

# IMC-4302A Wireless Networks

## Message Queuing Telemetry Transport (MQTT)

Sofiane Imadali, PhD [sofiane.imadali@orange.com](mailto:sofiane.imadali@orange.com)

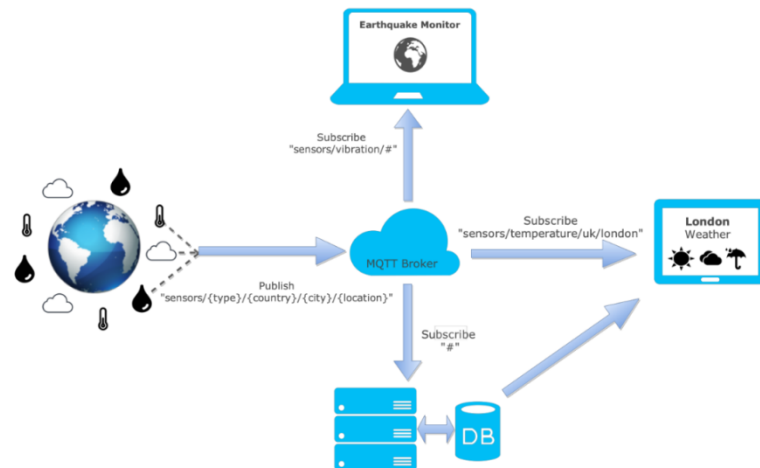


# Summary

- **Definitions**
  - Some history and generalities
- **Architecture**
  - The publish/subscribe model
  - Clients, brokers, and topics
- **MQTT Communication Patterns**
  - Point-to-Point, Broadcast, Fan-In
- **Best practices from the industry**
- **Example applications using MQTT**

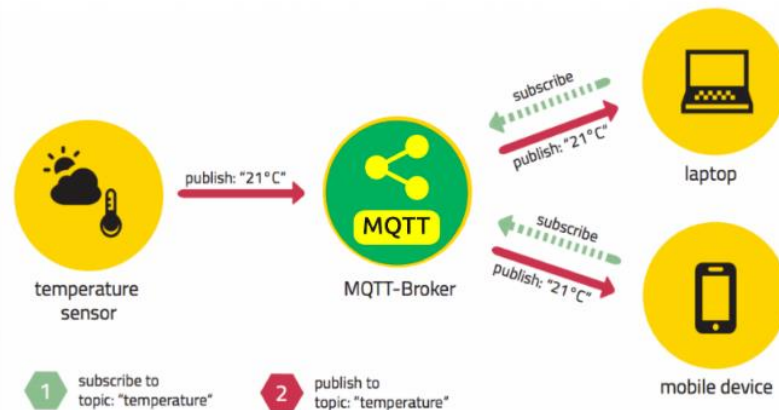
# Definitions (1/2)

- MQTT is a lightweight publish/subscribe messaging protocol designed for M2M (machine to machine) telemetry in low bandwidth environments.
  - Designed by Andy Stanford-Clark (IBM) and Arlen Nipper in 1999 for connecting Oil Pipeline telemetry systems over satellite.
  - it started as a proprietary protocol it was released Royalty free in 2010 and became an OASIS standard in 2014.
- MQTT stands for **M**essage **Q**ueuing **T**elemetry **T**ransport
  - ISO-ISO (IEC/PRF 20922) standard
  - Current version of the standard is v5.0 (March 2019)
  - Latest version before was v3.1.1. This is the version that's currently deployed



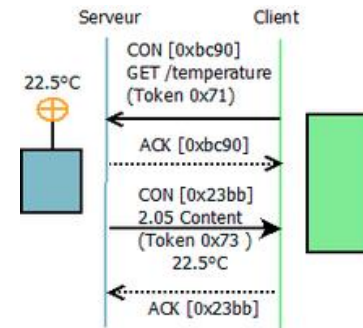
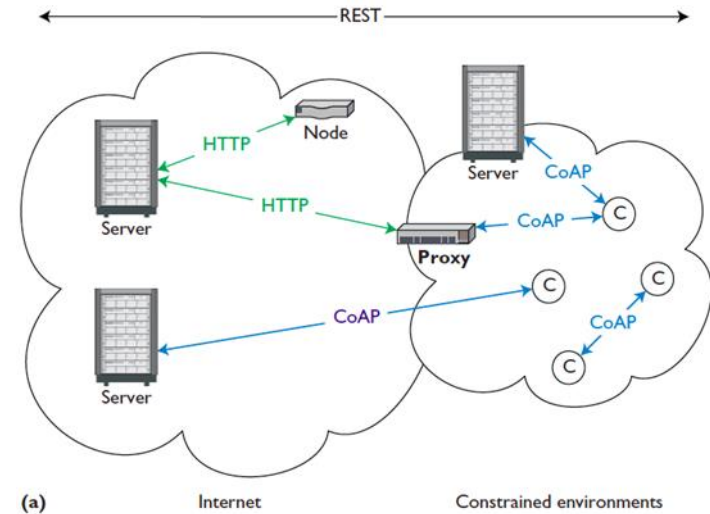
# Definitions (2/2)

- MQTT is becoming one of the main protocols for IOT (internet of things) deployments.
  - Proposed as a service by all major cloud providers: AWS, GCP, Azure
  - The version that is proposed by cloud providers is v3.1.1
- The biggest argument for the publish/subscribe topology is **« separation of concerns »**
  - Spatial concerns : The publisher and subscriber do not need to know each other
  - Temporal concerns : The publication and subscription do not need to execute in the same time
  - Synchronisation concerns : The publish and reception operation do not interfere



