, lo que crea un entorno aislado y separado del sistema operativo anfitrión (Windows). El servidor virtualizado se comporta como una entidad independiente, con su propia dirección IP, configuración de red y aplicaciones instaladas, como el Broker MQTT (Mosquitto), InfluxDB, Grafana y Telegraf.

Rol en la solución IoT:

- **Hospedaje del Broker MQTT y la Base de Datos**

  La máquina virtual con Ubuntu Server actúa como el servidor de alojamiento para el Broker MQTT (Mosquitto) y la base de datos InfluxDB. Estos servicios se ejecutan dentro del entorno virtualizado y están disponibles para su uso por parte de otros dispositivos y aplicaciones en la red local.

- **Comunicación con el ESP32y el Broker MQTT**

  El ESP8266 con el Sensor DHT11, al enviar datos de temperatura y humedad al Broker MQTT, se conecta a través de la red Wi-Fi del servidor virtualizado en Ubuntu Server. El Broker MQTT, que se ejecuta en el servidor virtual, recibe los mensajes del ESP8266 y los procesa para su posterior almacenamiento en InfluxDB.

- **Recopilación y Almacenamiento de Datos**

  Telegraf, que también se ejecuta en la máquina virtual, se encarga de recopilar los datos del Broker MQTT y enviarlos a InfluxDB para su almacenamiento. Telegraf actúa como el agente de recopilación de datos que conecta el Broker MQTT con la base de datos InfluxDB.

- **Visualización y Monitoreo**

  Grafana, otra de las aplicaciones instaladas en la máquina virtual, se conecta a InfluxDB para acceder a los datos almacenados y crea paneles y gráficos interactivos para visualizar y monitorear en tiempo real la temperatura y humedad capturada por el ESP8266 con el Sensor DHT11.

8. ESP32 y Sensor DHT11

8.1. Descripción del ESP32 como Microcontrolador

El ESP32 es un microcontrolador de bajo costo y bajo consumo de energía que incorpora un sistema en chip (SoC) con capacidad de conexión Wi-Fi. Es ampliamente utilizado en proyectos de IoT debido a su tamaño compacto, su facilidad de uso y su versatilidad para conectarse a redes inalámbricas.

Características clave del ESP32:

- **Procesador:** Está equipado con un procesador Tensilica Xtensa LX6 de 32 bits, disponible en versiones de uno o dos núcleos, que funciona a una frecuencia de hasta 240 MHz.
- **Memoria:** Incluye 520 KB de RAM y capacidad de memoria flash externa que varía según el módulo, generalmente entre 4 MB y 16 MB. También cuenta con 448 KB de memoria ROM para funciones internas.
- **Wi-Fi y Bluetooth Integrados:** El ESP32 tiene un módulo Wi-Fi 802.11 b/g/n integrado y soporte para Bluetooth v4.2 y BLE, lo que lo convierte en una excelente opción para proyectos IoT que requieren conectividad dual.

- **GPIO:** Dispone de pines de entrada/salida de propósito general (GPIO) que permiten interactuar con sensores y dispositivos externos. Hasta **36 pines** configurables para uso general.
- **Interfaces de Comunicación:** El ESP32 soporta interfaces de comunicación como I2C, SPI y UART, lo que facilita la conexión con sensores, pantallas y otros dispositivos.
- **ADC:** 18 canales ADC (12 bits, hasta 18 entradas).
- **DAC:** 2 canales DAC.
- **PWM:** Hasta 16 canales para control de motores o luces LED.
- **Programación:** Se puede programar el ESP32 utilizando el lenguaje de programación Arduino, lo que lo hace muy accesible para la comunidad de desarrolladores y entusiastas de IoT.

8.2. Conexión y Programación del ESP32 con el Sensor DHT11

El Sensor DHT11 es un sensor de temperatura y humedad que se puede utilizar junto con el ESP8266 para medir y enviar datos ambientales a través de la red Wi-Fi. A continuación, se describen los pasos para conectar y programar el ESP8266 con el Sensor DHT11:

1. Conexión del Sensor DHT11 al ESP32:
    - Conecte el pin de datos (DATA) del Sensor DHT11 al pin GPIO4 o D2 del ESP32.
    - Conecte el pin de alimentación (VCC) del Sensor DHT11 al pin de 3.3V del ESP32.
    - Conecte el pin de tierra (GND) del Sensor DHT11 al pin GND del ESP32.
2. Programación del ESP32 para Leer Datos del Sensor DHT11:
    - Utilice el entorno de desarrollo Arduino IDE para programar el ESP32.
    - Asegúrese de tener instalado el soporte para el ESP32 en el Arduino IDE siguiendo las instrucciones proporcionadas por el fabricante del ESP32.
    - Utilice una biblioteca como "DHT.h" para leer los datos del Sensor DHT11. Esta biblioteca proporciona funciones para leer la temperatura y la humedad del sensor.
    - Escriba el código para configurar la conexión Wi-Fi del ESP8266 y establecer la comunicación con el Broker MQTT para enviar los datos del sensor.
    - Programe el ESP32 para que lea los datos del Sensor DHT11 periódicamente y los envíe al Broker MQTT a través de la conexión Wi-Fi.
    - Compile y cargue el programa en el ESP32 utilizando el Arduino IDE.

Con el ESP32 conectado y programado correctamente con el Sensor DHT11, el dispositivo estará listo para medir la temperatura y la humedad y enviar los datos al servidor virtual a través de MQTT. Esta integración permitirá que los datos del sensor sean almacenados en InfluxDB y visualizados en tiempo real a través de Grafana, como parte de la solución completa de IoT.

8.3. Conexión y Programación del ESP32 con una Pantalla LCD

Para integrar una pantalla LCD con el ESP32, se utiliza una pantalla basada en el controlador HD44780, que puede ser controlada directamente o mediante un módulo adaptador I2C para reducir el número de pines requeridos.

Conexión básica (con módulo I2C):

- Conectar los pines SDA y SCL del módulo I2C al ESP32 (GPIO21 y GPIO22).
- Alimentar la pantalla con 5V y conectar GND.
- Instalar la librería LiquidCrystal_I2C en el Arduino IDE para facilitar la programación.

Código básico:
El ESP32 puede usar la pantalla LCD para mostrar datos como los valores de temperatura y humedad obtenidos del sensor DHT11 o cualquier información relevante. Esto mejora la visualización local en proyectos IoT.

8.4. Control del ESP32 con Relé y Programación de Control Automático

El ESP32 puede controlar dispositivos externos, como lámparas, motores o electrodomésticos, a través de módulos de relé. Además, es posible programar un sistema de control automático con activación y desactivación manual para proyectos avanzados de automatización.

Conexión básica del módulo relé con el ESP32:

- Conectar la entrada de control del módulo relé a un pin GPIO del ESP32 (por ejemplo, GPIO5).
- Alimentar el módulo relé con 5V y conectar GND.
- Conectar el dispositivo eléctrico al circuito del relé (NO/NC y COM).

Control automático con programación:
El ESP32 puede programarse para activar o desactivar el relé automáticamente según condiciones predefinidas, como lecturas de sensores (por ejemplo, temperatura o humedad). Un sensor como el DHT11 puede monitorear los valores, y el ESP32 decidirá si accionar el relé.

Activación y desactivación del modo automático:
Se puede implementar un interruptor o botón conectado a otro pin como GPIO23 del ESP32 para alternar entre modo automático y manual.

- En **modo manual**, el usuario controla directamente el estado del relé (encendido/apagado).
- En **modo automático**, el ESP32 activa o desactiva según el valor seteado .