



PET ENG.COMP.
**TECH
TALK**

Protocolo MQTT

Desenvolvendo seu Proprio Sistema IOT com MQTT, Azure Cloud e ESP32.



Temas Teóricos

- Introdução
- Criação
- Comparação MQTT vs HTTP
- Aplicações

- Arquitetura e Funcionamento
- Tópicos
- Quality Of Service (QOS)
- Segurança



Temas Práticos

- Criação da conta de Estudante na Azure
- Criar e configurar a Máquina Virtual
- Instalar o Broker Mosquitto
- Testando o Broker

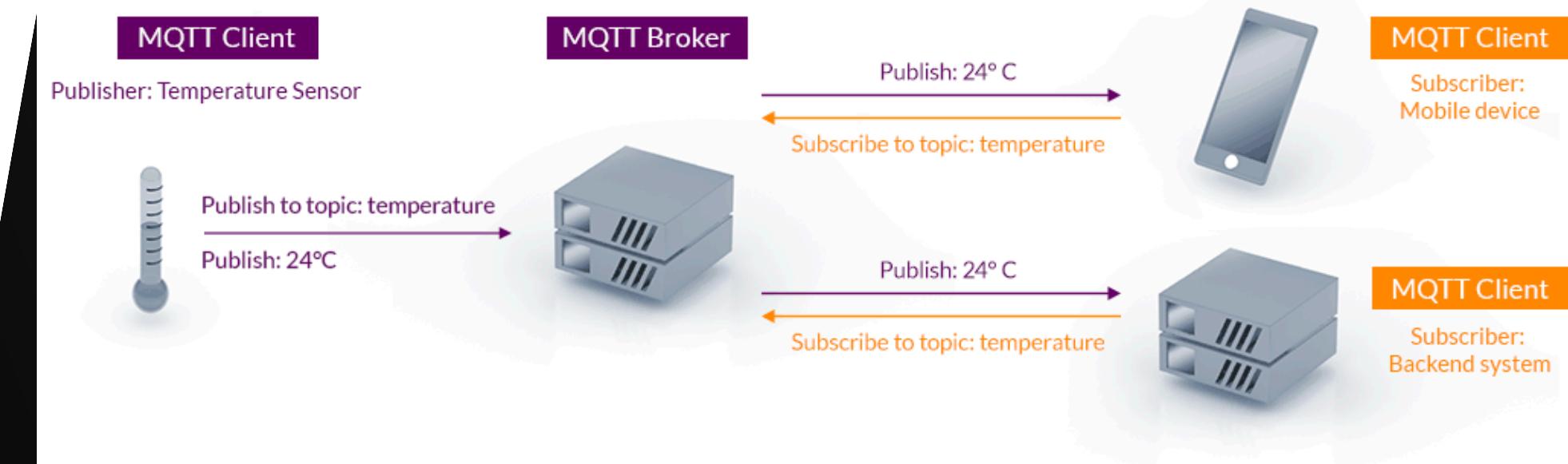
- Configurando o Aplicativo MQTT Dashboard no Celular
- Visão Geral do código ESP32
- Demonstração de Funcionamento





Message Queuing Telemetry Transport (Transporte de fila de mensagens de telemetria)

- Leve, Confiável e Escalável
- Voltado para M2M (Machine to Machine)
- Usa protocolo TCP/IP
- Presença de um servidor central (Broker)
- Baseado em Tópicos
- Paradigma Publish/Subscribe
- Baseada em Eventos

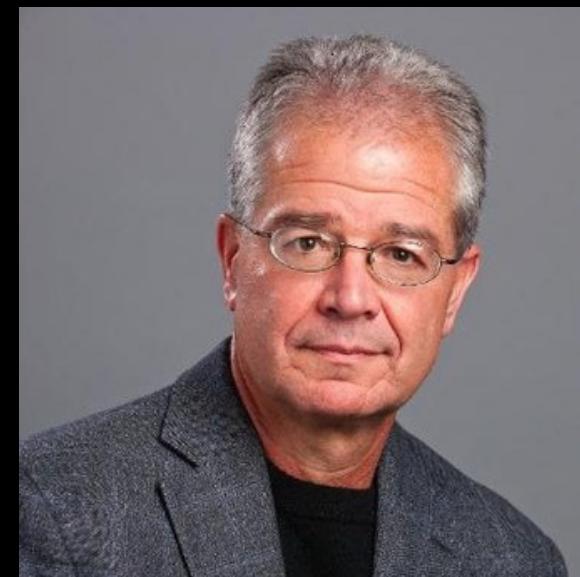




A criação do MQTT



[Andy Stanford-Clark](#)



[Arlen Nipper](#)



Linha do tempo



2010 – Open Sourcing

Protocolo liberado ao público sob a Licença Pública Eclipse.

2013 – Estandardização OASIS

A manutenção e desenvolvimento do MQTT agora é de responsabilidade da Organização para o Avanço de Padrões de Informação Estruturada.

2014 – Versão MQTT 3.1.1

Primeira grande versão do protocolo, em que ele foi melhor documentado e feito mais robusto e de fácil implementação.

2019 – MQTT Versão 5.0

Outra grande versão do protocolo onde foram adicionados como relatório de erros, subscrição compartilhada e expiração de mensagens.



HTTP VS MQTT

Características	HTTP	MQTT
Modelo de Comunicação	Requisição - Resposta (via URLs)	Publicação - Assinatura (via Tópicos)
Distribuição padrão de mensagens	Um para Um	Um para Vários
Uso de Recursos e Overhead	Alto (Cabeçalhos grandes e handshakes custosos)	Baixo (Cabeçalhos reduzidos e comunicação binária)
Consumo de Banda e Latência	Alta (a cada requisição geralmente é aberta uma nova conexão TCP)	Baixa (Conexão TCP persistente)
Persistência de dados	Não (precisa implementar)	Sim (Retenção de mensagens)
Garantia do recebimento de dados	Não (precisa implementar)	Sim (QoS 1 e 2)
Monitoramento de clientes	Não	Sim
Segurança	Sim (TLS/SSL, mas necessitaria implementar Autenticação)	Sim (TSL/SSL e Autenticação)
Escalabilidade	Escalável (mas necessitaria de diversas otimizações)	Escalável (Graças ao modelo de comunicação e broker)
Comunicação de dados grandes	Sim (Limitação na ordem de Gigabytes)	Não (Mensagem MQTT limitada a 256MB)
Ideal para	Dados de texto e arquivos grandes em menores frequências	Dados pequenos e binários em maiores frequências
Exemplos de Uso	Aplicações Web, APIs e Transferência de Arquivos	Aplicações M2M, IoT e Telemetria



Aplicações



Aplicações – Desenvolvimento

C

- [Eclipse Paho C](#)
- [Eclipse Paho Embedded C](#)
- [libmosquitto](#)
- [libemqtt](#) - an embedded C client
- [MQTT-C](#) - A portable MQTT C client for embedded systems and PCs alike.
- [wolfMQTT](#) - Embedded C client
- [SharkMQTT](#) - Embedded C client - [more information](#) - [documentation](#)
- [libcurl](#) - libcurl has basic support for publish and subscribe.
- [MQTT over lwIP](#) - MQTT C client for embedded systems using FreeRTOS, lwIP and mbedTLS
- [libsmartfactory](#) - easy to use library for different Smart Factory/Industry 4.0 technologies including a MQTT client implementation
- [libumqtt](#) - A Lightweight and fully asynchronous MQTT client C library based on libev
- [libmqttcli](#) - Easy to use, flexible and powerfull client library

Javascript / Node.js

- [Ascoltatori](#) - a node.js pub/sub library that allows access to Redis, AMQP, MQTT, and ZeroMQ with the same API.
- [Eclipse Paho HTML5 JavaScript over WebSocket](#).
- [IBM-provided PhoneGap / Apache Cordova MQTT plug-in for Android](#) - JavaScript API is identical to Eclipse Paho HTML5 JavaScript
- [mqtt.js](#)
- [node_mqtt_client](#)

Python

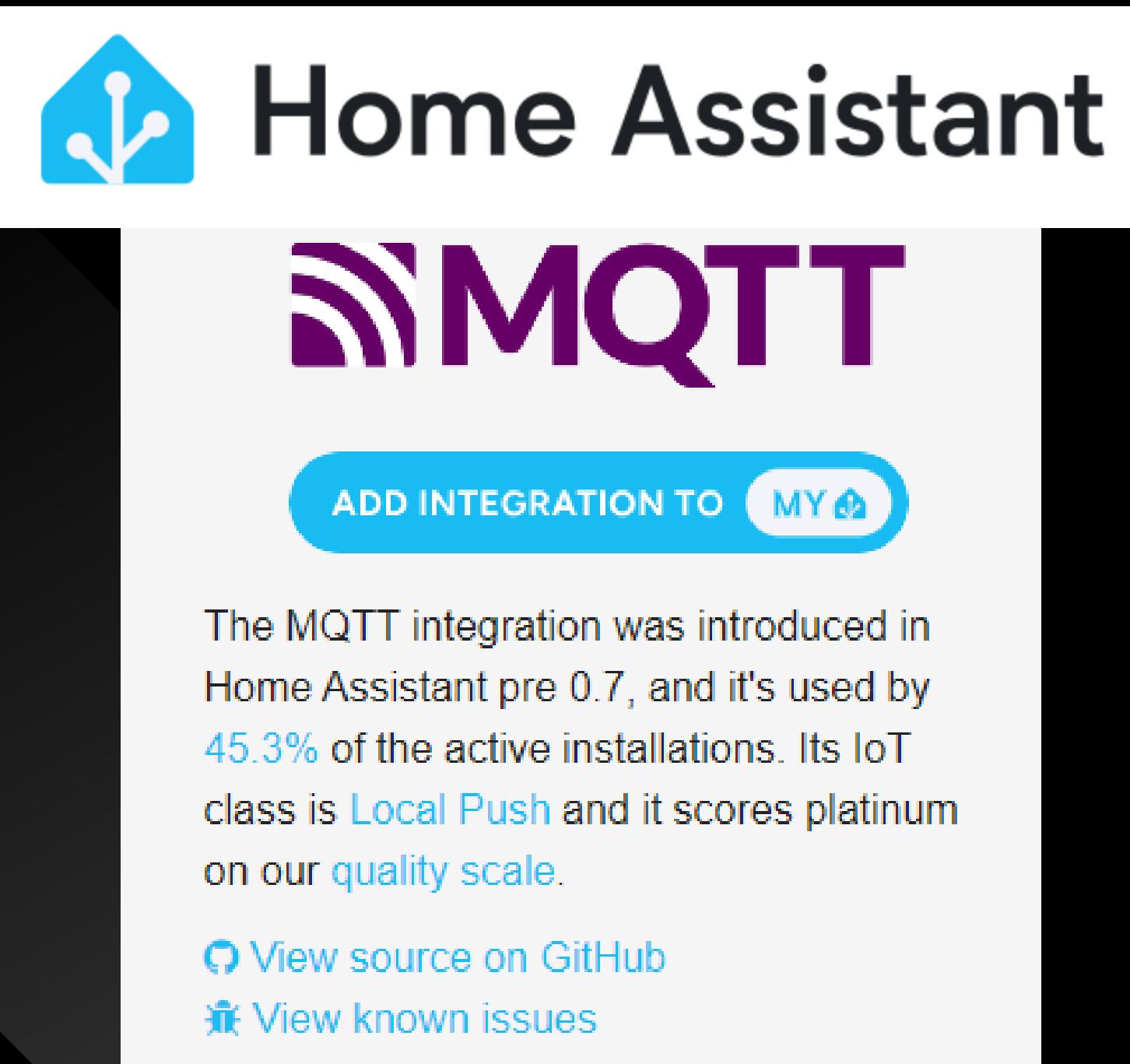
- [Eclipse Paho Python](#) - originally the mosquitto Python client
- [gmqtt](#)
- [nyamuk](#)
- [MQTT for twisted python](#)
- [HBMQTT](#)
- [mqttools](#)

