**1.MQTT介绍**

Message Queuing Telemetry Transport，消息队列遥测传输协议,是一种基于发布/订阅（publish/subscribe）模式的“轻量级”通讯协议，该协议构建于TCP/IP协议上，由IBM在1999年发布。MQTT最大优点在于，可以以极少的代码和有限的带宽，为连接远程设备提供实时可靠的消息服务。作为一种低开销、低带宽占用的即时通讯协议，使其在物联网、小型设备、移动应用等方面有较广泛的应用。



实现MQTT协议需要客户端和服务器端通讯完成，在通讯过程中，MQTT协议中有三种身份：发布者（Publish）、代理（Broker）（服务器）、订阅者（Subscribe）。其中，消息的发布者和订阅者都是客户端，消息代理是服务器，消息发布者可以同时是订阅者。

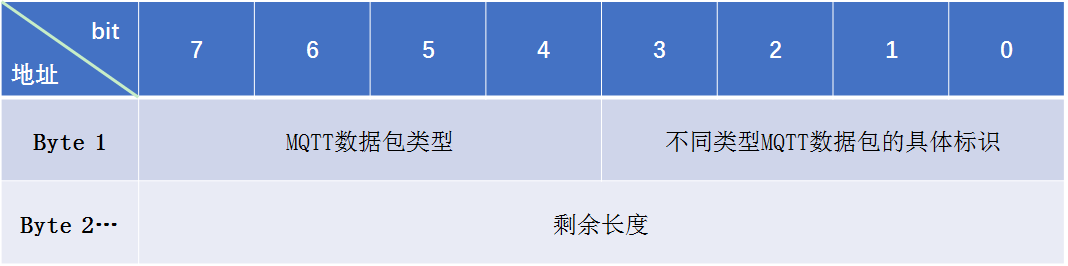
**2.MQTT协议详解**

**2.1 协议的整体结构：**



**2.1.1 固定header：**

大小：2bytes



Byte 1：

4~7位：MQTT数据包类型，总共能表示16中类型如下：



Byte1：

0~3： 标识位，主要是用在publish类型的消息体中。





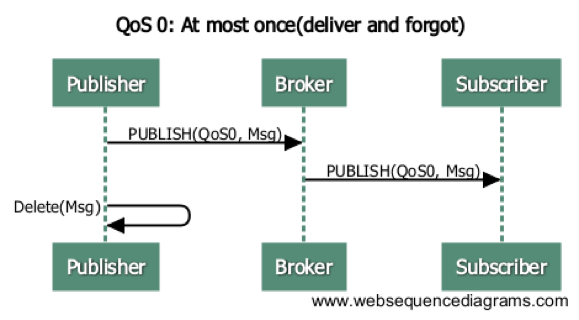
QoS：发布消息的服务质量，即：保证消息传递的次数

00：最多一次，即：<=1

