Mittuniversitetet
MID SWEDEN UNIVERSITY

MQTT

Experimental Work in Intelligent IoT Networks

Stefan Forsström
Department of Computer and Electrical Engineering (DET)

Mittuniversitetet
MID SWEDEN UNIVERSITY

Overview

- History and Background
- Structure and Features
- MQTT Messages Details
- Tips for the Lab Project

Mittuniversitetet
MID SWEDEN UNIVERSITY

History

- Message Queue Telemetry Transport (MQTT)
  - Was invented in 1999 by
    - Andy Stanford-Clark (IBM)
    - Arlen Nipper (Arcom, now Cirrus Link)
- The use case was to create a protocol for minimal battery loss and minimal bandwidth, to connect oil pipelines over satellite Internet
- They specified the following goals of the protocol:
  - Simple to implement
  - Provide a Quality of Service Data Delivery
  - Lightweight and Bandwidth Efficient
  - Data Agnostic
  - Continuous Session Awareness

Mittuniversitetet
MID SWEDEN UNIVERSITY

History

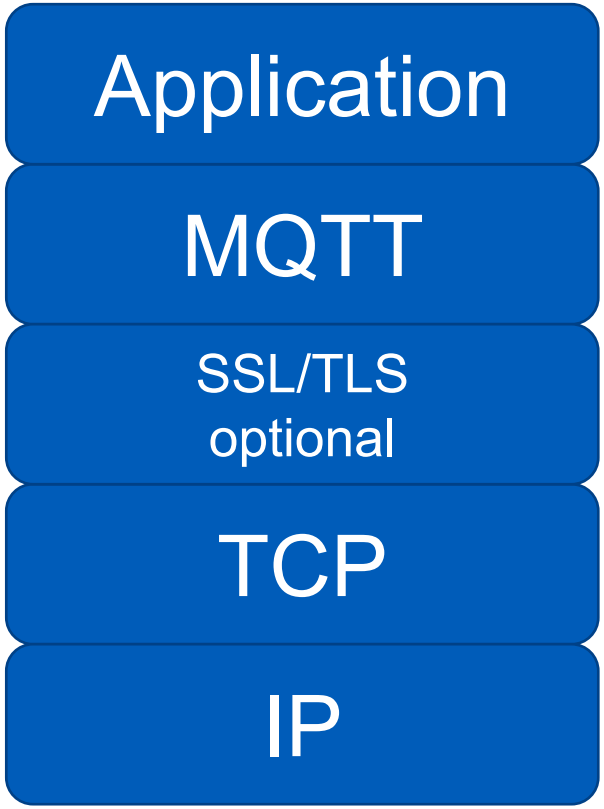
- It is an publish-subscribe "lightweight" messaging protocol
  - But it fits the IoT very well
- Highly centralized with a coordinating broker server
  - Consumers subscribe to topics
  - Which producers can publish to
  - And the broker unicasts the data to the subscribers
- Not really “lightweight”, since it is based on TCP
  - Comparable to REST, about the same
  - But has built in publish subscribe features

