MQTT : Message Queuing Telemetry Transport

Version du 06 avril 2020

Licence et droits d'utilisation

Vous êtes libre de :

- Copier, distribuer, afficher et présenter ce support.
- Créer des variantes de ce support.
- Utiliser à des fins commerciales ce support.

En respectant les conditions suivantes :

- Attribution : vous devez mentionner le nom de l'auteur.
- Share Alike : Si vous apportez des modifications à ce support vous devez appliquer la même licence.
- Vous devez afficher et expliquer clairement les termes de la licence de ce support.

L'entreprise OperaMetrix

- Entreprise de conseil en systèmes et logiciels informatiques fondée en 2019
- Locaux : Toulouse (France)
- Association avec FullSave, opérateur télécom et hébergeur toulousain

- Notre expertise : **Monitoring et Supervision des équipements industriels, télémétrie, accès et contrôle à distance, connexions multi-sites et gestion des alertes**

- Nos activités : étude, développement, hébergement, support et formation
- Notre engagement : développer de nouvelles solutions de monitoring et de supervision en parralèle des solutions existantes

Comment nous suivre ?

https://www.operametrix.fr

https://twitter.com/OperaMetrix

https://www.linkedin.com/company/operametrix

1 Introduction à MQTT

Origines, standard, acteurs importants
et objectifs du protocole

MQTT est un protocole de transport de message de type client/serveur orienté publish/subscribe

- Léger, ouvert, simple et défini pour être facile à implémenter
- Utilisable pour de multiples usages dont les communications M2M et pour l'IoT
- Basé sur TCP/IP avec support de TLS et de WebSocket

MQTT est implémenté au dessus de TCP/IP ou d'autres protocoles qui fournissent des communications ordonnées, sans pertes et bi-directionnelles.

- Couche de transport : TCP, WebSocket
- Keepalive de niveau MQTT
- Overhead très faible
- Format de charge utile non imposée

MQTT est défini par l'OASIS en deux versions, une version 3.1.1 puis une version 5.0 :

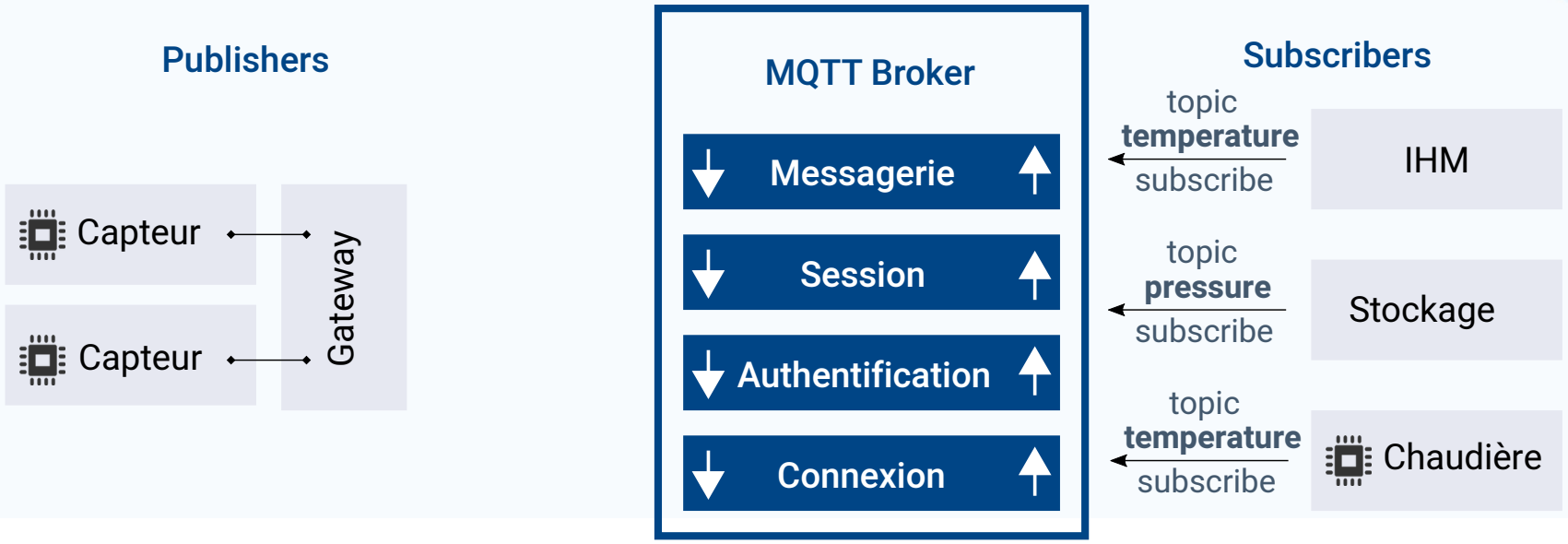
MQTT Version 3.1.1. Edited by Andrew Banks and Rahul Gupta.
29 October 2014. OASIS Standard.
http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html.
Latest version: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html.

MQTT Version 5.0. Edited by Andrew Banks, Ed Briggs, Ken Borgendale, and RahulGupta.
07 March 2019. OASIS Standard.
https://docs.oasis-open.org/mqtt/mqtt/v5.0/os/mqtt-v5.0-os.html.
Latest version: https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html.

