COMMUNICATING MOBILE TERMINALS REPORT

IoT, LoRa, WiFi, MQTT, SSL, ATECC508, Mongoose OS, Raspberry Pi ESP8266

Author
Viet-Huy HA

Contents

1 Introduction

The objective of this project is to establish communication between an MQTT client and an MQTT server (both implemented on Raspberry Pis) using LoRa technology. The MQTT client is connected to an ESP8266 module, and the MQTT server is configured to access the Internet. A critical step before initiating communication is the authentication process for both the client and server. For this purpose, the ATECC608 chip is employed to facilitate elliptic-curve cryptography, enabling secure authentication using their respective credentials. Furthermore, to safeguard the data exchanged between the two entities after successful authentication, AES encryption is utilized. The entire system's structure is depicted in Figure 1.

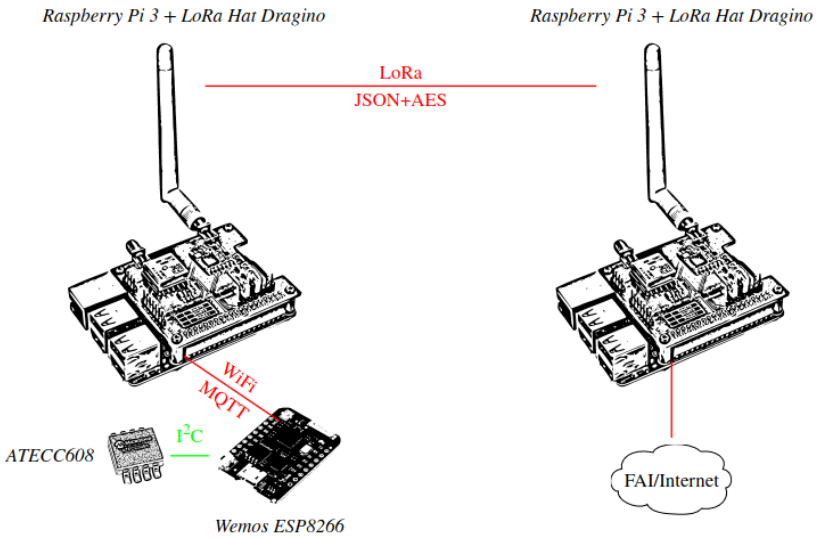

Figure 1: System Architecture

Then we set up devices as following:

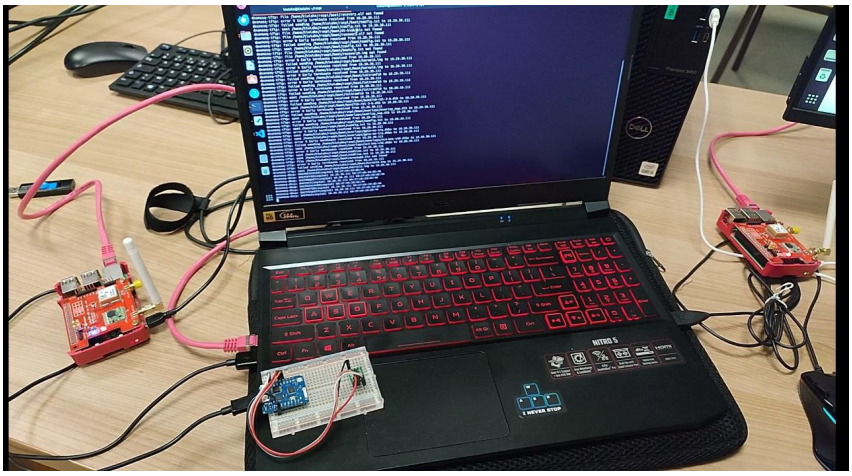

Figure 2: System Architecture

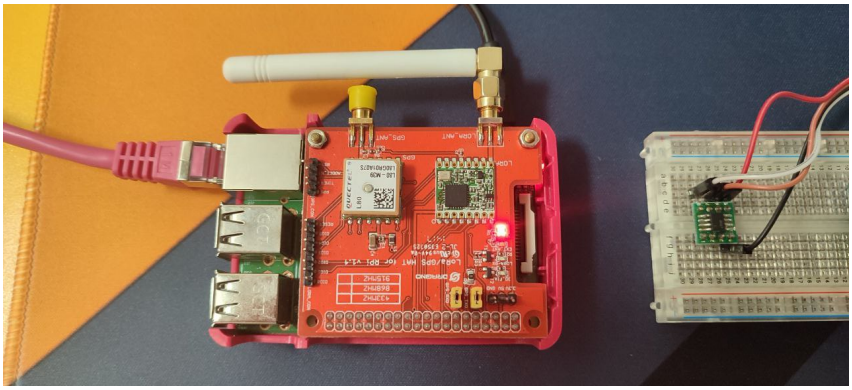

Figure 3: Raspberry Pi

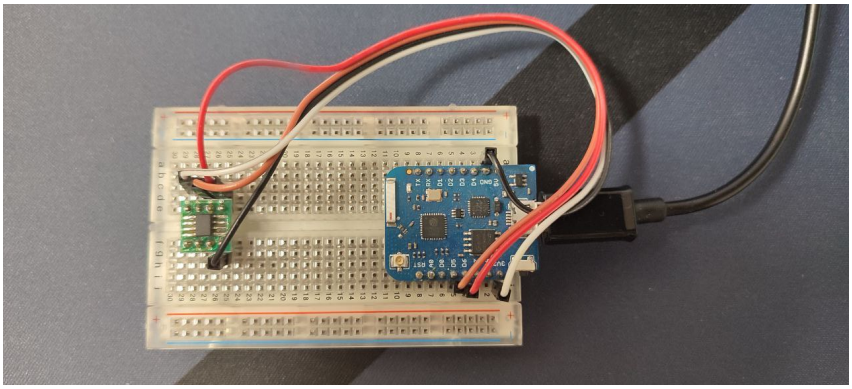

Figure 4: ESP8266

Within the confines of this report, we focus solely on the outcomes related to setting up communication, encryption, and data exchanges, as specified by the project requirements. For in-depth information on the implementation process, please consult the document titled `TMCProject_ConfigurationInstruction.pdf` and the demo video on Youtube here.

2 Communication

2.1 WiFi and MQTT

2.1.1 WiFi connection from ESP8266 to Raspberry Pi

Initially, we configured a WiFi access point on the Raspberry Pi by installing the hostapd and dnsmasq packages. The hostapd package enables the creation of a wireless hotspot, and dnsmasq allows for the setting up of a DNS and DHCP server.

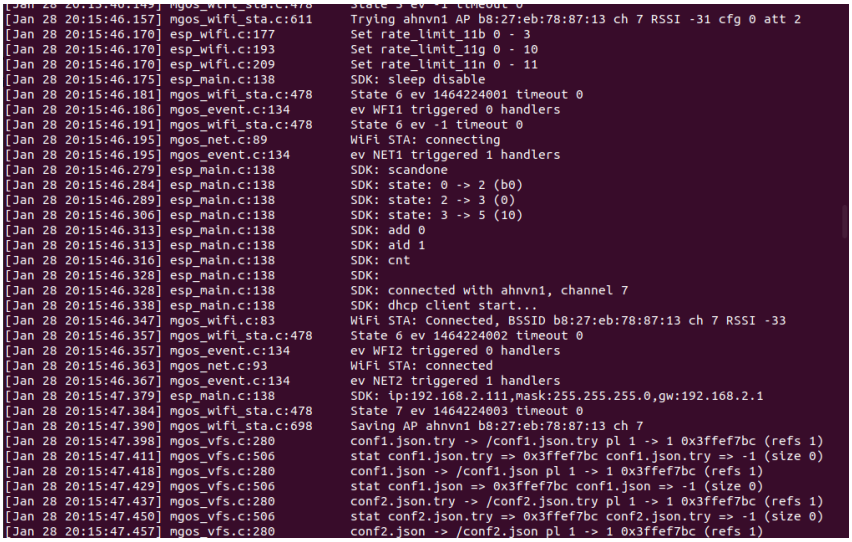

Figure 5: ESP8266 connects to WiFi

2.1.2 Connection between MQTT client and MQTT server

The MQTT client is an ESP8266 module designed to connect to an MQTT server, which is a Raspberry Pi in this scenario. The client is programmed to send messages with the content "Good job! You are connected" to the topic 'esp8266' every 5 seconds. Concurrently, the MQTT server is subscribed to the same topic to receive these messages.