Structure

TCP/IP Port: 1883

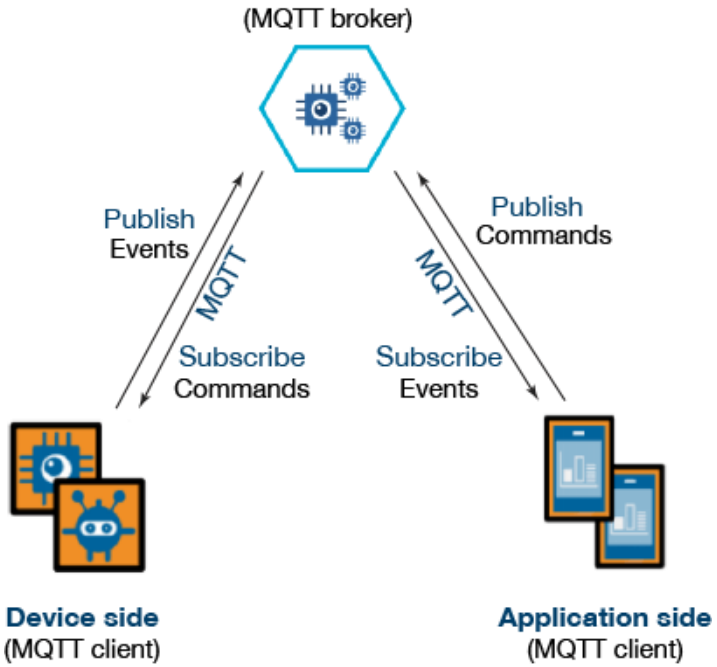
When running over SSL/TLS port: 8883

When running over Websockets port: 8000



Structure

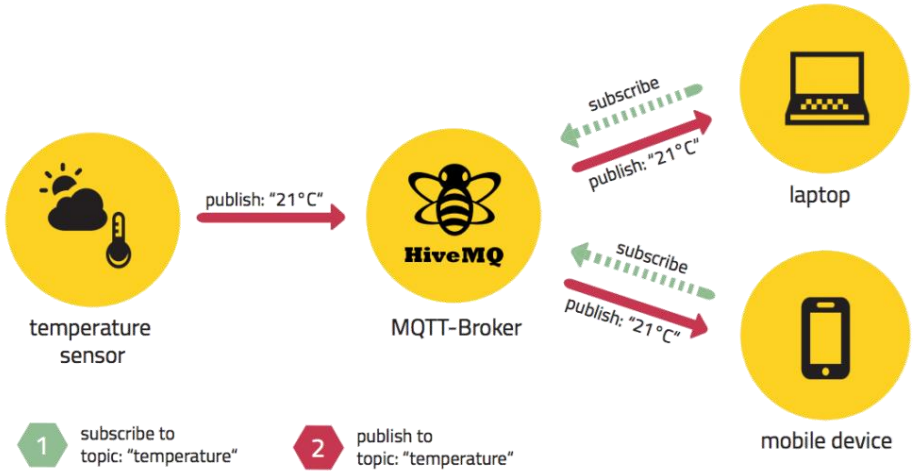
- MQTT consist of three parts:
  - Broker
  - Subscribers
  - Publishers



- Clients connect to a “Broker”
- Clients subscribe to topics e.g.,
- Clients can publish messages to topics:
  - All clients receive all messages published to topics they subscribe to
- Messages can be anything, Text, Images, etc.

Publish/Subscribe Concept

- Decoupled in space and time:
  - The clients does not need to know each others IP address and port and they do not need to be running at the same time

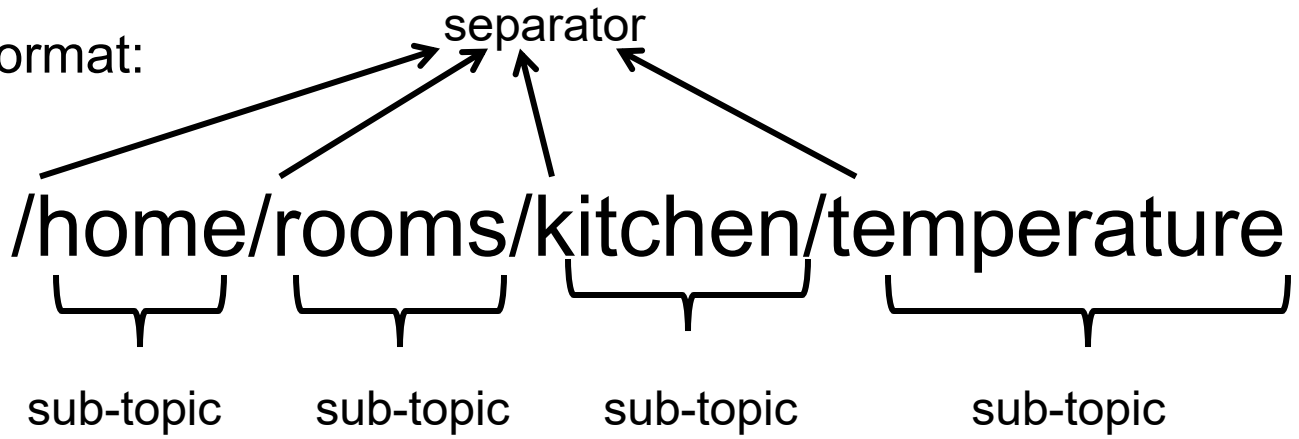


- The broker's IP and port must however be known by all clients

Topics

- Each published data specifies a topic
  - Each subscriber subscribed to that topic will receive it
  - Namespace hierarchy is used for topic filtering

- Topic format:



Mittuniversitetet
MID SWEDEN UNIVERSITY

Subscriptions

- Subscription types
  - Durable
    - If the subscriber disconnect messages are buffered at the broker and delivered upon reconnection
  - Non-durable
    - Connection lifetime is the subscription lifetime
    - When the TCP session breaks down, the subscription is closed