# An alternative: CoAP and HTTP

- CoAP is a web protocol used in M2M with constrained requirements
- Uses Asynchronous message exchange
- Has Low overhead and very simple to parse
- URI and content-type support
- Proxy and caching capabilities
- CoAP protocol uses two kinds of messages:
  - Confirmable message: used for reliable delivery of messages, the other party sends an acknowledge message (ACK) with the same ID
  - Non-confirmable message: unreliable messages that don't require an Acknowledge by the server

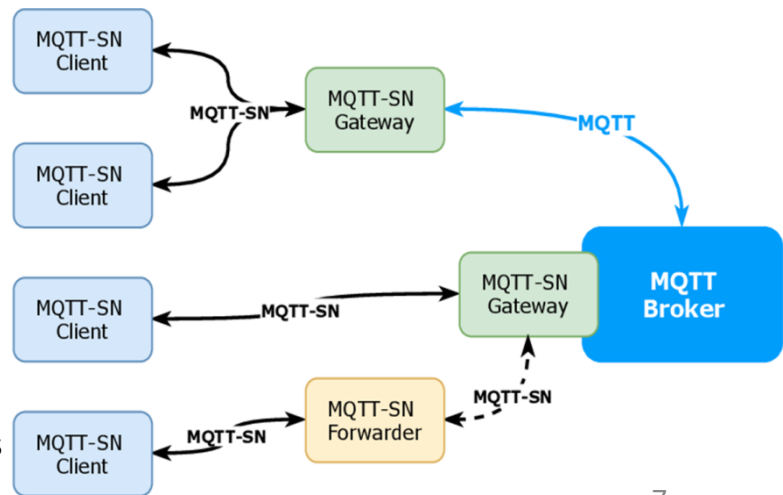


# MQTT and MQTT-SN

- We know 2 variants of MQTT depending on the domain it is used: MQTT or MQTT for sensor networks
- MQTT (up until version v5.0)
  - Primarily designed for wired connections
  - Works over TCP with 3 levels of QoS (0, 1, 2)
  - Defined over non-secure communication port 1883 TCP
  - Encrypted communications (secure-mqtt): TCP 8883(TLS v1.2, v1.1 or v1.0 with x509 certificates)
  - Port 443 TCP is used by major cloud platforms for Websockets
- MQTT For Sensor Networks (MQTT-SN, based mainly on)
  - Wireless radio links have higher failure rates than wired ones (fading, interference)
  - Wireless has a lower transmission rate.
  - Works over UDP, TCP stack is too complex for WSN
  - Designed for non-TCP/IP networks such as ZigBee
  - Defined over non-secure communication port 1883 UDP
  - Encrypted communications (secure-mqtt): TCP 8883(TLS v1.2, v1.1 or v1.0 with x509 certificates)

# MQTT vs MQTT-SN

- MQTT-SN is designed to work in the same way as MQTT
- Some differences
  - Reduces the size of the payload
  - Uses UDP to avoid the permanent connection that needs to be kept alive
  - Has a different architecture with 3 components: client, gateway, forwarder. The gateway plays the role of broker
  - Uses shorter topic names and predefined topics
  - Discovery process to discover Gateways using multicast messages. 2 models:
    - Advertising by a broker or Gateway
    - A Search by the client
  - Has an additional level of QoS 3 (or: -1)
    - doesn't require the set-up of an initial connection
    - requires the use of short topic names or pre-defines topics



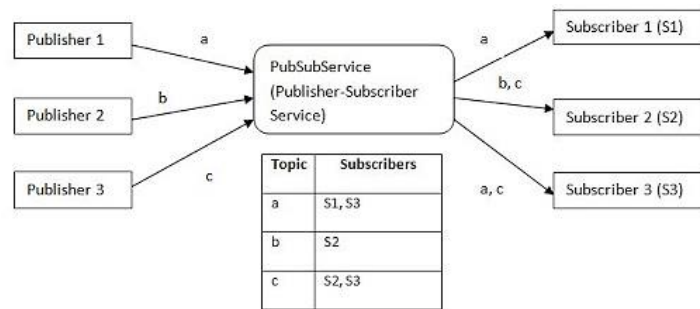
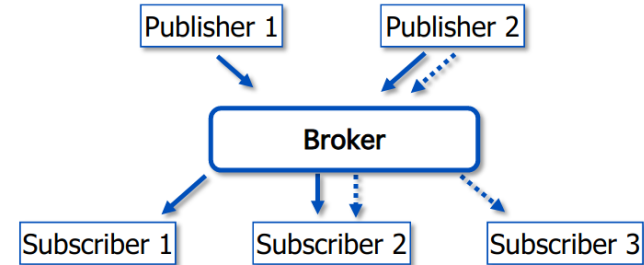
# Summary

- Is MQTT standard ? What is the latest version ? What do we use MQTT for ?
- What is the data exchange model used in MQTT ?
- Give two messages in MQTT and explain them ?
- How many QoS levels are there ? What is the difference between QoS1 and QoS2 ?
- What are the communication patterns used in MQTT ?
- I want to access the room 120 temperature from the building XYZ, what URI do I use ?
- I want to know the average temperature in the rooms of a certain building, what is the communication pattern used ? Where do I place a wildcard in the URI ?



