MQTT

- The protocol usually runs over TCP/IP.
- It was designed as an extremely lightweight publish/subscribe messaging transport.
- It is designed with a minimal protocol overhead.

★ Terminologies

Network Connection

- A construct provided by the underlying transport protocol that is being used by MQTT.
- It connects a peer to peer.
- It provides the means to send an ordered, lossless, stream of bytes in both direction.

Application Messages

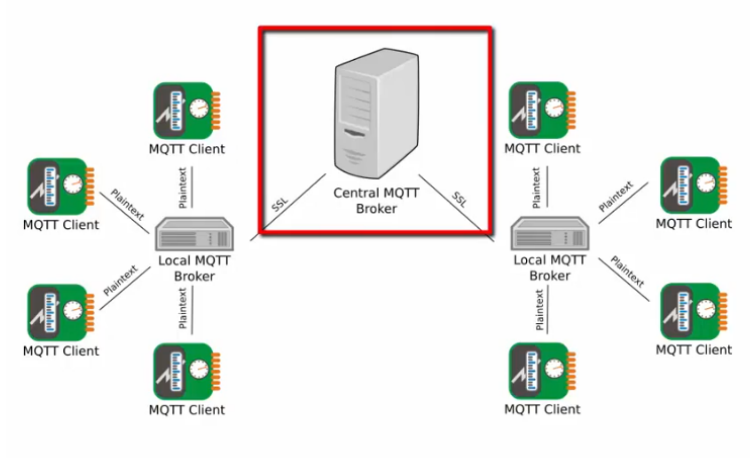
- The data carried by the MQTT protocol across the network for the application.
- When application message are transported by MQTT they have an associated:
  - Quality of Service
  - Topic Name

Client

- A client is a program that uses MQTT.
- A client always establishes the network connection to the server.
- It can:
  - Publish application messages that other client might be interested in.
  - Subscribe to application messages that it is interested in.
  - Disconnect from the server.

Server / Broker

- It is a program that acts as an intermediary between clients which publish application message and which have made subscriptions.



→ A Server:

- → Accepts Network Connections from client.
- → Accepth application messages published by clients.
- → Processes subscription and unsubscription request from client.
- → Forwards application message which matches client subscription.

Session

- → A session is a stateful interaction between a Client and a Server.
- → A session is typically stateful, meaning that at least one of the communicating parties needs to hold current state information and save information about the session history in order to be able to communicate, as opposed to stateless communication, where the communication consists of independent requests with responses.

Subscription

- → A subscription comprises a topic filter and a maximum quallity of service.
- → A subscription is associated with a single session.
- → A session can contain more than one subscription.
- → Each subscription within a session has different topic filter.

Topic name

- → The label attached to an application message which is matched against the subscriptions known to the server.
- → The server sends a copy of the application message to each client that has a matching subscription.

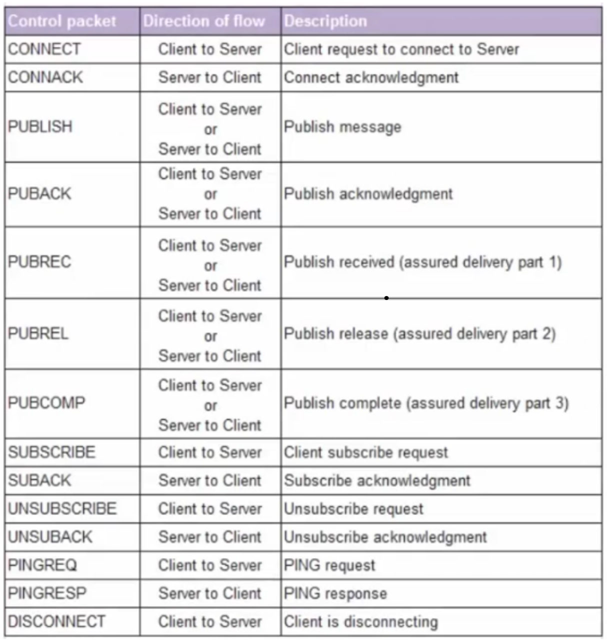
Topic filter

An expression contained in a subscription, to indicate an interest in one or more topics.

