**AULA 8 – MQTT**

**REDES**

**Message Queuing Telemetry Transport**

**Por que precisamos usá-lo?**

* HTTP é pesado para sensores
* Muitos dispositivos em IoT têm:
  + Pouco processamento e memória
  + Conexões intermitentes (Wi-Fi fraco, GPRS)
* É necessário algo leve, eficiente e tolerante a falhas.

O MQTT é uma alternativa para HTTP, usado em dispositivos mais fracos, ou com conexões fracas, pois é bem mais leve do que o HTTP. Os dispositivos não aguentariam processar os dados enviados por HTTP, apenas por MQTT.

**O que é?**

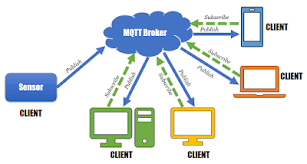
* Message Queuing Telemetry Transport
* Protocolo assíncrono baseado em publicação e assinatura (pub/sub) – Quando uma mensagem (informação) é enviada, existe um transmissor (quem envia) e o receptor – chamado de assinante - (quem recebe), um protocolo assíncrono, permite que os dois não estejam ligados entre si, assim, permitindo que o receptor receba a mensagem depois que foi enviada. (Síncrona é quando o receptor recebe a mensagem na mesma hora em que ela foi transmitida, como em uma conversa).
* Roda sobre TCP/IP
* Usa um intermediário chamado Broker, que recebe as mensagens do transmissor e quando o assinante quiser, ele distribui as mensagens

**Arquitetura MQTT**

Três elementos principais:

* **Broker:** Servidor intermediário – distribui as mensagens
* **Publisher(Publicador; Transmissor):** Quem envia dados/mensagens para o tópico
* **Subscriber(Assinante;Receptor):** Quem recebe dados/mensagens assinando um tópico

Exemplo de funcionamento da arquitetura MQTT:

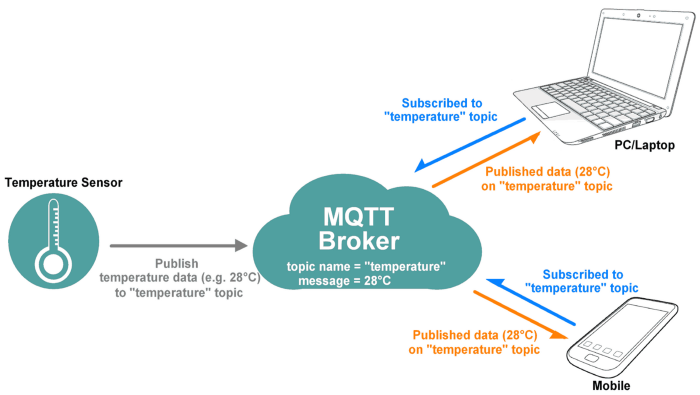


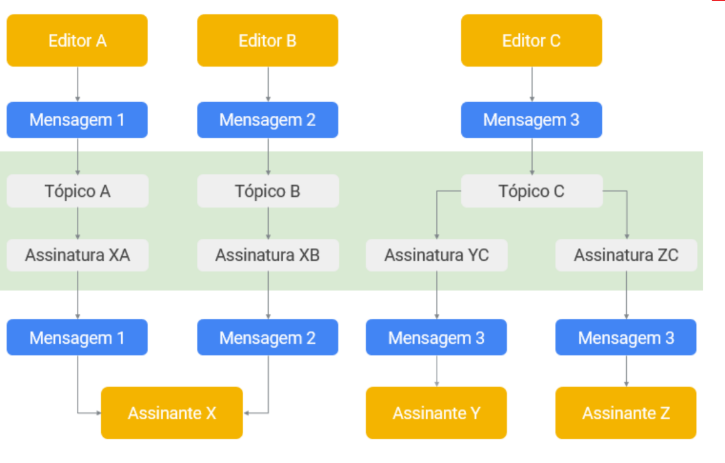
**Tópicos e Subtópicos no MQTT**

* Tópicos são canais de comunicação, ex: casa/sala/luz
* Hierarquia com “/” para organizar por local ou função
* Suporte a curingas:
  + + → um nível (ex: casa/+/luz)
  + # → todos os níveis seguintes (ex: casa/#)
    - Exemplo: SENAI/sala1/ar-condicionado -> Temp.; Modo; Movimento... SENAI/+/ar-condicionado

Quando um dado/mensagem é enviado, não é todo o Broker que recebe ele, mas apenas um tópico específico, como o tópico “Temperatura”. O assinante escolhe um tópico para receber mensagens.