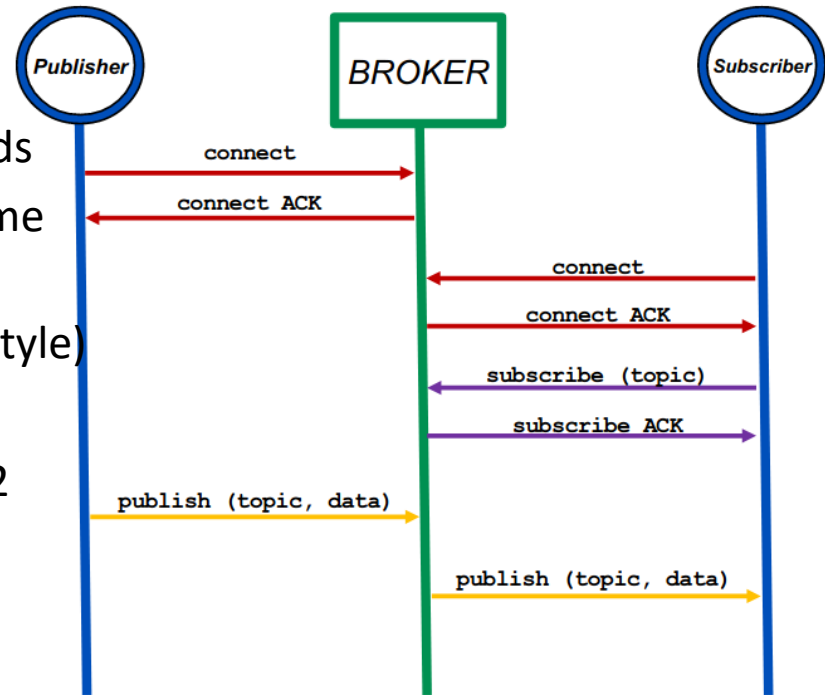
# Architecture: publish/subscribe

- Enable an application to announce events to multiple interested consumers asynchronously, without coupling the senders to the receivers
- This is also known as the producer/consumer pattern
- Some protocols also use this communication pattern
  - AMQP, XMPP (Jabber)
- In the Big data world, Kafka is the most well known tool to use this pattern



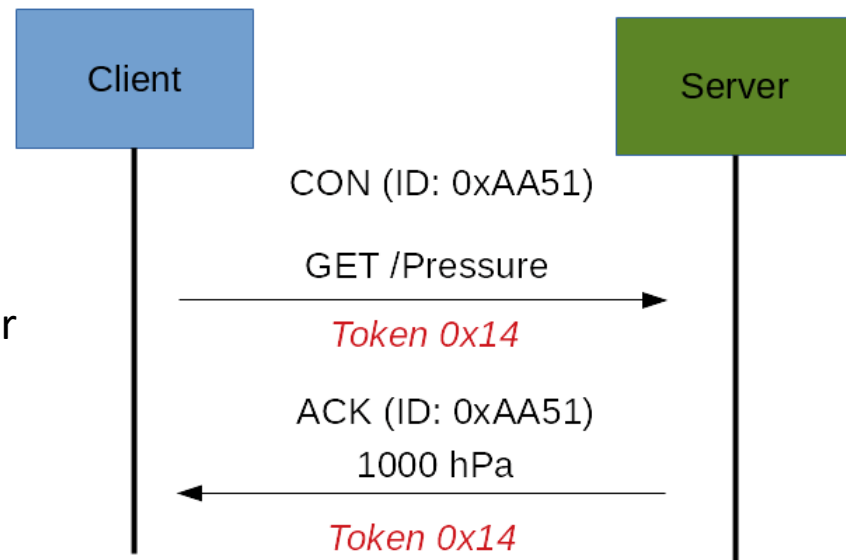
# Protocol: connection

- All clients are required to have unique names or Ids
- Clients can disconnect and reconnect with the same names/Ids
- Messages are ACKnowledged by the broker (TCP style)
- Publishing and Subscribing depends on the topic
- Clients can require a level of QoS between 0 and 2
- Clients can include a Last Will and Testament
  - Default message to be broadcast if the termination is not normal
  - If the publisher disconnects normally the last Will Message is not sent

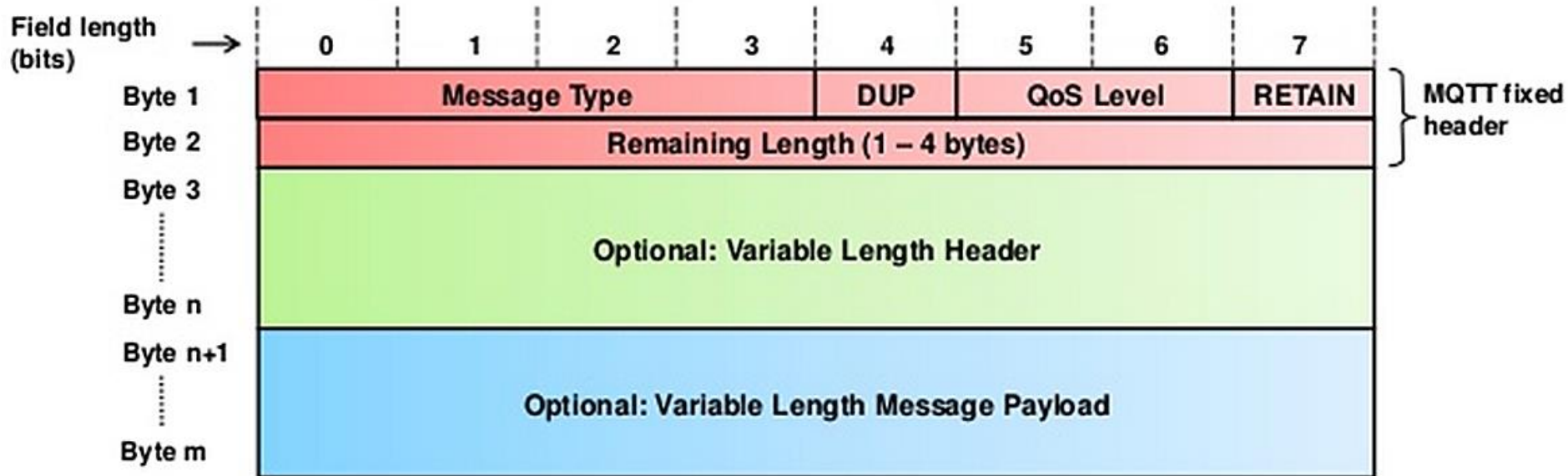


# Comparison with CoAP

- CoAP also has connected message exchanges
  - But also a disconnected mode with no ACK
- In the Server/Client dialogue, a message ID is included and is the same back and forth
- The communication style is based on REST
- IETF rfc7252 defines the protocol and behaviour