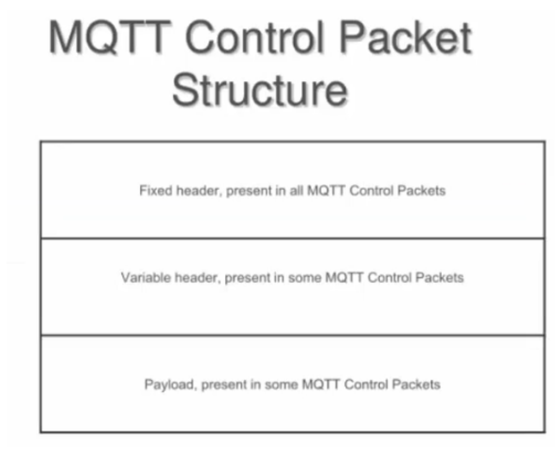
MQTT Control packet

Just a packet of information that is sent across the network connection.

MQTT specification define 14 different type of Control Packet.

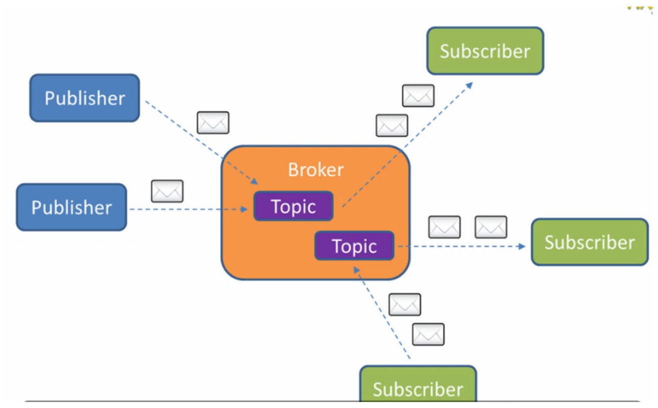


Control packet	Direction of flow	Description
CONNECT	Client to Server	Client request to connect to Server
CONNACK	Server to Client	Connect acknowledgment
PUBLISH	Client to Server or Server to Client	Publish message
PUBACK	Client to Server or Server to Client	Publish acknowledgment
PUBREC	Client to Server or Server to Client	Publish received (assured delivery part 1)
PUBREL	Client to Server or Server to Client	Publish release (assured delivery part 2)
PUBCOMP	Client to Server or Server to Client	Publish complete (assured delivery part 3)
SUBSCRIBE	Client to Server	Client subscribe request
SUBACK	Server to Client	Subscribe acknowledgment
UNSUBSCRIBE	Client to Server	Unsubscribe request
UNSUBACK	Server to Client	Unsubscribe acknowledgment
PINGREQ	Client to Server	PING request
PINGRESP	Server to Client	PING response
DISCONNECT	Client to Server	Client is disconnecting



MQTT Control Packet Structure

Fixed header, present in all MQTT Control Packets
Variable header, present in some MQTT Control Packets
Payload, present in some MQTT Control Packets



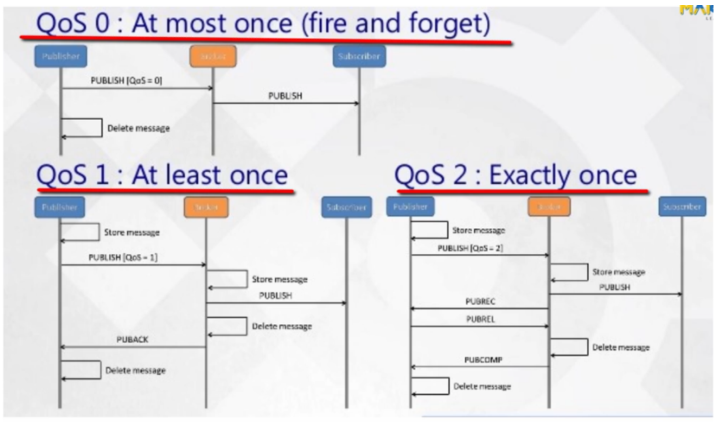
Quality of Service

MQTT defines three level of Quality of Service(QOS).