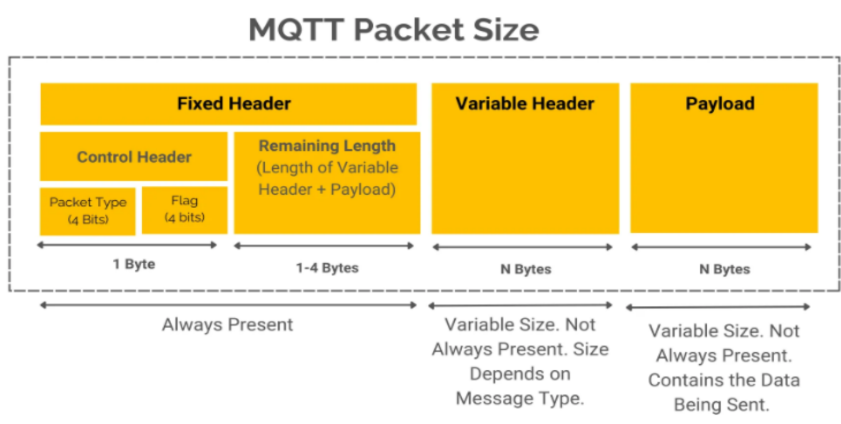
Exemplos:





**Por que o MQTT é ideal para IoT?**

* **Baixa sobrecarga de rede:** Mensagens simples, sem cabeçalhos pesados.
* **Baixo consumo de energia:** Dispositivos dormem e acordam para enviar dados.
* **Tolerância a falhas:** QoS, persistência e reconexão automática.
* **Escalável:** Milhares de sensores podem se comunicar com um único broker.

Cabeçalho do MQTT (Bem leve)

**Casos reais de uso do MQTT:**

* **Agricultura inteligente:** Sensores de solo publicam umidade para análise.
* **Casa inteligente:** Botões publicam comando casa/luz/sala 🡪 Ligado.
* **Monitoramento industrial:** Máquinas publicam alertas de falhas.
* **Veículos conectados:** Rastreadores publicam localização constantemente.

**Qualidade de Serviço (QoS)**

* MQTT têm 3 níveis de entrega:
  + **QoS 0:** Entrega única (sem confirmação)
  + **QoS 1:** Garante pelo menos uma entrega
  + **QoS 2:** Garante entrega **uma única vez**, sem duplicação
* Mais confiável = mais sobrecarga

**Persistência e sessões**

* **Sessões podem ser persistentes**: o broker guarda o estado do cliente
* Mensagens podem ser armazenadas até o cliente reconectar
* Uso do flag *retain* para manter última mensagem em um tópico

**Segurança no MQTT**

Por padrão, o MQTT não é criptografado, mas ele pode rodar sobre TLS

(como o HTTPS) para garantir confidencialidade e integridade dos dados.

Além disso você também pode:

* Autenticação por usuário e senha
* Controle de acesso por tópico

**Brokers MQTT**

* **Mosquitto:** Leve e Open-Source.
* **HiveMQ:** Profissional, possui painel e plugin.
* **EMQX:** Alta performance, escalável.
* **Brokers em nuvem:** AWS IoT, Azure IoT, Google Cloud IoT.

**Vantagens e Limitações do MQTT**

**Vantagens:**

* Baixo consumo de energia e dados.
* Escalável e desacoplado.
* Fácil de implementar.

**Limitações:**

* Funciona apenas com TCP/IP.
* Sem criptografia nativa (requer TLS).
* Simples demais para fluxos complexos.