# Protocol: message format



# Protocol: control messages (1/2)


Name	Value	Direction of flow	Description
Reserved	0	Forbidden	Reserved
CONNECT	1	Client to Server	Client request to connect to Server
CONNACK	2	Server to Client	Connect acknowledgment
PUBLISH	3	Client to Server or Server to Client	Publish message
PUBACK	4	Client to Server or Server to Client	Publish acknowledgment
PUBREC	5	Client to Server or Server to Client	Publish received (assured delivery part 1)
PUBREL	6	Client to Server or Server to Client	Publish release (assured delivery part 2)

# Protocol: control messages (2/2)

Name	Value	Direction of flow	Description
PUBCOMP	7	Client to Server or Server to Client	Publish complete (assured delivery part 3)
SUBSCRIBE	8	Client to Server	Client subscribe request
SUBACK	9	Server to Client	Subscribe acknowledgment
UNSUBSCRIBE	10	Client to Server	Unsubscribe request
UNSUBACK	11	Server to Client	Unsubscribe acknowledgment
PINGREQ	12	Client to Server	PING request
PINGRESP	13	Server to Client	PING response
DISCONNECT	14	Client to Server	Client is disconnecting
Reserved	15	Forbidden	Reserved

# Protocol: Connect message

- The first message from a client to the broker, to publish or subscribe is a **CONNECT** message
- Connect messages can contain:
  - Name or Ids
  - Username/password for basic auth
  - Last will and testament
  - Keep alive timeout
- **CONNECT** message is answered by the broker with a **CONNACK** message

MQTT-Packet:	
CONNECT 	
contains:	Example
<b>clientId</b>	"client-1"
<b>cleanSession</b>	true
<b>username</b> (optional)	"hans"
<b>password</b> (optional)	"letmein"
<b>lastWillTopic</b> (optional)	"/hans/will"
<b>lastWillQos</b> (optional)	2
<b>lastWillMessage</b> (optional)	"unexpected exit"
<b>lastWillRetain</b> (optional)	false
<b>keepAlive</b>	60

# Protocol: Publish message

- After connecting to the broker, a client may want to publish a metric
- The publish message may contain:
  - The packet ID
  - The topic name
  - QoS level
  - The payload
  - Retention and Duplicate
- PUBLISH has several linked possible answers: PUBACK, PUBREC, PUBREL, PUBCOMP



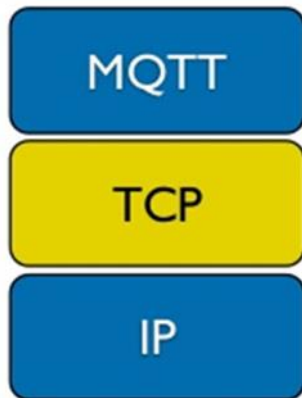


# Summary

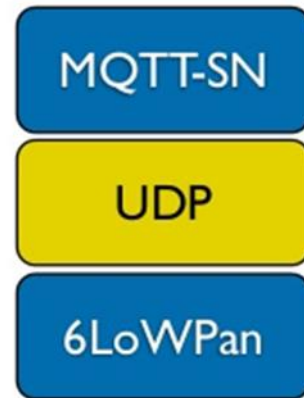
- Is MQTT standard ? What is the latest version ? What do we use MQTT for ?
- What is the data exchange model used in MQTT ?
- **Give two messages in MQTT and explain them ?**
- How many QoS levels are there ? What is the difference between QoS1 and QoS2 ?
- What are the communication patterns used in MQTT ?
- I want to access the room 120 temperature from the building XYZ, what URI do I use ?
- I want to know the average temperature in the rooms of a certain building, what is the communication pattern used ? Where do I place a wildcard in the URI ?

# Protocol: Layer 4 (and above)

- Uses TCP for communication
- Port 1883 by default
- Port 8883 for encrypted communications

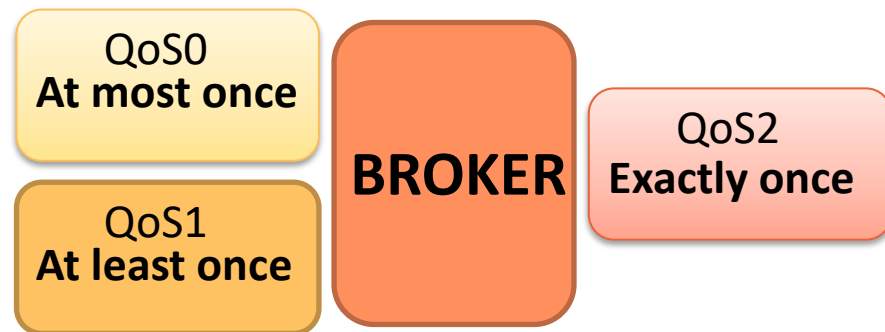


- Uses UDP for communication
- Port 1883 by default
- Port 8883 for encrypted communications



