2.4. GIAO THỨC MQTT

2.4.1. Các khái niệm cơ bản

MQTT được phát triển bởi IBM và Eurotech, phiên bản mới nhất là MQTT 3.1

MQTT (Giao vận tầm xa) là giao thức truyền message theo mô hình cung cấp/thuê bao publish/subcribe. Nó dựa trên một Broker (điểm trung gian) "nhẹ" (khá ít xử lý), và được thiết kế có tính mở (không đăng trưng cho ứng dụng nào), rất đơn giản và dễ để tích hợp. MQTT phù hợp cho các ứng dụng M2M(Mobile to mobile), WSN (Wireless Sensor Networks) hay IoT (Internet of Things). Những đặc trưng này khiến MQTT rất lý tưởng để sử dụng trong các môi trường bị giới hạn tài nguyên như:

- Những nơi mà giá mạng quá đắt hoặc băng thông thấp, hoặc độ tin cậy thấp
- Khi chạy trên một thiết bị nhúng bị giới hạn về tài nguyên tốc độ và bộ nhớ
- Các đặc trưng chính của giao thức bao gồm:
- Dạng truyền message cung cấp/thuê bao (publish/subcribe) cung cấp việc truyền tin phân tán 1-nhiều
- Việc truyền message là luôn không quan tâm đến nội dung truyền
- Dựa trên nền TCP/IP để cung cấp đường truyền
- Có 3 loại QoS được đưa ra:

+ "Hầu như chỉ 1 lần " : "At most once", message được truyền nhận dựa hoàn toàn vào tính tin cận của TCP/IP. Việc mất hoặc lặp message có thể xảy ra. Ở QoS này, có thể ví dụ 1 trường hợp sử dụng: như trong môi trường sensor mà việc mất 1 gói dữ liệu tại 1 thời điểm không ảnh hưởng đến toàn bộ quá trình.

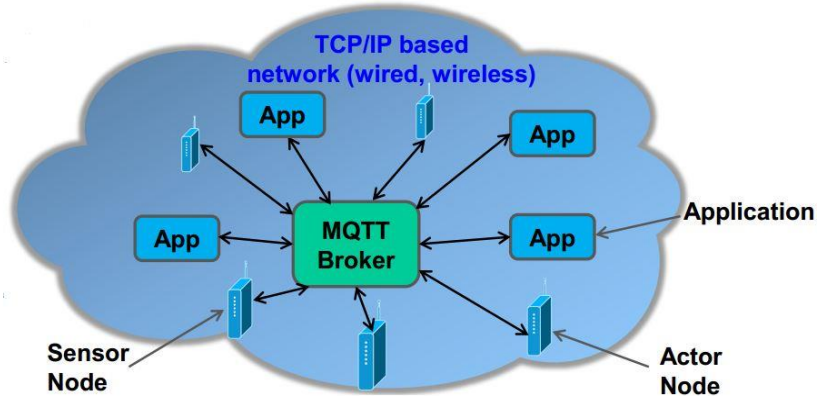
+ "Ít nhất 1 lần" : "At least once", các message được đảm bảo nhận được nhưng có thể xảy ra lặp

+ "Chính xác chỉ 1 lần" : "Exactly once", message được đảm bảo đến nơi đúng 1 lần. Ở level này, các hệ thống thanh toán, nơi mà việc lặp hay mất message

có thể gây ra việc tính tiền bị sai.

- Dữ liệu bao bọc dữ liệu truyền (overhead) nhỏ (độ dài cố định luôn là 2 byte), and là gia thức giảm đến mức tối thiểu traffic đường truyền.

- Một cơ chế để thông báo đến các thuê bao khi đường truyền bị đứt bất thường, sử dụng Last Will và Testament feature.