Java

- [ActiveMQ Client](#)
- [Eclipse Paho Java](#)
- [Fusesource mqtt-client](#)
- "MA9B" zip of 1/2 dozen mobile clients source code. Includes Android-optimized Java source that works with Android notifications, based on Paho
- [MeQanTT](#)
- [moquette](#)
- [MqttWk](#)
- [HiveMQ MQTT Client](#) - MQTT 5.0 and MQTT 3.1.1 compatible and feature-rich high-performance Java client library with different API flavours and backpressure support
- [IA92](#) - deprecated IBM IA92 support pack, use Eclipse Paho GUI client instead. A useful MQTT Java swing GUI for publishing & subscribing. The Eclipse Paho GUI is identical but uses newer client code
- [Qatja](#) is a Java client library for MQTT 3.1.1 with specific implementation for Android and Processing
- [Sentienz Akiro MQTT Client](#) - MQTT 3.1.1 compatible [Akiro MQTT broker](#) Java client with callbacks.
- [vertx-mqtt-client](#) is an open-source, high performance, non-blocking MQTT client built as a part of vert.x's JVM toolkit.
- [Xenqtt](#) - [documentation](#) Includes a client library, mock broker for unit/integration testing, and applications to support enterprise needs like using a cluster of servers as a single client, an HTTP gateway, etc.
- [Micronaut MQTT](#) - integration between [Micronaut Framework](#) and MQTT.etc.



Aplicações – Smart Home



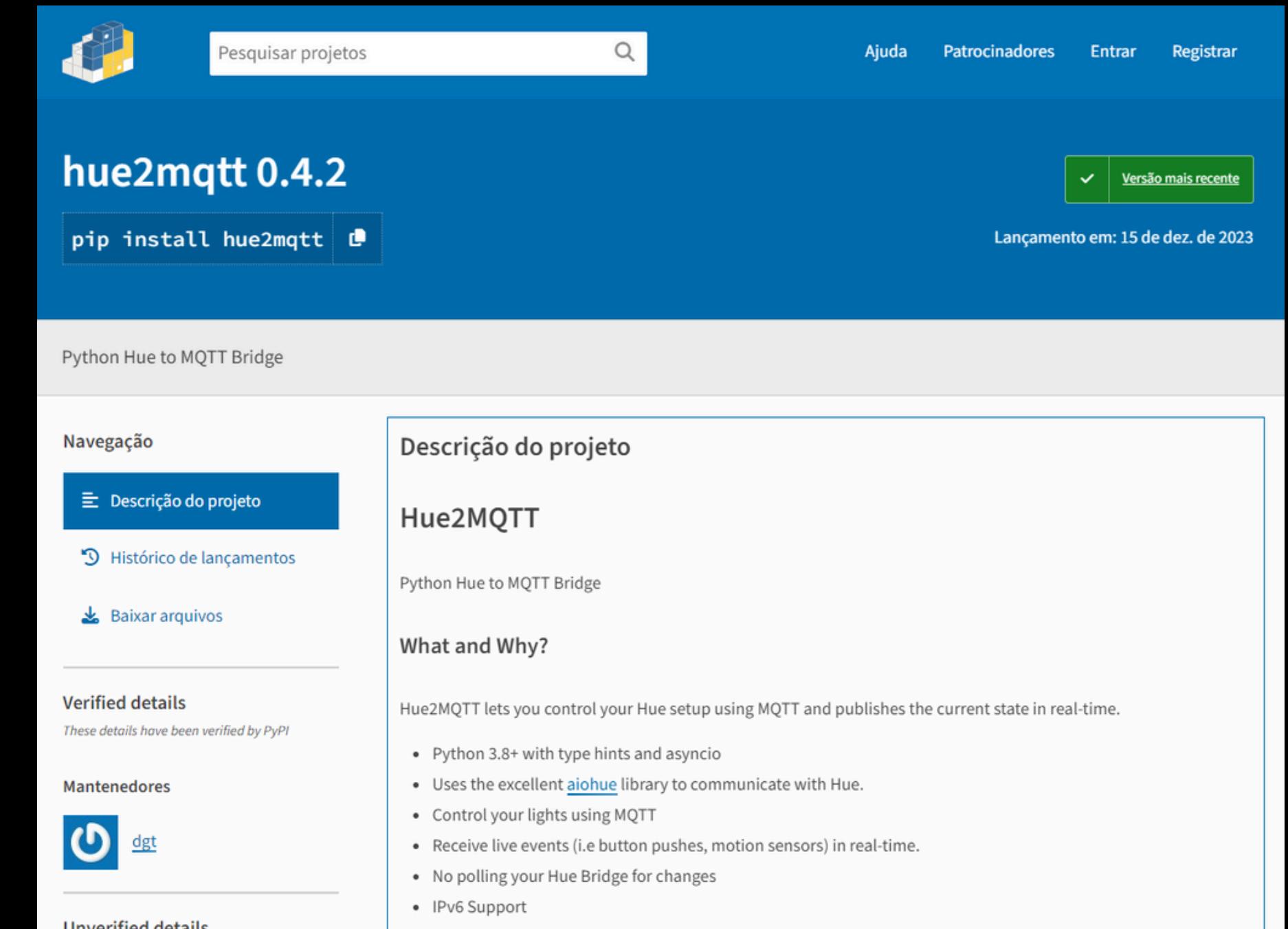
The screenshot shows the Home Assistant integration page for MQTT. At the top, there's a blue icon of a house with a tree inside. Below it, the text "Home Assistant" is displayed in large black font. Underneath, the "MQTT" logo is shown in purple. A prominent blue button at the bottom left says "ADD INTEGRATION TO MY HOME". To the right of this button, there's a small "MY HOME" icon. Below the button, a paragraph of text provides information about the integration, mentioning its introduction in pre 0.7, usage by 45.3% of active installations, and its IoT class being Local Push, with a platinum score on the quality scale. At the bottom, two links are provided: "View source on GitHub" and "View known issues".

The MQTT integration was introduced in Home Assistant pre 0.7, and it's used by 45.3% of the active installations. Its IoT class is Local Push and it scores platinum on our quality scale.

[View source on GitHub](#)

[View known issues](#)

www.home-assistant.io/integrations/mqtt/



The screenshot shows the PyPI project page for "hue2mqtt 0.4.2". At the top, there's a search bar with the placeholder "Pesquisar projetos" and a magnifying glass icon. The project name "hue2mqtt 0.4.2" is displayed prominently. Below the name, there's a button with the command "pip install hue2mqtt" and a link icon. To the right, there's a green button with a checkmark and the text "Versão mais recente". Further down, the text "Lançamento em: 15 de dez. de 2023" is visible. On the left, a sidebar titled "Navegação" contains links for "Descrição do projeto" (which is highlighted in blue), "Histórico de lançamentos", and "Baixar arquivos". Below this, sections for "Verified details" (with a note "These details have been verified by PyPI") and "Mantenedores" (listing "dgt" with a blue profile icon) are shown. On the right, a large section titled "Descrição do projeto" contains the project's title "Hue2MQTT", its description "Python Hue to MQTT Bridge", and a "What and Why?" section. The "What and Why?" section explains that Hue2MQTT lets you control your Hue setup using MQTT and publishes the current state in real-time, listing several bullet points about its features like Python 3.8+ support, aiohue library use, and IPv6 support.

hue2mqtt 0.4.2

`pip install hue2mqtt`

Lançamento em: 15 de dez. de 2023

Navegação

Descrição do projeto

Histórico de lançamentos

Baixar arquivos

Verified details

Mantenedores

Unverified details

Descrição do projeto

Hue2MQTT

Python Hue to MQTT Bridge

What and Why?

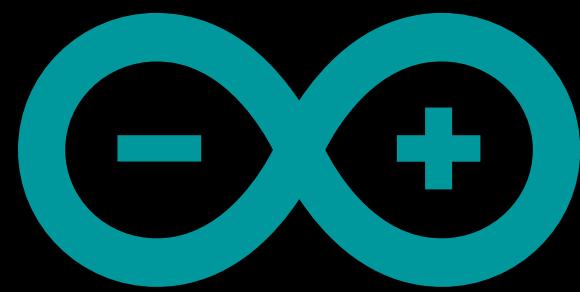
Hue2MQTT lets you control your Hue setup using MQTT and publishes the current state in real-time.

