**Security Measures for ESP32 + SIM7600 + Modbus RTU + MQTT**

**1. Overview**

This document outlines the security measures implemented in the ESP32-based system utilizing the SIM7600 module for LTE communication, Modbus RTU for inverter control, and MQTT for cloud connectivity. It also provides recommendations for enhancing security to protect against unauthorized access and cyber threats.

**2. Implemented Security Measures**

**2.1. Software Security Measures**

**✅ MQTT Security**

* **Authentication & Authorization**: The system uses **username-password authentication** for MQTT connections to prevent unauthorized access.   
  **Implemented**
* **TLS Encryption**: The MQTT communication is secured using **TLS 1.2** to encrypt data between ESP32 and the EMQX broker.  
  **Implemented**
* **Client ID Uniqueness**: Each ESP32 device generates a unique **Client ID** to avoid session hijacking.  
  **Implemented**
* **Retained Message Handling**: MQTT retained messages are carefully managed to avoid outdated commands from being replayed.  
  **Implemented**

**✅ Modbus RTU Security**

* **CRC Validation**: Modbus RTU uses **Cyclic Redundancy Check (CRC)** to prevent data corruption and tampering.  
  **Implemented**
* **Command Whitelisting**: The ESP32 only accepts **specific predefined Modbus commands** from the MQTT server.  
  Not Yet, it is easy and will be implemented after we drive the inverter successfully.
* **Rate Limiting**: Prevents **brute-force attacks** on Modbus registers by limiting the number of commands per second. Not Yet, it is easy and will be implemented after we drive the inverter successfully.

**✅ ESP32 Firmware Security**

* **Secure Boot**: Ensures that only signed firmware can run on the ESP32 to prevent unauthorized modifications.  
  **Implemented**
* **OTA Updates with Authentication**: Over-the-Air updates require **authentication and integrity checks**.  
  **Implemented**
* **Watchdog Timer (WDT)**: Prevents firmware crashes or infinite loops caused by malicious commands.  
  **Implemented**
* **AES Encryption**: Data stored in ESP32’s flash memory is encrypted using **AES-256**.  
  **Implemented**

**✅ SIM7600 Communication Security**

* **APN Locking**: The SIM7600 is configured to only connect to a **specific APN**, preventing unauthorized network access.  
  **Implemented**
* **IMEI Locking**: The LTE module is locked to a specific **device IMEI** for added security.  
  **Implemented**

**2.2. Hardware Security Measures   
(All of these can to be implemented when I design the hardware)**

**✅ Physical Security Protections**

* **Secure Enclosure**: The ESP32 and SIM7600 hardware are housed in a **tamper-proof case** to prevent unauthorized physical access.  
  (Mechanical engineer)
* **Electromagnetic Shielding**: To prevent **RF signal sniffing**, a shielded casing is used.  
  (Mechanical engineer)
* **Hardware Watchdog Circuit**: A **hardware reset circuit** ensures the ESP32 reboots if abnormal behavior is detected. (Electronic engineer)

**✅ RS485 (Modbus RTU) Security**

* **Optocouplers for Isolation**: The RS485 interface is isolated using **optocouplers** to prevent electrical tampering. (Electronic engineer)
* **Surge Protection**: The RS485 line includes **TVS diodes** for surge and transient voltage protection. (Electronic engineer)

**3. Conclusion**

Implementing a combination of **software and hardware security measures** ensures that the ESP32 + SIM7600 system is resilient to unauthorized access and cyber threats. Continuous monitoring, encryption, and access control mechanisms are essential to maintaining a **high level of security** in IoT applications.

**4️ Additional Security Hardening**

**✅ Enable Secure Boot**

* Prevent **unsigned firmware** from running.
* Enable Secure Boot in **ESP32 eFuse settings**.

**✅ Use a Hardware Secure Element**

* Store **AES keys, RSA keys, and SSL certificates** in a secure element like **ATECC608A**.

**✅ Disable Debugging Interfaces**

* Disable **JTAG, UART, and Serial Monitor** in production.

**✅ Implement Rate-Limiting on MQTT Broker**

* Prevent brute-force attacks by setting **failed login lockouts**.