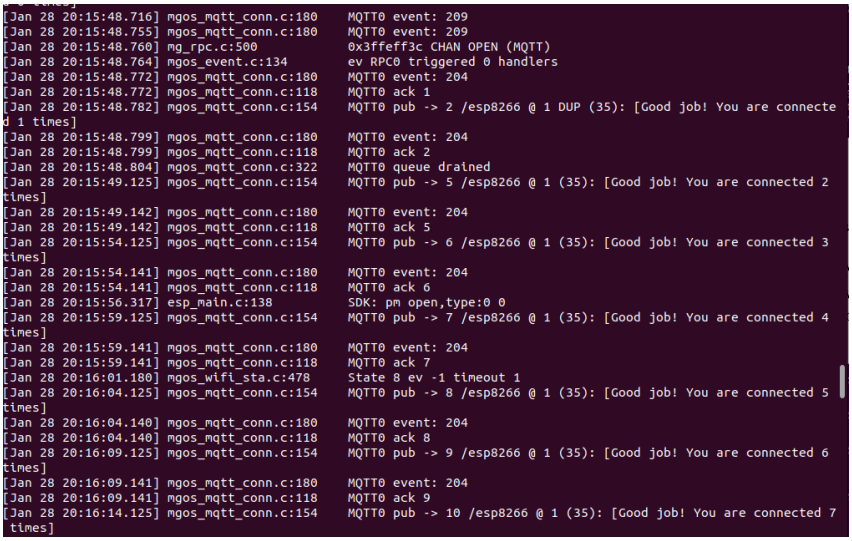

Figure 6: ESP8266 publishes to the topic `esp8266`

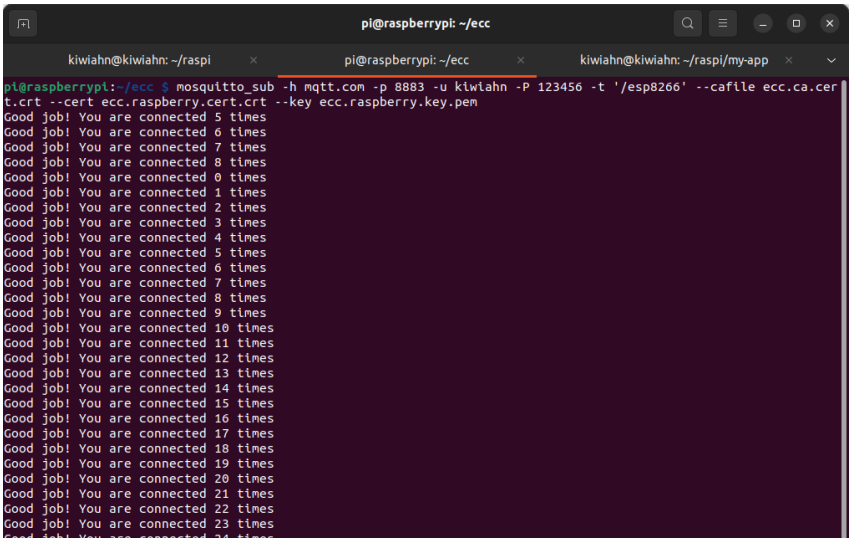

Figure 7: Raspberry Pi subscribes to the topic `esp8266`

2.2 LoRa

In the setup for LoRa communication, there are a LoRa client and a LoRa server, both of which are Raspberry Pi. The LoRa client is the Raspberry Pi that connects to the ESP8266. This client receives messages from the topic `esp8266` and forwards them to the LoRa server. The communication between these two devices is secured with AES encryption, details of which will be elaborated in the next section.

3 Encryption

3.1 Elliptic-curve cryptography

According to the project's architecture illustrated in Figure 1, the ESP8266 module is connected to the ATECC608 component. The ATECC608 is a secure element equipped with advanced ECC capabilities, facilitating encryption, signature, and verification operations efficiently.

In line with the project requirements, the connection between the ESP8266 and the Raspberry Pi is established using the TLS protocol after completing the authentication process, which involves certificate and ECC key exchanges. This ensures a secure communication channel between the devices, leveraging the strengths of ECC for IoT security.