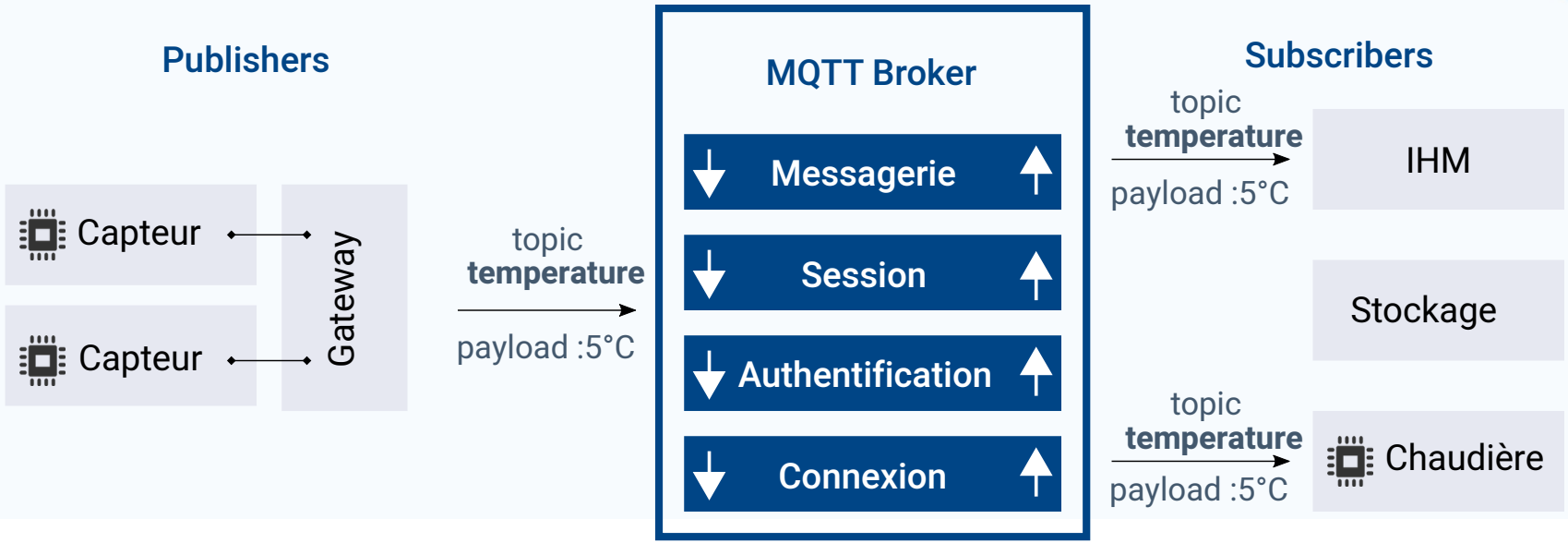
2 Concepts fondamentaux

Paradigme publish/subscribe, définition d'un client, d'un broker et des topics

Paradigme Publish/Subscribe



Paradigme Publish/Subscribe



Clients et serveurs

Un broker MQTT est un serveur TCP et les publisher/subscriber sont des clients TCP.

Un client peut être subscriber, publisher ou les deux !

Payload (charge utile)

MQTT n'impose pas de format de donnée pour la charge utile !

La seule limitation est la taille maximal autorisée qui est de 256 Mio par publication.

Le format des données peut être binaire, JSON, XML, string, protobuf etc ...

Topics

Un topic est constitué d'un ensemble de chaînes de caractères qui sont séparées par des slash.

Exemple : « maison/cuisine/four/temperature/1 », « client/1/site/toulouse/bureau/45 »

Topics wildcards

Afin de sélectionner plusieurs topics en un seul abonnement, il est possible d'utiliser des caractères spéciaux :

- '+' signifie qu'importe ce qu'il y a à ce niveau de l'arborescence
- '#' signifie qu'importe ce qu'il y a en dessous dans l'arborescence

<lieu> / <pièce> / <électroménager> / <capteur> / <nom>

Exemple 1 : « maison/cuisine/+/temperature/+ »

Exemple 2 : « maison/cuisine/four/# »

Utilitaires mosquitto-clients


```
root@server:~# apt install mosquitto mosquitto-clients

root@server:~# systemctl start mosquitto

root@server:~# mosquitto_sub -v -t topic/# -h iot.broker.net

root@server:~# mosquitto_pub -t topic/# -m hello -h iot.broker.net
```