Mittuniversitetet
MID SWEDEN UNIVERSITY

Publishing

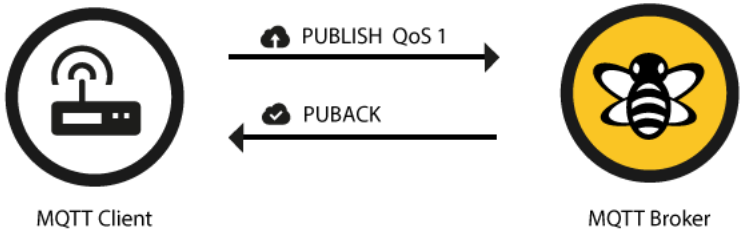
- It might also be the case that a published message never becomes consumed by any subscriber
  - The clients are unaware of the number of subscribers
- Message retention
  - Retained (a type of “persistent” message)
    - The subscriber upon first connection receives the last good publication (i.e., does not have to wait for new publication)
    - Only the most recent persistent message is stored and distributed
- Last Will and Testament (LWT)
  - A message published upon disconnecting a connection
  - Anybody subscribing to the LWT topic will know when a certain device (that registered a LWT) disconnected

Publishing “QoS” (Reliability)

- 0 – unreliable (aka “at most once”)
  - OK for continuous streams, least overhead (1 message)
  - “Fire and forget”
  - TCP will still provide reliability

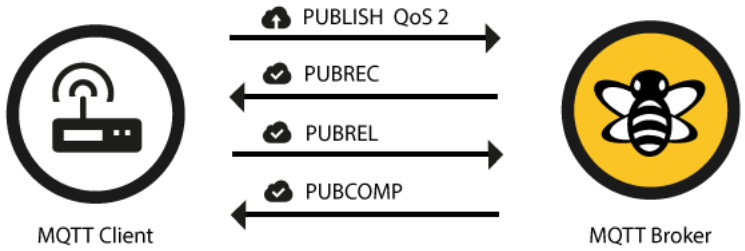
Publishing “QoS” (Reliability)

- 1 – delivery “at least once” (duplicates possible)
  - Used for alarms – more overhead (2 messages)
  - Contains message ID (to match with ACKed message)



Publishing “QoS” (Reliability)

- 2 – delivery “exactly once”
  - Utmost reliability is important – most overhead (4 messages) and slowest



Mittuniversitetet
MID SWEDEN UNIVERSITY

Security?

- All communication is done in clear text
  - Unless SSL/TLS is used
- There is a simple Client ID method used for recognizing users
- But there are also functions for username/password authentication
  - For privately accessing the broker etc.
- You can also control which clients are able to subscribe and publish to different topics
  - Using either the ClientID or username/password

Mittuniversitetet
MID SWEDEN UNIVERSITY

V3.1.1 and V.5 and MQTT-SN

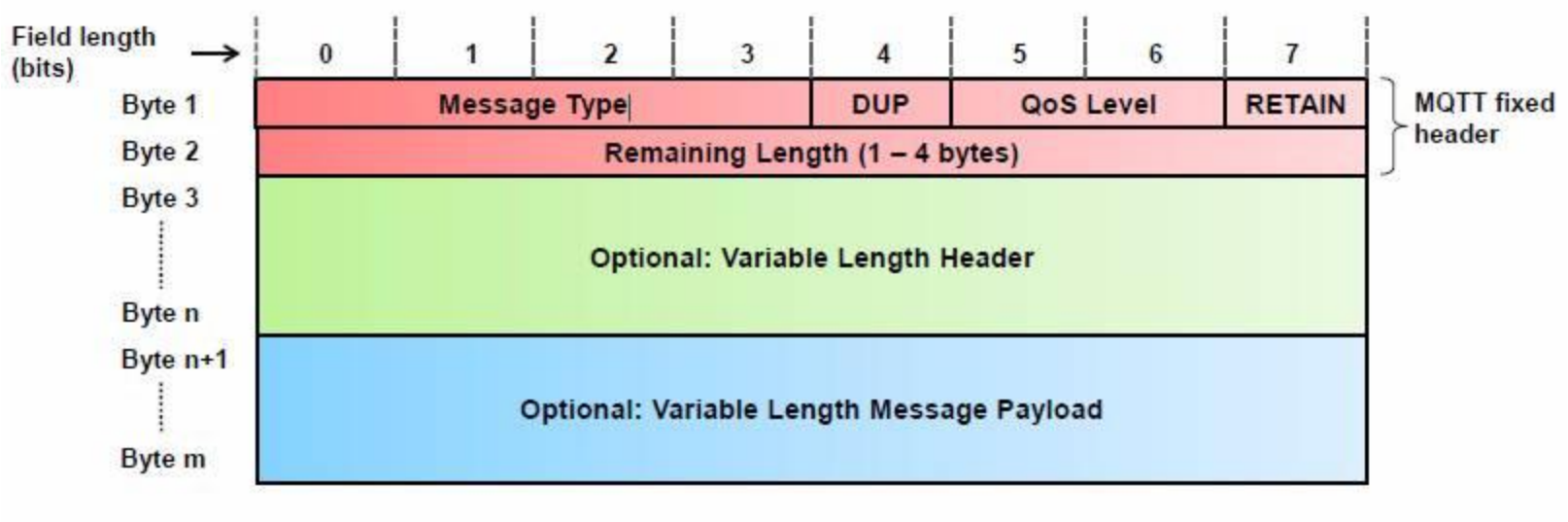
- MQTT version 3.1.1 was the standard for a long time
  - But has now been updated to v.5 (published 2019)
- In general it functions the same, but some changes to the protocol
  - Clean session/start
  - Client Restrictions/Limitations
  - Server Restrictions/Limitations
  - Will Delay Intervals
  - Server Redirect
  - Payload Format Indicator
  - Topic aliases
  - User Properties
  - Request Response
  - Non local publishing
  - Retained Message Control
  - Subscription Identifier
  - Shared Subscriptions
  - Reason Codes on All ACK Messages
  - Server Disconnect
- MQTT for Sensor Networks (MQTT-SN)
  - A more limited version of MQTT over UDP (not updated since 2013)

