



# IoT et protocole MQTT

QUE RETIRER DU CHALLENGE IOT DE LA CYBER NIGHT ?  
UNE INTRODUCTION À MQTT

Auteur : Delautre Benjamin

## Petit rappel du challenge

- Un schéma nous montrait différents appareils reliés entre eux, dont un relié à une pompe à eau vers une citerne. Il y était fait mention d'un certain port et d'un « broker ».
- Un « pirate » avait modifié le mode de fonctionnement d'une de ses pompes de manière à ce qu'elle continue de déverser de l'eau dans la citerne même en présence d'humains.
- Notre mission était d'empêcher cette citerne de se remplir pour sauver les ouvriers.

## Raison d'être de cette veille

- Quatre pensées me sont venues à l'esprit en lisant le pitch et le schéma :
  - Pourquoi le port 1883 ?
  - C'est quoi un broker ?
  - Quel rapport avec l'IoT ?
  - « Mon Dieu, pourquoi moi ? »
- Tant de sujets qu'il serait bon de creuser

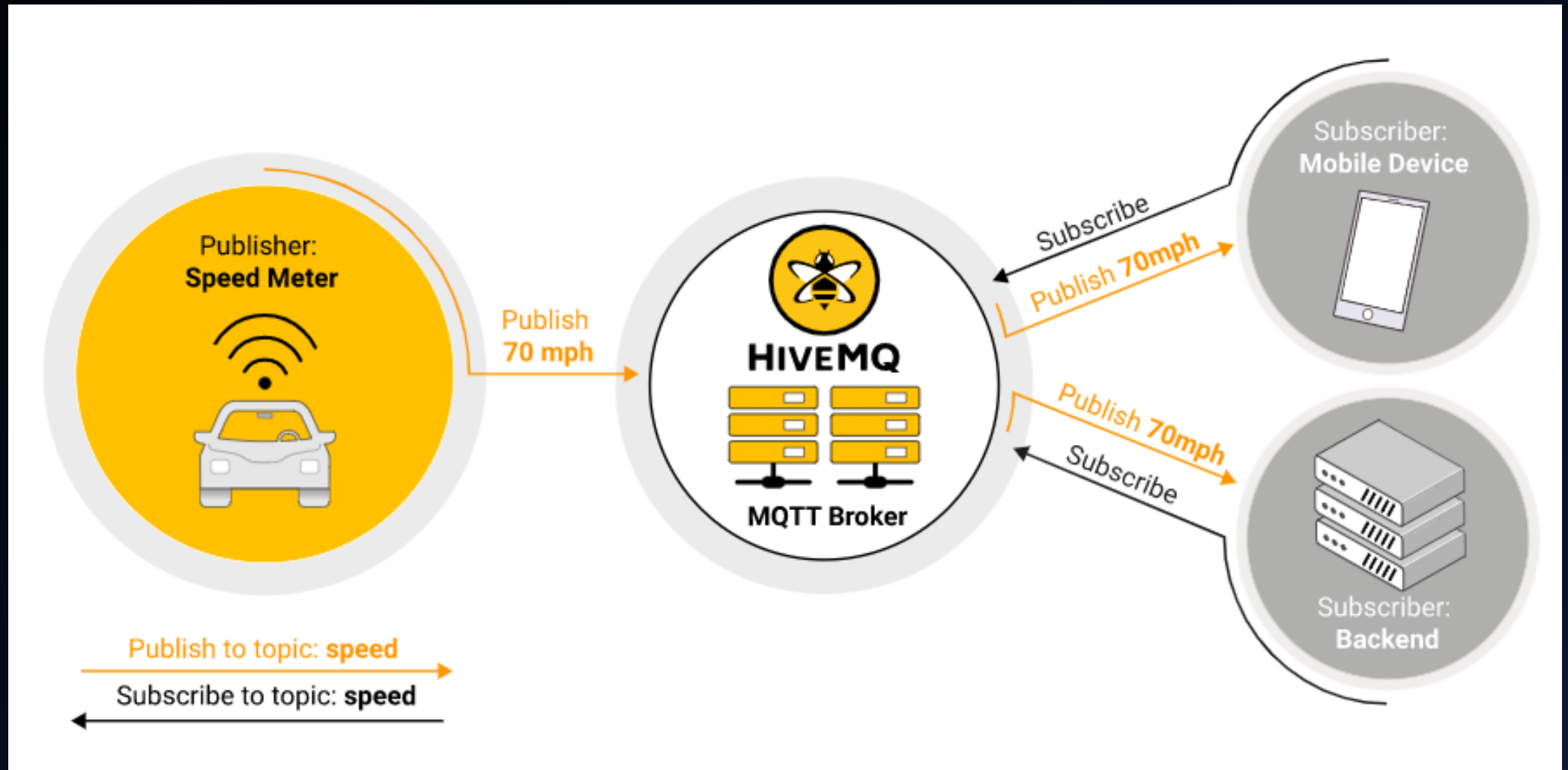
# Le port 1883 et le protocole MQTT

- MQTT est un protocole de transport de message inventé par Andy Stanford-Clark (IBM) et Arlen Nipper (Arcom/Cirrus Link).
- L'origine de l'acronyme est double :
  - « MQ » vient d'un système de messagerie d'IBM, les MQSeries.
  - « TT » est pour *Telemetry Transport*

