MQTT

Prof. Me. Hélio Esperidião

MQTT(Message Queuing Telemetry Transport)

Protocolo de comunicação M2M - Machine to Machine próprio para Internet of Things (IoT) que funciona sobre o protocolo TCP/IP.

MQTT se baseia na comunicação entre cliente e servidor, em que o cliente pode fazer “postagens” ou “ouvir” publicações. O servidor administra os dados a serem recebidos e enviados.

É utilizado um Paradigma chamado Publish-Subscribe. O protocolo se popularizou pela simplicidade, baixo consumo de dados e pela possibilidade comunicação bilateral.

História

Nos anos 90 a IBM criou o protocolo MQTT.

Nasceu devido à necessidade de um protocolo leve que conseguisse comunicar várias máquinas(microcontroladores) entre si.

O MQTT é uma das principais tecnologias que podem tanto participar quanto impulsionar o desenvolvimento de dispositivos de Internet das Coisas (IoT).

Publish-Subscribe

O Padrão Publish-Subscribe permite que dispositivos se inscrevam em canais e escutem todas as publicações desses canais.

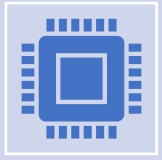
- Quem deseja comunicar outros dispositivos publica no canal (Publish)
- Quem fica esperando por novas publicações faz um Subscribe no canal

Dessa forma, o publisher e subscriber não precisam se conhecer diretamente e apenas precisam conhecer.

É preciso apenas conhecer o canal.

Broker

O Broker é o servidor de comunicação entre os dispositivos.



É ele que recebe os dados enviados pelos microcontroladores e é nele onde esses dados são tratados e passados adiante.

Broker

O Broker gerencia as inscrições em canais.

O Broker envia os dados para todos os inscritos.

O broker gerência as conexões no servidor.