Astuce : utiliser l'option -d pour l'analyse protocolaire

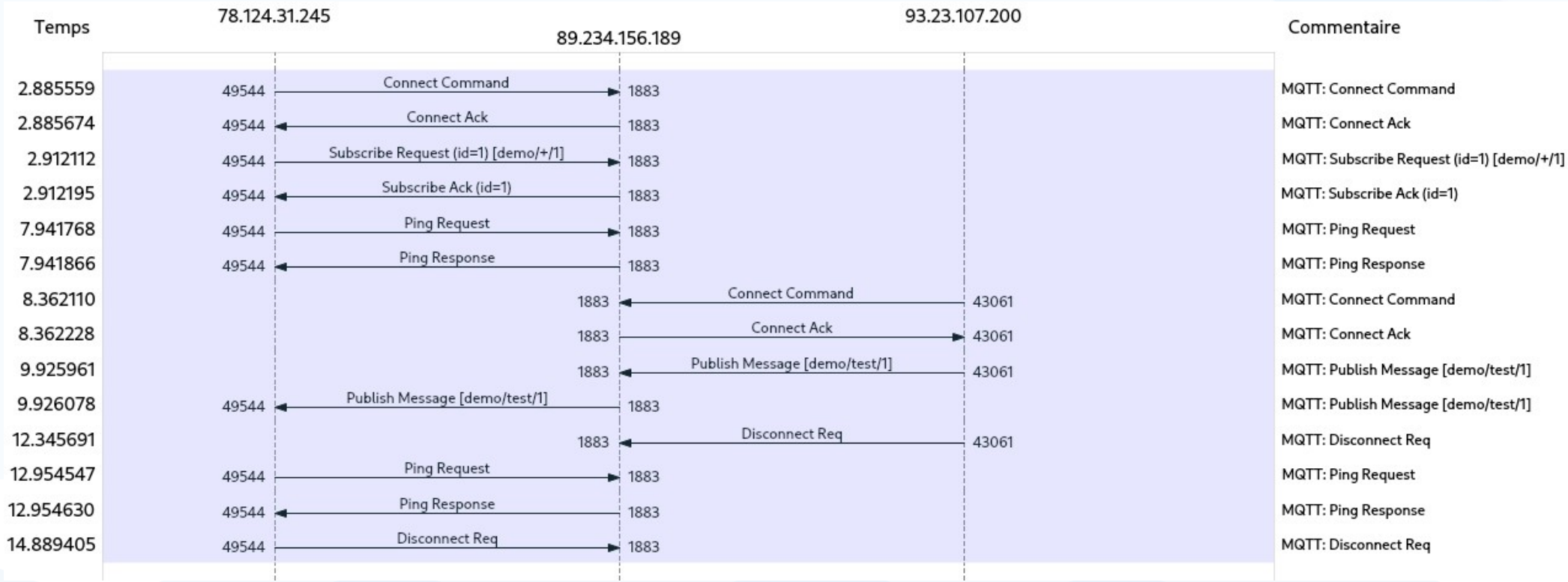
3 Analyse protocolaire MQTT

Codage des différents paquets via le protocole MQTT

Types de messages MQTT

• CONNECT,CONNACK	Requête de connexion d'un client au broker
• PUBLISH,PUBACK	Gestion d'une publication pour QoS 0 et 1
• PUBREC,PUBREL,PUBCOMP	Gestion d'une publication pour QoS 2
• SUBSCRIBE,SUBACK	Requête de souscription d'un client à un topic
• UNSUBSCRIBE,UNSUBACK	Reqûete de désabonnement d'un client à un topic
• PINGREQ,PINGRESP	Ping régulier permettant le mécanisme de keepalive
• DISCONNECT	Déconnexion d'un client
• AUTH	Gestion d'un algorithme d'authentification par challenge

Flux simple publish/subscribe



CONNECT


```
MQ Telemetry Transport Protocol, Connect Command
    Header Flags: 0x10, Message Type: Connect Command
        0001 .... = Message Type: Connect Command (1)
        .... 0000 = Reserved: 0
    Msg Len: 35
    Protocol Name Length: 4
    Protocol Name: MQTT
    Version: MQTT v3.1.1 (4)
    Connect Flags: 0x02, QoS Level:
        At most once delivery (Fire and Forget), Clean Session Flag
        0... .... = User Name Flag: Not set
        .0.. .... = Password Flag: Not set
        ..0. .... = Will Retain: Not set
        ...0 0... = QoS Level: At most once delivery (Fire and Forget) (0)
        .... .0.. = Will Flag: Not set
        .... ..1. = Clean Session Flag: Set
        .... ...0 = (Reserved): Not set
    Keep Alive: 5
    Client ID Length: 23
    Client ID: mosq-bFHY8QEyampvedac2O
```

CONNACK


```
MQ Telemetry Transport Protocol, Connect Ack
    Header Flags: 0x20, Message Type: Connect Ack
        0010 .... = Message Type: Connect Ack (2)
        .... 0000 = Reserved: 0
    Msg Len: 2
    Acknowledge Flags: 0x00
    Return Code: Connection Accepted (0)
```

0x00 Success	0x8C Bad authentication method
0x80 Unspecified error	0x90 Topic Name invalid
0x81 Malformed Packet	0x95 Packet too large
0x82 Protocol Error	0x97 Quota exceeded
0x83 Implementation specific error	0x99 Payload format invalid
0x84 Unsupported Protocol Version	0x9A Retain not supported
0x85 Client Identifier not valid	0x9B QoS not supported
0x86 Bad User Name or Password	**0x9C Use another server**
0x87 Not Authorized	**0x9D Server moved**
0x88 Server unavailable	0x9F Connection rate exceeded
0x89 Server busy	
0x8A Banned	

PING REQUEST


```
MQ Telemetry Transport Protocol, Ping Request
    Header Flags: 0xc0, Message Type: Ping Request
        1100 .... = Message Type: Ping Request (12)
        .... 0000 = Reserved: 0
    Msg Len: 0
```