- Python 3.8+ with type hints and asyncio
- Uses the excellent [aiohue](#) library to communicate with Hue.
- Control your lights using MQTT
- Receive live events (i.e button pushes, motion sensors) in real-time.
- No polling your Hue Bridge for changes
- IPv6 Support

pypi.org/project/hue2mqtt
github.com/trickeydan/hue2mqtt-python



Aplicações - Automação DIY



CLOUD

Arduino IoT JS

The `arduino-iot-js` NPM module is designed for communicating with the Arduino Cloud broker using the MQTT over Websocket protocol. It is primarily used to send and receive variable values.

Example:

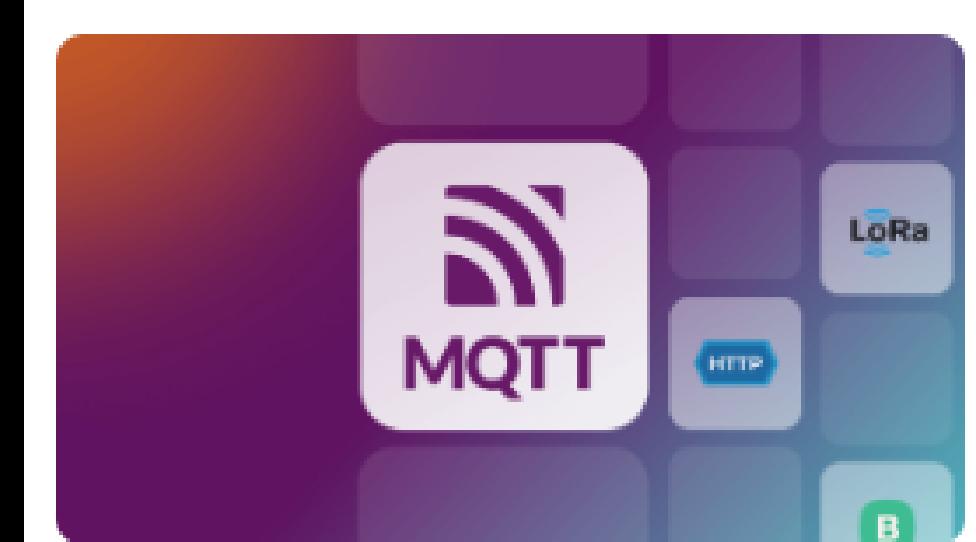
COPY

```
1 const { ArduinoIoTCloud } = require('@arduino/arduino-iot-js');
2
3 ArduinoIoTCloud.connect(options)
4   .then(() => {
5     console.log("Connected to Arduino Cloud");
6     return ArduinoIoTCloud.onPropertyValue(thingId, variableName, showUpdates = value =
7   })
8   .then(() => console.log("Callback registered"))
9   .catch(error => console.log(error));
```

