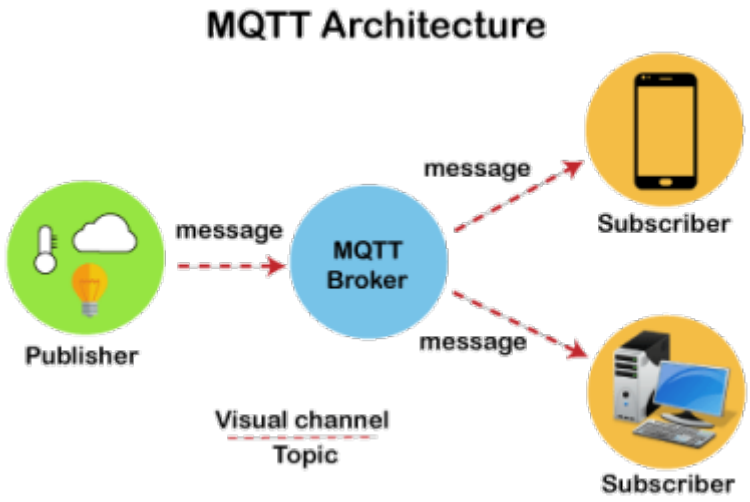
Protocole IOT MQTT

Qu’est ce que MQTT ?

MQTT (Message Queuing Telemetry Transport) est un service de messagerie TCP/IP où chacun pourra soit publier des messages, soit souscrire à un message, message de n’importe quelle nature que ce soit. Un broker reçoit et émet les messages.



Doc MQTT

- MQTT-fr en Pdf
- Presentation MQTT FR en pdf
- AWS et MQTT FR
- Doc MQTT

Liens Web

- https://www.hwlibre.com/fr/MQTT/
- https://www.lemagit.fr/conseil/Internet-des-Objets-bien-comprendre-MQTT
- https://www.framboise314.fr/utiliser-le-protocole-mqtt-pour-communiquer-des-donnees-entre-2-raspberry-pi/
- https://www.sla99.fr/2019/02/12/utiliser-mqtt-pour-la-domotique-a-la-maison/
- https://www.figer.com/Publications/domotique.htm
- https://docplayer.fr/142109461-Mqtt-message-queuing-telemetry-transport.html

Installer Mosquito

- Installer Mosquito sous linux FR
- Mosquito sur Raspberry FR
- Tutoriel pour l'installation de MQTT sur Rpi et connexion avec un Arduino en Wifi sur github FR
- Test mosquitto sur raspberry broker-and-client EN

- URL: https://linuxfr.org/news/transmission-de-donnees-de-capteurs-via-internet
- Title: Transmission de données de capteurs via internet

- Authors: tout

> Benoît Sibaud

- Date: 2022-05-21T20:43:38+02:00
- License: CC By-SA
- Tags: mqtt, iot et coap
- Score: 27

Cette dépêche passe en revue des solutions pour transmettre via internet des données de mesure issues de capteurs (domotique, machines instrumentées...).

Protocole MQTT (Message Queuing Telemetry Transport)

- Échanger des données de mesure entre des machines industrielles, des systèmes de domotique, des équipements pédagogiques, des logiciels, des espaces de stockage, des systèmes d'affichage (écran industriel, navigateur web, application sur smartphone...). Les objets connectés sont situés dans des réseaux privés distincts.
- Les échanges de données sont faits en temps réel. Certains flux peuvent être synchronisés. Il peut y avoir besoin d'enregistrer les données avant ou/et après leur transmission.
- Des traitements peuvent éventuellement être faits avant ou/et après une transmission :
  - Sélection (on ne transmet que les données qui changent) ;
  - Conversion des données, mise en forme, tri ;
  - Ajout d'informations (date, lieu, sujet...) ;
  - Simple affichage ou calculs, graphiques, intelligence artificielle...

Les besoins en sécurité :

- On doit pouvoir communiquer d'un réseau à l'autre en passant les différents filtrages (pare-feu, antivirus, translation d'adresse, proxy...).
- Les données doivent être transmises et éventuellement enregistrées avec des méthodes de chiffrement adaptées à chaque besoin de confidentialité.
- Les flux de données doivent être surveillés (détection de panne, code malveillant).
- Les objets connectés et les utilisateurs doivent être identifiés (qui sont-ils ?) et authentifiés (ils doivent légitimer leur demande d'accès). Les droits d'accès doivent être gérés.
- Les données enregistrées doivent être gérées (volume, obsolescence) et sauvegardées.

- Les applications doivent être mises à jour et sauvegardées.
- Les données ne doivent pas être à caractère personnel (anonymisation ou pseudonymisation). Même croisées avec d'autres données, elles ne doivent pas être identifiantes. Si les données ne sont pas à caractère personnel, elles ne doivent pas non plus mettre en danger un établissement (par exemple, donner des indications sur les dates et horaires de fermeture).