PING RESPONSE


```
MQ Telemetry Transport Protocol, Ping Response
    Header Flags: 0xd0, Message Type: Ping Response
        1101 .... = Message Type: Ping Response (13)
        .... 0000 = Reserved: 0
    Msg Len: 0
```

SUBSCRIBE REQUEST


```
MQ Telemetry Transport Protocol, Subscribe Request
    Header Flags: 0x82, Message Type: Subscribe Request
        1000 .... = Message Type: Subscribe Request (8)
        .... 0010 = Reserved: 2
    Msg Len: 13
    Message Identifier: 1
    Topic Length: 8
    Topic: demo/+/1
    Requested QoS: At most once delivery (Fire and Forget) (0)
```

SUBACK


```
MQ Telemetry Transport Protocol, Subscribe Ack
    Header Flags: 0x90, Message Type: Subscribe Ack
        1001 .... = Message Type: Subscribe Ack (9)
        .... 0000 = Reserved: 0
    Msg Len: 3
    Message Identifier: 1
    Granted QoS: At most once delivery (Fire and Forget) (0)
```

PUBLISH


```
MQ Telemetry Transport Protocol, Publish Message
    Header Flags: 0x30, Message Type: Publish Message
        0011 .... = Message Type: Publish Message (3)
        .... 0... = DUP Flag: Not set
        .... .00. = QoS Level: At most once delivery (Fire and Forget) (0)
        .... ...0 = Retain: Not set
    Msg Len: 14
    Topic Length: 11
    Topic: demo/test/1
    Message: 1
```

DISCONNECT


```
MQ Telemetry Transport Protocol, Disconnect Req
    Header Flags: 0xe0, Message Type: Disconnect Req
        1110 .... = Message Type: Disconnect Req (14)
        .... 0000 = Reserved: 0
    Msg Len: 0
```

0x00 Normal disconnection	0x94 Topic Alias invalid
0x04 Disconnect with Will Message	0x95 Packet too large
0x80 Unspecified error	0x96 Message rate too high
0x81 Malformed Packet	0x97 Quota exceeded
0x82 Protocol Error	**0x98 Administrative action**
0x83 Implementation specific error	0x99 Payload format invalid
0x87 Not authorized	0x9A Retain not supported
0x89 Server busy	0x9B QoS not supported
0x8B Server shutting down	**0x9C Use another server**
0x8D Keep Alive timeout	**0x9D Server moved**
0x8E Session taken over	0x9E Shared Subscriptions not supported
0x8F Topic Filter invalid	0x9F Connection rate exceeded
0x90 Topic Name invalid	**0xA0 Maximum connect time**
0x93 Receive Maximum exceeded	0xA1 Subscription identifiers not supported
	0xA2 Wildcard subscriptions not supported

4 Connexion et session

Gestion de la connexion au broker et allocation d'une session

ClientID

Un ClientID est un identifiant unique de session généré par le client ou par le broker.

```
[MQTT-3.1.3-5]
The Server MUST allow ClientIds which are between 1 and 23 UTF-8 encoded bytes
in length, and that contain only the characters
"0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ"
```

Connexion au broker


```
MQ Telemetry Transport Protocol, Connect Command
    Header Flags: 0x10, Message Type: Connect Command
        0001 .... = Message Type: Connect Command (1)
    Msg Len: 35
    Protocol Name Length: 4
    Protocol Name: MQTT
    Version: MQTT v3.1.1 (4)
    Connect Flags: 0x02, QoS Level:
        0... .... = User Name Flag: Not set
        .0.. .... = Password Flag: Not set
        ..0. .... = Will Retain: Not set
        ...0 0... = QoS Level: At most once delivery (Fire and Forget) (0)
        .... .0.. = Will Flag: Not set
        .... ..1. = Clean Session Flag: Set
        .... ...0 = (Reserved): Not set
    Keep Alive: 5
    Client ID Length: 23
    Client ID: mosq-bFHY8QEyampvedac2O
```

Session d'un client

Une session est un ensemble d'états et d'options sauvegardés dans la mémoire du broker au sujet d'un client.

Une session peut être persistante entre plusieurs connexions !
La persistance est gérée via le bit « clean start ».

5 Authentification et ACL

Comment gérer l'accès aux ressources d'un broker et limiter l'usage

Authentification en v3.1.1


```
MQ Telemetry Transport Protocol, Connect Command
    Header Flags: 0x10, Message Type: Connect Command
    Msg Len: 56
    Protocol Name Length: 4
    Protocol Name: MQTT
    Version: MQTT v3.1.1 (4)
 (Fire and Forget), Clean Session Flag
        1... .... = User Name Flag: Set
        .1.. .... = Password Flag: Set
        ..0. .... = Will Retain: Not set
        ...0 0... = QoS Level: At most once delivery (Fire and Forget) (0)
        .... .0.. = Will Flag: Not set
        .... ..1. = Clean Session Flag: Set
        .... ...0 = (Reserved): Not set
    Keep Alive: 60
    Client ID Length: 23
    Client ID: mosq-mgjaogVPyADnAcVzwJ
    User Name Length: 9
    User Name: ngonzalez
    Password Length: 8
    Password: p4ssw0rd
```

Nouveautés de la version 5

Définition d'un message de type « AUTH » qui permet de faire des challenges cryptographiques lors de la phase de connexion. SCRAM-SHA-1 pour SCRAM avec SHA-1 ou GS2-KRB5 pour Kerberos.

