IoT
MQTT

Dr. Sarwan Singh
Deputy Director(S)
NIELIT Chandigarh

Agenda

- Histroy
- Types, Usage
- Commands

The term Internet of Things was first used by **Kevin Ashton** in 2009 for interconnecting physical devices over the internet. The basic idea is very simple: *Physical devices can exchange data between each other or being controlled by others.*

History

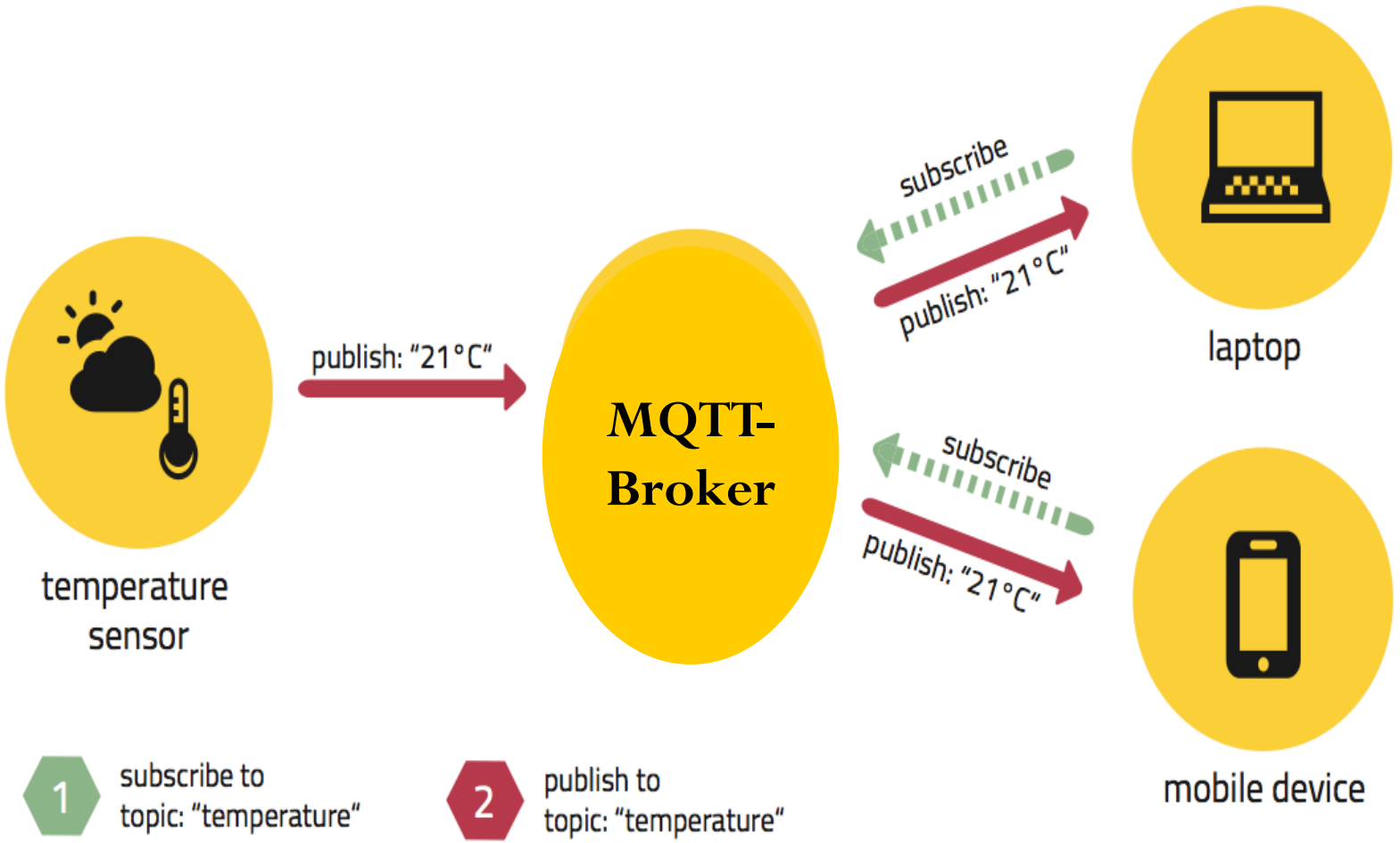
- MQTT was developed by Andy Stanford-Clark (IBM) and Arlen Nipper (Eurotech; now Cirrus Link) in 1999 for the monitoring of an oil pipeline through the desert.
- The goals were to have a protocol, which is bandwidth-efficient and uses little battery power, because the devices were connected via satellite link and this was extremely expensive at that time.