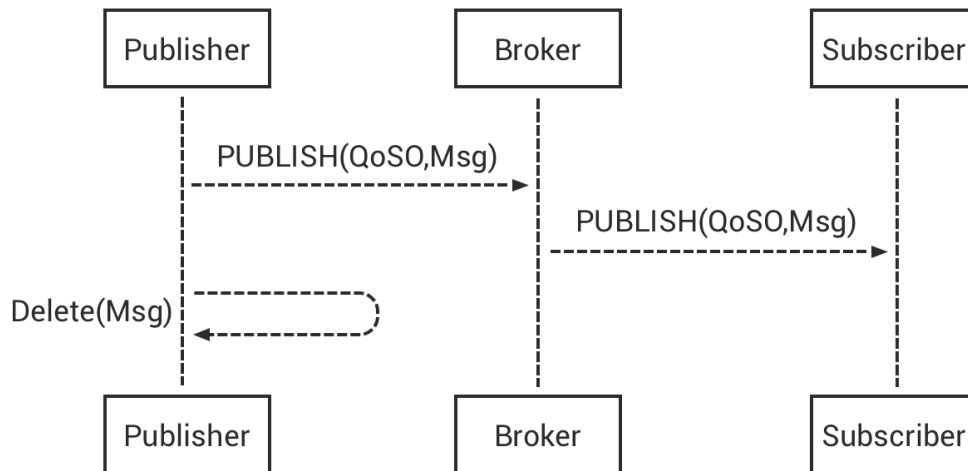
# Protocol: QoS (1/4)

- Quality of Service in the MQTT protocol is an important feature around which the protocol is designed
- 3 levels can be found
- For levels 1 and 2, the broker has to have the retention feature (a database) to store messages



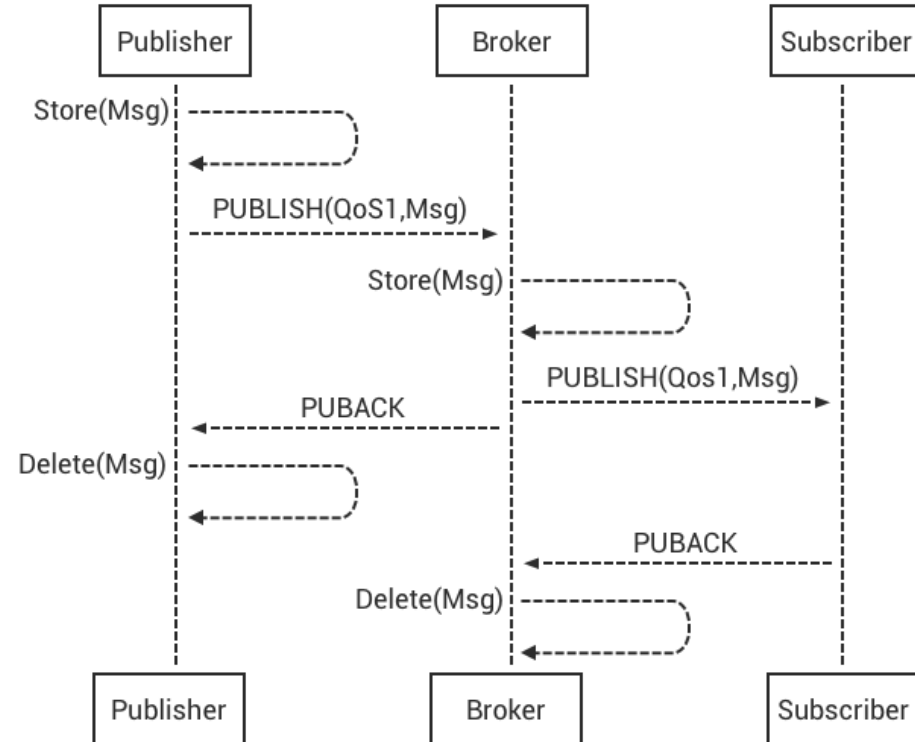
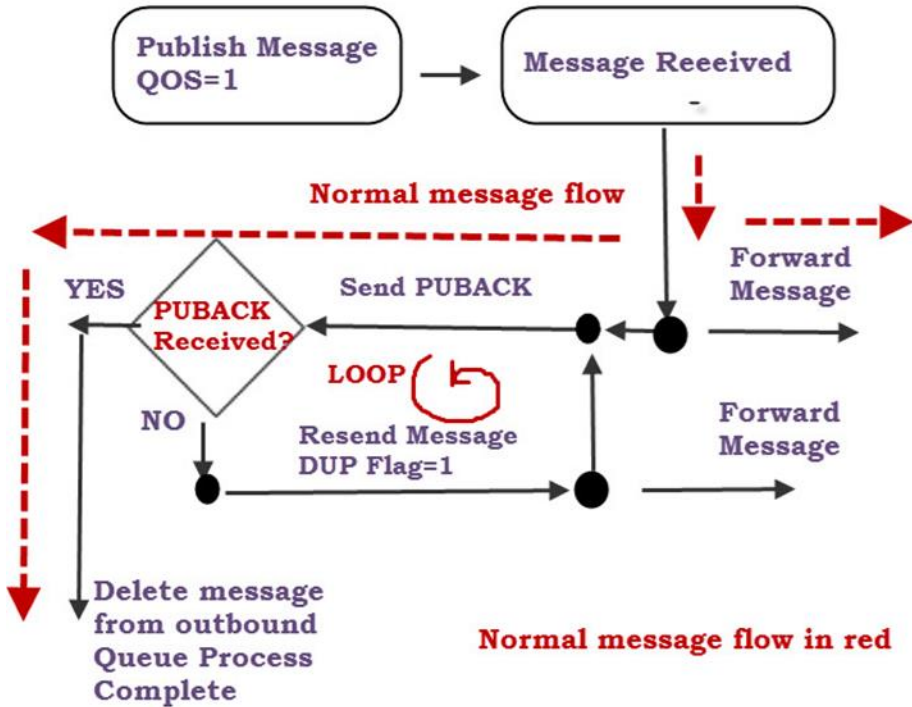
# Protocol: QoS (2/4)