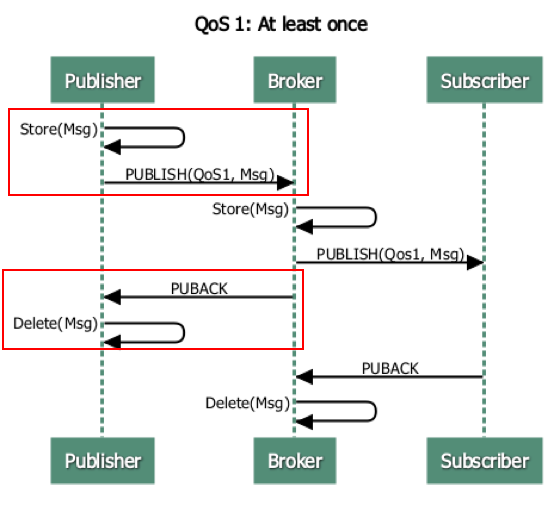
01：至少一次，即：>=1

10：一次，即：=1

在Qos0情况下，Broker可能没有接受到消息,在客户端只有一个publish类型的消息。



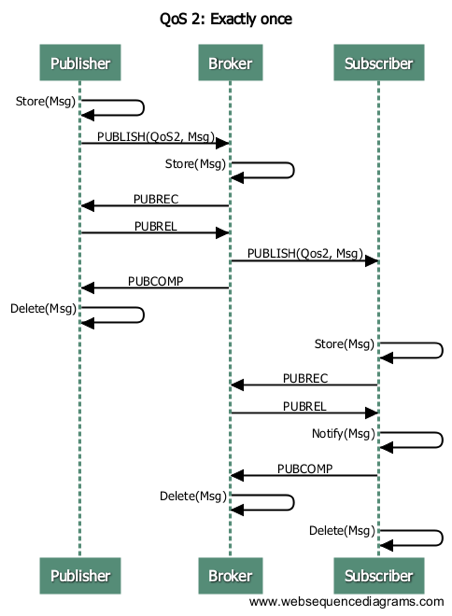
在Qos1情况下，发送者最少发送一次消息，确保消息到达Broker，Broker需要返回确认消息PUBACK，Broker可能接受到重复消息。



publisher store msg -> publish ->broker （传递message）

broker -> puback -> publisher delete msg （确认传递成功）

Qos2使用两阶段确认来保证消息的不丢失和不重复。在Qos2情况下，Broker肯定会收到消息，且只收到一次。



publisher store msg -> publish ->broker -> broker store msgID（传递message）

broker -> pubreceived （确认传递成功）

publisher -> pubrelease -> broker delete msgID （告诉broker删除msgID）

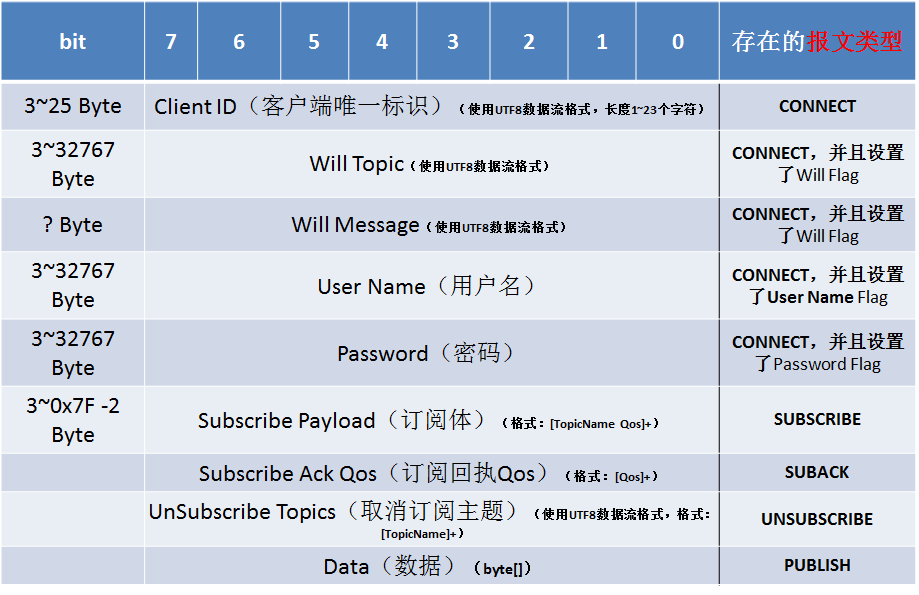
broker -> pubcomplete -> publisher delete msg （告诉publisher删除msg）

