

Proyecto final

Daniela López y Yamil Hallal



Universidad autónoma de Guadalajara

IoT

**MQTT broker and client**

MQTT (Message Queuing Telemetry Transport) es un protocolo de comunicación ligero y eficiente diseñado para facilitar la comunicación entre dispositivos en redes de baja potencia y ancho de banda limitado.

MQTT utiliza un modelo de publicación/suscripción en el que los dispositivos envían y reciben mensajes a través de un intermediario centralizado conocido como el broker MQTT. Los clientes pueden suscribirse a uno o más temas (topics) para recibir mensajes relevantes y también pueden publicar mensajes en los temas para enviar información a otros clientes interesados.

MQTT se caracteriza por su simplicidad, bajo consumo de ancho de banda y recursos, y su capacidad para funcionar en redes con conexiones poco confiables. Es ampliamente utilizado en aplicaciones de IoT, automatización del hogar, monitoreo remoto, telemetría, y otras áreas en las que se requiere una comunicación eficiente entre dispositivos.

**Desarrollo del MQTT**

A lo largo del proyecto final, se desarrolló la imitación de un broker tipo MQTT, por medio del archivo “Servidor.c” que es donde se lleva a cabo la aceptación de conexiones y el dilo que realiza los pings request de manera continua. Asimismo, el archivo “RcvCast.c” representa otro hilo que se ejecuta de manera paralela, en el que se “cachan” todos los paquetes que se va recibiendo y se clasifican, para así, poder enviar el paquete que representa su contraparte o confirmación de llegada.

Por otra parte, el cliente realiza sus funciones principales de conexión y ping por medio del archivo “Cliente.c” y el casteo de receive y menú de funciones por medio de “ClientAction.c”. Todos y cada uno de los frames y funciones globales se encuentran declarados en el archivo “libs.h”.

**Conclusión**

A lo largo de este proyecto, tuvimos principalmente el problema del casteo de los paquetes recibidos (en un buffer) a su estructura correspondiente, por lo que se tuvo que hacer las condicionales tomando en cuenta cada byte del buffer. Sin embargo, se logró obtener un resultado satisfactorio a pesar de las dificultades.