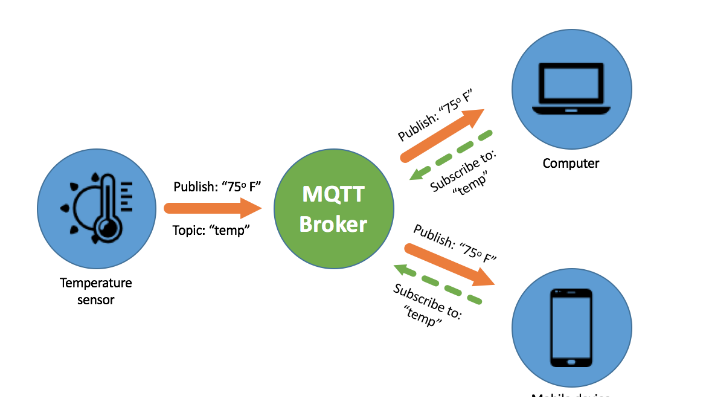
MQTT

MQTT awalnya dikembangkan oleh IBM pada tahun 1999 untuk melakukan monitoring terhadap sebuah pipa minyak yang ada di suatu tempat yang jauh dan sulit dijangkau. Tujuan dari proyek ini adalah untuk mempunyai sebuah *protocol,*yang sangat *bandwidth-efficient*dan mengonsumsi tenaga batere se-sedikit mungkin, hal ini karena perangkat terkoneksi melalui jaringan satelit yang sangat mahal pada jaman itu.

Protocol ini menggunakan konsep *publish/subscribe*. Berbeda dengan HTTP yang menggunakan konsep*request/response*. Pubsub adalah *event-driven* dan memungkinkan dari *server* untuk mengirim pesan ke *client* kapanpun dibutuhkan. Pusat komunikasi ada di MQTT *broker*, *broker* ini bertanggung jawab atas terkirimnya semua pesan termasuk jalur distribusinya. Setiap *client* yang mengirim pesan ke *broker*, termasuk juga mengirimkan *topic* ke dalam pesan tersebut. *Topic* merupakan bagian dari *routing information*untuk broker-nya. Tiap klien yang menginginkan menerima pesan, bisa meng-*subscribe* ke suatu *topic* tertentu dan broker akan mengirimkan semua message yang cocok dengan pola (*pattern*) *topic* tersebut kepada *client*yang sesuai. Mekanisme tersebut membuat klien tidak perlu untuk tau satu sama lain untuk bisa berkomunikasi, melainkan mereka cukup berkomunikasi menggunakan *topic*. Arsitektur semacam ini membuat solusi percakapan bisa dikembangkan dengan skala besar (*scalable*) karena tidak memerlukan *dependencies* antara *data producers*dan *data consumers.*



Komponen Dalam MQTT

1. Publisher dan Subscriber

Komponen paling jelas terlihat dalam sebuah pubsub system adalah subject yang melakukan publisher dan subscriber. Dalam MQTT, kita memerlukan mqtt client untuk bisa melakukan publish ataupun subscribe. Library client MQTT sudah banyak tersedia di berbagai bahasa pemrograman. Salah satu yang paling terkenal adalah [Paho](https://www.eclipse.org/paho/), yang dikembangkan oleh Eclipse, tersedia dalam berbagai macam platform ataupun bahasa pemrograman.

1. Message

Dalam contoh yang sama di atas, message dikirim melalui publisher dengan nilai Hello there! . Message ini kemudian akan diterima oleh client yang melakukan subscribe, dan suatu callback on\_message akan dipanggil. Sehingga kita bisa melakukan penanganan terhadap nilai dari pesan tersebut.

Apabila diamati lebih dalam, nilai dari message yang dikirim mengandung nilai dari topic. Seperti yang sudah dijelaskan di atas, MQTT client akan mengirim message dan topic kepada broker sehingga broker MQTT akan melakukan penanganan terhadap routing pesan tersebut sesuai dengan topic yang dimiliki.

1. Topic

Dalam contoh di atas, topic mempunyai nilai paho/ruby/test. Di MQTT, topic merupakan string akan tetapi ada beberapa character yang mempunyai arti khusus. Seperti misalnya character / mempunyai arti seperti halnya directory. Kemudian ada lagi character + yang merupakan character wildcard dalam 1 level directory dan juga # yang merupakan character wildcard dalam multiple level directory.

1. QoS

QoS mempunyai singkatan Quality of Service (kualitas pelayanan). Dalam MQTT, ada 3 macam QoS, yaitu 0, 1, atau 2.

1. QoS 0 Sama dengan halnya kita melakukan fire-and-forget. Artinya tidak ada jaminan kalau pengiriman pesan tersebut akan sampai ke subscriber.
2. QoS 1 : Pesan akan dijamin untuk sampai di minimal 1x ke subscriber yang sedang meng-subscribe.
3. QoS 2 : Pesan akan dijamin untuk sampai tepat 1x ke subscriber yang sedang meng-subscribe.
4. Broker

Bisa dikatakan broker merupakan bagian paling fundamental dalam MQTT, karena tanpa adanya broker, semua hal di atas tidak akan bisa dilakukan.

MQTT broker sudah diimplementasi oleh banyak orang sesuai dengan kebutuhan masing-masing. Ada banyak open-sourced implementasi yang bisa kita nikmati untuk bisa menjalankan MQTT broker di server kita. Ada pula enterprise version atau managed service MQTT apabila kita tidak mau repot untuk mengurus MQTT tersebut.

Tautan dari Wikipedia berikut mempunyai beberapa daftar implementasi MQTT. Beberapa broker implementasi yang populer di antaranya : Mosquitto (C based), EMQ (Erlang based), Mosca (Javascript based), VerneMQ (Erlang based), HiveMQ (Java based), dll.

Port yang biasa digunakan broker MQTT adalah 1883, dan 1885 (untuk TLS). Kemudian beberapa broker MQTT juga sudah mendukung integrasi langsung ke protocol Websocket, sehingga bisa lebih mudah kompatibel dengan aplikasi web. Biasanya yang digunakan adalah port websocket tersebut misalnya 8080, atau 8000, atau bahkan bisa 1884 dan 1886 (untuk TLS) agar mirip dengan protocol yang umum di MQTT, tergantung selera.

Apabila hanya untuk mencoba mencicipi MQTT, kita bisa saja menggunakan free online MQTT yang bisa langsung digunakan. Seperti misal contoh di source code di atas, kita menggunakan iot.eclipse.org. Ada beberapa online broker yang lain :