Hình 2.1. Ví dụ về kết nối trong mạng lưới MQTT

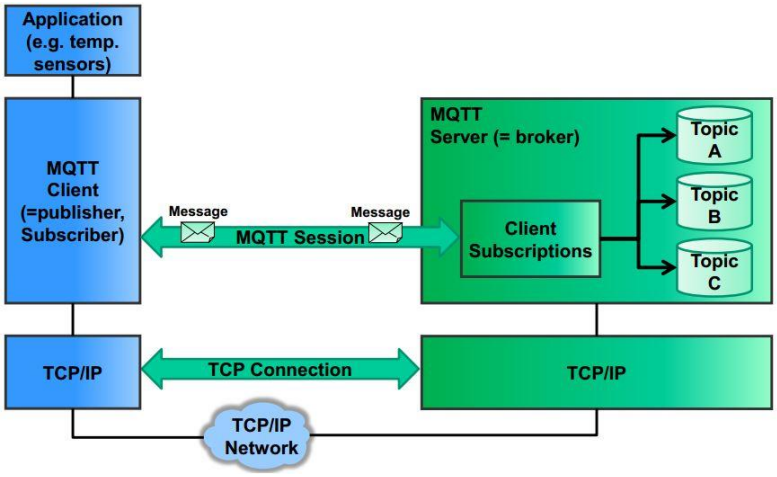
- Light sensor liên tục gửi dữ liệu về broker.

- Ứng dụng điều khiển tòa nhà nhận dữ liệu từ broker để quyết định trạng thái các thiết bị trong nhà.

- Ứng dụng gửi tín hiệu điều khiển actor node thông qua broker.

2.4.2. Mô hình MQTT

Các thành phần chính của MQTT là clients, servers (=brokers), sessions, subscriptions và topics



Hình 2.2. Mô hình cơ bản của giao thức MQTT

MQTT client (publisher, subscriber): Client thực hiện subscribe đến topics để publish và receive các gói tin.

MQTT server (broker): Servers thực hiện run các topic, đồng thời nhận subscriptions từ clients yêu cầu các topics, nhận các messages từ clients và forward chúng.

Topic: Về mặt kỹ thuật, topics là các hàng đợi chứa message. Về logic, topics cho phép clients trao đổi thông tin và dữ liệu.

Session: Một session được định nghĩa là kết nối từ client đến server. Tất cả các giao tiếp giữa client và server đều là 1 phần của session.

Subscription: Không giống như sessions, subscription về mặt logic là kết nối từ client đến topic. Khi thực hiện subscribed đến topic, client có thể trao đổi messages với topic. Subscriptions có thể ở trạng thái 'transient' hoặc 'durable', phụ thuộc vào cờ clean session trong gói Connect.

Message: Messages là các đơn vị dữ liệu được trao đổi giữa các topic clients.

2.4.3. Các gói tin quan trọng của giao thức MQTT

a. Định dạng của message

* Phần header cố định

Tất cả các message luôn chứa phần cố định theo bảng 2.4

Bảng 2.1 Header cố định

bit	7	6	5	4	3	2	1	0
byte 1	Loại Message				Cờ DUP	QoS level		RETAIN
byte 2	Độ dài còn lại							

Byte 1 : Chứa loại Message và các cờ (DUP, QoS level, and RETAIN).

Byte 2 : (Ít nhất 1 byte) quy định độ dài còn lại.

Loại Message

Vị trí: byte 1, bits 7-4.

Một số 4-bit không dấu diễn tả các giá trị được miêu tả dưới bảng 2.5

Bảng 2.2. Loại message

Từ gợi nhớ	Giá trị thứ tự	Miêu tả
Reserved	0	Chưa dùng
CONNECT	1	Client yêu cầu kết nối đến Server
CONNACK	2	Kết nối được chấp nhận
PUBLISH	3	Xuất bản message
PUBACK	4	Xuất bản message được chấp nhận
PUBREC	5	Xuất bản đã được nhận (đảm bảo nhận được part 1)
PUBREL	6	Xuất bản release (đảm bảo nhận được part 2)
PUBCOMP	7	Xuất bản hoàn thành (đảm bảo nhận được part 3)
SUBSCRIBE	8	Yêu cầu subcribe từ client
SUBACK	9	Yêu cầu subcriber được chấp nhận
UNSUBSCRIBE	10	Yêu cầu unsubcribe
UNSUBACK	11	Yêu cầu unsubcribe được chấp nhận
PINGREQ	12	Request PING
PINGRESP	13	Response PING
DISCONNECT	14	Client đang mất kết nối
Reserved	15	Reserved

