MQTT com reconexão automática

O broker cai, o firmware continua - manual prático



Placa	ESP32 ou ESP32-S3
Framework	Arduino Core, via PlatformIO
Biblioteca	PubSubClient (única dependência externa)
Broker	Mosquitto local, HiveMQ ou equivalente
Nível	Intermediário
Tutorial online	iiot.esp32.tec.br/tutoriais/reconexao-automatica-do-mqtt

Sobre a validação. O padrão de guarda e as cadências reproduzem o que roda no firmware do AnalictY Device Studio. O Last Will é um acréscimo que aquele firmware ainda não usa. O código deste pacote não foi validado em bancada contra um broker real: teste no seu ambiente antes de usar em instalação que importe.

1. O laço que você não deve usar

Procure por reconexão MQTT no ESP32 e você encontra alguma variação disto:

```
while (!client.connected()) {
if (client.connect("esp32")) {
Serial.println("conectado");
} else {
delay(5000);
}
}
```

Funciona na bancada, com o broker no mesmo computador. Em campo é uma armadilha.

Enquanto esse laço roda, o firmware inteiro está parado. O servidor web não responde, o botão físico não é lido, o Modbus não é consultado, a máquina de estados do processo não avança. Se o broker ficar fora do ar por dez minutos, o dispositivo fica dez minutos **morto** - não desconectado, morto.

> Pior: em muitos projetos isso alimenta o watchdog de forma irregular e o ESP32 reinicia sozinho. O diagnóstico vira 'o dispositivo está reiniciando', e ninguém liga o reset a queda do broker.

A inversão: guarda em vez de laço

O padrão correto não tem laço. A função é chamada a cada volta do loop e decide sozinha se deve agir:

```
if (mqtt.connected() || WiFi.status() != WL_CONNECTED) {
return;
}
if (millis() - ultimaTentativaMqtt < INTERVALO_MQTT_MS) {
return;
}
ultimaTentativaMqtt = millis();
```

A primeira guarda desiste em dois casos: já conectado, ou Wi-Fi caído. Tentar MQTT sem rede é desperdício garantido - a chamada bloqueia até estourar o tempo limite do socket, por nada.

A segunda é o coração do padrão. Se a última tentativa foi há menos de cinco segundos, volta imediatamente. É isso que substitui o delay(5000) sem parar o firmware: em vez de dormir, a função retorna em microssegundos e o loop segue fazendo tudo o mais.

2. Cadências, ordem e keepalive

Custos diferentes, intervalos diferentes

Operação	Intervalo	Por que
Wi-Fi	30 s	liga radio, varre, negocia autenticação e espera DHCP: caro em tempo e energia
MQTT	5 s	handshake TCP sobre rede que já está de pe: barato, e cada segundo fora é telemetria perdida

A regra geral: quanto mais cara a operação, maior o intervalo entre tentativas.

A ordem no loop

```
void loop() {
garantirWifi();
conectarMqtt();
if (mqtt.connected()) {
mqtt.loop();
}
unsigned long agora = millis();
if (agora - ultimaTelemetria >= INTERVALO_TELEMETRIA_MS) {
ultimaTelemetria = agora;
publicarTelemetria();
}
}
```

O Wi-Fi vem antes do MQTT porque o segundo depende do primeiro. É repare que não existe nenhum delay no laço inteiro: todo o agendamento é por comparação de millis().

mqtt.loop() é obrigatório

É ele que processa mensagens recebidas e, principalmente, responde ao keepalive do broker. Sem chama-lo com regularidade, o broker considera o cliente morto e derruba a conexão - produzindo um ciclo de reconexão que parece problema de rede e é problema de código.

3. Client id e Last Will

O client id duplicado é o falso diagnóstico

O protocolo exige identificador único por cliente. Se dois dispositivos conectam com o mesmo id, **o broker derruba o primeiro** para aceitar o segundo. O primeiro reconecta e derruba o segundo. E assim indefinidamente.

O resultado é um par de dispositivos conectando e caindo em ciclo, sem erro nos logs além de desconexões. É facilmente confundido com Wi-Fi instável.

> O firmware deste pacote deriva o id do MAC, que é único de fábrica. Se você usa um literal como "esp32" - o valor que aparece em quase todo exemplo - o problema surge no dia em que a segunda placa entrar em operação.

Last Will: avisar que morreu

Há um caso que a reconexão não resolve: o dispositivo perde energia. Ele não tem como avisar ninguém, e o sistema do outro lado continua exibindo o último valor recebido como se fosse atual.

