What is mqtt

Describe, background, good stuff

Real world applications

//What does it do

How does it connect

Use what technique,

How does it receive data

# Background

MQTT(MQ Telemetry Transport) is a light weight event and message oriented protocol which allows devices to asynchronously communicate efficiently across constrained remote networks. It is designed for connections with remote locations where only a ‘small code footprint’ is required or where the network bandwidth is limited.

MQTT was built in 1999, with the intention to simplify the connection between the M2M would to the ‘internet of things’. Its use wasn’t controversial until IBM created an open source MQTT. IBM’s version of MQTT had added extra aspect into the design, including a open standard and a list of favourable, neat features.

MQTT’s favourable features provide great advantage in growing its popularity in such short order. Its low code footprint makes it possible to be put onto very small devices. It is efficient which favours in battery saving. MQTT’s low cost consumption makes is possible to conduct rapid, real-time communication with low network usage, with extremely low latency, the smallest fixed header it requires could be as little as two bytes, which is very small compared to other messaging protocols, like http. This light weight protocol is also easy to understand and simple to implement. Change this sentence so that it fits into the paragraph…

MQTT has gotten many acknowledgment as a valid protocol recently. A major one comes from Facebook. Facebook messenger is built upon MQTT, MQTT acts as a low intensive protocol, it provides low latency for about a billion users and makes sure that the messages show up instantly without destroying device battery and network.

The Amazon Web Services has announced Amazon loT based on MQTT.

Home users also receive great benefit from MQTT as many Smart Home systems and alarm systems are also built upon MQTT.

# What does it do

MQTT is designed for obtaining controlled data events from constrained networks and constrained systems into somewhere useful.

It is a messaging protocol which implement the publish/subscriber messaging (one to many) mechanism.

Say something about p/s mechanism here‼

# Usage

|  |  |
| --- | --- |
| **Term** | **Definition** |
| Topic | A structured string that defines a location in a namespace with ‘/’ used to delimit levels of tat namespace’s hierarchy. |
| Message | Strings or files in bytes that need to be sent from the publish to the consumers. |
| Publisher | Uploader of the message. |
| Server | Broker of the message sending media, listens for incoming publish requests and pushes them to corresponding consumers. |
| Consumer | Receiver of the message. |
| Subscription | A consumer can subscribe to one or more topic that they are interested in, in order to receive future updates published to that topic. |

The publish publishes a message on a Topic.

Usage:

Publisher publish message on a topic

Topic = key/ subject of interest

Once topic is published, goes into server / broker

Zero or more Consermuer connected to server express interest in receiving messages on a given topic

The server reevied request, it pushes message to consumers, if 0 consumers, the messages get discarded, if more than one, all will receive

Durable and non durable subscriptions

Durable:

Once a subs is make to broker, if I am connected , I would get message imm, if I am offline, message is stored at broker , will get them when I get online

Non durable:

Only last the life thime the subsers are connected to the broker

Another feature

Retaining message

If message is marked as retained by publisher, Broker remembers the last message on topic, so when subser first subs, they get last topic sent to them , instead of waiting for the next new one

Push messages to both direction, from prod to server, if theres a client waiting for that message, broker pushes the message across the network, so there is no polling, good mechism for timely delivery of event messages and data between produer and consumers

Polling Is expensive

Two models for working with mqtt, Great for sometime connected model and always online clients≤== timely delivery, good, realtime

Feature:

Last willing testiment

Server see device disconnect abnormally, server publish message on behave of device ,say this device is disconnected, to let other users see it aand save it or fix it

API:

Create a mqtt client oobject

Java client in paho, instantiate client obj tell what sever to connect to

Client id must be unique, if two tries to connect, the first gets kicked out, bad for atuo signins

Specify connection options and connect to it

// Keepalive() use to detect if connect breaks out , smaller number = quicker to find out but expensive

client.connect(opts)

connect with option

sending message:

creating a message:

create mqt messae, “dfgsd”.getBytes()

everything is in bytes in MQTT

can setRetained(true)

publish

sending a message:

need to know what topic to send to

so get topic and publish

the returned delivery token given by the publish is used to determine when delivery is complete,

asynchronous callback = delivery token

callback()

when msg get to server, a callback listener is notified, and returns token, to tell delivery is completed

connectionLost()

if connection breaks, client will be notified and ask if wanted to reconnect and resubcript

messageArrived()

be told topic, message content, process message in this method

hwo to subcribe:

client.subcribe(“sdf”);

client.unsubcribe(“dfsg”);

Describe clodu mqtt dashboard