- Các cờ

Bit còn lại của byte đầu chứa các trường DUP, QoS và RETAIN. Vị trí các bit

và ý nghĩa được miêu tả trong bảng 2.6

Bảng 2.3. Bảng các cờ

Vị trí bit	Tên viết gọn	Miêu tả
3	DUP	Nhận lặp lại
2-1	QoS	Quality of Service (chất lượng dịch vụ)
0	RETAIN	cờ RETAIN

*** DUP :**Vị trí byte 1, bit 3.

Cờ này được bật khi client hoặc server đang cố chuyển lại một gói PUBLISH, PUBREL, SUBSCRIBE hoặc UNSUBSCRIBE. Giá trị này được sử dụng trong các mesage mà có QoSS lớn hơn 0 và yêu cầu ACK. Khi bit DUP được set, phần header thay đổi sẽ chứa Message ID. Nhìn vào giá trị này sẽ biết được gói tin đã nhận được trước đó hay không. Nó không nên sử dụng để tin ngay rằng có duplicates hay không.

*** QoS :** Vị trí byte 1, bits 2-1.

Cờ này sẽ cho biết độ đảm bảo việc nhận message PUBLISH. Giá trị của QoS được miêu tả trong bảng 2.7

Bảng 2.4. Giá trị QoS

Giá trị QoS	bit 2	bit 1	Miêu tả		
0	0	0	Cùng lắm là 1 lần	Gửi rồi quên ngay	<=1
1	0	1	Ít nhất 1 lần	Xác nhận bằng ACK	>=1
2	1	0	Chính xác 1 lần	Nhận đảm bảo	=1
3	1	1	Chưa dùng		

*** RETAIN**: Vị trí byte 1, bit 0.

Cờ này chỉ được sử dụng ở message PUBLISH. Khi client gửi 1 message PUBLISH đến server, nếu cờ Retain được set (1), thì server phải hiểu rằng nên giữ message này ngay cả sau khi chuyển nó đến các subcribers hiện tại. Khi có 1 subcription mới được thiết lập trên 1 topic, message cuối cùng của topic đó nên được gửi đến subscriber với 1 trường Retain được set trong header. Nếu không có

messsage nào còn, thì không cần gửi gì hết.

Trường này sẽ có ích khi publisher gửi message để báo "report bằng ngoại lệ", thỉnh thoảng là nơi giữa các message. Điều này cho phép những subcribers mới nhanh chóng nhận dữ cần thiết. Trường hợp mà server chuyển tiếp nội dung vừa nhận được từ một Publisher thì trường Retain sẽ không được set. Điều này sẽ giúp phân biệt được message có từ trước với message mới được publish lên. Message Retained sẽ được giữ thậm chí sau khi restart lại server. Server sẽ xóa message được retained nếu nó nhận được một message với payload bằng zero.

* Độ dài còn lại

Vị trí: byte 2.

Miêu tả độ dài bao gồm cả phần header và payload có trong message. Việc encoding với độ dài thay đổi sử dụng 1 byte để miêu tả độ dài, vì thế độ dài tối đa sẽ là 127. Những message dài hơn sẽ được miêu tả theo cách sau. 7 bít được dùng để miêu tả giá trị, bít còn lại dùng để miêu tả phía sau còn byte nào miêu tả trường này hay không. Mỗi byte tiếp sau đó cũng như vậy 7 bit để lưu giá trị, 1 bít gọi là bit tiếp tục. Giá trị được tính bằng cách nhân giá trị được diên tả bởi 7 bit và lũy thừa tăng dần của 128. Ví dụ miêu tả độ Remain Length = 64, ta chỉ cần 1 byte, trong đó 7 bytes để miêu tả giá trị 64, 1 bit còn lại bằng 0. Một ví dụ nữa, giá trị là 321 chẳng hạn 321 = 65*128^0 + 2* 128^1, ta cần 2 byte để biểu diễn. Byte đầu chứa giá trị 65 trong 7 bit và bit còn lại là 1. Byte thứ 2 chứa giá trị 2 ở 7 bit và 1 bit chứa giá trị bằng 0.