Mittuniversitetet
MID SWEDEN UNIVERSITY

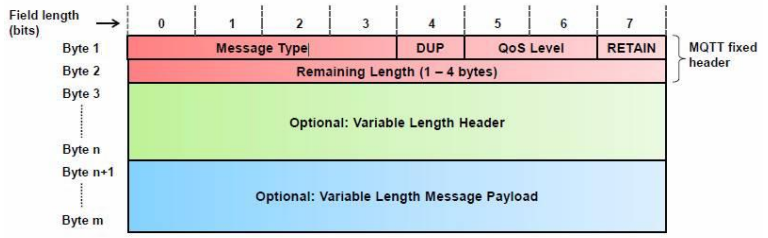
MQTT Messages v5

Mittuniversitetet
MID SWEDEN UNIVERSITY

MQTT Message Format

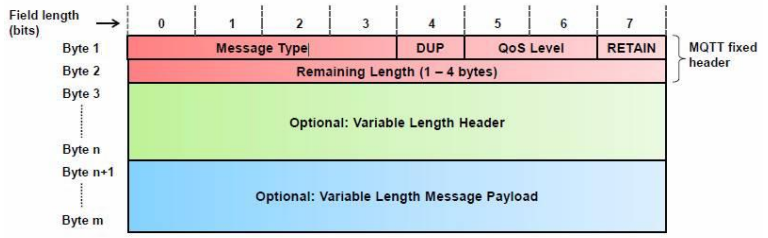


- Shortest message is two bytes (red fields)



Message Types

Message fixed header field	Description / Values	
Message Type	0: Reserved	8: SUBSCRIBE
	1: CONNECT	9: SUBACK
	2: CONNACK	10: UNSUBSCRIBE
	3: PUBLISH	11: UNSUBACK
	4: PUBACK	12: PINGREQ
	5: PUBREC	13: PINGRESP
	6: PUBREL	14: DISCONNECT
	7: PUBCOMP	15: AUTH
DUP	Duplicate message flag. Indicates to the receiver that this message may have already been received. 1: Client or server (broker) re-delivers a PUBLISH, PUBREL, SUBSCRIBE or UNSUBSCRIBE message (duplicate message).	
QoS Level	Indicates the level of delivery assurance of a PUBLISH message. 0: At-most-once delivery, no guarantees, «Fire and Forget». 1: At-least-once delivery, acknowledged delivery. 2: Exactly-once delivery. Further details see MQTT QoS.	
RETAIN	1: Instructs the server to retain the last received PUBLISH message and deliver it as a first message to new subscriptions. Further details see RETAIN (keep last message).	
Remaining Length	Indicates the number of remaining bytes in the message, i.e. the length of the (optional) variable length header and (optional) payload. Further details see Remaining length (RL).	