Permission d'utiliser uniquement un mot de passe afin de pouvoir utiliser des tokens (OAuth 2.0)

Access Control List

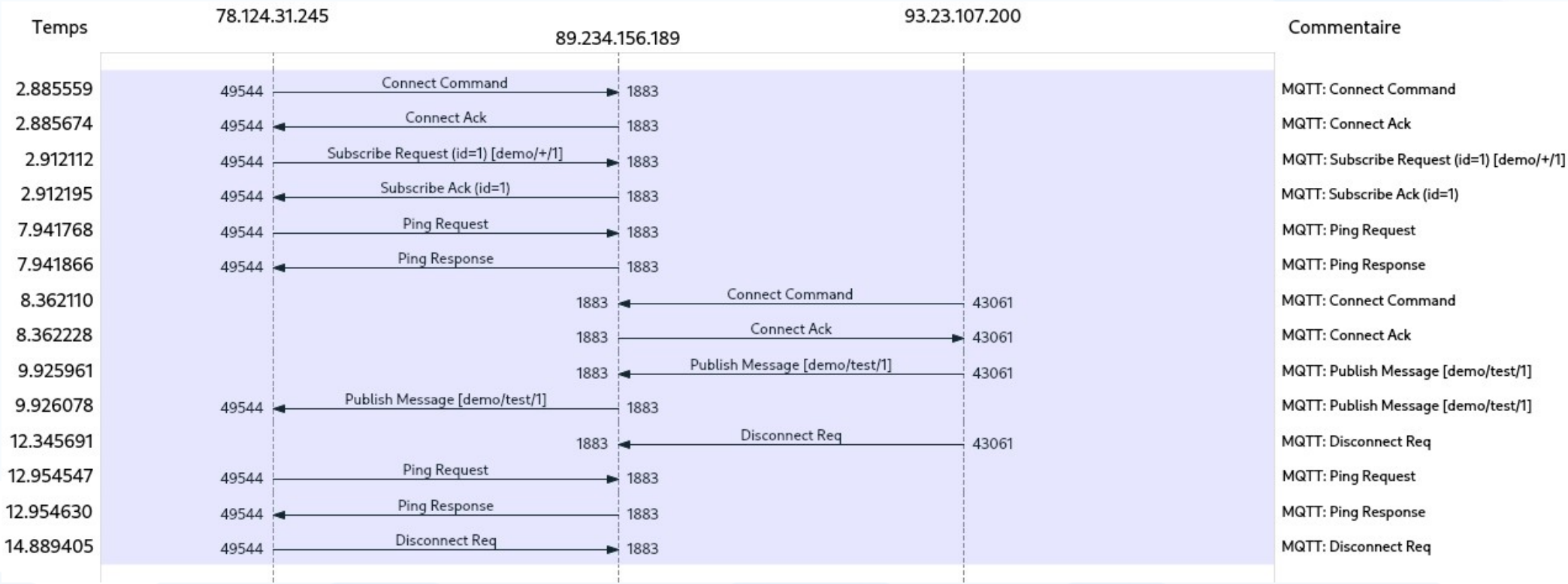
Les ACLs ne sont pas définies dans le protocole et sont implémentées dans le broker pour définir l'accès aux topics.

Association d'un clientID ou d'un utlisateur à un ensemble de topics accessibles en lecture, écriture ou les deux.

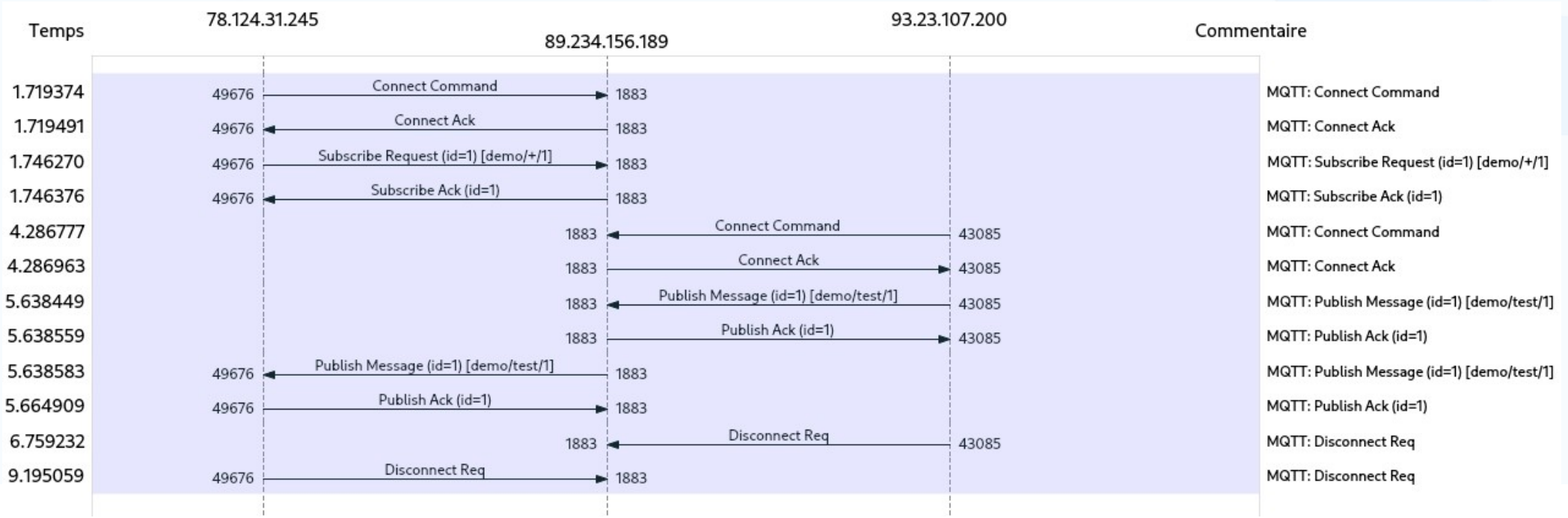
6 Qualité de service (QoS)