<https://docs.arduino.cc/arduino-cloud/api/api-overview/>



Blynk



MARCH 28, 2024

Blynk Expands Protocol Range:
Introducing MQTT Support

<https://blynk.io/blog/blynk-expands-protocol-range-introducing-mqtt-support>



Aplicações – Automotivas



www.hivemq.com/case-studies/bmw-mobility-services/



Aplicações – Logística

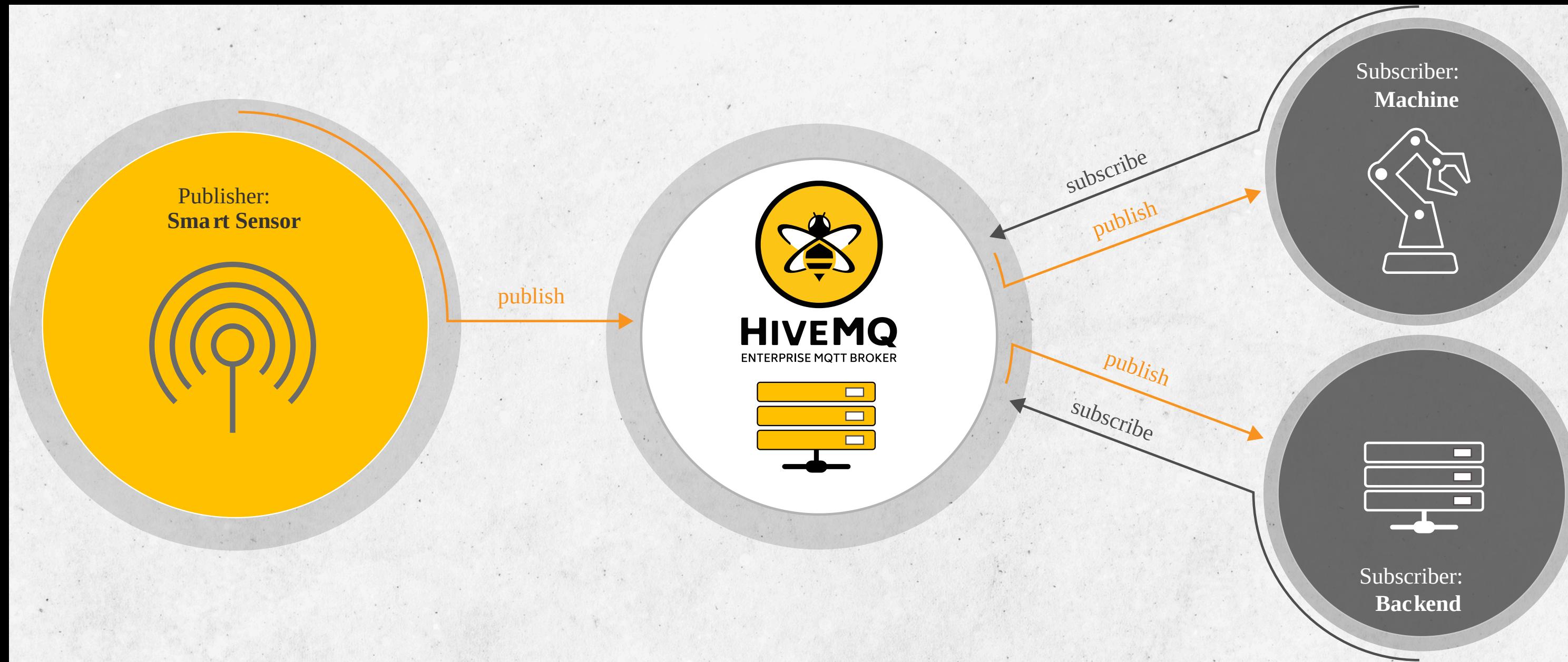


www.hivemq.com/case-studies/matternet

www.mttr.net



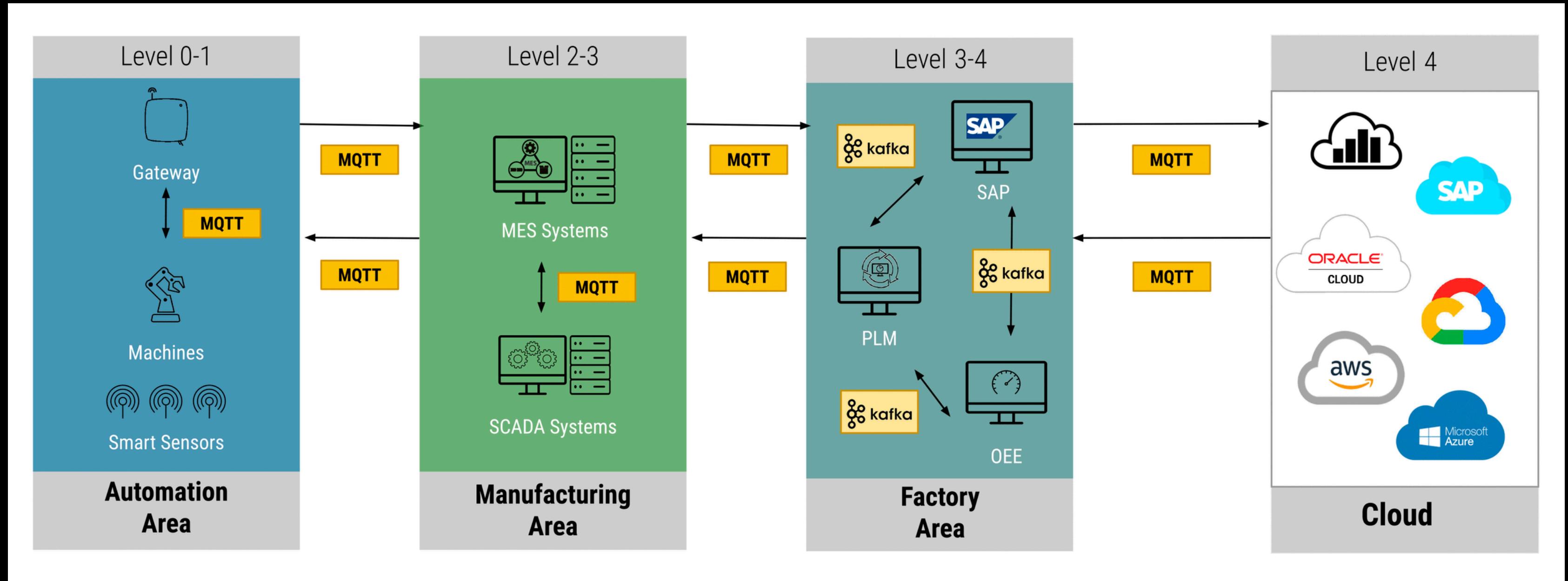
Aplicações – Indústria



<https://www.hivemq.com/resources/modernizing-the-manufacturing-industry/>



Aplicações - Indústria



<https://www.hivemq.com/resources/modernizing-the-manufacturing-industry/>

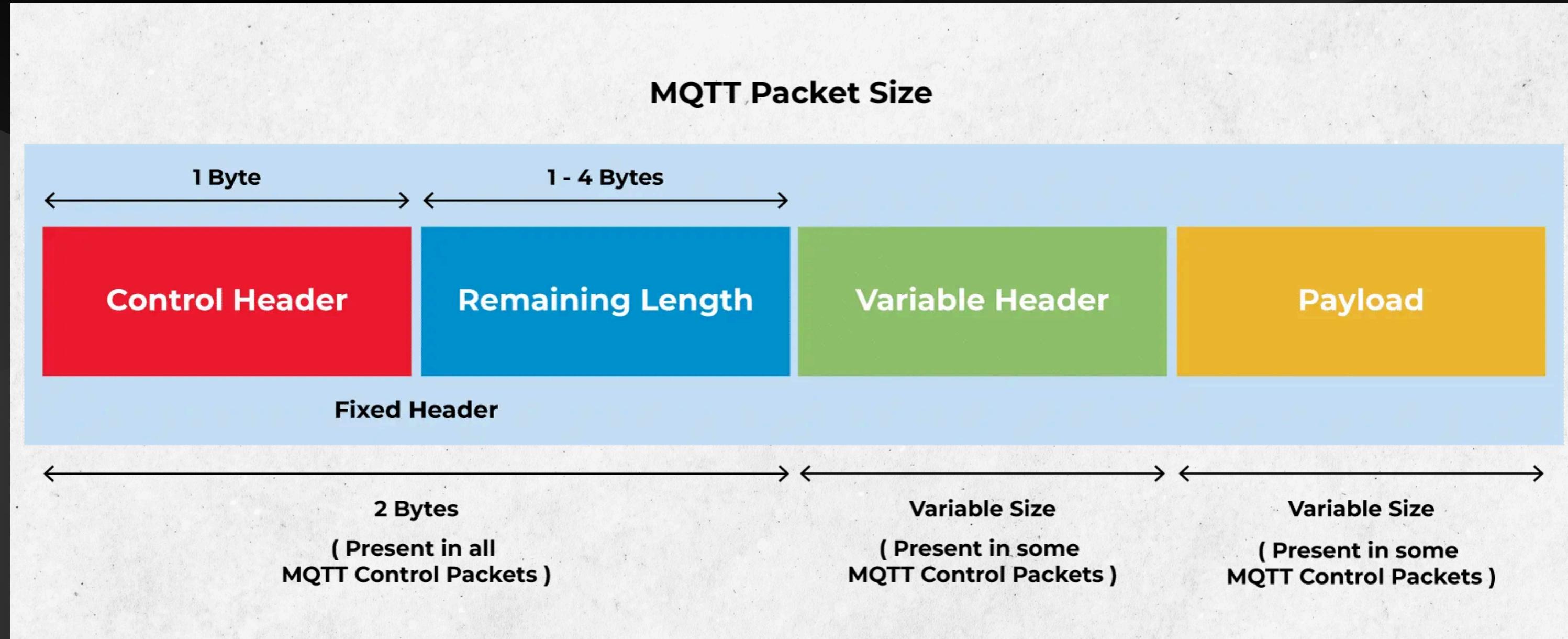


Arquitetura e Funcionamento



Arquitetura e Funcionamento

Um pacote MQTT é formado por:

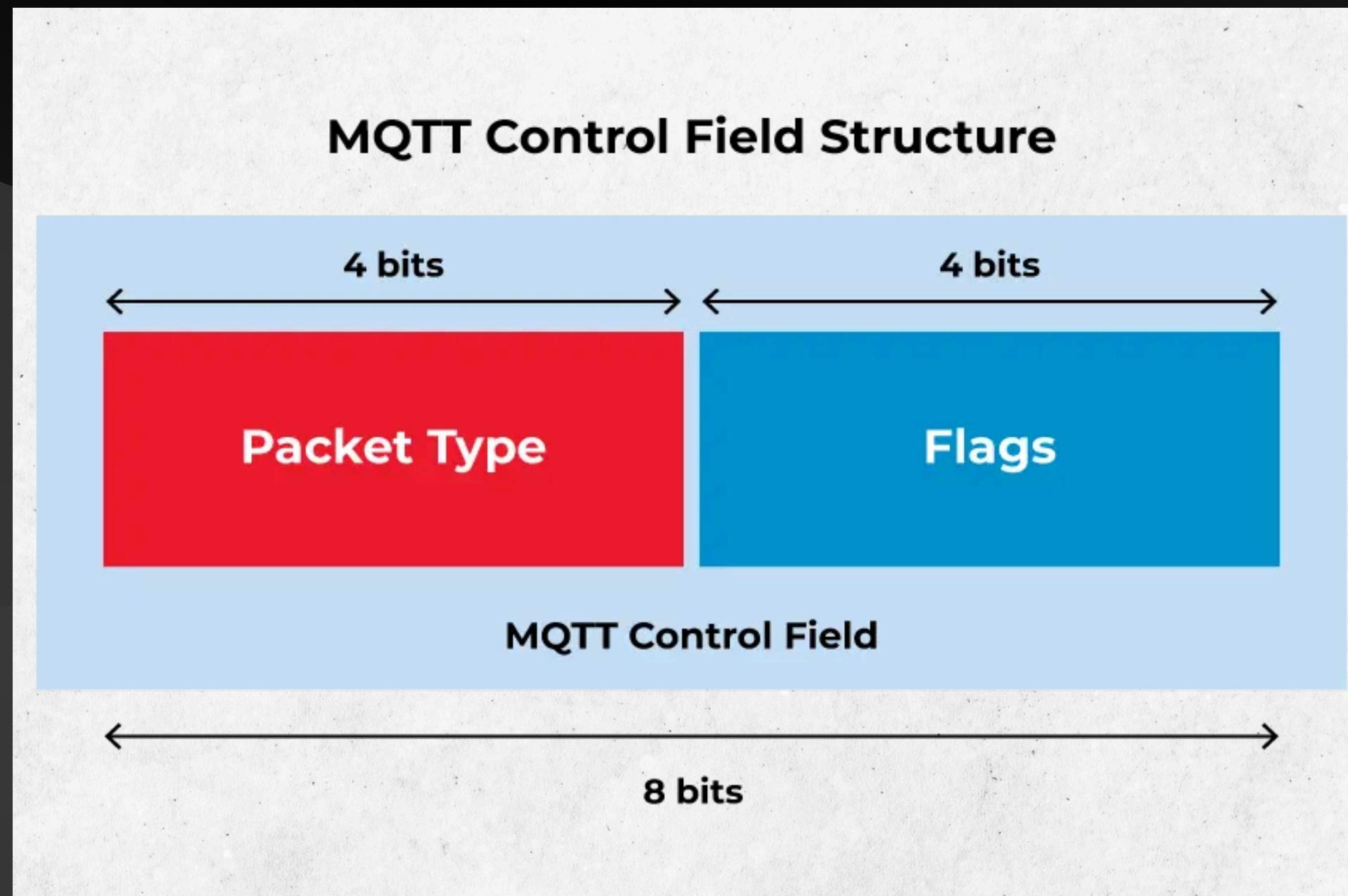


<https://www.bevywise.com/blog/understanding-mqtt-protocol-packet-format/>



Arquitetura e Funcionamento

Control Field:



Tipos de Pacote:

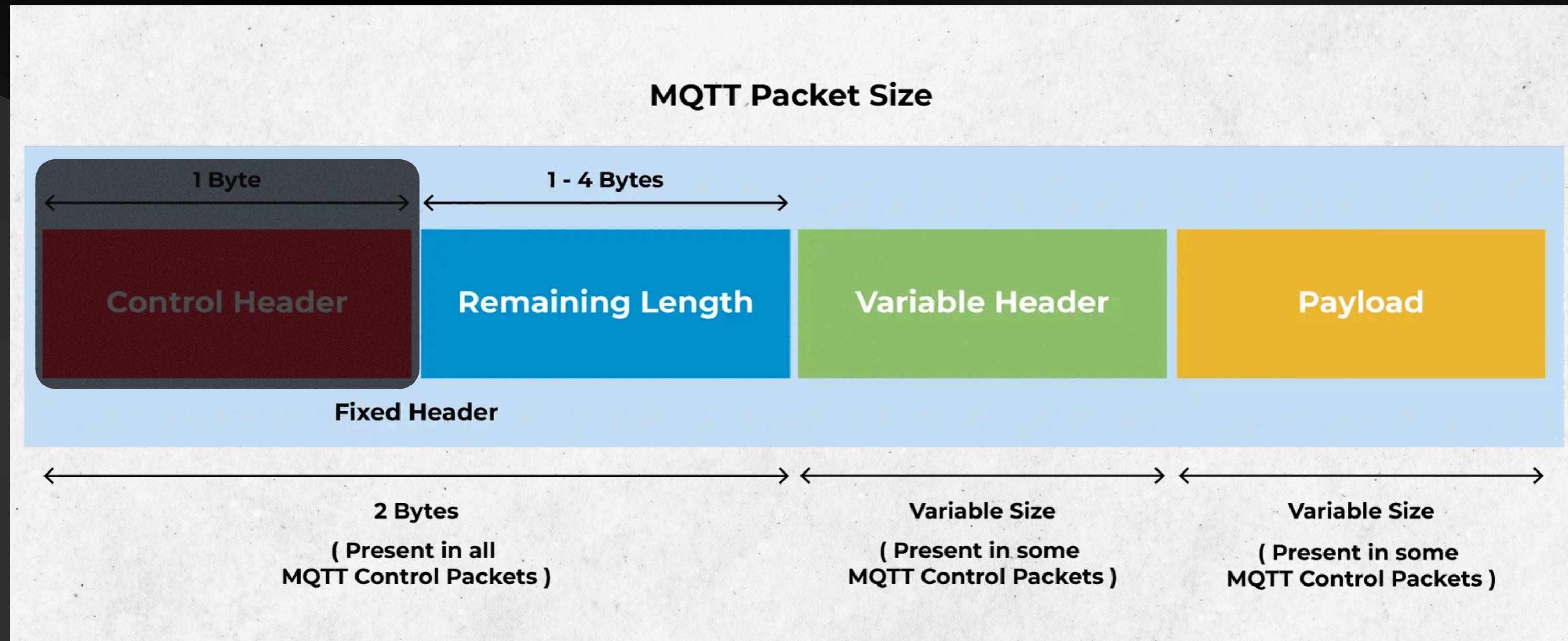
Nome	Valor	Direção	Descrição
CONNECT	1	Cliente -> Broker	Cliente requisita conexão
CONNACK	2	Broker -> Cliente	Reconhecimento de Conexão
PUBLISH	3	Ambas	Publicação de mensagem
SUBSCRIBE	8	Cliente -> Broker	Requisição de inscrição
PINGREQ	12	Cliente -> Broker	Teste de conexão com o broker*
DISCONNECT	14	Cliente -> Broker	Cliente está desconectando