Connect Message

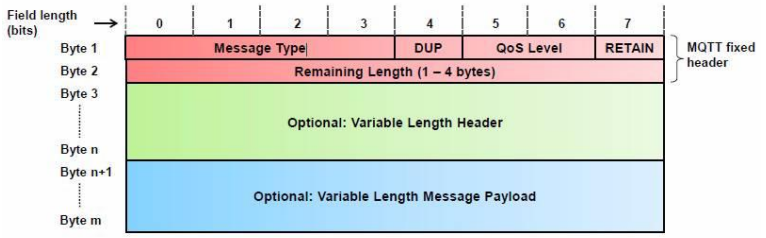
- Message nr 1
- Protocol name
- Flags
- Keep alive
- Payload
  - can include client ID length and name

Bit	7	6	5	4	3	2	1	0
Byte 1	MQTT Control Packet type (1)				Reserved			
	0	0	0	1	0	0	0	0
Byte 2...	Remaining Length							

	Description	7	6	5	4	3	2	1	0
Protocol Name									
byte 1	Length MSB (0)	0	0	0	0	0	0	0	0
byte 2	Length LSB (4)	0	0	0	0	0	1	0	0
byte 3	'M'	0	1	0	0	1	1	0	1
byte 4	'Q'	0	1	0	1	0	0	0	1
byte 5	'T'	0	1	0	1	0	1	0	0
byte 6	'T'	0	1	0	1	0	1	0	0
byte 7	Version(5)	0	0	0	0	0	1	0	1

Bit	7	6	5	4	3	2	1	0
	User Name Flag	Password Flag	Will Retain	Will QoS		Will Flag	Clean Session	Reserved
Byte 8	X	X	X	X	X	X	X	0

Bit	7	6	5	4	3	2	1	0
byte 9	Keep Alive MSB							
byte 10	Keep Alive LSB							



Connect Ack Message

- Message nr 2
- SP: Session Present Bit
- Return code
- Properties

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet Type (2)				Reserved			
	0	0	1	0	0	0	0	0
byte 2	Remaining Length (2)							
	0	0	0	0	0	0	1	0

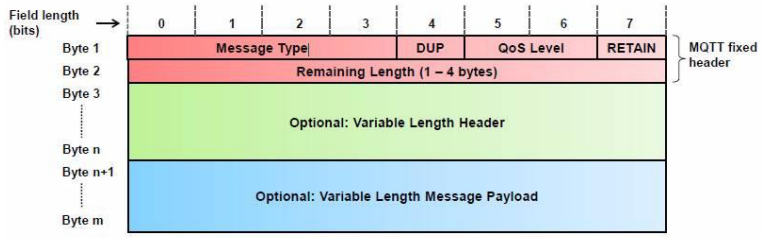
	Description	7	6	5	4	3	2	1	0
Connect Acknowledge Flags		Reserved							SP[1]
byte 1		0	0	0	0	0	0	0	X
Connect Return code									
byte 2		X	X	X	X	X	X	X	X
byte		X	X	X	X	X	X	X	X

Value	Return Code Response	Description
0	0x00 Connection Accepted	Connection accepted

128	0x80	Unspecified error	The Server does not wish to reveal the reason for the failure, or none of the other Reason Codes apply.
129	0x81	Malformed Packet	Data within the CONNECT packet could not be correctly parsed.
130	0x82	Protocol Error	Data in the CONNECT packet does not conform to this specification.
131	0x83	Implementation specific error	The CONNECT is valid but is not accepted by this Server.
132	0x84	Unsupported Protocol Version	The Server does not support the version of the MQTT protocol requested by the Client.
133	0x85	Client Identifier not valid	The Client Identifier is a valid string but is not allowed by the Server.
134	0x86	Bad User Name or Password	The Server does not accept the User Name or Password specified by the Client

135	0x87	Not authorized	The Client is not authorized to connect.
136	0x88	Server unavailable	The MQTT Server is not available.
137	0x89	Server busy	The Server is busy. Try again later.
138	0x8A	Banned	This Client has been banned by administrative action. Contact the server administrator.
140	0x8C	Bad authentication method	The authentication method is not supported or does not match the authentication method currently in use.
144	0x90	Topic Name invalid	The Will Topic Name is not malformed, but is not accepted by this Server.
149	0x95	Packet too large	The CONNECT packet exceeded the maximum permissible size.
151	0x97	Quota exceeded	An implementation or administrative imposed limit has been exceeded.

151	0x97	Quota exceeded	An implementation or administrative imposed limit has been exceeded.
153	0x99	Payload format invalid	The Will Payload does not match the specified Payload Format Indicator.
154	0x9A	Retain not supported	The Server does not support retained messages, and Will Retain was set to 1.
155	0x9B	QoS not supported	The Server does not support the QoS set in Will QoS.
156	0x9C	Use another server	The Client should temporarily use another server.
157	0x9D	Server moved	The Client should permanently use another server.
159	0x9F	Connection rate exceeded	The connection rate limit has been exceeded.