QoS 0: At most once (deliver and forget)

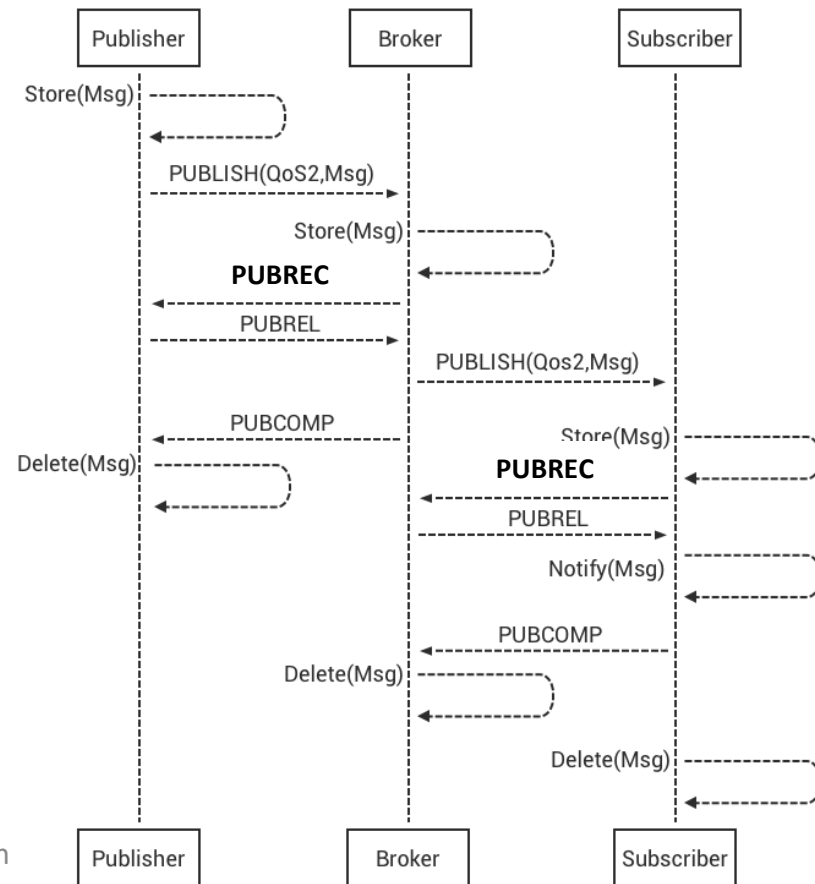


# Protocol: QoS (3/4)

QoS 1:AT least once



QoS 2:Exactly once



# Summary

- Is MQTT standard ? What is the latest version ? What do we use MQTT for ?
- What is the data exchange model used in MQTT ?
- Give two messages in MQTT and explain them ?
- **How many QoS levels are there ? What is the difference between QoS1 and QoS2 ?**
- What are the communication patterns used in MQTT ?
- I want to access the room 120 temperature from the building XYZ, what URI do I use ?
- I want to know the average temperature in the rooms of a certain building, what is the communication pattern used ? Where do I place a wildcard in the URI ?

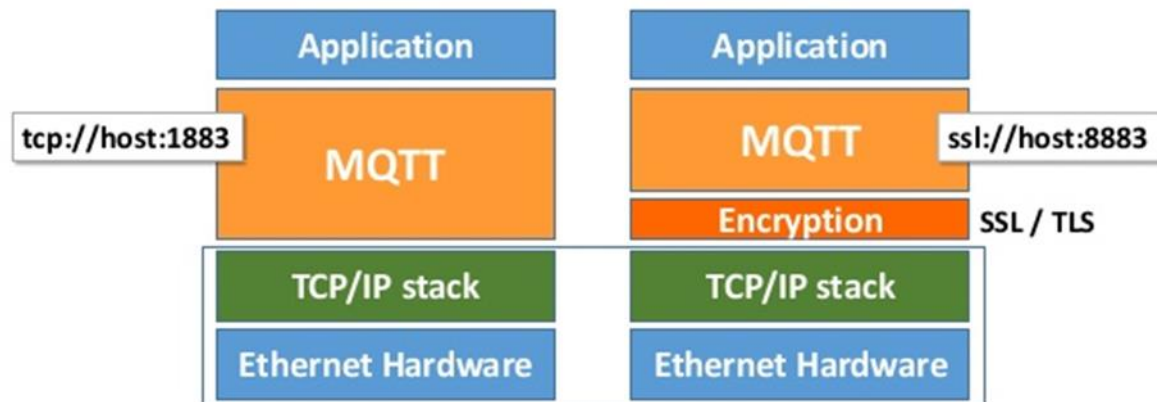
# Protocol: QoS and message retention

- A retained message is a normal MQTT message with the **retained flag set to true**.
  - The broker will **store the last retained** message and the corresponding QoS for that topic
  - Each client that subscribes to a topic pattern, which matches the topic of the retained message, will receive the message immediately after subscribing.
  - For each topic only one retained message will be stored by the broker.
- **Retained messages can help newly subscribed clients to get a status update immediately after subscribing to a topic and don't have to wait until a publishing clients send the next update.**
  - In other words a retained message on a topic is the last known good value, because it doesn't have to be the last value, but it certainly is the last message with the retained flag set to true.