QOS defines how hard the broker/client will try to ensure that a message is received.

Message may be sent at any QOS level, and client may attempt to subscribe to topic at any QOS level.

This means that the clinet chooses the maximum QOS.



→ Higher levels of QOS are more reliable, but involves higher latency and have higher bandwidth requirements.

a) QOS 0

⇒ The broker/client will deliver message once, with no confirmation.

b) QOS 1

→ The broker/client will deliver message atleast once, with confirmation required.

c) QOS2

⇒ The broker/client will deliver message exactly once by using 4 step handshake.

Keep-alive

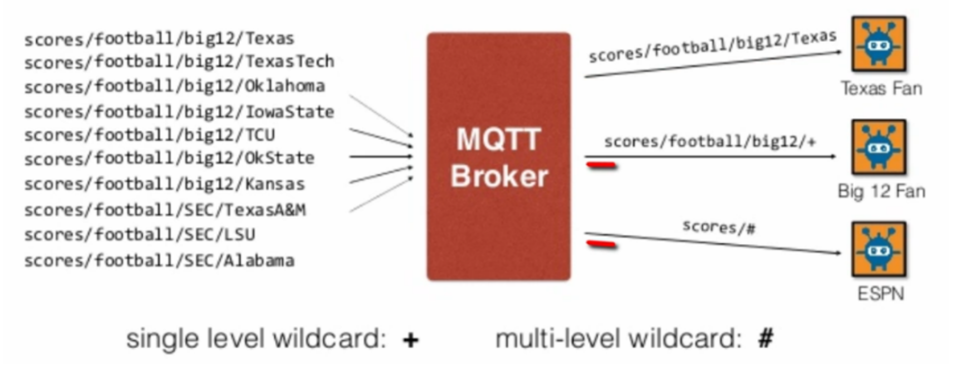
- The keep-alive functionality assures that the connection is still open and both broker and client are connected to one another.
- The client ensures that the interval between Control Packets being sent does not exceed the keep alive value.
  - The PINGREQ is sent by the client, to indicate to the broker that the client is still alive, even if hasent sent any other packets.
  - When receiving a PINGREQ the broker replies with a PINGRESP packet to indicate its availability to the client.

Last will and Testament

- This feature in MQTT notify other clients about an ungracefully disconnected client.
- Each client can specify its last will message when connecting to a broker.
- The stored LWT message will be discarded if a client disconnects gracefully by sending a DISCONNECT message.

Wildcard

When a client subscribes to a topic it can use the exact topic the message was published or it can subscribe to more topic at once by using wildcard.