HTTP pourrait convenir mais il faut construire un serveur avec des API et c'est un travail de titan. Je n'ai pas trouvé de solution clé en main sous licence libre ou Open Source.

MQTT est la solution idéale. Ce protocole est simple à implémenter (80 pages de spécification) et très répandu dans tous les domaines : domotique et industrie. On le trouve intégré dans quasiment tous les produits de communication récents. Son autre avantage est qu'on ouvre toujours les sessions depuis l'intérieur du réseau privé donc normalement, on n'a pas à modifier le réseau privé pour que ça fonctionne.

AMQP est similaire à MQTT. Il est peu répandu à part dans le secteur bancaire mais il faudra surveiller son évolution. Il y a jusqu'à 3 fois plus de latence qu'avec MQTT.

OPC-UA est une solution performante pour faire communiquer des machines directement entre elles, dans un réseau privé. Il est difficile à implémenter (1 200 pages de spécification) et nécessite d'accéder depuis l'extérieur à un serveur situé dans le réseau privé, d'où sa faible utilisation, même dans l'industrie. Heureusement, on peut l'interfacer avec MQTT.

Le protocole MQTT est un protocole de communication couramment utilisé dans l'internet des objets (IoT) en raison de sa simplicité, de sa basse consommation et de ses performances dans les réseaux à faible débit.

Le protocole MQTT fonctionne comme un service de messagerie instantanée, avec un système de **publication** de messages et d'**abonnement** (en anglais, « publish/subscribe » ou « Pub/Sub »). Des logiciels « clients MQTT » se connectent à un « serveur MQTT » aussi appelé « courtier MQTT » (« MQTT broker » en anglais). Chaque client publie des messages vers le serveur, en associant un « sujet » (« topic » en anglais) à chaque message. Chaque client peut aussi s'abonner à un ou plusieurs sujets. Le serveur ne lui envoie alors que les messages associés à ces sujets :

Les logiciels clients MQTT sont de plus en plus souvent intégrés dans les capteurs, les automates programmables industriels, les interfaces, les passerelles, les afficheurs, les logiciels... Il faut demander aux fournisseurs si la version du protocole MQTT qu'ils utilisent est la même que celle du serveur MQTT et si leur produit peut communiquer avec des serveurs MQTT publics (certains produits ne communiquent qu'avec le serveur MQTT du fournisseur).

Si le serveur MQTT intègre aussi le protocole de communication WebSocket, on peut utiliser un navigateur web pour publier et s'abonner, le navigateur web fait office de client MQTT.

Les sujets des messages sont organisés de manière hiérarchique. Exemple : gendarmerie/drone1/altitude, gendarmerie/drone1/rotor, gendarmerie/drone2/altitude. Pour recevoir les messages de tous les capteurs du drone 1, on s'abonne au sujet gendarmerie/drone1/#. Pour recevoir les messages concernant l'altitude de tous les drones, on s'abonne au sujet gendarmerie/+/altitude.

Les données transmises dans les messages MQTT sont au format « JavaScript Object Notation » ou «

JSON ». Exemple de fichier JSON : > {

```
  "fruits": [
    { "kiwis": 3,
      "mangues": 4,
      "pommes": null
    },
    { "panier": true }
  ],
  "légumes": {
      "patates": "amandine",
      "poireaux": false
    },
    "viandes": ["poisson","poulet","bœuf"]
}
```

Pour transmettre une image, il faut d'abord la transformer en texte avec un codage « base64 ». Cela augmente la taille du message de 33%.

La qualité de Service (QoS) :

Le protocole MQTT dispose d'un mécanisme de qualité de service (« Quality of Service » ou « QoS » en anglais) qui garantit la livraison des messages au client en cas de défaillance d'un appareil ou de la liaison.

Il y a trois niveaux de qualité :

- QoS 0 « At most once » : le message n'est envoyé qu'une seule fois. En cas de défaillance, il se peut qu'il ne soit pas reçu.
- QoS 1 « At least once » : le message est envoyé jusqu'à ce que sa réception soit garantie. En cas de défaillance, le message peut être reçu en double.
- QoS 2 « Exactly once » : chaque message est garanti d'être réceptionné et il n'est réceptionné qu'une seule fois.

Plus le niveau de qualité est élevé, plus la charge réseau augmente.

La vérification de l'intégrité des données n'est pas prévue dans la norme MQTT. On peut éventuellement l'ajouter dans les applications utilisant MQTT (somme de contrôle).

La sécurité :

Le protocole MQTT peut s'utiliser avec le protocole de transport chiffré TLS.

Le protocole MQTT gère l'authentification des clients par mot de passe, certificat, délégation d'authentification.

