ITESO, Universidad Jesuita de Guadalajara

Entrega 2 Proyecto
IoT para Linux Embebido

Rodrigo Ramos Romero IE715082
Daniel Jiram Rodriguez Padilla IE703331
12/11/2025

Zamora Davalos Ricardo

Proyecto: MQTT con reconocimiento de comandos por voz

Tabla de Contenidos

Definición de la entrega

La segunda entrega de proyecto añade la funcionalidad del Hardware que necesita el proyecto, micrófono y botones. Se trabaja sobre la implementación de MQTT de la entrega 1, en esta entrega se puede escuchar publicaciones a los 4 tópicos del proyecto, y grabar el comando de voz que será implementado en la entrega 3.

Entrega 1

- Utilizar Mosquitto para la creación del Broker de MQTT a través del cual se comunicará la aplicación y todos los dispositivos.
- Probar este bróker con comandos desde otra CMD en el host de Windows.
- Crear la aplicación dentro de "MQTT Panel" en el celular y probar la comunicación con este bróker de MQTT.
- Lograr la conexión con el celular e interactuar con este utilizando comandos de Publish y Subscribe.

Entrega 2

- Agregar implementación del micrófono.
- Botón "Play" para grabar un comando de voz en formato .wav "Comando.wav". graba por 2 segundos.
- Botón "Next" para escuchar el "Comando.wav" y verificar que se haya grabado lo que queremos.
- Todo esto encima de la aplicación MQTT de la entrega 1

Aplicación

Se creció el proyecto a 5 archivos en total. Buttons .c y .h, MQTT .c y .h, main.c

MQTT.h

Contiene los tópicos para subscribirnos, el QoS, Keep alive, puerto del Servidor (1883) y la IP del host. Básicamente, todo lo que necesita una aplicación de MQTT para poder ser editado rápidamente.

MQTT.c

Contiene toda la funcionalidad MQTT del proyecto. Desde las conexiones con el Broker, hasta el código para subscribirnos y publicar. Se junto todo en un mismo código, ya que se espera que el proyecto final tenga varios ejecutables y librerías. Tratamos de encapsular totalmente MQTT en solo estos dos códigos.

Buttons.c

Dependiendo del Ev.value detecto al presionar el botón hace una función en específico:

- Play: Grabar comando de voz por 2 segundos, guardar como "Comando.wav".
- Next: Reproducir "Comando.wav", únicamente para que el usuario verifique que se grabó lo que esperaba.

Buttons.h
Únicamente contiene el prototipo de la función de botones para ser llamada por el main

Main.c
Llama las funciones de MQTT para inicializar la conexión al bróker, entra a un loop para seguir conectado al bróker mientras escucha los botones.

La aplicación funciona de esta manera:

1. Manda el paquete de conexión creado con apuntadores.
2. Espera el acknowledge, que el bróker haya permitido establecer una conexión.
3. Hace las subscripciones a los tópicos seleccionados.
4. Entra al loop de main, donde espera mensajes de MQTT en los tópicos subscritos, o un input del usuario para poder mandar un mensaje propio de Publish.
5. Escucha a los botones mientras esta en el loop, graba o reproduce el comando.

Resultados

Las pruebas se realizaron primero probando el bróker creado de MQTT con Mosquitto, usando pings, subscribes y publishes desde consola en esta primera foto vemos el servidor funcionando perfectamente y como se necesita para la aplicación.