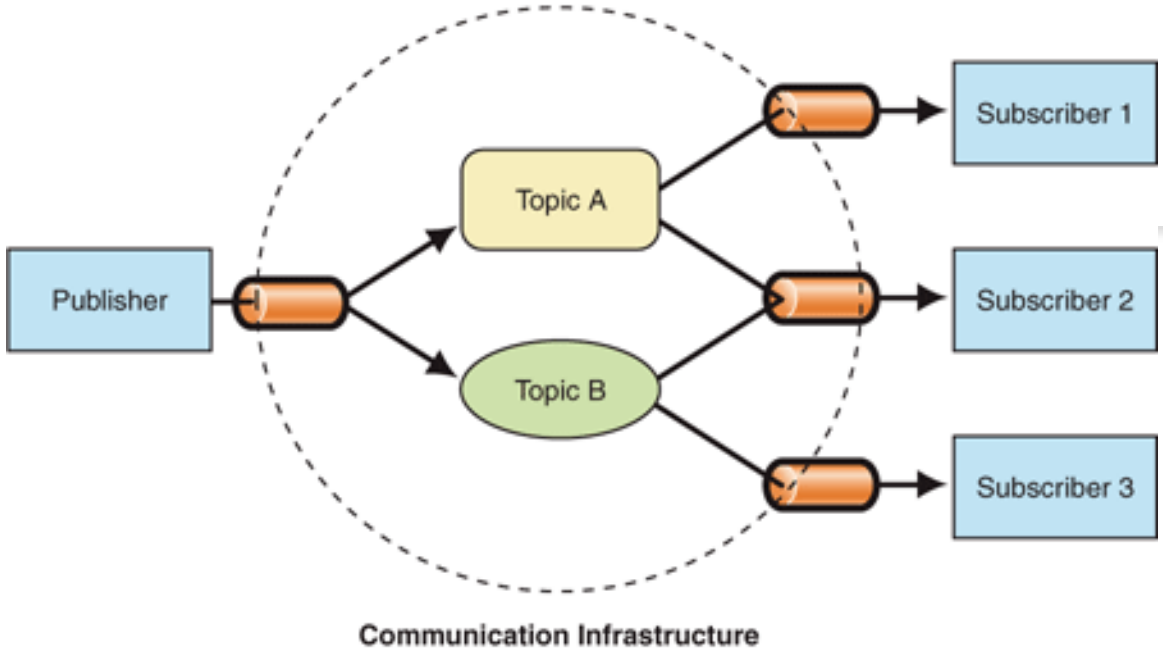
Broker, subscribers e publishers

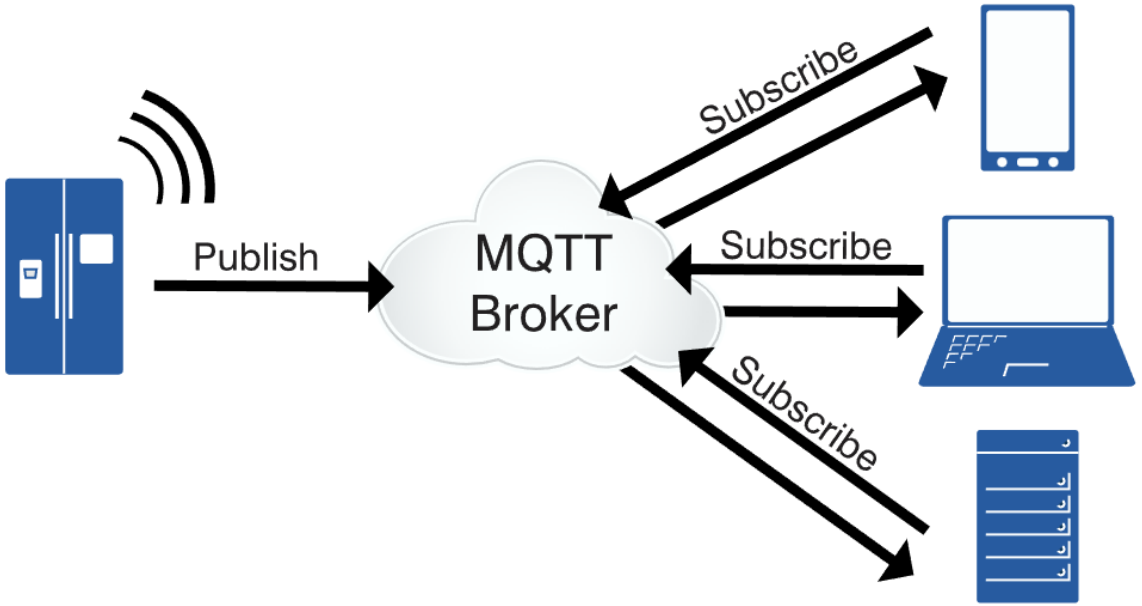
- O Broker é um intermediador entre os clientes do protocolo.
  - É ele quem vai receber as informações de um cliente publisher e vai enviar essa informação aos clientes subscribers inscritos no tópico referente àquela informação.
- O Cliente Publisher é aquele que envia ao broker.
  - Um exemplo seria um sensor, que envia os dados captados (como temperatura), ao broker que, por sua vez, enviará esse dado aos clientes subscribers.
- Cliente Subscriber é o cliente que recebe uma informação do broker.
  - Ele deve se inscrever em tópicos definidos pelo broker, cada um sobre uma certa informação, que por sua vez foi captada por um cliente publisher.
- Vale dizer que o cliente pode ser, ao mesmo tempo, publisher e subscriber, ou seja, enviar e consumir dados ao/do broker.

Publisher e Subscriber ao mesmo tempo

É comum em alguns casos os sistemas terem atuações tanto de Publisher quanto de Subscriber.

Ou seja, hora recebe dados de um canal hora escreve em um canal.


Publisher
Topic A
Topic B
Subscriber 1
Subscriber 2
Subscriber 3
Communication Infrastructure