Publish Message

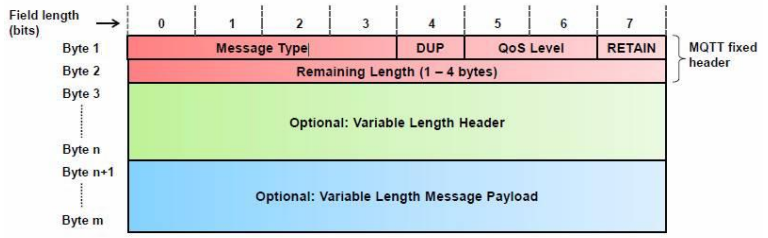
- Message nr 3

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type (3)				DUP flag	QoS level		RETAIN
	0	0	1	1	X	X	X	X
byte 2	Remaining Length							

- Topic Length
- The topic characters
- Packet ID

	Description	7	6	5	4	3	2	1	0
Topic Name									
byte 1	Length MSB (0)	0	0	0	0	0	0	0	0
byte 2	Length LSB (3)	0	0	0	0	0	0	1	1
byte 3	'a' (0x61)	0	1	1	0	0	0	0	1
byte 4	'/' (0x2F)	0	0	1	0	1	1	1	1
byte 5	'b' (0x62)	0	1	1	0	0	0	1	0
Packet Identifier									
byte 6	Packet Identifier MSB (0)	0	0	0	0	0	0	0	0
byte 7	Packet Identifier LSB (10)	0	0	0	0	1	0	1	0

byte 1 byte 2 bytes 3..N	Payload

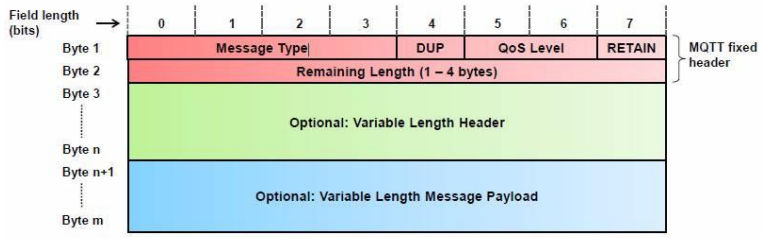


Publish Message

- The QoS level specifies the answer

QoS Level	Expected Response
QoS 0	None
QoS 1	PUBACK Packet
QoS 2	PUBREC Packet

- I leave these for self study:
  - PUBACK – Publish acknowledgemen (QoS 1)
  - PUBREC – Publish received (QoS 2 publish received, part 1)
  - PUBREL – Publish release (QoS 2 publish received, part 2)
  - PUBCOMP – Publish complete (QoS 2 publish received, part 3)
  - AUTH – Extended authentication exchange



Subscribe Message

- Message nr 8

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type (8)				Reserved			
	1	0	0	0	0	0	1	0
byte 2	Remaining Length							

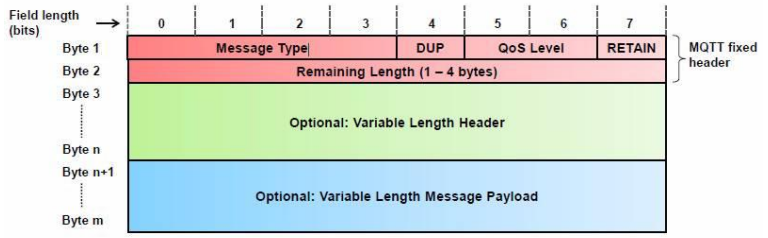
- Package ID
- Properties

	Description	7	6	5	4	3	2	1	0
Packet Identifier									
byte 1	Packet Identifier MSB (0)	0	0	0	0	0	0	0	0
byte 2	Packet Identifier LSB (10)	0	0	0	0	1	0	1	0
byte 3	Property Length (0)	0	0	0	0	0	0	0	0

- Payload
  - With options

Description	7	6	5	4	3	2	1	0
Topic Filter								
byte 1	Length MSB							
byte 2	Length LSB							
bytes 3..N	Topic Filter							
Subscription Options								
	Reserved		Retain Handling		RAP	NL	QoS	
byte N+1	0	0	X	X	X	X	X	X

- Many subs in one message



Subscribe Ack Message

- Message nr 9
- Packet ID to ACK
- Return code

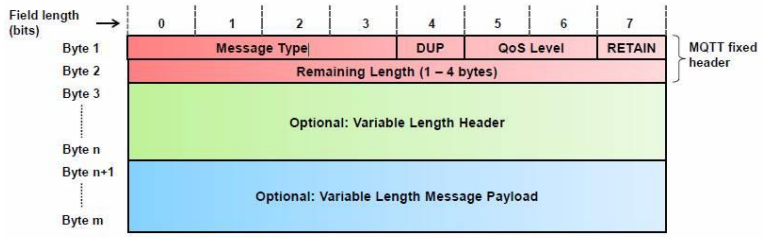
Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type (9)				Reserved			
	1	0	0	1	0	0	0	0
byte 2	Remaining Length							