Exemplo: 0b00110011 → PUBLISH, DUP=0, QoS=1, Retain=1



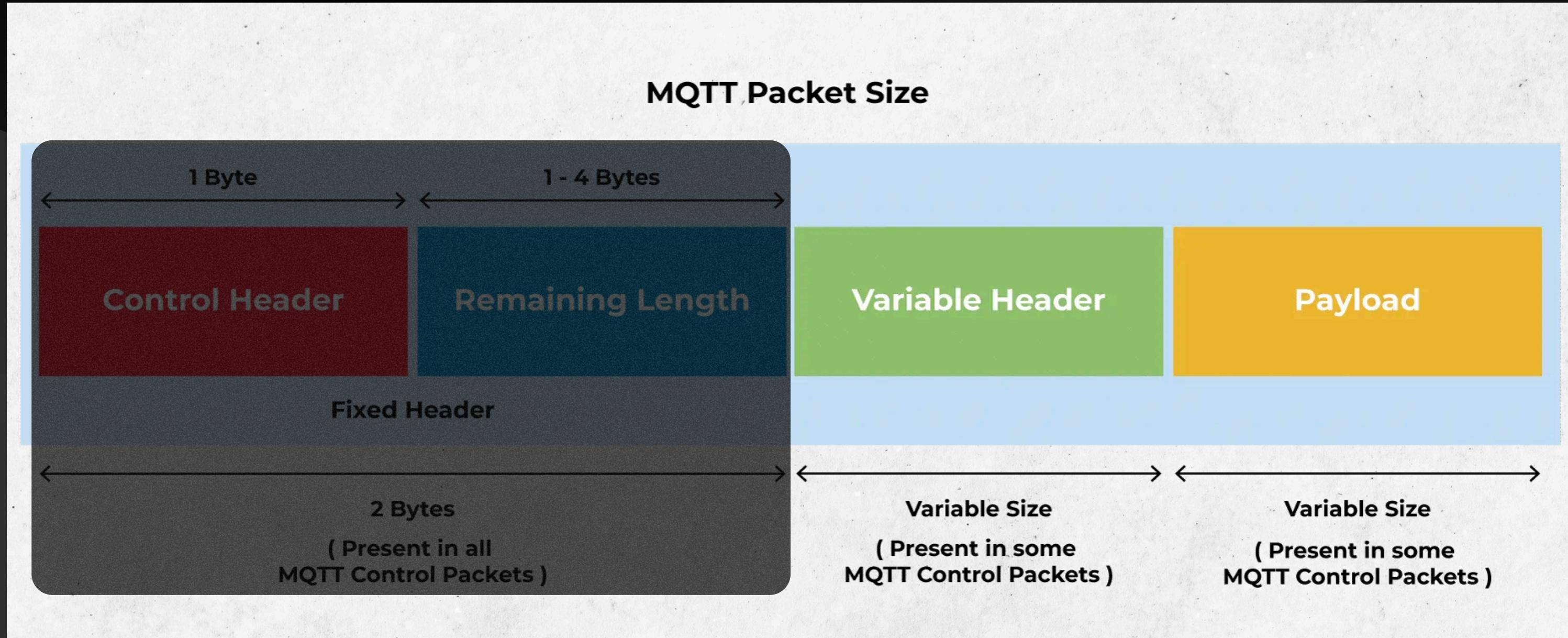
Arquitetura e Funcionamento

Remaining Length: Informa a quantidade restante de bytes da mensagem MQTT, ou seja, das sessões Variable Header e Payload



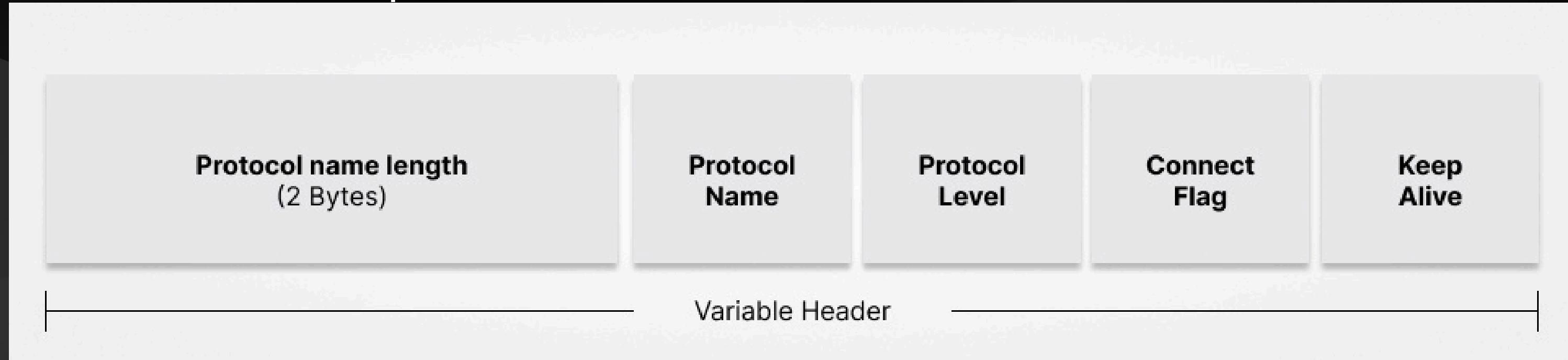
Arquitetura e Funcionamento

Variable Header: Carrega informações adicionais do pacote MQTT e por isso os tipos de informações presentes e seu tamanho variam por consequência



Arquitetura e Funcionamento

Variable Header de um pacote CONNECT:

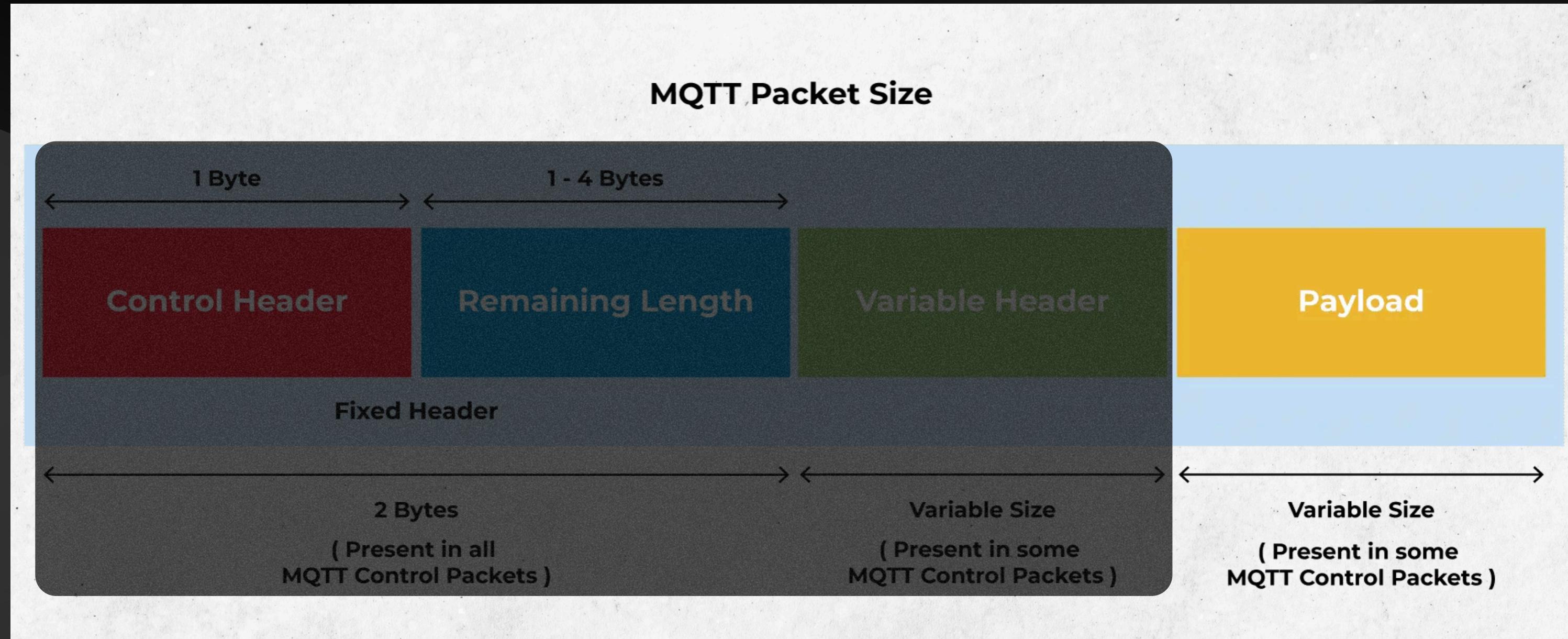


<https://cedalo.com/blog/mqtt-packet-guide/>



Arquitetura e Funcionamento

Payload: É a representação em binário da mensagem transmitida via MQTT, obviamente seu tamanho varia dependendo do pacote e também pode ser inexistente.