# Protocol: Security

- Uses TCP for communication
  - Port 1883 by default
  - No encryption, clear messages
- Uses SSL/TLS over TCP
  - Port 8883 by default
  - You can use 443 with websockets
  - End-to-end Encryption

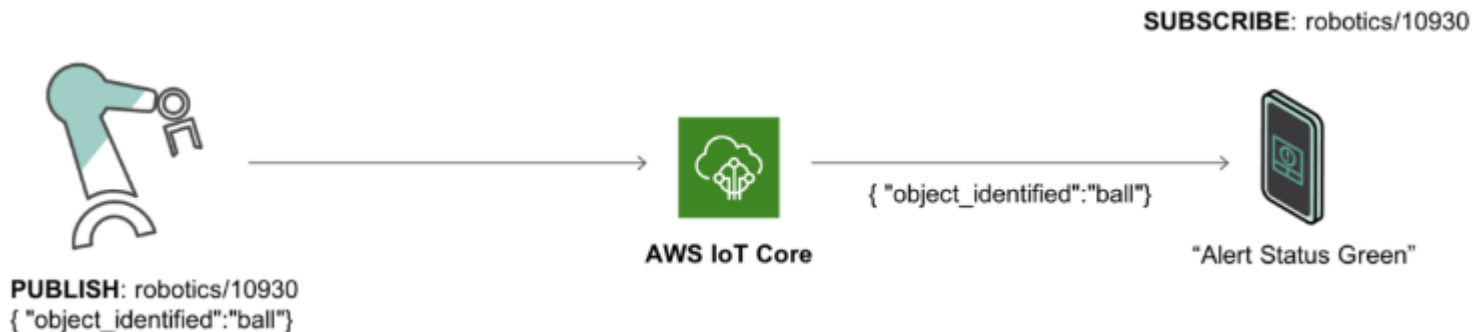


**Both communication modes are supported by major cloud providers**

# Communication patterns (1/3)

## Point-to-Point

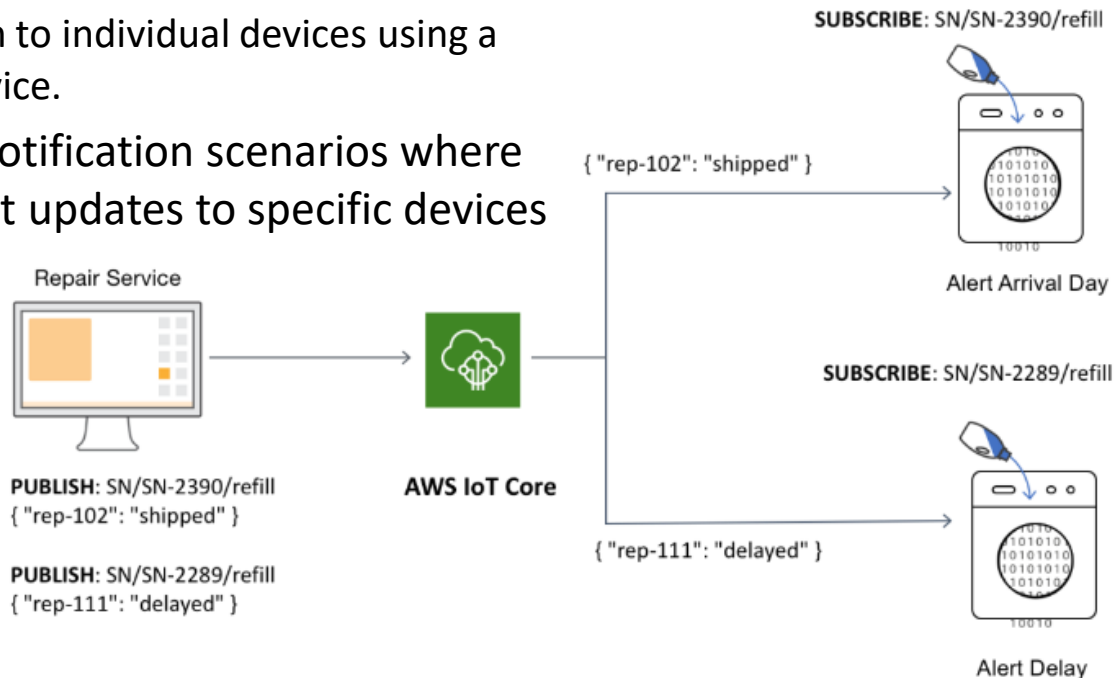
- Two things use a single MQTT topic as the communication channel.
  - The thing that receives the event subscribes to an MQTT topic.
  - The thing that sends the message publishes to the same known MQTT topic.



# Communication patterns (1/3)

## Point-to-Point

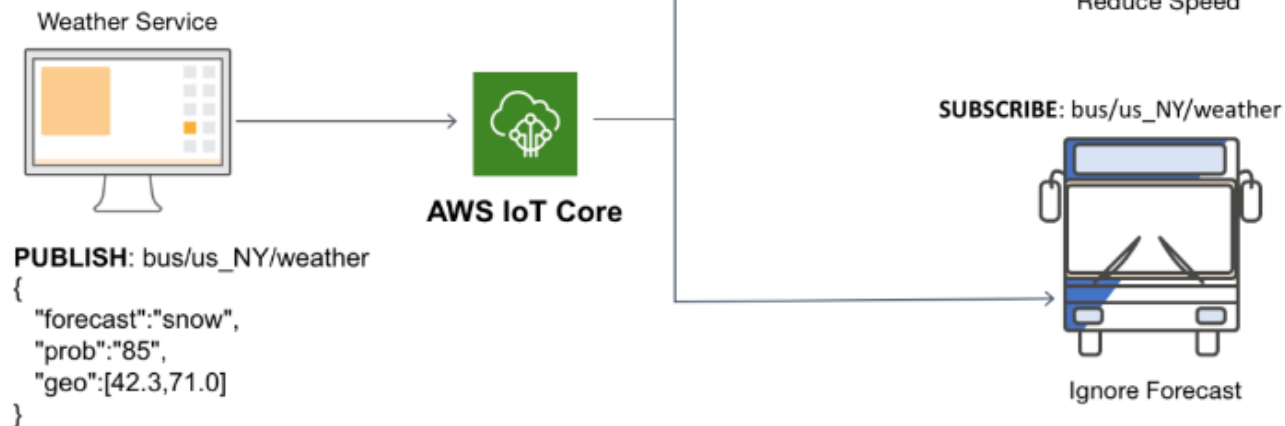
- Point-to-point is also used in one-to-many communication :
  - a single publisher can publish to individual devices using a different MQTT topic per device.
- This approach is common in notification scenarios where an administrator sends distinct updates to specific devices