Bit	7	6	5	4	3	2	1	0
byte 1	Packet Identifier MSB							
byte 2	Packet Identifier LSB							

Bit	7	6	5	4	3	2	1	0
	Return Code							
byte 1	X	0	0	0	0	0	X	X

Value	Hex	Reason Code name	Description
0	0x00	Granted QoS 0	The subscription is accepted and the maximum QoS sent will be QoS 0. This might be a lower QoS than was requested.
1	0x01	Granted QoS 1	The subscription is accepted and the maximum QoS sent will be QoS 1. This might be a lower QoS than was requested.
2	0x02	Granted QoS 2	The subscription is accepted and any received QoS will be sent to this subscription.

128	0x80	Unspecified error	The subscription is not accepted and the Server either does not wish to reveal the reason or none of the other Reason Codes apply.
131	0x83	Implementation specific error	The SUBSCRIBE is valid but the Server does not accept it.
135	0x87	Not authorized	The Client is not authorized to make this subscription.
143	0x8F	Topic Filter invalid	The Topic Filter is correctly formed but is not allowed for this Client.
145	0x91	Packet Identifier in use	The specified Packet Identifier is already in use.
151	0x97	Quota exceeded	An implementation or administrative imposed limit has been exceeded.
158	0x9E	Shared Subscriptions not supported	The Server does not support Shared Subscriptions for this Client.
161	0xA1	Subscription Identifiers not supported	The Server does not support Subscription Identifiers; the subscription is not accepted.
162	0xA2	Wildcard Subscriptions not supported	The Server does not support Wildcard Subscriptions; the subscription is not accepted.



Unsubscribe Message

- Message nr 10

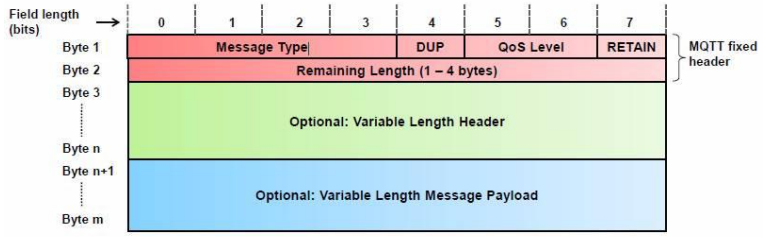
Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type (10)				Reserved			
	1	0	1	0	0	0	1	0
byte 2	Remaining Length							

- Packet ID

Bit	7	6	5	4	3	2	1	0
byte 1	Packet Identifier MSB							
byte 2	Packet Identifier LSB							

- Payload
  - Can have many unsub in one message

	Description	7	6	5	4	3	2	1	0
Topic Filter									
byte 1	Length MSB (0)	0	0	0	0	0	0	0	0
byte 2	Length LSB (3)	0	0	0	0	0	0	1	1
byte 3	'a' (0x61)	0	1	1	0	0	0	0	1
byte 4	'/' (0x2F)	0	0	1	0	1	1	1	1
byte 5	'b' (0x62)	0	1	1	0	0	0	1	0



Unsubscribe Ack Message

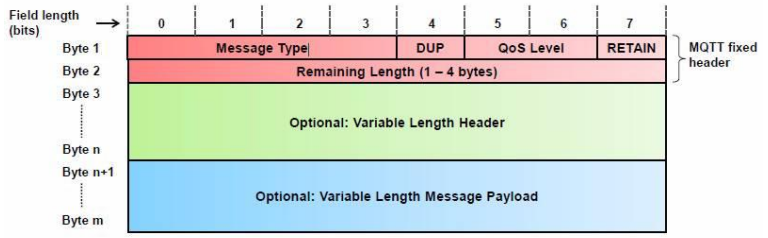
- Message nr 11
- Packet ID to ACK
- Reason Code

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type (11)				Reserved			
	1	0	1	1	0	0	0	0
byte 2	Remaining Length							

Bit	7	6	5	4	3	2	1	0
byte 1	Packet Identifier MSB							
byte 2	Packet Identifier LSB							

Bit	7	6	5	4	3	2	1	0
	Return Code							
byte 1	X	0	0	0	0	0	X	X

Value	Hex	Reason Code name	Description
0	0x00	Success	The subscription is deleted.
17	0x11	No subscription existed	No matching Topic Filter is being used by the Client.
128	0x80	Unspecified error	The unsubscribe could not be completed and the Server either does not wish to reveal the reason or none of the other Reason Codes apply.
131	0x83	Implementation specific error	The UNSUBSCRIBE is valid but the Server does not accept it.
135	0x87	Not authorized	The Client is not authorized to unsubscribe.
143	0x8F	Topic Filter invalid	The Topic Filter is correctly formed but is not allowed for this Client.
145	0x91	Packet Identifier in use	The specified Packet Identifier is already in use.