## Pourquoi le MQTT

- Il a été inventé en 1999 pour un projet visant à surveiller un oléoduc dans le désert via un satellite.
- Les chercheurs avaient besoin d'un protocole plus « léger » que le http et à faible consommation d'énergie.
- Par défaut, son port est le 1883. Mais aussi le 8883 si on utilise TLS/ SSL pour plus de sécurité.

# Comment ça marche ?



# Comment ça marche ?

- Contrairement au http qui emploie un modèle requête-réponse qui lie les clients (relation client-serveur), le **modèle publish/subscribe** découple le client qui envoie le message du client qui reçoit le message.
- Les deux clients ne sont, pour ainsi dire, pas conscients de l'existence de l'autre. Car la communication est gérée par un intermédiaire : **le broker**.
- Le travail du broker est de filtrer tous les messages reçus et de les distribuer correctement à tous les subscribers.

# Oui, d'accord. Mais c'est quoi un broker ?

- Le **broker est un serveur** jouant le rôle principal dans un réseau basé sur MQTT.
- Le broker est principalement chargé de **recevoir tous les messages** des clients éditeurs, de **les filtrer**, de décider quel abonné est intéressé puis **d'envoyer les messages aux clients abonnés**.
- Le broker MQTT doit être connecté en permanence à Internet afin que chaque fois qu'il y a une demande du client pour se connecter, publier un message ou s'abonner à un sujet, le broker MQTT puisse traiter la demande immédiatement.



# Liste de brokers

- Quelques brokers dispos :
  - HiveMQ
  - Mosquitto
  - CloudMQTT
  - Adafruit
  - ...
- Deux catégories de brokers : **Self-hosted brokers** (peut être installé sur son propre serveur) et **Cloud based brokers** (hardware fourni par le vendeur, le client est responsable de l'installation et la gestion du software)



# Susbscriber ? Topic ? Message ?

+ New Subscription

# QoS 0

Plaintext

All | Received | Published

Topic: finemeca/mobile/write QoS: 0

```
{"gateway":120,"idRPC":100,"DQ1":false,"DQ2":false,"DQ0":false}
```

2022-11-18 16:41:50:378

Topic: finemeca/mobile/write QoS: 0

```
{"gateway":120,"idRPC":100,"DQ1":false,"DQ2":false,"DQ0":false}
```

2022-11-18 16:41:50:414

Topic: finemeca/mobile/write QoS: 0

```
{"gateway":120,"idRPC":100,"DQ2":true,"DQ1":false,"DQ0":true}
```

2022-11-18 16:41:51:102

Topic: finemeca/mobile/write QoS: 0

```
{"gateway":120,"idRPC":100,"DQ2":true,"DQ1":false,"DQ0":true}
```

2022-11-18 16:41:51:401

Topic: finemeca/mobile/write QoS: 0

```
{"gateway":120,"idRPC":100,"DQ1":false,"DQ2":false,"DQ0":false}
```

2022-11-18 16:41:51:861

Payload: JSON QoS: 0 Retain Meta

finemeca/mobile/write

```
{"gateway":120,"idRPC":100,"DQ1":false,"DQ2":false,"DQ0":false}
```

← →

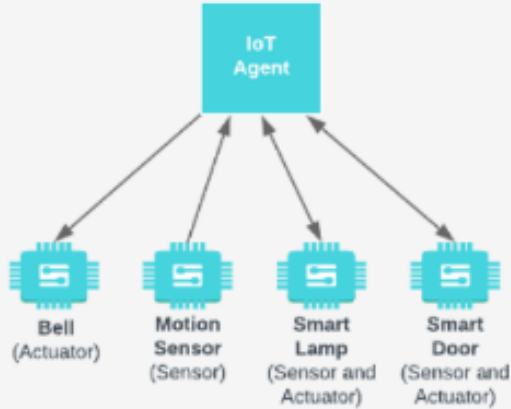
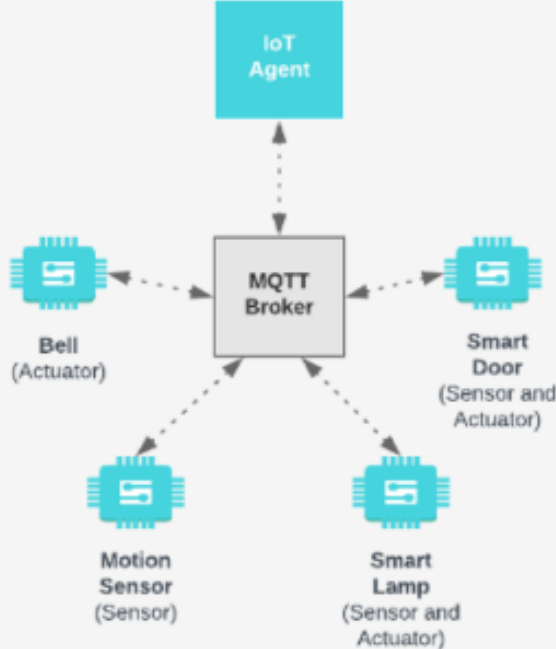
# Susbscriber ? Topic ? Message ?