Source : hivemq.com

What is MQTT

1. MQTT is a binary client-server publish/subscribe messaging transport protocol
2. It is lightweight, open, simple, and easy to implement.
3. Designed with a minimal protocol overhead,
4. This protocol is a good choice for a variety of (M2M) and IoT applications
5. MQTT utilizes many characteristics of the TCP transport.
6. So the minimum requirement for using MQTT is a working TCP stack, which is now available for even the smallest microcontrollers.
7. The most recent version of MQTT is 3.1.1, which has many improvements over the first public MQTT release, MQTT 3.1.

MQTT working



Source : hivemq.com

MQTT working

- The protocol uses a publish/subscribe architecture in contrast to HTTP with its request/response paradigm.
- Publish/Subscribe is event-driven and enables messages to be pushed to clients.
- The central communication point is the MQTT broker, it is in charge of dispatching all messages between the senders and the rightful receivers.
- Each client that publishes a message to the broker, includes a topic into the message. The topic is the routing information for the broker.

Source : hivemq.com

- Each client that wants to receive messages subscribes to a certain topic and the broker delivers all messages with the matching topic to the client. Therefore the clients don't have to know each other, they only communicate over the topic. This architecture enables highly scalable solutions without dependencies between the data producers and the data consumers.

MQTT Message Types

- MQTT has 14 different message types.
- Typically, end users only need to employ the CONNECT, PUBLISH, SUBSCRIBE, and UNSUBSCRIBE message types.
- The other message types are used for internal mechanisms and message flows.

MESSAGETYPE	DESCRIPTION
CONNECT	Client request to connect to Server Connection Acknowledgement
CONNACK	Connection Acknowledgement
PUBLISH	A message which represents a new/separate publish
PUBACK	QoS 1 Response to a PUBLISH message
PUBREC	First part of QoS 2 message flow
PUBREL	Second part of QoS 2 message flow
PUBCOMP	Last part of the QoS 2 message flow
SUBSCRIBE	A message used by clients to subscribe to specific topics

MESSAGETYPE	DESCRIPTION
SUBACK	Acknowledgement of a SUBSCRIBE message
UNSUBCRIBE	A message used by clients to unsubscribe from specific topics
UNSUBACK	Acknowledgement of an UNSUBSCRIBE message
PINGREQ	Heartbeat message
PINGRESP	Heartbeat message acknowledgement
DISCONNECT	Graceful disconnect message sent by clients before disconnecting.

Topics

- A topic is a simple string that can have more hierarchy levels, which are separated by a slash.
- A sample topic for sending temperature data of the living room could be ***house/living-room/temperature***

wildcard : ***house/+/temperature***

- my/test/topic
- my/+/topic

multilevel wildcard (#)