Exemples de serveurs MQTT (version 5.0), sous licence Open Source :

Mosquitto : https://mosquitto.org (MQTT, MQTT + TLS, WebSocket, WebSockets + TLS). Disponible dans un boîtier sur rail DIN : https://www.directautomation.com.au/mqtt-box.html. EMQX : https://emqx.io (MQTT, MQTT + TLS, WebSocket, WebSockets + TLS). HiveMQ : https://hivemq.com (MQTT, MQTT + TLS, WebSocket, WebSockets + TLS). VerneMQ : https://github.com/vernemq/vernemq (MQTT, MQTT + TLS, WebSocket, WebSockets + TLS). KMQTT :

https://github.com/davidepianca98/KMQTT (MQTT, MQTT + TLS, WebSocket, WebSockets + TLS apparemment pas disponible). NanoMQ : https://nanomq.io

Les plateformes IoT sont des applications web qui intègrent un serveur MQTT et éventuellement d'autres serveurs de communication pour l'IoT, par exemple MQTT-SN pour les réseaux avec une mauvaise qualité de transmission. Elles permettent de gérer facilement les comptes utilisateurs, les droits d'accès des clients MQTT, de visualiser les publications et les abonnements, de paramétrer le stockage sur des bases de données locales ou distantes etc.

Plateformes IoT sous licence Open Source :

- Kuzzle : https://kuzzle.io
- Openremote : https://openremote.io
- ThingsBoard : https://thingsboard.io
- SiteWhere : https://sitewhere.io/fr/ (conçue pour être installée dans le cloud)
- Mainflux : https://docs.mainflux.io

Plateformes IoT sous licences non libres (liste non exhaustive, il en existe des centaines) :

- Orange : https://liveobjects.orange-business.com
- Google : https://cloud.google.com/solutions/iot
- Amazon : https://docs.aws.amazon.com/fr_fr/iot/latest/developerguide/mqtt.html
- IBM : https://internetofthings.ibmcloud.com
- Microsoft Azure : https://docs.microsoft.com/fr-fr/azure/iot-hub/iot-hub-mqtt-support
- Scaleway : https://www.scaleway.com/fr/iot-hub/
- Siemens : https://siemens.mindsphere.io/en
- Kaa Iot : https://www.kaaiot.com/advantages/platform
- ThingSpeak : https://thingspeak.com, de MathWorks, éditeur de MATLAB et Simulink.
- CloudMQTT : https://cloudmqtt.com.
- Ignition IIoT d'Inductive Automation, pour l'industrie : https://inductiveautomation.com/distributor-landing/francais.
- Cogent, pour l'industrie : https://cogentdatahub.com/products/datahub-smart-mqtt-broker/.
- Flow Software : https://www.flow-software.com/information-platform.

Quand on choisit une plateforme IoT, il faut vérifier qu'elle soit compatible avec les protocoles de communication qu'on utilise et qu'elle intègre ou permette de communiquer avec les services dont on a besoin (stockage, tableaux de bord, calculs…).

Des prestataires développent des plateformes IoT sur mesure :

- WMW : https://www.wmw-hub.com
- ProcessOne : https://www.process-one.net/en/ejabberd/ (MQTT + XMPP + SIP).

Clients et passerelles MQTT :

La liste des clients MQTT est consultable sur https://mqtt.org/software/. Pour écrire un client MQTT :

- Node-RED : https://nodered.org
- Kura : https://eclipse.org/kura/

MQTT Websocket Toolkit permet d'utiliser un navigateur web comme client MQTT : https://www.emqx.com/en/mqtt/mqtt-websocket-toolkit

MQTT2Excel enregistre un flux MQTT dans un fichier Excel : https://github.com/gsampallo/mqtt2excel. Streamsheets est tableur qui peut s'abonner à des flux MQTT et les afficher dans des feuilles de calcul : https://github.com/eclipse/streamsheets. Afficher un flux de données dans LibreOffice Calc ou dans Google Sheets : https://www.mathieupassenaud.fr/iot-platform/.

ChirpStack est une passerelle LoRaWAN -> serveur MQTT : https://www.chirpstack.io. Elle est utilisée par exemple dans la passerelle 1Gate d'ATIM : https://atim.com.

Outils de test et de développement, documentation :

Serveurs MQTT de test :

- https://www.emqx.com/en/mqtt/public-mqtt5-broker
- https://test.mosquitto.org(ne fonctionne pas en permanence, ne pas utiliser pour une démonstration)
- http://www.mqtt-dashboard.com
- http://mqtt.psykokwak.com:8080

MQTT Tools : https://github.com/eerimoq/mqttools

MQTT Data Generator : https://github.com/ufthelp/MQTT-Generator

Les spécifications du protocole MQTT sont publiées par l'organisme de normalisation OASIS, la dernière version est la version 5.0 :
https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=mqtt

La version 3.1.1 des spécifications de MQTT est parfois complétée avec les spécifications Eclipse Sparkplug : https://sparkplug.eclipse.org.

Discussions sur le canal reddit r/MQTT : https://www.reddit.com/r/MQTT/

Discussions sur le canal IRC #mqtt : https://libera.chat