Trường này được biểu diễn tối đa trong 4 byte. Tức là cho độ dài cho phép sẽ là đến 268 435 455 (256 MB).

Bảng 2.5. Bảng miêu tả độ dàu ứng với số byte

Số byte	Độ dài min	Độ dài max
1	0 (0x00)	127 (0x7F)
2	128 (0x80, 0x01)	16 383 (0xFF, 0x7F)

3	16 384 (0x80, 0x80, 0x01)	2 097 151 (0xFF, 0xFF, 0x7F)
4	2 097 152 (0x80, 0x80, 0x80, 0x01)	268 435 455 (0xFF, 0xFF, 0xFF, 0x7F)

b. CONNECT - Client yêu cầu connect đến server

Khi một một kết nối TCP/IP được thiết lập từ client đến server, thì một session ở mức protocol cũng được tạo sử dụng luồng CONNECT. Server sẽ gửi message CONNACK để trả lời message CONNECT từ client. Nếu server không nhận được mesage CONNET từ client trong một khoang thời gian nào đó sau khi thiết lập kết nối TCP/IP, thì server nên đóng kết nối đó lại. Nếu client không nhận được một message CONNACK từ server trong một khoảng thời gian nhất định, thì client cũng nên đóng kết nối đó lại, và restart session bằng một socket mới đến server rồi tiếp tục gửi yêu cầu kết nối bằng gói CONNECT. Trong cả 2 trường hợp trên, thời gian chờ để nhận được message CONNECT hoặc CONNACK phụ thuộc vào ứng dụng và điều kiện kết nối. Nếu một client kết nối bằng một Client ID đang được kết nối với Server rồi, thì client trước đó phải được disconnect bắt buộc bởi server trước khi thực hiện luồng CONNECT với client mới. Nếu client gửi một một message CONNECT không hợp lệ, server nên đóng kết nối luôn. Không hợp ở đây bao gồm việc khác nhau về Protocol Name hoặc Protocol Version Numbers. Nếu server đã parse message CONNECT rồi mới phát hiện ra có một trường nào đó không hợp lệ, nó nên gửi lại message CONNACK chứa nội dung mã có nội dung "Kết nối bị từ chối: phiên bản protocol không được chấp nhận" trước khi hủy kết nối này.

c. CONNACK - Acknowledge connection request

Message CONNACK được gửi bởi server như để trả lời một yêu cầu a CONNECT từ client. Mỗi giá trị trả về của Connack được chỉ ra dưới bảng 2.9.

Bảng 2.6. Ý nghĩa gói CONNACK

Enumeration	HEX	Meaning

0	0x00	Kết nối được chấp nhận
1	0x01	Kết nối bị từ chối: Phiên bản protocol không được chấp nhận.
2	0x02	Kết nối vị từ chối: Định danh không hợp lệ
3	0x03	Kết nối vị từ chối: server không phục vụ được
4	0x04	Kết nối bị từ chối: Sai user name hoặc password
5	0x05	Kết nối bị từ chối: Chưa được xác thực
6-255		Dành cho tương lai

Mã trả về là 2 sẽ (định danh bị từ chối) sẽ được gửi nếu nếu định danh duy nhất cho 1 client có độ dài không nằm trong khoảng từ 1 đến 23.

d. PUBLISH - Message Publish

Một message PUBLISH được gửi từ một client đến server để phân tán chúng đến các subscriber cần chúng. Mỗi message luôn gắn liền với một topic name (cũng được hiểu là Subject hoặc Channel). Topic là có dạng không gian phân cấp, nó định nghĩa ra một cách phân loại nguồn thông tin để cho các subscribers có thể subscribe nếu muốn.Một message được published đến một topic nào đó sẽ được phân phát đến các subscriber quan tâm đến topic đó .

Nếu 1 client subscribe một hoặc nhiều topic, thì mọi message được published đến những topic đó được gửi bởi server đến client như là một message PUBLISH. Response cho PUBLISH message phụ thuộc vào level của QoS.