Exemplo: 0x43 0x65 0x64 0x61 0x6c 0x6f → Cedalo



Arquitetura e Funcionamento

Instagram

VS

MQTT

Contas

Tópicos

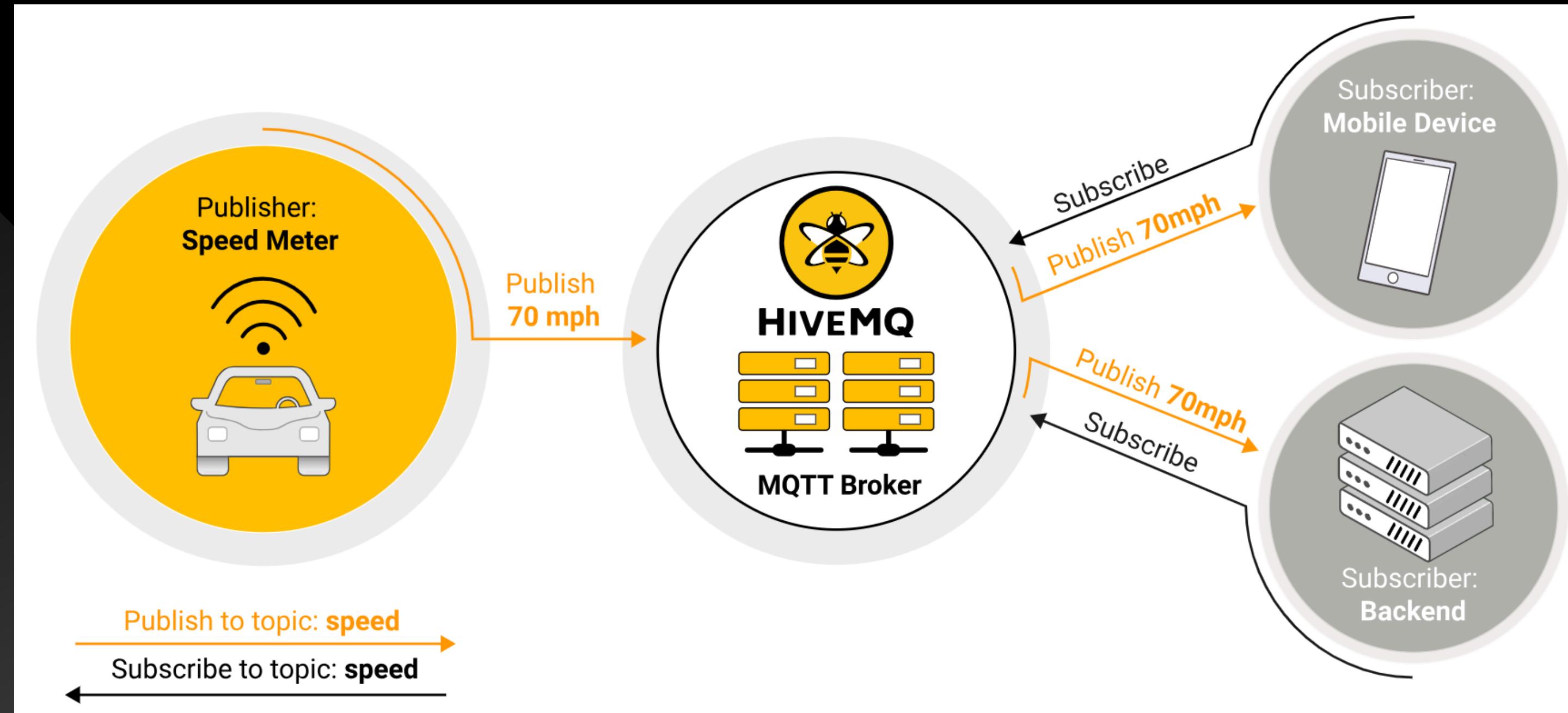
Influenciadores
&
Seguidores

Publicação
&
Inscrição

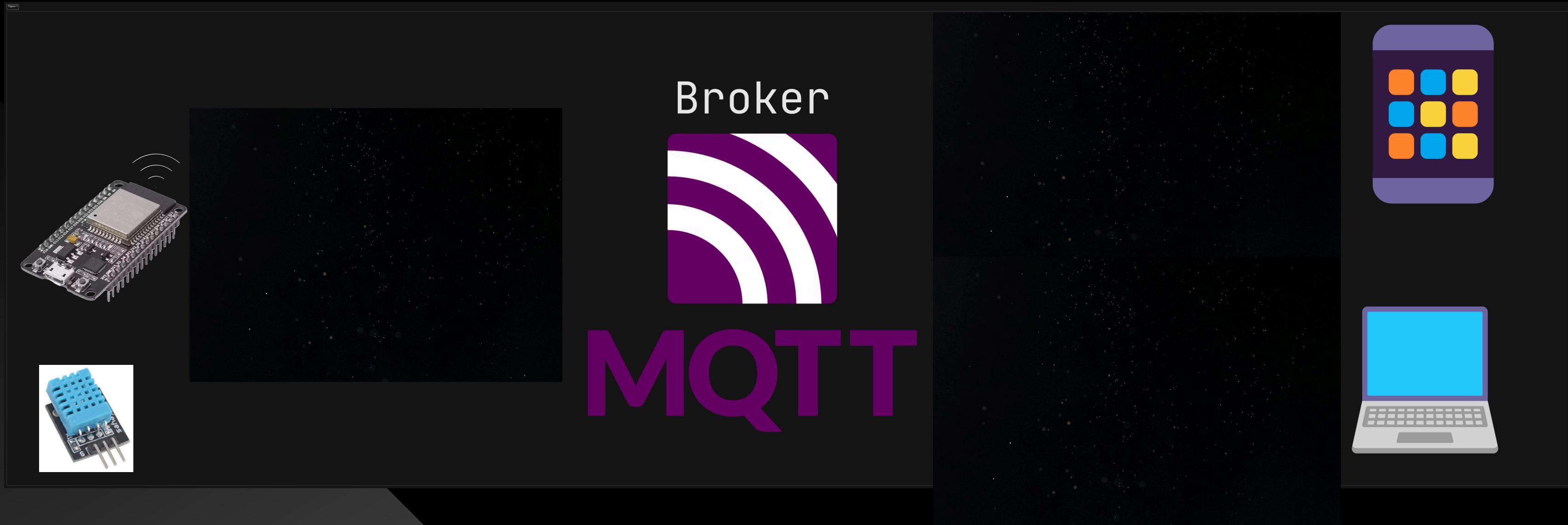


Arquitetura e Funcionamento

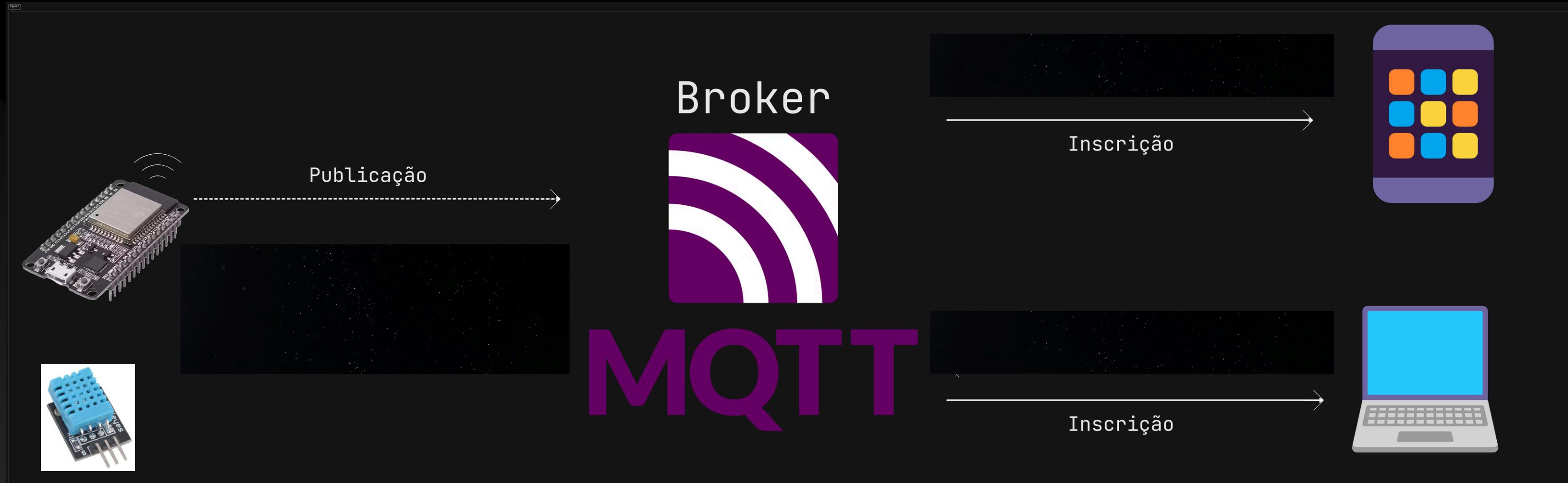
Arquitetura de Publicação e Inscrição:



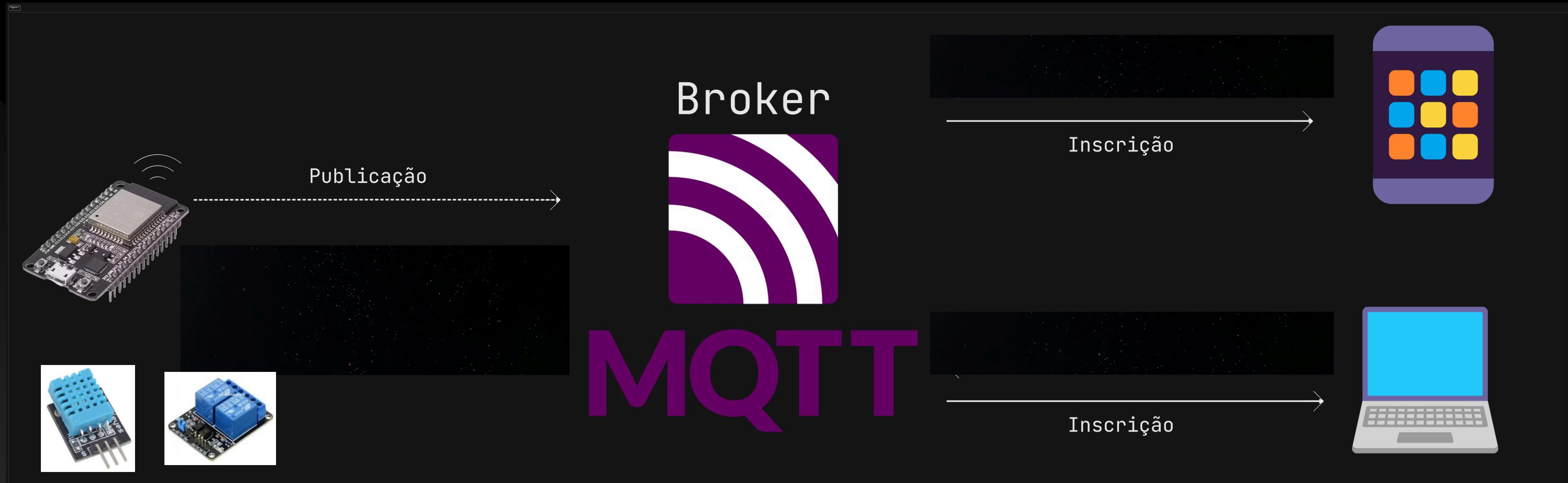
Arquitetura e Funcionamento



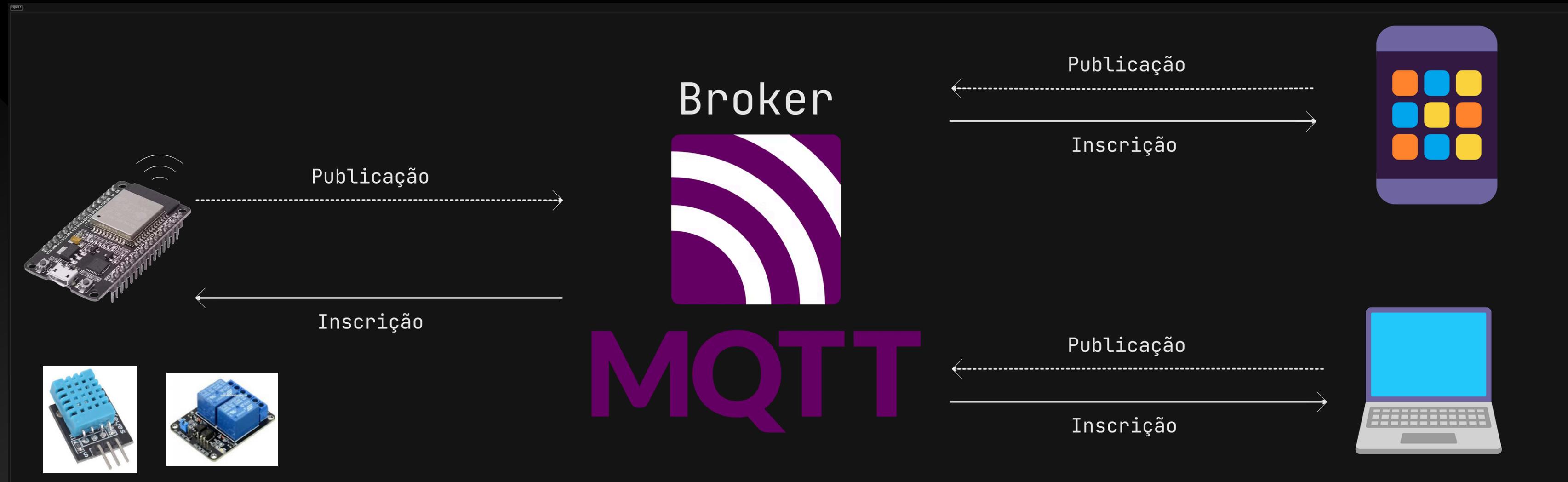
Arquitetura e Funcionamento



Arquitetura e Funcionamento



Arquitetura e Funcionamento



Tópicos

São cruciais para o funcionamento do protocolo MQTT, pois controlam o fluxo de mensagens definindo os interesses dos clientes inscritos

- String composta por níveis separados por barras “/”
- Criados Dinamicamente pelo Broker
- São sensíveis a capitalização e podem incluir espaços em branco*
- O uso de caracteres fora da tabela ASCII não é recomendado

- Começar ou terminar tópicos com barra não é recomendado
- Tópicos começados em “&SYS” retornam informações do próprio broker
- Presença de Tópicos Curinga (Wildcards) para facilitar a inscrição em diversos tópicos

Exemplos:

myhome/groundfloor/livingroom/temperature

Germany/Bavaria/car/2382340923453/latitude

5ff4a2ce-e485-40f4-826c-b1a5d81be9b6/status



Tópicos Curingas (Wildcards)

São utilizados por clientes que desejam se inscrever em uma variedade de tópicos que seguem um determinado padrão ao mesmo tempo.

+

Wildcard de nível único que deve ser usado dentro do tópico

Esse wildcard substitui um nível, ou seja, esse nível pode ser qualquer e o cliente se inscreverá em todos os que obedecerem esse padrão

casa/terreo/+/temperatura

- ✓ casa/terreo/cozinha/temperatura
- ✓ casa/terreo/quarto/temperatura
- ✗ casa/andar1/sala/luminosidade
- ✗ casa/andar1/quarto/temperatura
- ✗ casa/terreo/cozinha/geladeira/temperatura

#

Wildcard multi nível que deve ser usado ao final do tópico

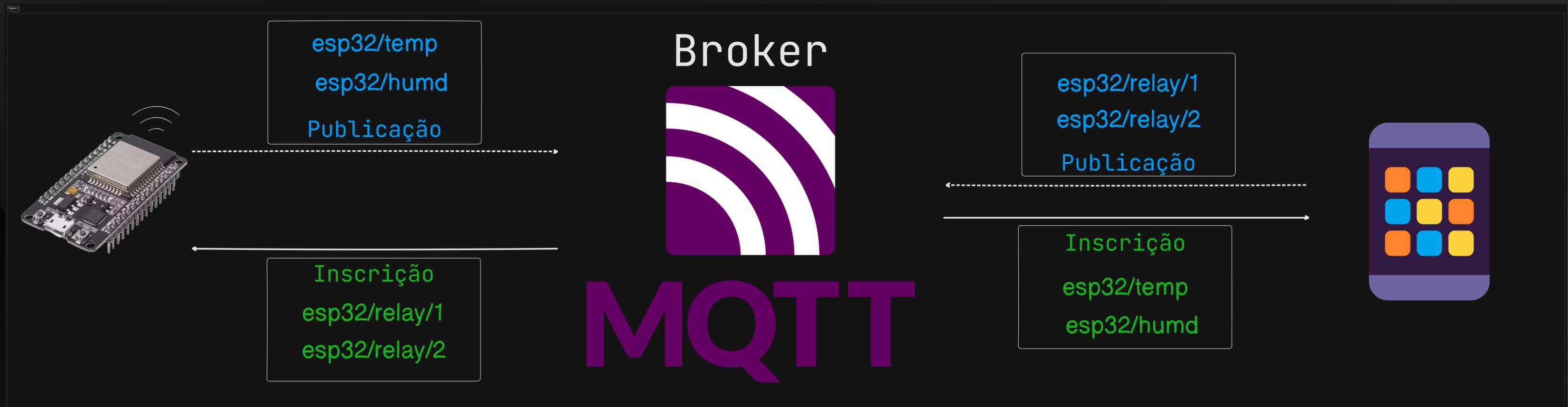
O cliente que usar esse wildcard se inscreve em todos os topics os quais o início correspondem com o tópico curinga

casa/terreo/#

- ✓ casa/terreo/sala/temperatra
- ✓ casa/terreo/quarto/temperatura
- ✓ casa/terreo/sala/luminosidade
- ✗ casa/andar1/cozinha/temperatura
- ✗ torreeifel/terreo/temperatura



Esquema de Tópicos do Projeto Prático



Quality Of Service (QoS)

Existem 3 níveis de QoS e em cada um a mensagem será recebida:

0

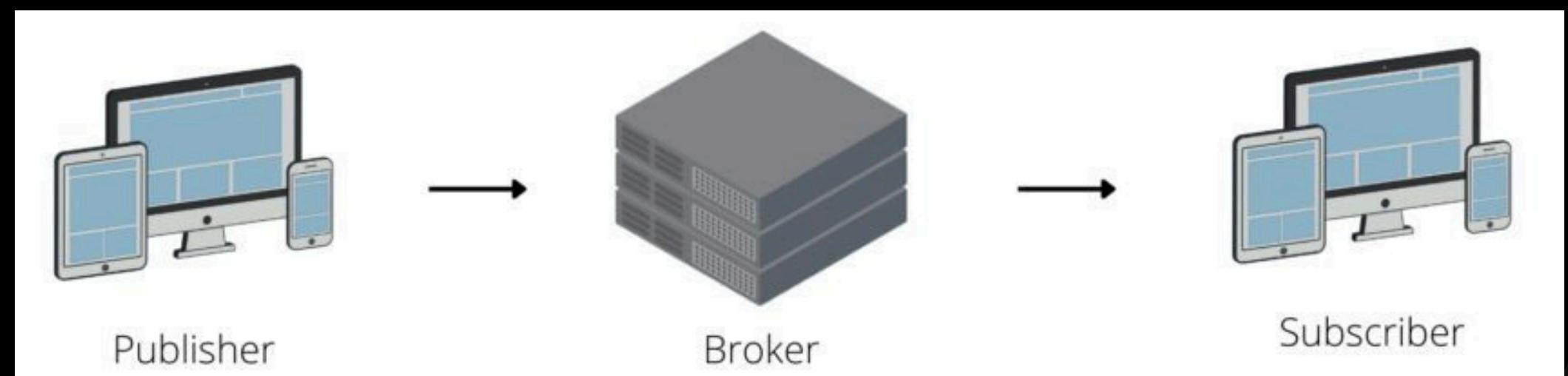
Ao máximo uma vez

1

Ao menos uma vez

2

Exatamente uma vez



QoS 0 – Ao máximo uma vez

Esse nível é conhecido como “lançar e esquecer” (“fire and forget it”), pois **caso não haja o recebimento da mensagem em seu primeiro envio, ela será perdida**



Fonte: <https://www.hivemq.com/blog/mqtt-essentials-part-6-mqtt-quality-of-service-levels/>



QoS 1 – Ao menos uma vez

Nesse nível a **mensagem é lançada e é aguardado a confirmação de recebimento**, que se não chegar, fará com que a mesma mensagem seja reenviada.