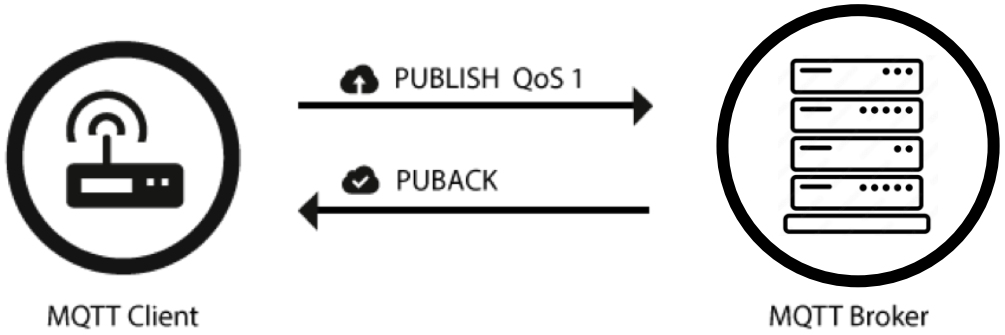

Publish
MQTT
Broker
Subscribe
Subscribe
Subscribe

QoS 0 - No máximo uma vez

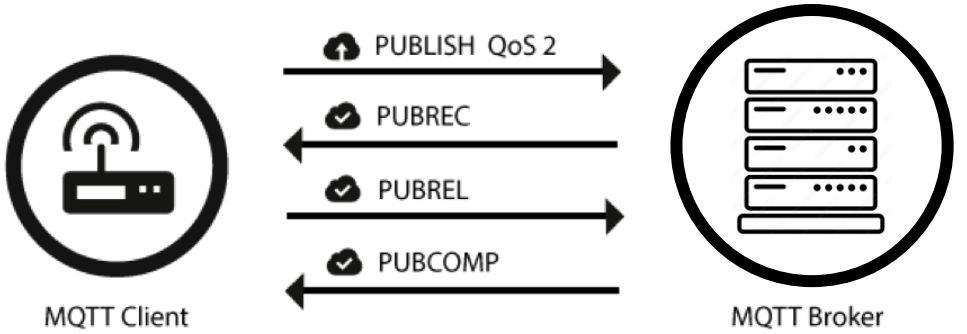
- Conhecido como *fire and forgot* (atirar e esquecer)
- a mensagem é enviada apenas uma vez e não haverá passos seguintes, dessa forma a mensagem não será armazenada, nem haverá um feedback para saber se ela chegou ao destinatário.
- Esse modo de transferência é o mais rápido, porém o menos seguro.

QoS 1 - Pelo menos uma vez

- Nesse modo de transferência, a mensagem é entregue pelo menos uma vez, havendo uma espera da recepção de feedback da entrega da mensagem, o chamado PUBACK.
- Não recebendo o PUBACK, a mensagem continuará sendo enviado até que haja o feedback. Nesse QoS pode acontecer da mensagem ser enviada diversas vezes e ser processada diversas vezes.



QoS 2 - Exatamente uma vez



- A mensagem é entregue exatamente uma vez, necessitando que a mensagem seja armazenada localmente no emissor e no receptor até que seja processada.
- É necessário o envio de 2 pares de request-response(chamado de four-part handshake),
  - 1 - o envio da mensagem(PUBLISH),
  - 2 - a resposta de recepção(PUBREC),
  - 3 - o aviso do recebimento do PUBREC(PUBREL)
  - 4 - e a confirmação de que o processo foi concluído e pode ser feita a exclusão(PUBCOMP).
- Após o recebimento do PUBREL, o receiver pode excluir a mensagem e ao sender receber o PUBCOMP ele poderá excluir a mensagem.

Vantagens do MQTT

- Baixo consumo de memória, baixa necessidade de processamento para o envio de mensagem e baixo consumo de banda.
- Como o **publisher** não envia a informação direto para os subscribers, ele não precisa guardar a informação de todos os seus subscritores e nem precisa fazer várias envios de informação(uma para cada subscriber).
  - Apenas é necessário que ele realize um envio de informação para o broker com a informação que ele quer que seja enviada daquele tópico, dessa forma o processamento realizado e o consumo de memória do Publisher pode ser reduzido.
- O header de uma mensagem no protocolo MQTT é muito menor do que um Header no protocolo HTTP, o que economiza muito o consumo de banda.

Transmissão de Dados

O protocolo MQTT utiliza outro protocolo chamado TCP para a transmissão de dados.

Vantagens do MQTT sobre o HTTP

- Baixo consumo de memória,
- Baixa necessidade de processamento para o envio de mensagem
- Baixo consumo de banda.

Como começar a usar o protocolo MQTT

Existem inúmeras implementações tanto para brokers como para clientes MQTT em diversas linguagens, tais como Python, JavaScript, C#, entre outros.

Uma dessas implementações mais famosas é o broker Mosquitto.

Mosquitto é um dos brokers mais utilizados atualmente devido à sua simplicidade e facilidade de implementação, esta tecnologia open source ganhou um grande espaço no mercado de IoT.