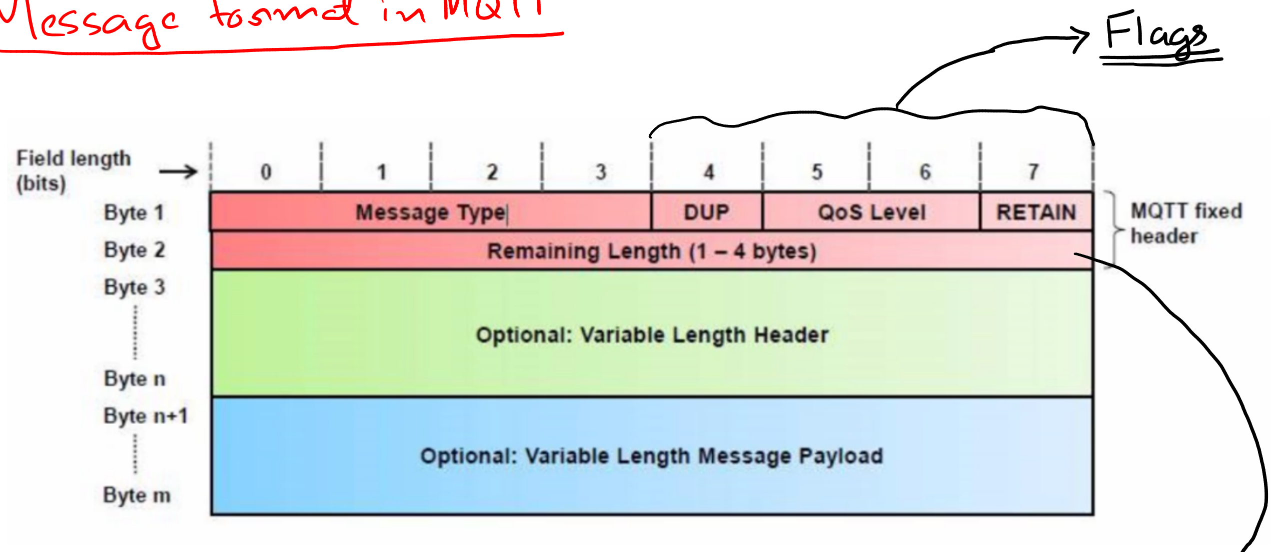


scores/football/big12/Texas
scores/football/big12/TexasTech
scores/football/big12/Oklahoma
scores/football/big12/IowaState
scores/football/big12/TCU
scores/football/big12/OkState
scores/football/big12/Kansas
scores/football/SEC/TexasA&M
scores/football/SEC/LSU
scores/football/SEC/Alabama
MQTT Broker
scores/football/big12/Texas
Texas Fan
scores/football/big12/+
Big 12 Fan
scores/#
ESPN
single level wildcard: +
multi-level wildcard: #

Retained message

- A retained message is a normal MQTT message with the retained flag set to true.
- The broker will store the last retained message and corresponding QOS of the topic.
- Each client that subscribes to a topic pattern, which matches the topic of the retained message will receive the message immediately after subscribing.
- Retained message on a topic is the last known good value.
  - It does not have to be the last value, but it certainly is the last message with the retained flag set to true.

★ Message format in MQTT



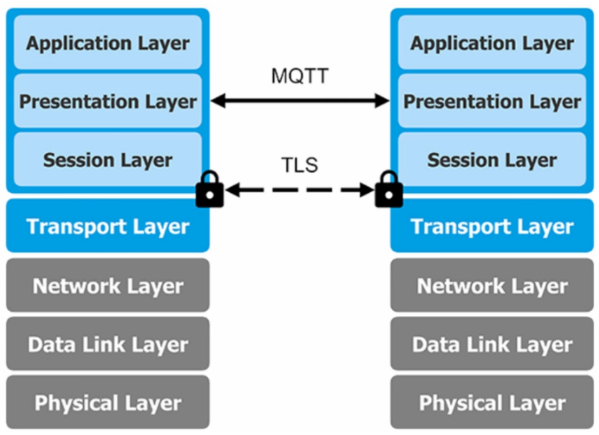
Represents the number of bytes remaining within the current message, including data in the variable header and the payload.

★ Mosquitto MQTT Broker

⇒ It is an opensource message broker that implements the MQTT protocol versions 3.1 and 3.1.1.

⇒ Added features in mosquitto that are not described in the MQTT specification:

- → MQTT bridge, to allow mosquito to connect to other MQTT brokers.
  - → This allow sharing of client data between two brokers.
- → Ability to secure connections using SSL/TSL.
- → User authorization and authentication, the ability to restrict user access to MQTT topic.



- Starting Mosquitto: $ suod /etc/init.d/mosquitto start
- Stoping Mosquitto: $ suod /etc/init.d/mosquitto stop
- Mosquitto Status : $ suod /etc/init.d/mosquitto status

Mosquitto-clients {Contains command line tools for publishing and subscribing on MQTT topics.}

→ mosquitto_pub

```
mosquitto_pub -h 192.168.1.1 -p 1885 -t sensors/temperature -m "1266193804 32"
```
{Example}

→ mosquitto_sub

```
mosquitto_sub -t sensors/temperature -q 1
```
{Example}