- my/#
- my/+/+
- +/#
- #

Topics

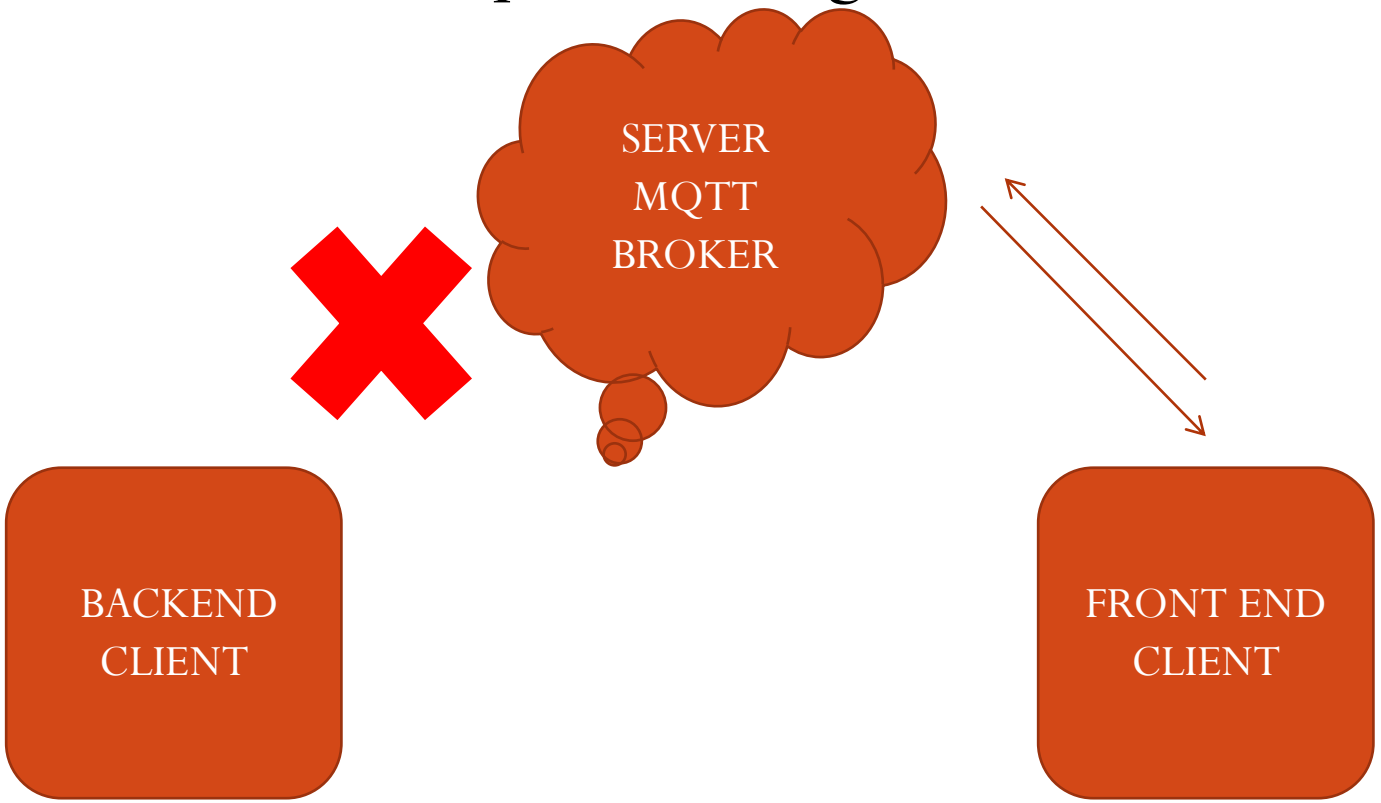
VALID MQTT TOPIC EXAMPLES

- my/test/topic
- my/+/topic
- my/#
- my/+/+
- +/#
- #

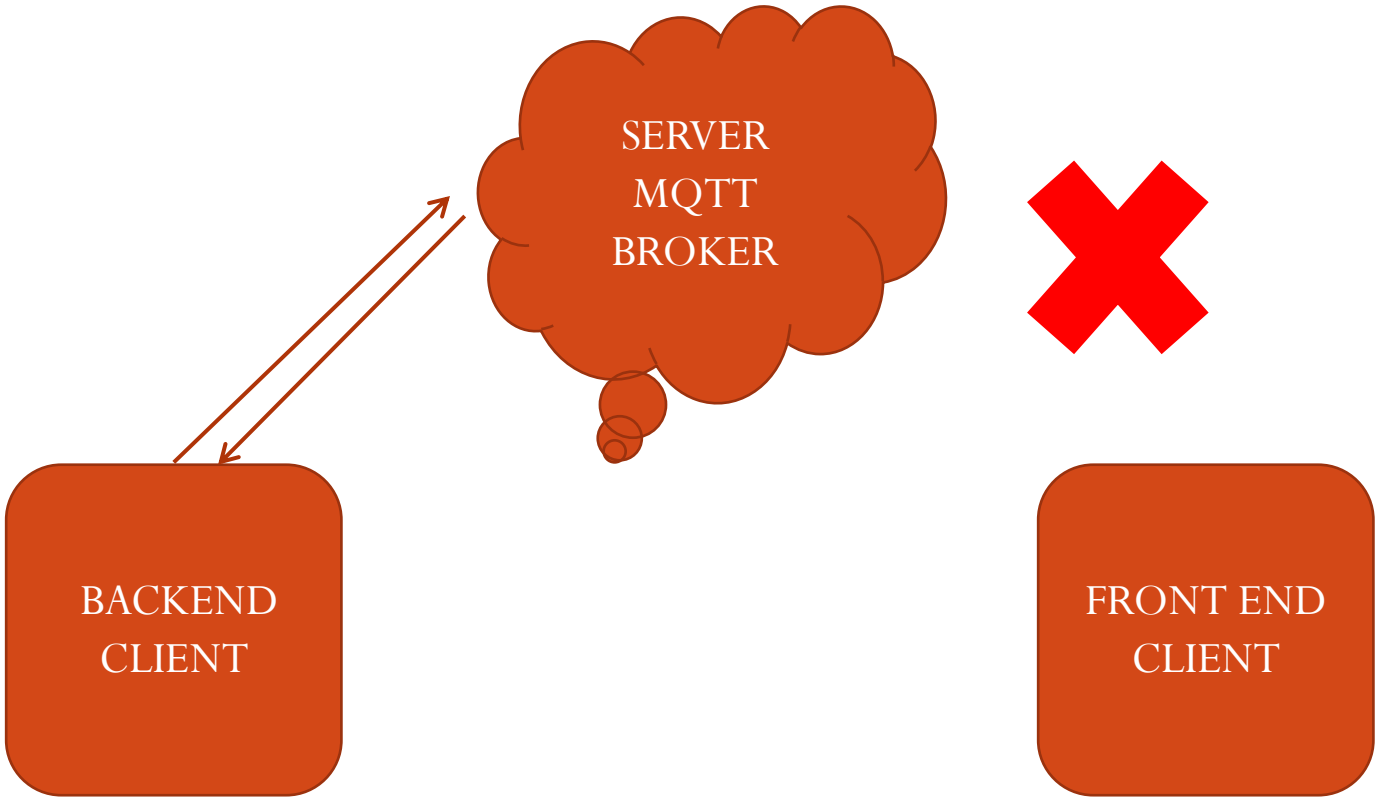
MQTT Features (A) LWT (B) Retain (C) QoS