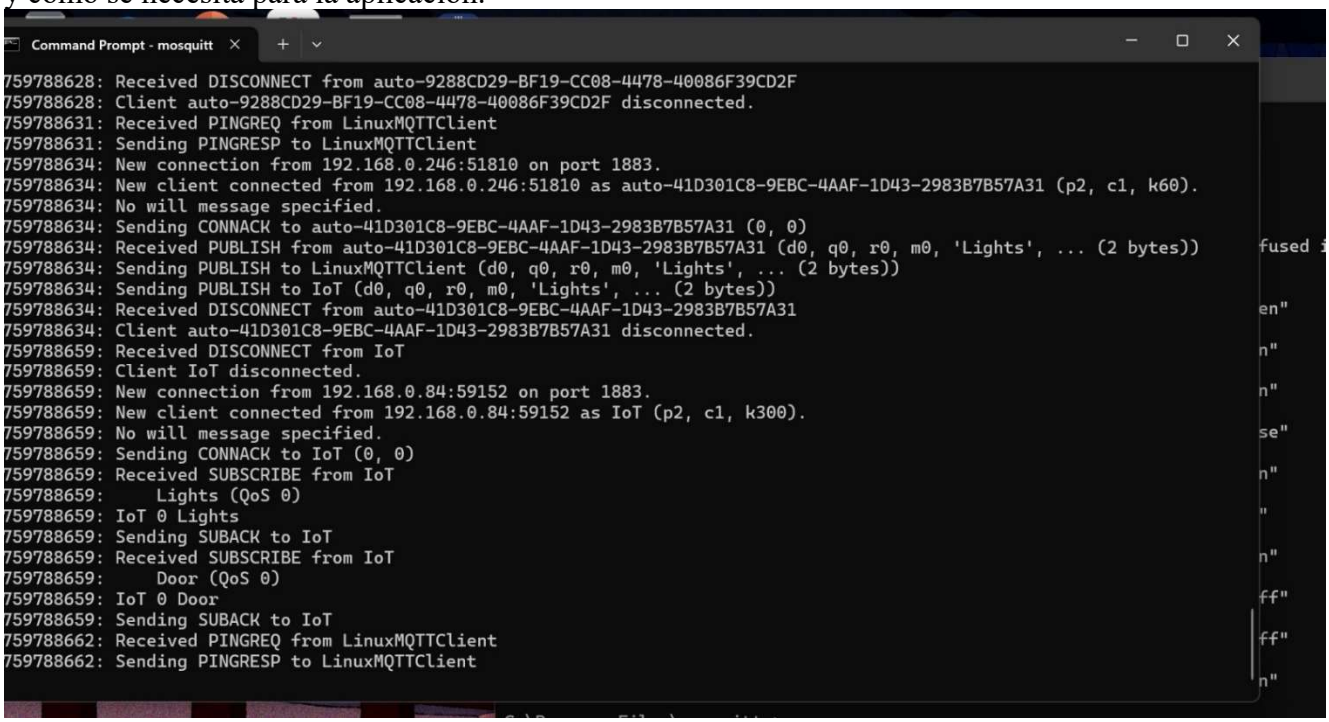

Figura 1: MQTT Bróker corriendo en Windows CMD

Teniendo un bróker de MQTT funcional, el siguiente paso es preparar la interfaz. La meta del proyecto es lograr una interfaz completa con el celular, imprimiendo información de estados en este, y permitiendo actualizar estados y salidas del sistema total desde el celular. Se utilizo la aplicación “MQTT Panel” y esta es la interfaz creada:

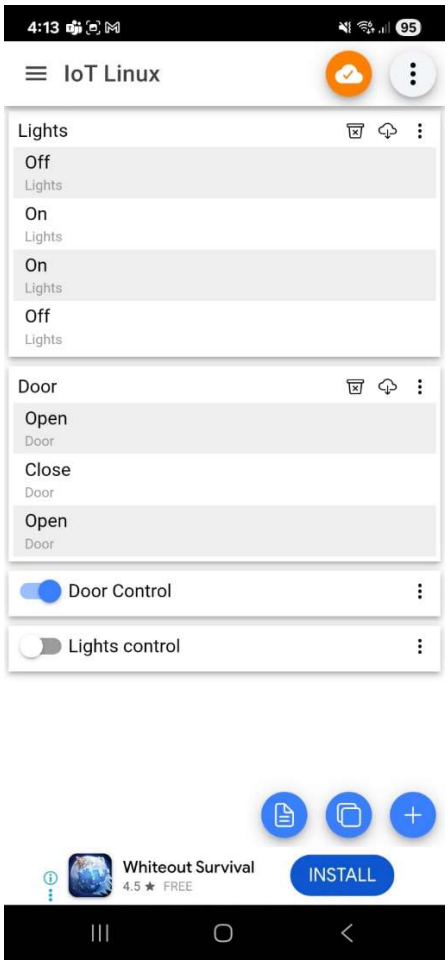

Figura 2: Aplicación de apoyo en el celular, MQTT Panel

Aquí podemos ver ya la aplicación funcionando. Imprimimos los últimos 3 estados o cambios de los tópicos “Door” y “Lights”. Los switches nos permiten actualizar manualmente el estado.

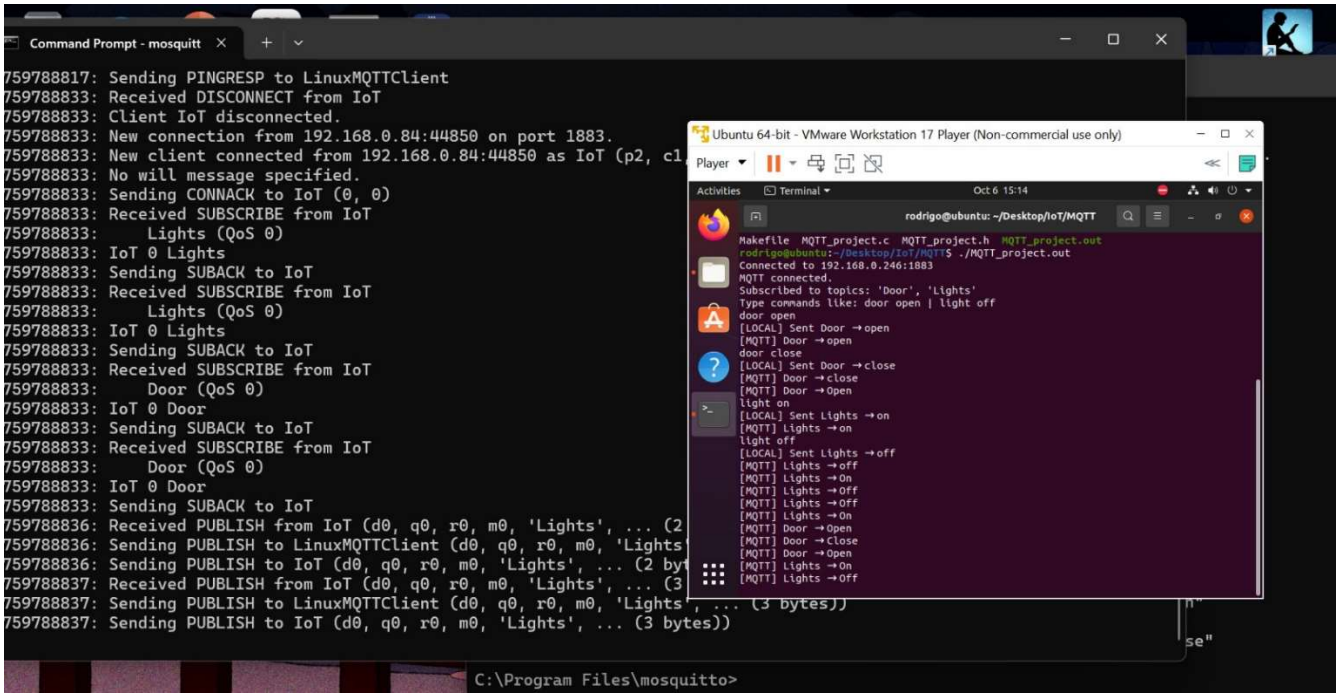

Figura 3: Aplicación funcionando en Consola

Finalmente, la aplicación funciona como fue esperado y definido en las metas. Se puede actualizar en la misma aplicación el estado de los dos tópicos o funcionalidades.

Debido a que los resultados de la entrega 2 son de voz, no se agregaron nuevas fotos como evidencia.