Bảng 2.7. Giá trị mức QoS

QoS Level	Giá trị trả về mong muốn
QoS 0	None
QoS 1	PUBACK
QoS 2	PUBREC

Các message PUBLISH được gửi từ Publisher đến server, hoặc là từ Server đến Subsciber. Việc tiếp thu khi nó nhận được messsage phụ thuộc vào QoS level

của message:

QoS 0 : Phải đảm bảo chắc chắn rằng các message đến các bên quan tâm.

QoS 1: Giữ một bản của messsage trên bộ nhớ ngoài, tất nhiên cũng phải đảm báo nó đến các bên quan tâm, và trả lại một message PUBACK đến bên gửi.

QoS 2 : Giữ lại một bản trên bộ nhớ ngoài, đừng gửi nó đến các bên quan tâm , trả về message PUBREC đến bên gửi.

Ở đây, nếu server nhận được message, các bên quan tâm nghĩa là các subscriber đến topic của message PUBLISH đó. Nếu một subscriber nhận một message, thì các bên quan tâm ám chỉ đến các ứng dụng bên phía client mà đã subcribe mọt hoặc nhiều topics và đang đợi một message đến từ server.

Nếu một server mà không xác nhận được một PUBLISH từ một client, thì nó không có cách nào để thông bao đến client đó. Vì thế nó phải đảm bảo một mức hiểu biết nhất định, tùy thuộc vào các rule QoS, client sẽ *không* được báo rằng việc xác thực message PUBLISH nó gửi đi.

e. SUBSCRIBE - Subscribe to named topics

Message SUBSCRIBE cho phép client subscribe một hoặc nhiều topics với server. Message được published lên server sẽ được chuyển đến client bằng message PUBLISH . Message SUBSCRIBE cũng chỉ ra QoS level mà subscriber muốn nhận message. Khi nhận được một message SUBSCRIBE message, server trả lời bằng message SUBACK.

Một server bắt đầu gửi một message PUBLISH cho yêu cầu của về subscription của client thậm chí trước khi client nhận được message SUBACK. Nếu một server không xác nhận một yêu cầu SUBSCRIBE từ client, thì nó không có cách nào thông tin cho client. Vì thế nó tạo ra message SUBACK như là để xác nhận, và client sẽ *không* được thông báo rằng nó không được xác thực.

Một server có quyền chỉ định một level of QoS thấp hơn client yêu cầu. Điều này có thể xảy ra nếu server không thể cung cấp các levels of QoS cao hơn. Ví dụ,

nếu server không cung cấp một cơ chế lưu trữ tin cậy nào đó nó có thể chỉ cấp cho các subscriptions với QoS là 0.

f. PINGREQ - PING request

Message PINGREQ có nghĩa là message "Kết nối vẫn tốt đúng không?" được gửi từ một client đã kết nối đến server.

g. PINGRESP - PING response

Một message PINGRESP được gửi từ server cho một message PINGREQ và nó có nghĩa là "OK".