Byte2...ByteN：剩余长度，可变头部和载荷的字节数之和。

**2.1.2 MQTT可变头**

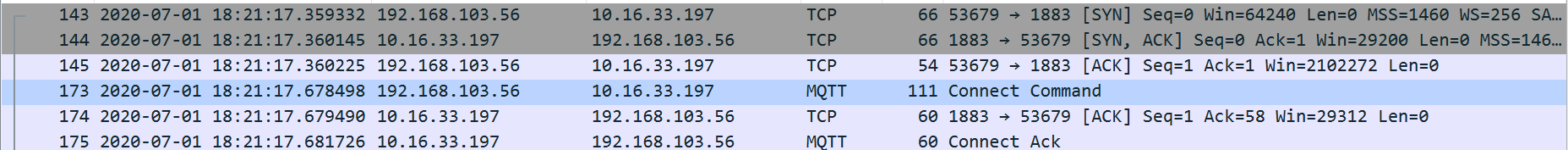


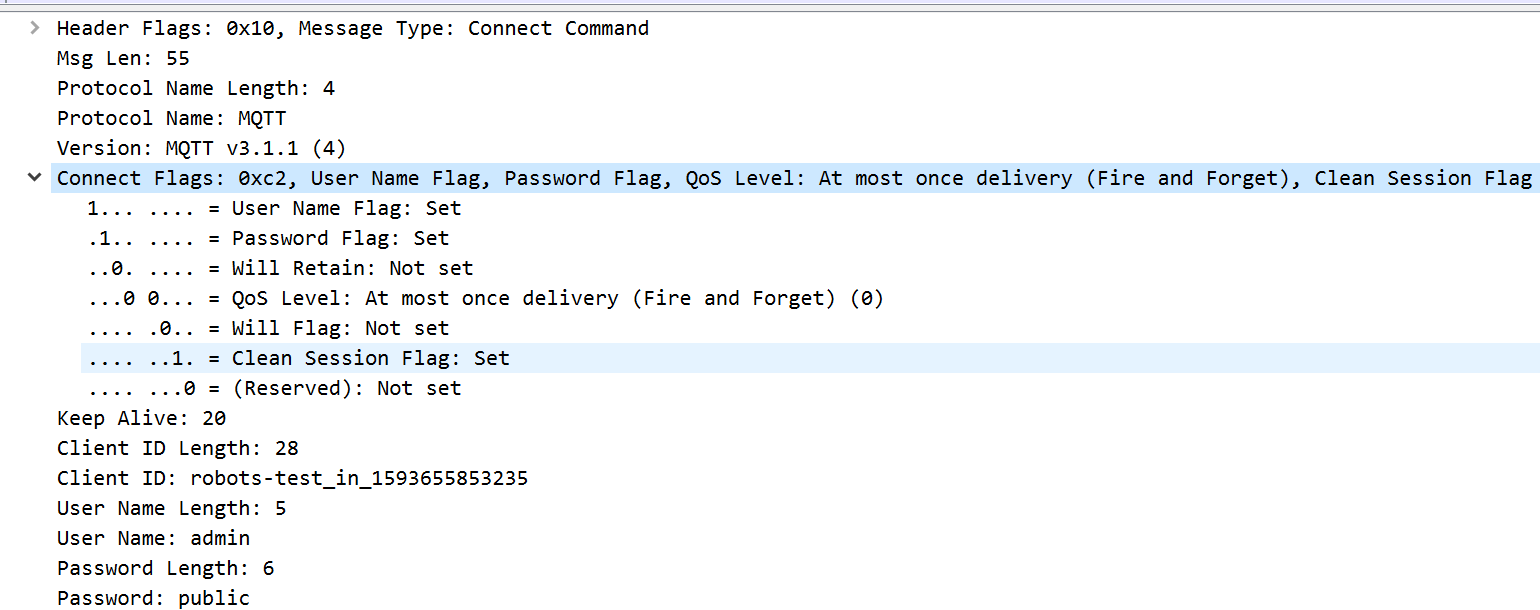
**2.1.3 Payload消息体**



**2.2协议的方法：**

**2.2.1 Connect:**





cleanSession：

会话是指订阅和发布的时候和代理服务器建立的连接。

此标志指定连接是否是持久性的。持久会话会将所有订阅和可能丢失的消息（具体取决于 QoS） 都存储在代理中。

Qos的值大于等于1时对持久性的订阅发布的消息是保存在代理服务器的。

订阅客户端可能不在线，一些mqqt消息发布之后希望不在线的订阅端在上线之后仍然能收到这些消息。

**注意：CleanSession为false的时候如果该会话对应的订阅需要添加/删除主题可能会失效。**

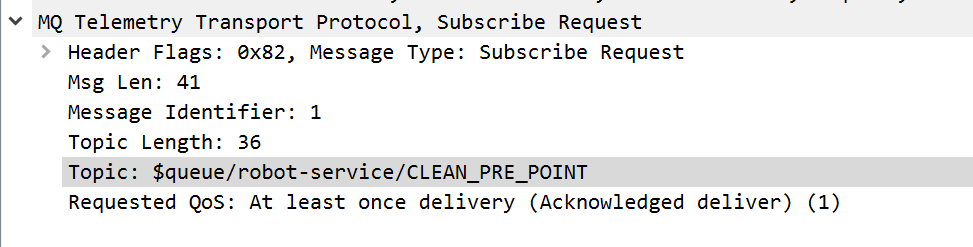
Username:连接代理服务器的用户名

Password:连接代理服务器的密码

keepAlive: 发送心跳消息的间隔

**2.2.2 Subscribe：**





主题(Topic)通过/分割层级，支持+, #通配符:

`+`: 表示通配一个层级，例如a/+，匹配a/x, a/y

`#`: 表示通配多个层级，例如a/#，匹配a/x, a/b/c/d

通过topic中的通配符可以支持一对多的发布消息。

Qos用于broker推送给订阅端时候指定消息的质量。

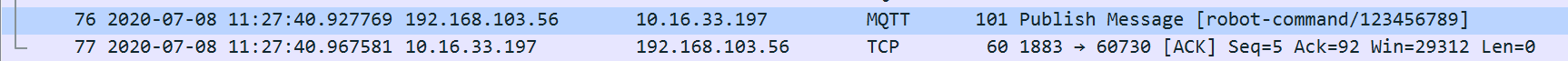
**目前主题的通配符只有在订阅的时候使用。Publish主题使用通配符会报以下的错误：**

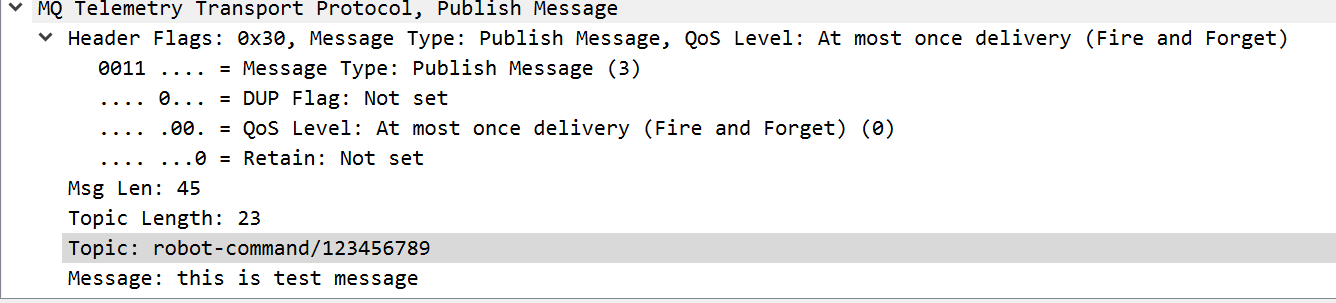


**2.2.3 Publish：**

根据Qos来发送不同格式的publish消息：

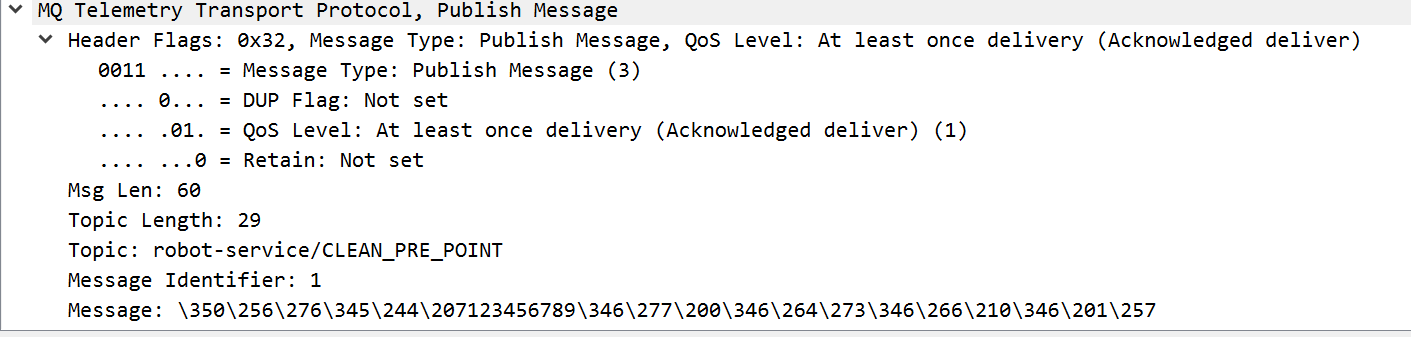
Qos0：



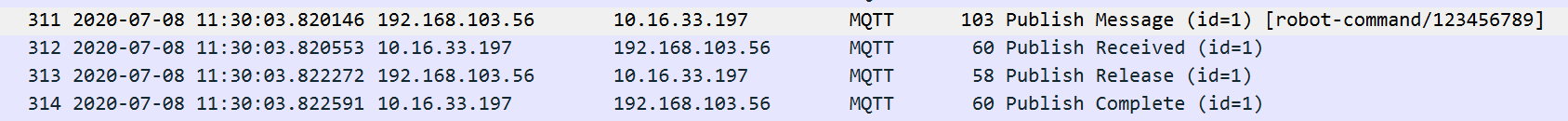


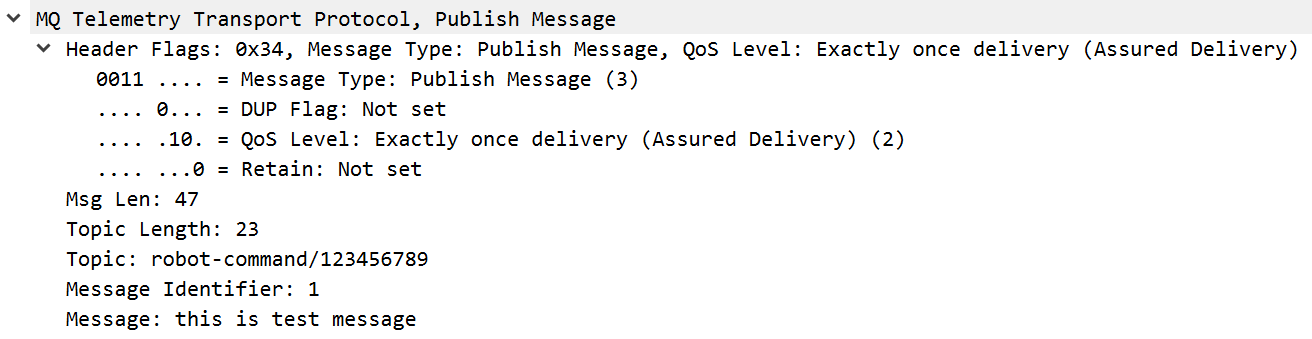
Qos1:





Qos2:





**3.client端上下线监控**

3.1 MQTT协议-遗嘱机制