O Last Will and Testament é a mensagem que o broker publica **por você** se a conexão cair sem despedida:

```
mqtt.connect(clienteId, usuario, senha,
topicoStatus, 1, true, "offline");
mqtt.publish(topicoStatus, "online", true);
```

Os parâmetros extras são o tópico do testamento, a qualidade de serviço, o flag de retenção e a mensagem. O efeito é um tópico de presença confiável: enquanto o dispositivo estiver vivo ele diz online; se sumir, o próprio broker publica offline depois do keepalive expirar.

O painel do outro lado passa a distinguir dado velho de dispositivo caído - uma distinção que salva plantão.

4. Os códigos rc

Quando a conexão falha, mqtt.state() devolve um número. É a informação mais útil do diagnóstico, e quase todo tutorial imprime sem explicar. A leitura prática: negativos são problema de rede ou endereço; positivos são recusa do broker, que recebeu sua conexão e disse não.

rc	Significado	Causa típica
-4	Tempo limite estourado	broker inalcançável, IP ou porta errados
-3	Conexão perdida	a rede caiu no meio da operação
-2	Falha na conexão	DNS não resolveu ou TCP recusado
-1	Desconectado localmente	chamada de disconnect() no próprio código
0	Conectado	-
1	Protocolo recusado	versão incompatível com o broker
2	Client id rejeitado	caractere invalido ou identificador longo demais
3	Broker indisponível	serviço no ar, mas recusando conexões
4	Credencial incorreta	usuário, senha ou token expirado
5	Não autorizado	credencial válida, sem permissão no tópico

5. O teste que realmente importa

Conectar quando tudo está bem é o caso fácil. O que separa firmware resiliente de firmware de bancada é o comportamento na falha.

Derrubar o broker

Com tudo funcionando, pare o broker. O comportamento correto é: o dispositivo detecta a queda, passa a tentar de cinco em cinco segundos, imprime o rc de cada tentativa, e **continua fazendo todo o resto**. Se o firmware travar aqui, o padrão de guarda não está aplicado.

Subir o broker de volta

A reconexão deve acontecer em até cinco segundos, sem intervenção e sem reiniciar a placa.

Derrubar o Wi-Fi

Desligue o roteador. A guarda de WiFi.status() deve impedir tentativas de MQTT enquanto não houver rede: você não deve ver rc negativos em sequência, e sim silêncio do lado MQTT enquanto o Wi-Fi tenta a cada trinta segundos. Ao religar, a sequência correta é Wi-Fi primeiro, MQTT logo depois.

Verificar o keepalive

Deixe rodando vinte minutos sem publicar nada. Se a conexão cair sozinha nesse intervalo, o mqtt.loop() não está sendo chamado com regularidade - procure por algum delay longo no laço.

6. Solução de problemas

Sintoma	Causa provável	O que fazer
Trava quando o broker cai	Laço while ou delay na conexão	Aplique o padrão de guarda da seção 1.
Reconecta e cai em ciclo	Client id duplicado	Confirme identificador diferente em cada placa.
Cai sozinho após alguns minutos	Keepalive sem resposta	Chame mqtt.loop() com regularidade; remova delays longos.
rc=-2 constante	Nome do broker não resolve	Teste com o IP direto para separar DNS de endereço.
rc=4 ou rc=5	Credencial	Em broker com token, verifique validade antes do código.
Publica mas não recebe	Faltou reinscrever após reconectar	A inscrição não sobrevive a queda: subscribe dentro do sucesso do connect.
Falha ao publicar carga grande	Buffer do PubSubClient	O padrão é 256 bytes. Use setBufferSize para aumentar.
Funciona com um, falha com varios	Client id ou limite do broker	Além do id, confira o limite de conexões configurado no broker.

7. Checklist final

	Item
[ ]	Nenhum laço while de espera na rotina de conexão
[ ]	Nenhum delay() no loop principal
[ ]	Guarda de WiFi.status() antes de tentar MQTT
[ ]	Intervalos separados: Wi-Fi mais longo que MQTT
[ ]	Client id único por dispositivo, derivado do MAC
[ ]	mqtt.loop() chamado a cada volta, quando conectado
[ ]	subscribe dentro do bloco de sucesso do connect
[ ]	Last Will registrado no connect, com retenção
[ ]	Publicação de online logo após conectar
[ ]	Teste com broker derrubado: firmware seguiu funcionando
[ ]	Teste de religar broker: reconectou em até 5 segundos
[ ]	Teste com Wi-Fi derrubado: sem rc negativos em sequência
[ ]	Teste de 20 minutos ocioso: conexão permaneceu de pe