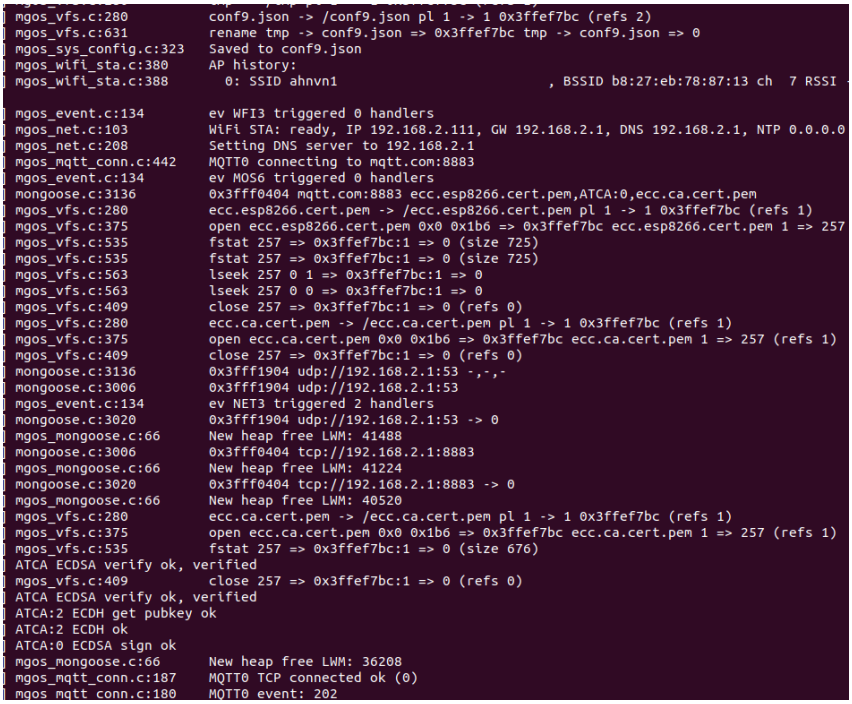

Figure 8: Authentication using TLS protocol

3.2 AES

The communication process between the LoRa client and the LoRa server involves encrypting messages sent by the LoRa client using Advanced Encryption Standard (AES) with a key shared between the two devices. Once the LoRa server receives the encrypted message, it proceeds to decrypt the data using the same shared key. After decryption, the server displays the received information. This method ensures that the data exchanged over the LoRa network remains confidential and secure from unauthorized access.

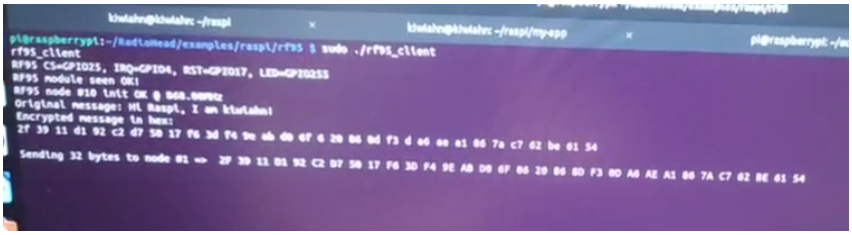

Figure 9: Message from LoRa client

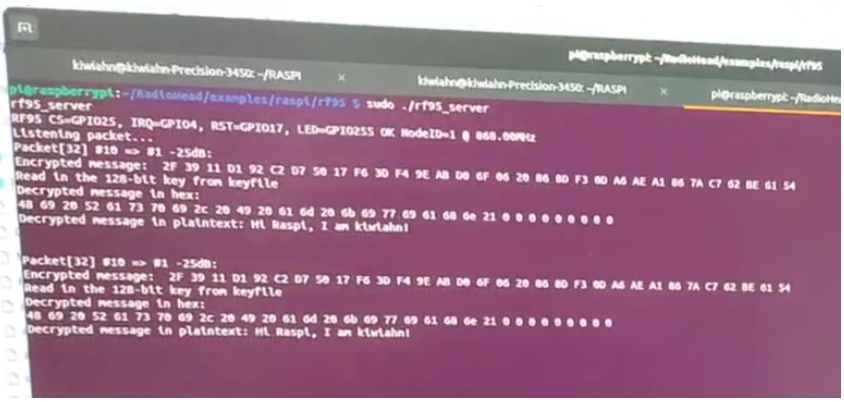

Figure 10: Received message on LoRa server

4 Conclusion

In this project, we have successfully implemented the specified architecture, facilitating secure communication between the ESP8266 and a Raspberry Pi, as well as between two Raspberry Pi. This setup simulates a real-world IoT structure, where messages from embedded devices are relayed to an MQTT server on the Internet through an intermediate server. Despite encountering numerous technical challenges, we managed to fulfill all project requirements within the constrained timeframe. These difficulties, however, contributed significantly to our learning, enhancing our knowledge and skills in a way that will greatly benefit our personal development and professional careers.