Quelles sont les différentes qualités de service proposées dans le standard ?

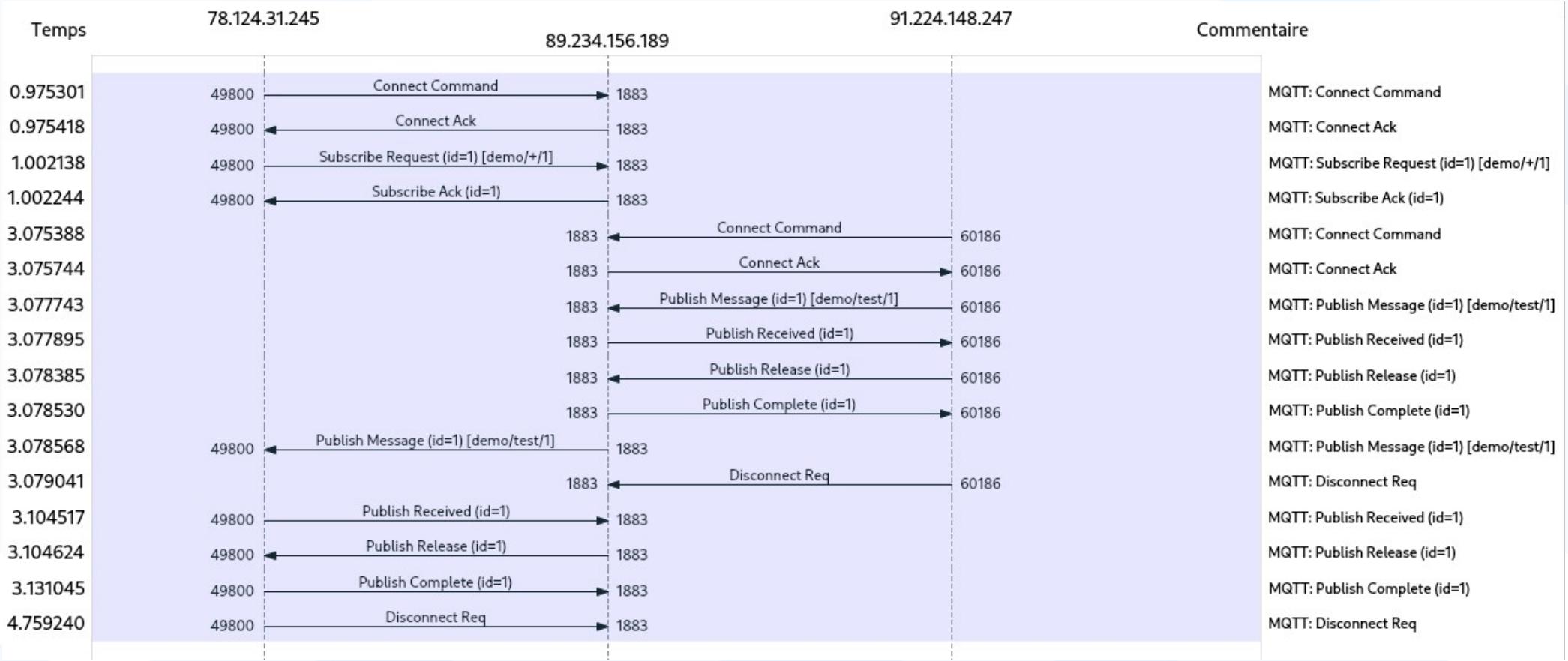
QOS 0 : fire and forget



QOS 1 : at least once delivery



QOS 2 : at most once delivery



7 Retain, Last Will et Testament

Mémorisation de certains messages

Messages Retain

La définition d'un message comme « Retain » permet de garder en mémoire dans le broker le dernier message de ce topic.

Ce mécanisme permet à un subscriber de recevoir lors de sa connexion le dernier message des Topics auxquels il s'abonne.

Très utile pour les IHMs ou pour retenir le statut d'un client.

Last Will / Testament

Un client peut lors de sa connexion indiquer un message à publier sur un topic si une déconnexion anormale se produisait.