Mosquitto

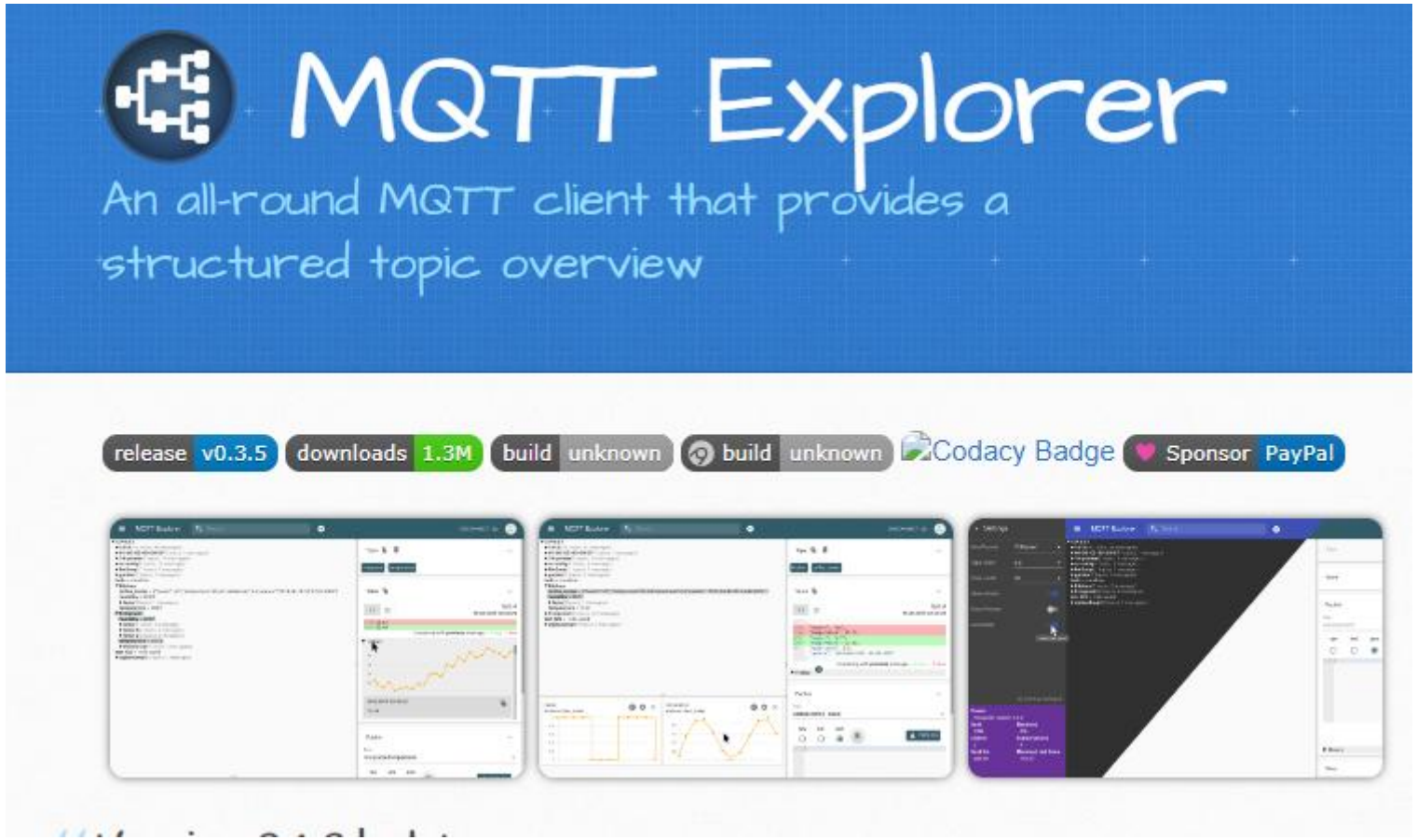
- O "Mosquitto" pode ser facilmente baixado para vários sistemas operacionais.
- Os arquivos de download podem ser encontrados na página oficial do broker.
- Ao baixar o broker escolhido, este vai ser utilizado para rodar o servidor onde o sistema e o relacionamento entre os clientes e o broker será hospedado.

Outros tipos de Brokers

Host/Endereço	Porta(s)	Observações
test.mosquitto.org	1883 / 8883 / 80 / 443	Broker oficial da Eclipse. Sem autenticação, ideal para testes.
broker.hivemq.com	1883 / 8000 (WebSocket)	Sem autenticação. Recomendado apenas para testes e prototipagem.
mqtt.fluux.io	1883 / 8883	Sem autenticação. Público e estável.
mqtt.swifitch.cz	1883	Sem autenticação. Pouco conhecido, mas funcional.
mqtt.eclipseprojects.io	1883 / 8883	Mantido pela Eclipse Foundation. Sem autenticação.

Software mqtt explorer

- http://mqtt-explorer.com/



Servidor de teste

- Host: iotsistemas.com.br
- Usuario: univap
- Senha: univap