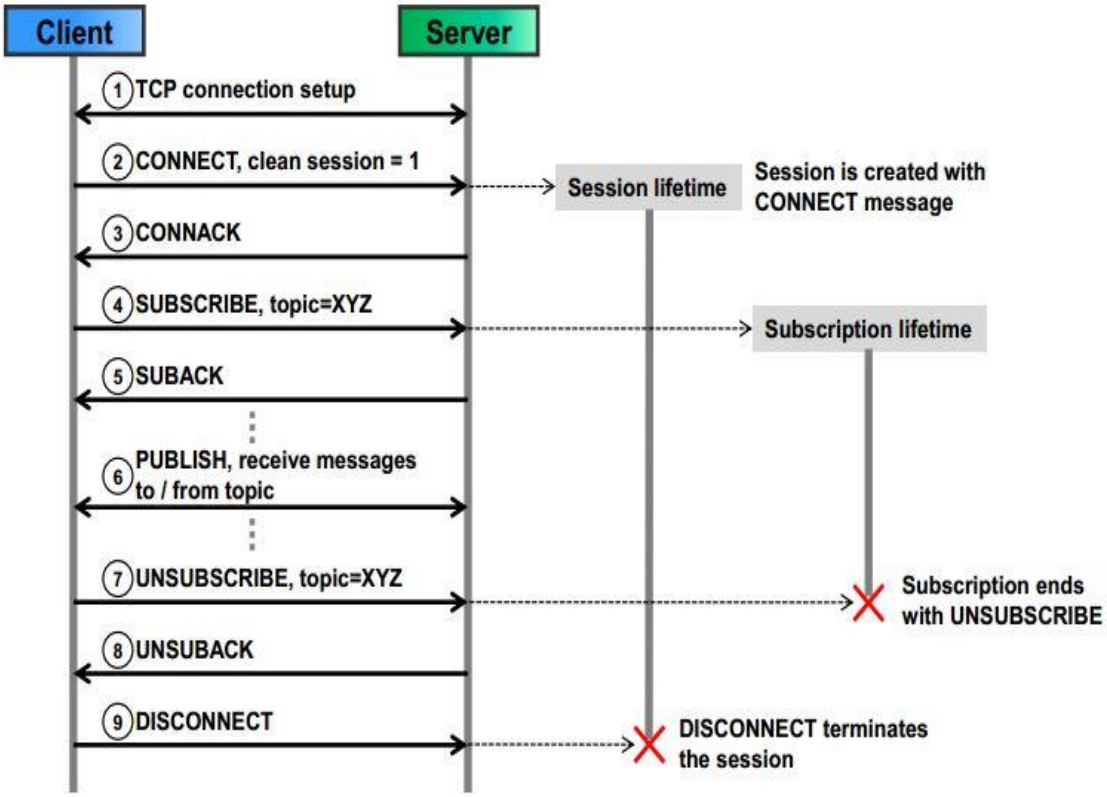
h. DISCONNECT - Disconnect notification

Message DISCONNECT được gửi từ client đến server để báo rằng nó sẽ đóng kết nối TCP/IP đang kết nối. Cái này cho phép một clean disconnection, chứ không chỉ là hủy kết nối. Một server không nên để việc đóng kết nối này cho phía client sau khi nhận message DISCONNECT.

2.4.4. Quy trình truyền nhận dữ liệu chính trong MQTT

a. CONNECT and SUBSCRIBE message sequence

Trường hợp 1: Session và subscription được thiết lập với clean session flag = 1 (transient subscription)



Hình 2.3. Session và subscription được thiết lập với clean session flag = 1

Quy trình truyền nhận dữ liệu trong MQTT khi session flag = 1:

- Client và Server được kết nối qua giao thức TCP.

- Client gửi gói tin CONNECT yêu cầu kết nối đến Server, clean session = 1. Đây là thời điểm đánh dấu session được thiết lập.

- Server gởi gói CONNACK xác nhận thiết lập kết nối thành công.

- Client thực hiện SUBSCRIBE đến topic XYZ. Đây là thời điểm bắt đầu timeout của một subscription.

- Server gởi gói SUBACK xác nhận quá trình subscription.

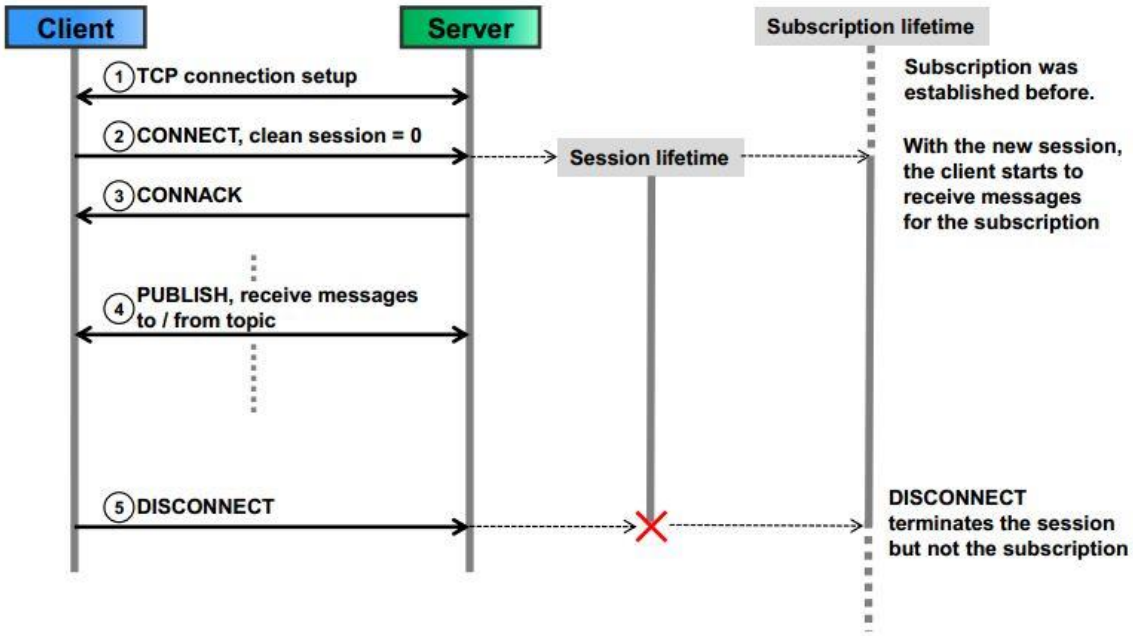
- Client PUBLISH để gởi topic - message đến server.