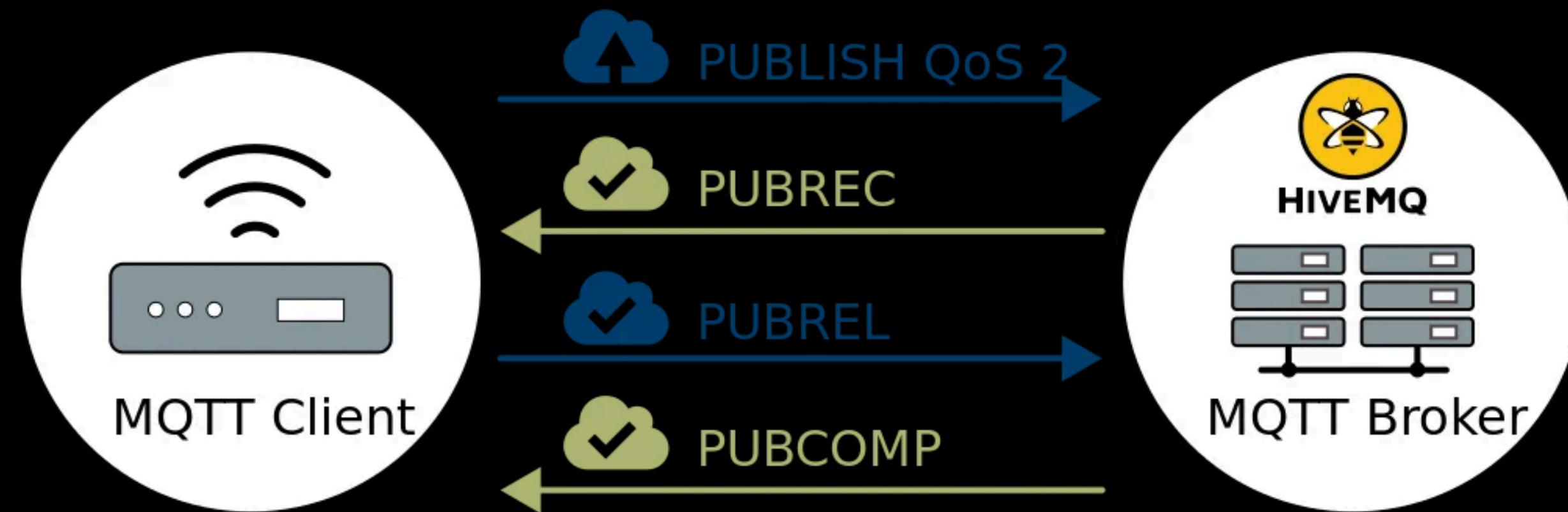


Fonte: <https://www.hivemq.com/blog/mqtt-essentials-part-6-mqtt-quality-of-service-levels/>



QoS 2 – Exatamente uma vez

Nesse nível a **mensagem é lançada e um processo de handshake entre as partes é iniciado** a fim de garantir o recebimento da mensagem sem reenviá-la.



Fonte: <https://www.hivemq.com/blog/mqtt-essentials-part-6-mqtt-quality-of-service-levels/>



Segurança

A Segurança do protocolo MQTT é garantida por:

Autenticação

Onde ocorre a identificação do cliente através de:

- 1) Usuário e Senha**
- 2) Serviços Externos (OAuth, JWT e etc)**

Autorização

É o processo de permitir ou negar o acesso do cliente a um ou mais recursos através de:

ACL (Access Control List)

Criptografia

A Criptografia garante que os pacotes MQTT não sejam lidos por terceiros através de:

SSL/TLS

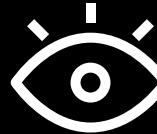


Conceitos Adicionais do Protocolo MQTT



Mensagens Persistentes

São mensagens que ficam gravadas em um tópico e que serão entregues ao cliente no momento em que ele se inscrever ao tópico.



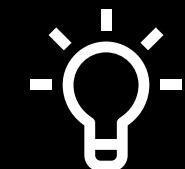
Intervalo de Keep Alive

Refere-se ao intervalo em segundos em que o cliente enviará um PINGREQ ao Broker para testar a conexão e informa-lo que o cliente ainda está conectado.



Sessões Clean e Sessões Persistentes

Opção configurada pelo cliente que informa ao Broker se ele deve gravar metadados desse cliente e oferecer funcionalidades adicionais.



Mensagens Will

São mensagens enviadas automaticamente pelo Broker quando o cliente não responde em um tempo além do intervalo de Keep Alive a fim de informar a ausência do cliente.



Projeto Prático



Criação Da Máquina Virtual

- Criação da VM na Azure com Ubuntu Server 22.04 LTS
- Configuração de Rede



Deploy do Broker Mosquitto

- Configuração e Deploy o Broker MQTT na recém criada VM



Configuração do App IoT MQTT Panel

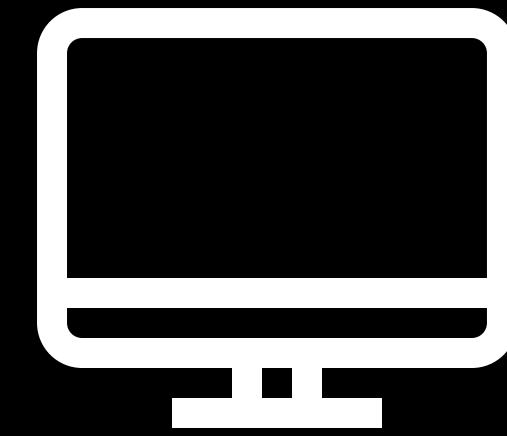
- Configuração do Cliente no Celular com o APP IoT MQTT Panel



Configuração do ESP32

- Visão Geral do circuito e código do ESP32





Mãos à Obra

Proximos passos...

- Criptografar a comunicação entre Clientes e Broker com SSL/TLS
- Implementar ACL (Access Control List) para controle de acesso e permissões
- Alterar configurações de rede da VM Azure e liberar o acesso apenas à clientes com IPs específicos
- Buscar uma biblioteca para ESP ou Arduino que suporte os outros níveis de QoS
- Comunicar nosso broker com outras ferramentas de automação residencial como Home Assistant, IFTT, Node-RED e etc.

(...)



Fontes:

- <https://mqtt.org/>
- <https://www.gta.ufrj.br/ensino/eel878/redes1-2019-1/vf/mqtt/>
- https://docs.aws.amazon.com/pt_br/iot/latest/developerguide/mqtt.html
- <https://www.bevywise.com/blog/understanding-mqtt-protocol-packet-format/> <https://www.hivemq.com/blog/mqtt-essentials-part-5-mqtt-topics-best-practices/>
- <https://github.com/mqtt/mqtt.org/wiki/SYS-Topics>
- <https://www.hivemq.com/blog/mqtt-essentials-part-6-mqtt-quality-of-service-levels/> https://www.youtube.com/watch?v=2S_kZo_ElxY <https://cedalo.com/blog/mosquitto-docker-configuration-ultimate-guide/>
- <https://pubsubclient.knolleary.net/>



Obrigado pela Atenção!

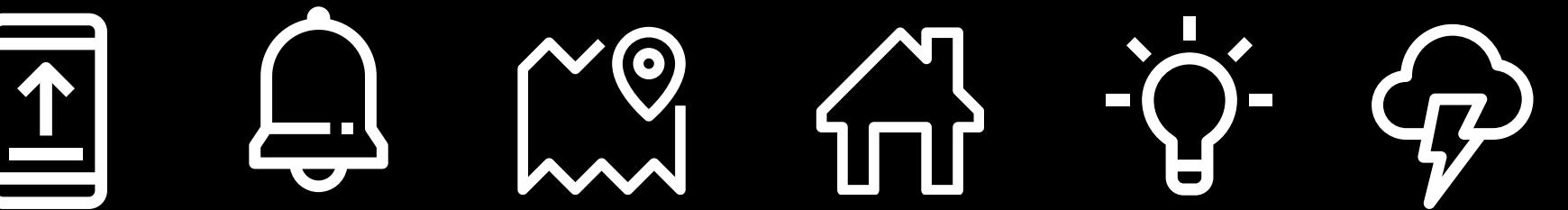
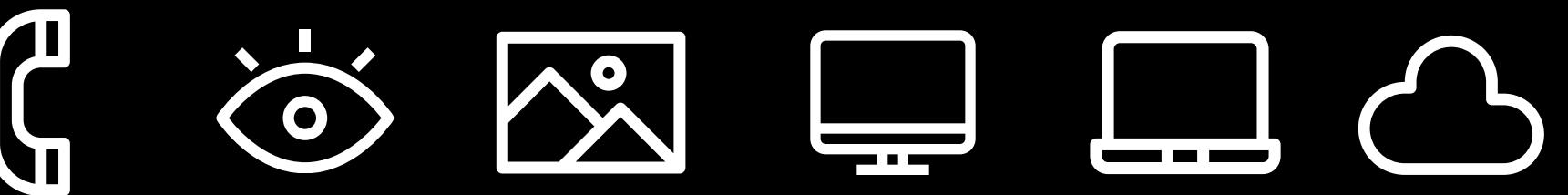
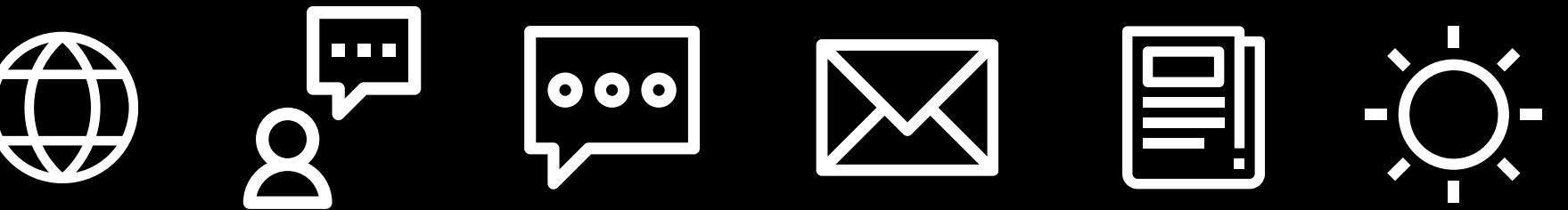
Foi um prazer poder **compartilhar** um pouco desse **conhecimento** com vocês.

Dúvidas? Entre em Contato!

- LinkedIn: www.linkedin.com/in/cizelucas
- Discord (nome de usuário): **czlcs**
- Instagram: <https://www.instagram.com/cize.lucas/>



Recursos



Use estes recursos de design na sua apresentação do Canva. Bom design!

Não se esqueça de excluir este slide antes de fazer sua apresentação.