- En haut à gauche de l'image, on peut voir que j'ai une subscription au **topic « # »**. Qui est une convention pour **recevoir les messages de tous les topics**.
- La pompe envoyait des messages dans le **topic « finemeca/mobile/write »** :
  - {"gateway":120,"idRPC":100,"DQ2":true,"DQ1":false,"DQ0":true}
- Avec mon équipier, on a envoyé un message dans le même topic pour tourner toutes les valeurs en false :
  - {"gateway":120,"idRPC":100,"DQ2":false,"DQ1":false,"DQ0":false}

# Susbscriber ? Topic ? Message ?

- Le broker a plusieurs options de filtres pour que les subscribers ne reçoivent que les messages qui les intéressent :
  - Filtre basé sur le **sujet (topic)**
  - Filtre basé sur le **contenu (content)** – Attention, l'inconvénient est que le contenu du message doit être connu à l'avance et ne peut être chiffré ou changé facilement.
  - Filtre basé sur le **type** - Lorsque des langages orientés objet sont utilisés, le filtrage basé sur le type/classe d'un message (événement) est une pratique courante.

# Comment ça marche ?

HTTP Transport	MQTT Transport
 <p>The diagram shows an IoT Agent at the top, connected by solid arrows to four IoT devices below: Bell (Actuator), Motion Sensor (Sensor), Smart Lamp (Sensor and Actuator), and Smart Door (Sensor and Actuator). Each device is represented by a blue icon with a white 'S' and a chip-like border.</p>	 <p>The diagram shows an IoT Agent at the top, connected by a dashed arrow to a central MQTT Broker. The MQTT Broker is then connected by dashed arrows to four IoT devices: Bell (Actuator), Motion Sensor (Sensor), Smart Lamp (Sensor and Actuator), and Smart Door (Sensor and Actuator). Each device is represented by a blue icon with a white 'S' and a chip-like border.</p>
IoT Agent communicates with IoT devices <b>directly</b>	IoT Agent communicates with IoT devices <b>indirectly</b> via an MQTT Broker
<b>Request-Response</b> Paradigm	<b>Publish-Subscribe</b> Paradigm
IoT Devices must always be ready to receive communication	IoT Devices choose when to receive communication
Higher Power Requirement	Low Power Requirement

# Avantages

- Les avantages du MQTT pour les communications de machine à machine :
  - Évolutivité, pour connecter de plus en plus de clients.
  - Découplage entre clients, pour moins de dépendance.
  - Simplicité.
  - Légèreté pour ne pas consommer trop de ressources (bien qu'avec la sécurité TLS / SSL ça monte).
  - Économe en énergie et n'a pas besoin d'une large bande passante (idéal pour les connexions lentes, comme certaines connexions sans fil).

# Sources

- <https://www.paessler.com/it-explained/mqtt#:~:text=in%20your%20browser.-,History,their%20data%20to%20remote%20servers>.
- [https://behrtech.com/blog/mqtt-in-the-iot-architecture/#:~:text=MQTT%20was%20invented%20in%201999,Data%20Acquisition%20\(SCADA\)%20systems](https://behrtech.com/blog/mqtt-in-the-iot-architecture/#:~:text=MQTT%20was%20invented%20in%201999,Data%20Acquisition%20(SCADA)%20systems).
- <https://www.journaldunet.fr/web-tech/dictionnaire-de-l-iot/1440686-mqtt-comment-fonctionne-ce-protocole/>
- <https://www.hivemq.com/blog/mqtt-essentials-part-1-introducing-mqtt/>
- <https://www.hivemq.com/blog/introducing-the-mqtt-security-fundamentals/>
- <https://www.planete-domotique.com/blog/2021/03/17/protocole-mqtt-iot-domotique/>
- [https://www.hwlibre.com/fr/mqtt/?utm\\_content=buffer6e011&utm\\_medium=social&utm\\_source=facebook.com&utm\\_campaign=buffer](https://www.hwlibre.com/fr/mqtt/?utm_content=buffer6e011&utm_medium=social&utm_source=facebook.com&utm_campaign=buffer)
- <https://devopedia.org/mqtt>