# Communication patterns (2/3)

## Broadcast

- Broadcast patterns are used for one-to-many messaging.
  - Sends the same message to a large fleet of devices
  - Multiple devices subscribe to the same MQTT topic, and the sender publishes a message to that shared top
- Typical use: send a notification to devices based on the category in a geolocation



# Communication patterns (3/3)

## Fan-In

- Many-to-one communication → reverse of the broadcast pattern
- Multiple devices publish on a shared topic with a single subscriber to that topic
- The subscriber has the added ability to leverage wildcards to facilitate aggregation of telemetry

**PUBLISH:** robotics/building1234/5678  
{ "status": "green" }



{ "status": "green" }



{ "status": "red" }

**PUBLISH:** robotics/building1234/1234  
{ "status": "red" }



AWS IoT Core

{ "status": "green" }  
{ "status": "red" }



AWS IoT Rule



Amazon Kinesis

**RULE SELECT:** robotics/building1234/+

# Summary

- Is MQTT standard ? What is the latest version ? What do we use MQTT for ?
- What is the data exchange model used in MQTT ?
- Give two messages in MQTT and explain them ?
- How many QoS levels are there ? What is the difference between QoS1 and QoS2 ?
- **What are the communication patterns used in MQTT ?**
- I want to access the room 120 temperature from the building XYZ, what URI do I use ?
- I want to know the average temperature in the rooms of a certain building, what is the communication pattern used ? Where do I place a wildcard in the URI ?

# MQTT Topics (1/2)

- MQTT Topics are structured in a hierarchy similar to folders and files in a file system using the forward slash ( / ) as a delimiter
- Allow to create a user friendly and self descriptive naming structures
- Topic names are:
  - Case sensitive
  - Use UTF-8 strings.
  - Must consist of at least one character to be valid.


 myhome / groundfloor / livingroom / temperature

- One special topic: \$SYS
  - Except for the \$SYS topic, there is no default or standard topic structure

Special \$SYS/ topics

\$SYS/broker/clients/connected  
 \$SYS/broker/clients/disconnected  
 \$SYS/broker/clients/total  
 \$SYS/broker/messages/sent  
 \$SYS/broker/uptime

# MQTT Topics (2/2)

- Topic subscriptions can have wildcards. These enable nodes to subscribe to groups of topics that don't exist yet, allowing greater flexibility in the network's messaging structure.
  - '+' matches anything at a given tree level
  - '#' matches a whole sub-tree
- Subscribing to topic **house/#** covers:
  - house/room1/main-light
  - house/room1/alarm
  - house/garage/main-light
  - house/main-door

**Multi-Level  
Wildcard**
- Subscribing to topic **house/+/main-light** covers:
  - house/room1/main-light
  - house/room2/main-light
  - house/garage/main-light

**Single-Level  
Wildcard**
- But doesn't cover
  - house/room1/side-light
  - house/room2/side-light



# Best practices for Topics (1/2)

- Ensure MQTT topic levels only use lowercase letters, numbers, and dashes
  - MQTT topics are case sensitive
- Ensure MQTT topic levels structure follows a general to specific pattern.
  - For example, an HVAC system is associated with an IoT platform named hv100, is located in the basement of building bld1518, and has a Thing Name of hvac719
- Include any relevant routing information in the MQTT topic
  - The IoT application identifier, any groups the device may be a part of, such as installed location, and the unique identity of your IoT device

hv100/bld1518/basement/hvac719

# Best practices for Topics (2/2)

- Use the IoT Thing Name as the MQTT client ID for connecting as a device over MQTT
  - Use the unique Thing Name per device as the MQTT client ID so you are more easily able to correlate information
- Include the Thing Name of the device in any MQTT topic the device uses for publishing or subscribing to its data to track messages destined for a particular device
- Include additional contextual information about a specific message in the payload of the MQTT message
  - For scalability reasons, do not allow a single device to subscribe to a shared topic that is being published to by a large number of other devices
- Never allow a device to subscribe to all topics using #, and only use multi-level wildcard subscriptions in IoT rules
  - use single level wildcards (+) and not multi-level wildcards to avoid unintended consequences such as the creation of new topics

# Summary

- Is MQTT standard ? What is the latest version ? What do we use MQTT for ?
- What is the data exchange model used in MQTT ?
- Give two messages in MQTT and explain them ?
- How many QoS levels are there ? What is the difference between QoS1 and QoS2 ?
- What are the communication patterns used in MQTT ?
- I want to access the room 120 temperature from the building XYZ, what URI do I use ?
- I want to know the average temperature in the rooms of a certain building, what is the communication pattern used ? Where do I place a wildcard in the URI ?

# Conclusion

- MQTT is the true standard for IoT communications
- It is based on the publish/subscribe pattern
- It is widely used by the industry and has been adopted by all major players
- The QoS feature is very important to cover all the possible use cases
- When you implement the protocol for your industry, you should:
  - Pay extra attention to security
  - Take the best practices at heart with regards to topics and system design