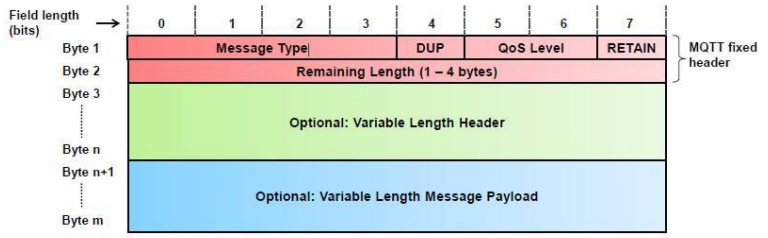
PingReq/PingResp Message

- Ping Request

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type (12)				Reserved			
	1	1	0	0	0	0	0	0
byte 2	Remaining Length (0)							
	0	0	0	0	0	0	0	0

- Ping Response

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type (13)				Reserved			
	1	1	0	1	0	0	0	0
byte 2	Remaining Length (0)							
	0	0	0	0	0	0	0	0



Disconnect Message

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type (14)				Reserved			
	1	1	1	0	0	0	0	0
byte 2	Remaining Length							

- Disconnect

	Description	7	6	5	4	3	2	1	0
Disconnect Reason Code									
byte 1		0	0	0	0	0	0	0	0

- And many more disconnect codes…

Value	Hex	Reason Code name	Sent by	Description
0	0x00	Normal disconnection	Client or Server	Close the connection normally. Do not send the Will Message.
4	0x04	Disconnect with Will Message	Client	The Client wishes to disconnect but requires that the Server also publishes its Will Message.
128	0x80	Unspecified error	Client or Server	The Connection is closed but the sender either does not wish to reveal the reason, or none of the other Reason Codes apply.
129	0x81	Malformed Packet	Client or Server	The received packet does not conform to this specification.
130	0x82	Protocol Error	Client or Server	An unexpected or out of order packet was received.
131	0x83	Implementation specific error	Client or Server	The packet received is valid but cannot be processed by this implementation.
135	0x87	Not authorized	Server	The request is not authorized.
137	0x89	Server busy	Server	The Server is busy and cannot continue processing requests from this Client.
139	0x8B	Server shutting down	Server	The Server is shutting down.

Mittuniversitetet
MID SWEDEN UNIVERSITY

Tips for the Lab Project

Mittuniversitetet
MID SWEDEN UNIVERSITY

Tips for the Lab Project

- Start by using a normal MQTT Client, connect to an MQTT broker
  - For example: broker.mqttdashboard.com
  - Observe the messages in Wireshark (tcp.port == 1883)
- Then create your own TCP socket listening on port 1883
  - And connect your MQTT client to it instead
  - Observe, listening, and answer to incoming packets
- Since you are the broker, most answers will be short ACKs
  - The Connection ACK is only 5 bytes, for example:
    - Byte 1: (0010 0000) Packet Type 2, not DUP, QoS 0, no RETAIN
    - Byte 2: (0000 0011) 3 remaining bytes
    - Byte 3: (0000 0000) No session present
    - Byte 4: (0000 0000) return code 0, success
    - Byte 5: (0000 0000) no properties

Mittuniversitetet
MID SWEDEN UNIVERSITY

Tips for the Lab Project

- Start by answering the connect packages
  - And then make your program answer the ping packets
  - Otherwise, all your clients will time out after a while
- If you receive a subscribe, save that socket and topic to a list/map
  - When you receive a publish, send it to all sockets on the topic list
  - Unsubscribe removes socket from the topic list
- Wireshark is your friend
  - To see how other MQTT client's messages looks like is very good for you to learn, debug and compare to yourself to them
- There is no delimiter of payload or between messages
  - Make sure you have the right remainder packet length
  - It is the only way for the system to distinguish between two messages

Mittuniversitetet
MID SWEDEN UNIVERSITY

Tips for the Lab Project

- The MQTT 5.0 Oasis Standard
  - https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html

- A good online reference:
  - http://www.steves-internet-guide.com/mqtt-basics-course/

Mittuniversitetet
MID SWEDEN UNIVERSITY

Contact Information

STEFAN FORSSTRÖM
Assoc. Prof. in Computer Engineering

MID SWEDEN UNIVERSITY
Department of Computer and Electrical Engineering (DET)
Campus Sundsvall, Room L426
Email: stefan.forsstrom@miun.se