MQTT com TLS no ESP32

Valide o certificado do broker e pare de usar setInsecure - manual prático

Placa	ESP32 ou ESP32-S3
Framework	Arduino Core, via PlatformIO
Bibliotecas	PubSubClient (única dependência externa) e WiFiClientSecure (do núcleo)
Broker	Eclipse Mosquitto 2.x, porta 8883
Ferramenta no computador	OpenSSL
Nível	Intermediário
Tutorial online	iiot.esp32.tec.br/tutoriais/mqtt-com-tls-no-esp32

Sobre a validação. O código deste pacote segue a API do WiFiClientSecure e do PubSubClient do núcleo Arduino-ESP32, mas **não foi compilado nem testado em placa** ao escrever o kit. Compile no seu ambiente e rode os testes negativos da seção 5 antes de usar em instalação que importe.

1. A criptografia que não protege

O exemplo mais copiado da internet chama **setInsecure()** no WiFiClientSecure. A conexão na porta 8883 passa e o tráfego sai cifrado, mas o ESP32 aceita o certificado de **qualquer** servidor que responda no endereço. Quem estiver no meio do caminho apresenta um certificado próprio e recebe o usuário e a senha do MQTT.

Propriedade	O que significa	setInsecure()	setCACert()
Sigilo	Ninguém lê o tráfego	Sim	Sim
Integridade	Ninguém altera no caminho	Sim	Sim
Autenticação do servidor	Você fala mesmo com o seu broker	**Não**	Sim

A terceira linha é a que impede o ataque de homem no meio, e é a única que o setInsecure() descarta.

2. As quatro armadilhas

Armadilha	Sintoma	Saída
Relógio em 1970	Todo certificado parece 'ainda não válido'; handshake falha	Sincronizar NTP antes do TLS; o firmware não tenta sem hora válida
Nome do broker	Falha de verificação ao conectar por IP	Usar nome DNS e colocá-lo no certificado (subjectAltName)
Memória	rc=-2 sem motivo claro, heap baixo	Medir ESP.getFreeHeap() antes e depois; deixar margem
O que gravar	Firmware para de conectar quando o certificado do broker é renovado	Gravar a CA, não o certificado do servidor; planejar a troca da CA via OTA

3. CA e certificado do broker

Numa rede interna, uma CA privada resolve e não custa nada. O script **certs/gerar-certificados.sh** executa os comandos abaixo, com o nome do broker como argumento:

```
openssl genrsa -out ca.key 4096
openssl req -x509 -new -nodes -key ca.key -sha256 -days 3650 -subj "/CN=IIoT Lab CA"
-out ca.crt
openssl genrsa -out broker.key 2048
openssl req -new -key broker.key -subj "/CN=broker.lab.local" -out broker.csr
printf "subjectAltName=DNS:broker.lab.local" > san.ext
openssl x509 -req -in broker.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out
broker.crt -days 825 -sha256 -extfile san.ext
```

Arquivo	Para onde vai	Sensível?
ca.crt	Firmware (include/ca_cert.h) e broker	Não, é público
broker.crt	Broker	Não
broker.key	Broker, legível só pelo usuário do Mosquitto	**Sim**
ca.key	Guardar offline. Nunca no broker nem em repositório	**Sim, o mais crítico**

4. Mosquitto e prova do certificado

Aplique o arquivo **mosquitto/tls.conf**:

```
listener 8883
cafile /etc/mosquitto/certs/ca.crt
certfile /etc/mosquitto/certs/broker.crt
keyfile /etc/mosquitto/certs/broker.key
allow_anonymous false
password_file /etc/mosquitto/passwd
listener 1883 127.0.0.1
```

Prove o certificado **antes** de tocar no ESP32:

```
openssl s_client -connect broker.lab.local:8883 -CAfile ca.crt
```

No fim da saída deve aparecer **Verify return code: 0 (ok)**. Outro código significa problema no certificado ou no nome, e o ESP32 só repetiria o erro com menos informação. Teste também a autenticação:

```
mosquitto_pub -h broker.lab.local -p 8883 --cafile ca.crt -u esp32-lab -P SENHA_MQTT -t
lab/teste -m ok
```

5. Firmware e testes negativos

O **src/main.cpp** segue o padrão de guarda do artigo de reconexão MQTT: a função de conexão é chamada a cada volta do loop() e decide sozinha. Há uma guarda a mais, a da hora válida. Em caso de falha, o log mostra o rc, o texto do mbedTLS (lastError) e o heap livre.

```
if (mqtt.connected() || WiFi.status() != WL_CONNECTED) return;
if (time(nullptr) < MIN_VALID_EPOCH) return;
if (millis() - lastAttemptMs < RETRY_MS) return;
```

> O handshake TLS é bloqueante e pode levar de centenas de milissegundos a alguns segundos. Se houver tarefa crítica de tempo, rode o MQTT em uma tarefa separada do FreeRTOS e confira se o watchdog tolera o tempo.

Os três testes negativos

Se a validação de verdade está ligada, ela precisa reprovar o que não é seu. Faça os três; todos devem **falhar**:

Teste	Resultado esperado
Trocar o host do firmware por um IP que não está no certificado	Falha de verificação do certificado
Alterar uma letra do conteúdo de ca.crt em include/ca_cert.h	Falha de verificação do certificado
Bloquear o NTP (porta 123 no roteador) e reiniciar	Nenhuma tentativa de conexão; nunca conecta com a hora de 1970

> Se algum dos três conectar, a validação não está ativa.

6. Quando algo não funciona

Sintoma	Causa provável	O que fazer
rc=-2 e erro X.509 em lastError()	CA errada, nome não bate ou certificado vencido	Repetir o openssl s_client
rc=-2 sem texto e heap baixo	Falta memória para o handshake	Tirar bibliotecas, reduzir buffers, mais RAM
Nenhuma tentativa no Serial	Guarda da hora barrando	Conferir o acesso ao NTP ou usar NTP local
Funciona na bancada, falha no cliente	DNS do broker não resolve ou porta 8883 bloqueada	Conferir DNS e firewall
Funcionou por meses e parou	Certificado do broker ou da CA vencido	Renovar no servidor; a CA igual não exige tocar nos dispositivos

7. O que este kit não cobre

- **TLS mútuo** (certificado por dispositivo): require_certificate e use_identity_as_username no Mosquitto; setCertificate() e setPrivateKey() no ESP32.

- **Permissões por tópico** (ACL do broker): TLS prova quem conecta, não limita o que publica.

- **Proteção da senha no dispositivo**: criptografia da flash e boot seguro do ESP32.

8. Checklist

	Item
[ ]	ca.key guardada offline, fora do broker e de repositórios
[ ]	Certificado do broker com o nome DNS no subjectAltName
[ ]	openssl s_client terminou com Verify return code: 0 (ok)
[ ]	mosquitto_pub pela 8883 com a CA funcionou
[ ]	ca.crt colado em include/ca_cert.h; senhas de exemplo trocadas
[ ]	Conectou e lab/esp32/status virou online
[ ]	Os três testes negativos falharam como esperado
[ ]	Heap livre medido com a aplicação completa
[ ]	Data de validade da CA e do certificado do broker anotada, com plano de troca