C’est le broker lui même qui publie ce message si la connexion se termine anormalement.
Exemple : connexion TCP fermée sans DISCONNECT, keepalive expiré etc …

Nouveauté de la version 5 : il est possible de définir un délai avant la publication du testament

8 Autres nouveautés de la v5

Shared Subscriptions, Request-Response Pattern et Topic Alias

Propriétés MQTT dans le header

0x01 Payload Format Indicator
0x02 Message Expiry Interval
0x03 Content Type
0x08 Response Topic
0x09 Correlation Data
0x0B Subscription Identifier
0x11 Session Expiry Interval
0x12 Assigned Client Identifier
0x13 Server Keep Alive
0x15 Authentication Method
0x16 Authentication Data
0x17 Request Problem Information
0x18 Will Delay Interval
0x19 Request Response Information
0x1A Response Information
0x1C Server Reference
0x1F Reason String
0x21 Receive Maximum
0x22 Topic Alias Maximum
0x23 Topic Alias
0x24 Maximum QoS
0x25 Retain Available
0x26 User Property
0x27 Maximum Packet Size
0x28 Wildcard Subscription Available
0x29 Subscription Identifier Available
0x2A Shared Subscription Available

Request-Response Pattern

Un client peut publier un message sur un topic avec comme propriétés :

- Un « Response Topic » : topic sur lequel doit être publié la réponse
- Une « Correlation Data » : donnée qui va servir de corrélation entre la requète et la réponse

Le subscriber qui reçoit la requète la traite et publie la réponse dans le topic indiqué précédemment. Il copie dans les propriétés la « Correlation Data ».

L'initiateur qui a souscrit à ce topic précédemment reçoit la réponse.

Topic Alias

Un client peut déclarer au broker un « topic alias » qui est une correspondance entre un topic et un entier.

Lors des futurs échanges entre le broker et le client cet entier peut être utilisé en remplacement de la chaîne de caractères du topic.

Shared subscriptions

Il est défini un type particulier de topics qui permettent de faire de la répartition de charge entre différents subscribers.

```
$share/{ShareName}/{filter}

$share : préfix obligatoire des topics partagés
ShareName : identifiant d'un pool de subscribers pour le load balancing
Filter : topic utilisé pour la publication des messages
```

9 Mise en production

Retour d'expérience sur la mise en production d'une architecture MQTT

Choix d'un broker MQTT

Liste des meilleurs brokers MQTT du marché :

Eclipse Mosquitto : développé par la fondation Eclipse, open source sous licence Apache v2.0

VerneMQ : développé par VerneMQ/Erlio, open source sous licence Apache v2.0

HiveMQ : développé par dc-square GmbH sous licence propriétaire

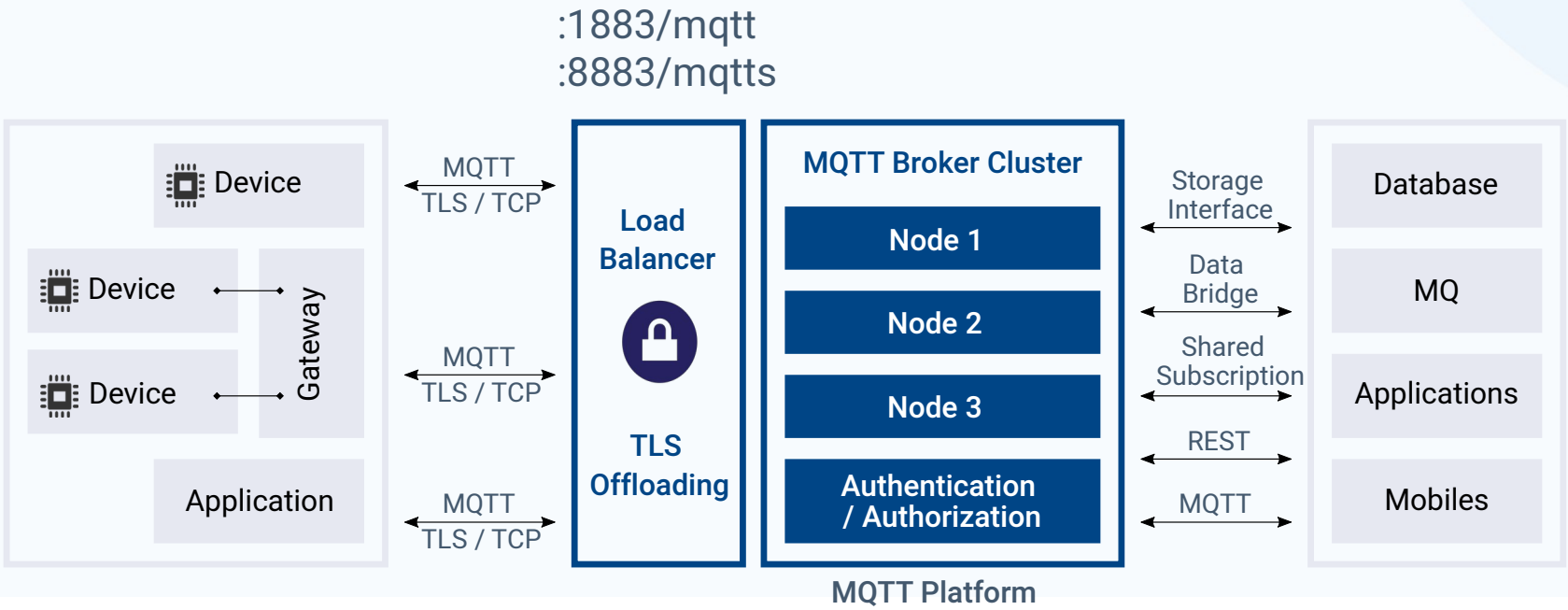
EMQ X : développé par EMQ, open source sous licence Apache v2.0