- Sau khi nhận đủ thông tin, client gửi gói UNSUBSCRIBE topic XYZ để kết thúc quá trình Subscribe.

- Server trả về gói UNSUBACK.

- Client gửi gói DISCONNECT để kết thúc session truyền thông.

Trường hợp 2: Session và subscription được thiết lập với clean session flag = 0 (durable subscription).



Hình 2.4. Session và subscription được thiết lập với clean session flag = 0

Quy trình truyền nhận dữ liệu trong MQTT khi session flag = 0:

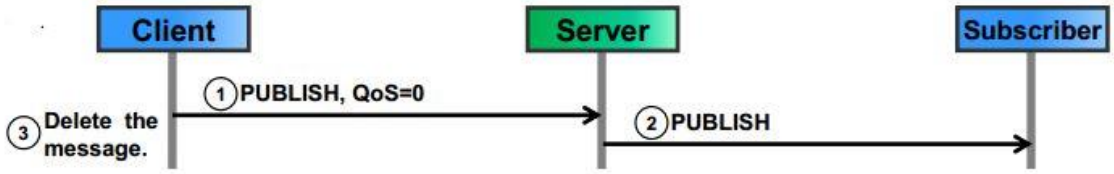
- Subscription lifetime đã được thiết lập trước.
- Client và Server được kết nối qua giao thức TCP.
- Client gửi gói tin CONNECT yêu cầu kết nối đến Server, clean session = 0. Đây là thời điểm đánh dấu session được thiết lập.
- Server gởi gói CONNACK xác nhận thiết lập kết nối thành công.
- Client PUBLISH để gởi topic - message đến server.
- Client gửi gói DISCONNECT để kết thúc session truyền thông.

b. PUBLISH message flows

QoS level 0: At most once delivery.

Message được phân phối dựa trên best efforts của tầng mạng TCP/IP bên dưới. Một response sẽ không được định nghĩa trong giao thức. Các message đến

server hoặc chỉ 1 lần hoặc không bao giờ.