MQTT 协议层面获取设备上下线状态，可利用 MQTT 协议遗嘱机制获取设备状态，设备连接时启用遗嘱，在连接到Broker时，设置 Will-Topic、Will-Payload等属性，当Client异常断开时（即设备没有发送Disconnect报文），Broker会发布遗嘱 Topic 与 Payload，但遗嘱机制明显弊端是，只有异常断开才会获得下线状态，同时需要设备端订阅连接的时候去设置遗嘱消息。

3.2 监听EMQ X 系统主题

EMQ X Broker 上下线状态主题：

上线主题：$SYS/brokers/<node>/clients/<clientid>/connected

下线主题：$SYS/brokers/<node>/clients/<clientid>/disconnected

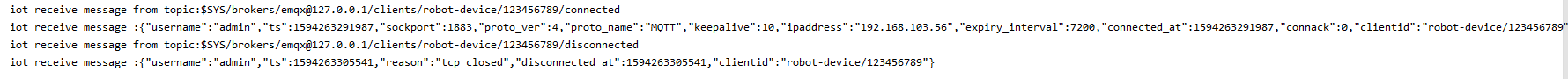
需要在broker代理做如下的操作打开client订阅client上下线的系统主题：

sudo vi /etc/emqx/acl.conf

{allow, all, subscribe, ["$SYS/brokers/+/clients/#"]}.

sudo systemctl restart emqx.service

如下图所示就是一次client : robot-device/123456789上线下线的消息：



**4.MQTT使用**