Choix d'un broker MQTT

Implementation	MQTT-SN (MQTT v1.2)	MQTT 3.1	MQTT 3.1.1	MQTT 5.0	SSL/TLS	TCP	WS/WSS
EMQ	Yes	Yes	Yes	Yes	Yes	Yes	Yes
flespi		Yes	Yes	Yes[42]	Yes	Yes	
HiveMQ		Yes (only for broker)	Yes	Yes [43]	Yes	Yes	Yes
IBM WIoTP Message Gateway		Yes	Yes	Yes	Yes	Yes	Yes
JoramMQ	Yes	Yes	Yes	Yes	Yes	Yes	Yes
M2Mqtt		Yes	Yes		Yes	Yes	
Machine Head							
moquette		Yes	Yes			Yes	
Mosquitto		Yes	Yes	Yes	Supports certificate-based and pre-shared-key-based SSL/TLS, general support for SSL/TLS across bridges[44]	Yes	Yes
MQTT-C		Yes	Yes		Yes	Yes	
net-mqtt		Yes	Yes	Yes	Yes	Yes	Yes
Paho MQTT	Yes	Yes	Yes	Yes (only in C and Java client library) [45]	Yes	Yes	Yes
Solace PubSub+			Yes		Yes	Yes	Yes
Thingstream	Yes			Yes	Yes	Yes	
VerneMQ		Yes	Yes	Yes	Yes	Yes	Yes
wolfMQTT	Yes		Yes	Yes	Yes	Yes	
Bevywise Networks	Yes	Yes	Yes		Yes	Yes	Yes

https://en.wikipedia.org/wiki/Comparison_of_MQTT_implementations

Architecture d'un cluster MQTT



Choix d'un backend d'Authn/Authz

Liste des backends généralement disponibles :

- **Fichier**
- **Redis**
- **Postegresql**
- **MongoDB**
- **Administration externe** … ???

Monitoring et alerting

VerneMQ propose nativement une interface Prometheus !

Cette interface très riche permet de surveiller de nombreuses métriques :

- Nombre de connexions
- Nombre de sessions
- Débit échangés
- Informations sur le nombre d'authentifications

etc ...

10 Problématiques de sécurité

Notes sur la gestion de clientID, le chiffrement TLS et le proxying

Différentes menaces de sécurité

- Compromission des objets
- Accès aux données dans l'objet ou dans le serveur
- Utilisation des effets de bords du protocol (exemple : « timing attacks »)
- Attaque par déni de service (création de socket ou de session)
- Interception, altération, re-routage, man-in-the-middle dans le réseau
- Injection de paquets de contrôle MQTT (spoofing)
- ClientID hijacking

http://docs.oasis-open.org/mqtt/mqtt-nist-cybersecurity/v1.0/mqtt-nist-cybersecurity-v1.0.pdf

1. Identification des risques

- Inventaire des logiciels et des matériels
- Carte du réseau à jour
- Définition de la surface d’attaque
- Versionnement des suites cryptographiques
- Localisation du flot de données
- Localisation des clients, des brokers et des applications
- Identification des communications internes et externes

2. Protection

- Formation des utilisateurs
- Authentification clients ↔ serveur
- Usage de PKI (TLS, VPN …)
- Chiffrement des payloads et vérification de l'intégrité
- Non répudiation des messages
- Utilisation de générateurs de nombres aléatoires « sécurisés »
- Entière compatibilité avec le standard
- Mécanisme de déconnexion automatique des connexions fantômes
- Rate limiting et black listing des IP
- Renégociation régulière des sessions (clés de session)
- Utilisation de hardware sécurisé avec sécurisation des certificats (TPM)

3. Détection

- Tentatives répétées de connexion au broker
- Déconnexions anormales
- Vérification de la présence des clients
- Surveillance physique du hardware
- Tentatives répétées d’authentification
- Topic scanning
- Envois de messages à des topics sans subscribers …
- Clients qui se connectent mais n’envoient pas de données

4. Répondre à l'attaque

- Révocation des certificats
- Révocation des credentials
- Déconnexion physique des hardwares
- Blocage des canaux de télémétries compromis
- Ajout de règles dans le firewall
- Extinction des serveurs et des brokers compromis

5. Restauration après incident

- Reinstallation des services compromis
- Faire une reconstitution de l'incident
- Utilisation d'un site alternatif en mode dégradé
- Passe sur les règles des firewall
- Création des nouveaux certificats
- Inspection physique des matériels
- Déploiement des sauvegardes
- Passe sur la gestion des outils cryptographiques

Félicitation, vous êtes enfin MQTT Senior Expert !