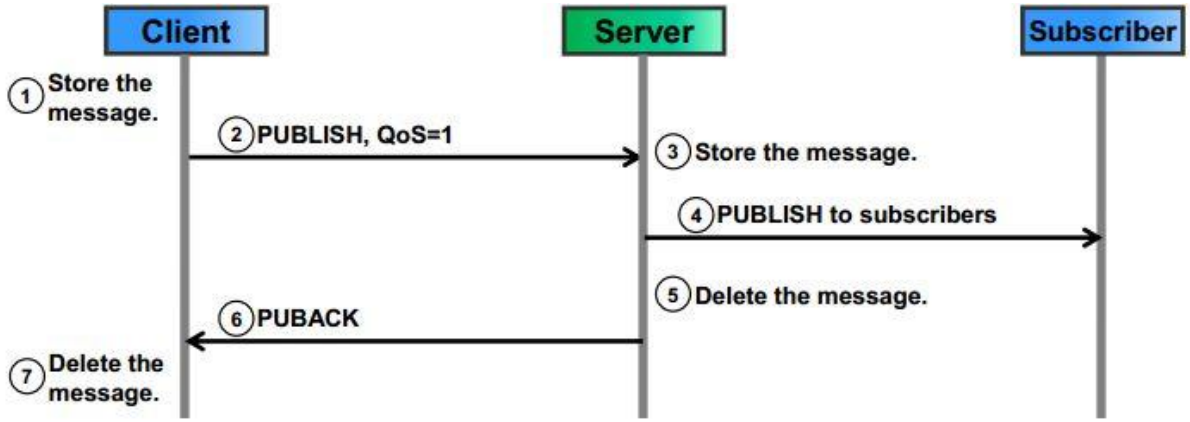


Hình 2.5. QoS mức 0

QoS level 1: At least once delivery

Việc nhận được message bên phía server được xác nhận bởi một message PUBACK. Nếu có lỗi do kết nối hoặc gửi đến device, hoặc message xác nhận không nhận được sau một khoảng thời gian nhất định, sender sẽ gửi lại message và set DUP bit trong phần header của message header. Message đến server ít nhất 1 lần. Cả message SUBSCRIBE và message UNSUBSCRIBE đều sử dụng QoS level là 1.

Khi nhận được một message lặp lại từ phía client, server sẽ publish các message đến các subscribers, và gửi một message PUBACK khác.



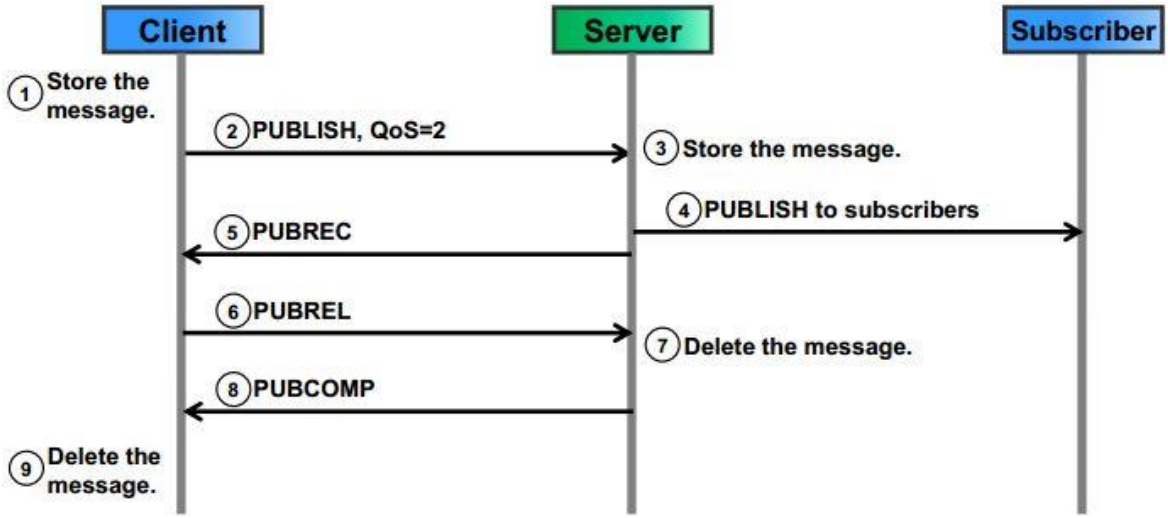
Hình 2.6. QoS mức 1

QoS level 2: Exactly once delivery

Một luồng được thêm vào luồng QoS level bằng 1 ở trên để đảm bảo rằng message bị lặp lại không bị chuyển đến ứng dụng. Đây là mức độ cao nhất khi khi phân phối message, không message lặp nào được chấp nhận. Nhờ đó mà lưu lượng

mạng sẽ tăng lên.

Nếu phát hiện lỗi, hoặc sau một khoảng thời gian nhất định, luồng protocol sẽ được thực hiện lại từ kết quả của message xác nhận cuối cùng; hoặc là PUBLISH , hoặc là PUBREL. Luồng protocol đảm bảo rằng message đến các subscriber chỉ đúng 1 lần.



Hình 2.7. QoS mức 2