(A) Last Will And Tastment (LWT)

LWT- which helps detecting failures of other client



(B) RETAIN



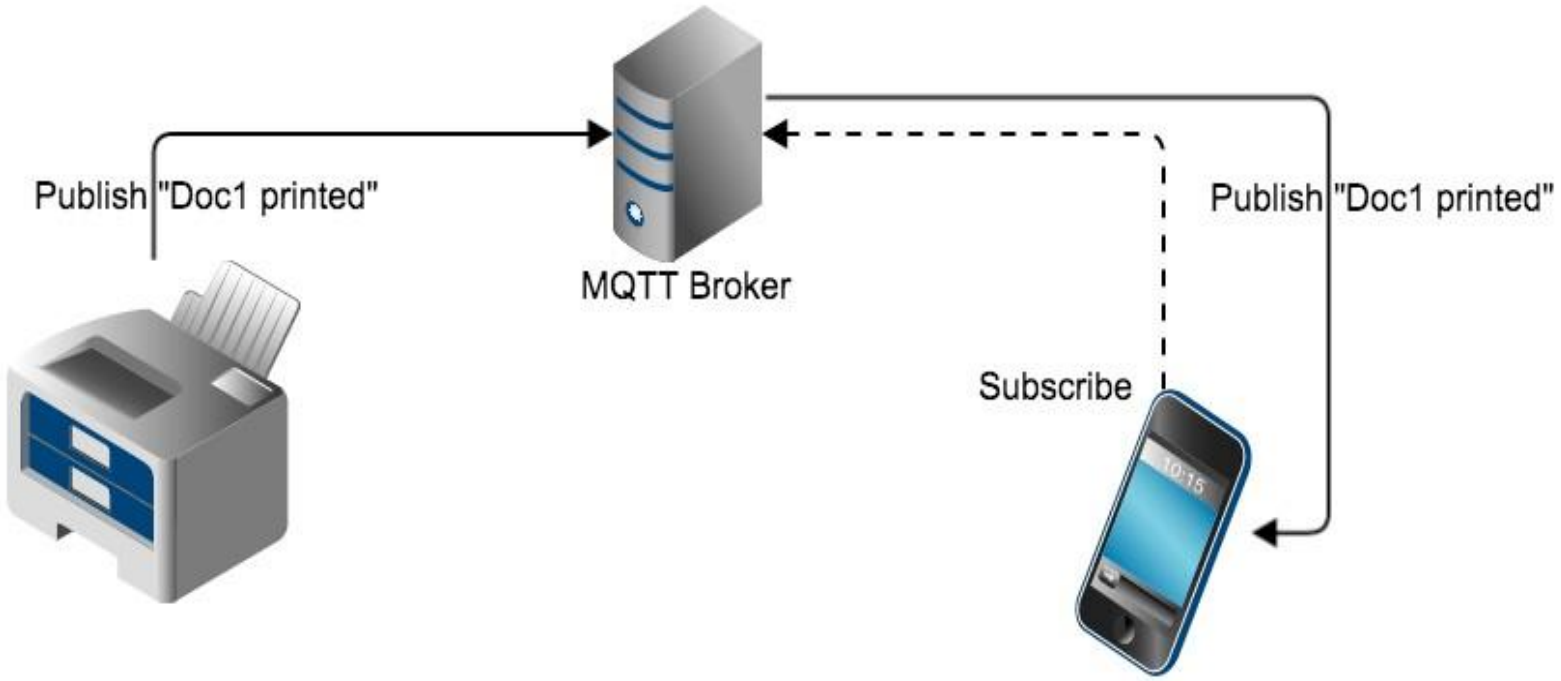
(C) Quality Of Service (QoS)

- QoS 0 – Atleast Once Delivery
- QoS 1 – Atmost Once Delivery
- QoS 2 – Exactly Once Delivery

Broker Setup

Broker Needed and what it is ?

Publish / Subscribe



Publish / Subscribe