MQTT初始篇笔记整理

MQTT简介

- MQTT(Message Queuing Telemetry Transport,消息队列遥测传输),基于TCP/IP 协议栈而构建,虽然叫消息队列遥测传输，但是她与消息队列毫无关系,她是一个IBM开发的客户端服务端架构的 发布/订阅模式的消息传输协议;她的设计思想是轻巧、开放、简单、规范、易于实现，因此MQTT比较适合在物联网环境(IoT,Internet of Things)以及机器与机器的通信(M2M,Machine to Machine)等受限的环境下。

MQTT的优势

- MQTT是一种轻量级、灵活的网络协议，轻量级保证其可在严重受限的设备硬件和高延迟/带宽有限的网络上实现；她的灵活性使得IoT设备和服务的多样化应用场景提供支持成为可能
  - 1、使用发布/订阅消息模式，提供一对多的消息发布，解除应用程序耦合；
  - 2、对负载内容屏蔽的消息传输；
  - 3、使用 TCP/IP 提供网络连接；
  - 4、有三种消息发布服务质量：
    - “至多一次” 服务质量级别0 QoS0，消息将最多传递一次，消息发布完全依赖底层 TCP/IP 网络。会发生消息丢失或重复。这一级别可用于如下情况，环境传感器数据，丢失一次读记录无所谓，因为不久后还会有第二次发送。她是最快的传输方式，有时成为"触发并忘记"。消息最多传递一次
    - “至少一次” 服务质量级别1 QoS1，消息会始终至少传递一次，确保消息到达，但消息重复可能会发生。
    - “只有一次” 服务质量级别2 QoS2，确保消息到达一次。这一级别可用于如下情况，在计费系统中，消息重复或丢失会导致不正确的结果。她是最安全也是最慢的传输方式。
  - 5、小型传输，开销很小（固定长度的头部是 2 字节），协议交换最小化，以降低网络流量；
  - 6、使用 Last Will 和 Testament 特性通知有关各方客户端异常中断的机制；

MQTT与其他网络协议的对比

- 与HTTP协议对比
  - 1. HTTP 是一种同步协议。客户端需要等待服务器响应。Web 浏览器具有这样的要求，但它的代价是牺牲了可伸缩性。在 IoT 领域，大量设备以及很可能不可靠或高延迟的网络使得同步通信成为问题。异步消息协议更适合 IoT 应用程序。传感器发送读数，让网络确定将其传送到目标设备和服务的最佳路线和时间。
  - 2. HTTP 是单向的。客户端必须发起连接。在 IoT 应用程序中，设备或传感器通常是客户端，这意味着它们无法被动地接收来自网络的命令。
  - 3. HTTP 是一种 1-1 协议。客户端发出请求，服务器进行响应。将消息传送到网络上的所有设备上，不但很困难，而且成本很高，而这是 IoT 应用程序中的一种常见使用情况。
  - 4. HTTP 是一种有许多标头和规则的重量级协议。它不适合受限的网络。

- 与AMQP对比
  - 企业中间件系统中使用的最流行的消息协议被称为 AMQP（高级消息排队协议）。但是，在高性能环境中，计算能力和网络延迟通常不是问题。AMQP 致力于在企业应用程序中实现可靠性和互操作性。它拥有庞大的特性集，但不适合资源受限的 IoT 应用程序。

- 与XMPP对比
  - XMPP（Extensible Messaging and Presence Protocol，可扩展消息和状态协议）是一种对等即时消息 (IM) 协议。它高度依赖于支持 IM 用例的特性，比如存在状态和介质连接。与 MQTT 相比，它在设备和网络上需要的资源都要多得多。

发布订阅模型

- MQTT 协议在网络中定义了两种实体类型：一个消息代理和一些客户端。代理是一个服务器，它从客户端接收所有消息，然后将这些消息路由到相关的目标客户端。客户端是能够与代理交互来发送和接收消息的任何事物。客户端可以是现场的 IoT 传感器，或者是数据中心内处理 IoT 数据的应用程序。
  - 1.客户端连接到代理。它可以订阅代理中的任何消息 “主题”。此连接可以是简单的 TCP/IP 连接，也可以是用于发送敏感消息的加密 TLS 连接。
  - 2.客户端通过将消息和主题发送给代理，发布某个主题范围内的消息。
  - 3.代理然后将消息转发给所有订阅该主题的客户端。

- 因为 MQTT 消息是按主题进行组织的，所以应用程序开发人员能灵活地指定某些客户端只能与某些消息交互。就是订阅相应的主题后才会收到相应的主题下的消息。

MQTT 控制报文格式

- 2.1 MQTT控制报文的结构 Structure of an MQTT Control Packet

图例 2.1 –MQTT控制报文的结构

Fixed header	固定报头，所有控制报文都包含
Variable header	可变报头，部分控制报文包含
Payload	有效载荷，部分控制报文包含

- 2.2 固定报头 Fixed header

2.2 固定报头 Fixed header

每个MQTT控制报文都包含一个固定报头。图例 2.2 -固定报头的格式 描述了固定报头的格式。

图例 2.2 -固定报头的格式

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT控制报文的类型				用于指定控制报文类型的标志位			
byte 2...	剩余长度							

- 2.2.1 MQTT控制报文的类型 MQTT Control Packet type

2.2.1 MQTT控制报文的类型 MQTT Control Packet type

位置：第1个字节，二进制位7-4。

表示为4位无符号值，这些值的定义见 表格 2.1 -控制报文的类型

表格 2.1 -控制报文的类型

名字	值	报文流动方向	描述
Reserved	0	禁止	保留
CONNECT	1	客户端到服务端	客户端请求连接服务端
CONNACK	2	服务端到客户端	连接报文确认
PUBLISH	3	两个方向都允许	发布消息
PUBACK	4	两个方向都允许	QoS 1消息发布收到确认
PUBREC	5	两个方向都允许	发布收到（保证交付第一步）
PUBREL	6	两个方向都允许	发布释放（保证交付第二步）
PUBCOMP	7	两个方向都允许	QoS 2消息发布完成（保证交互第三步）
SUBSCRIBE	8	客户端到服务端	客户端订阅请求
SUBACK	9	服务端到客户端	订阅请求报文确认
UNSUBSCRIBE	10	客户端到服务端	客户端取消订阅请求
UNSUBACK	11	服务端到客户端	取消订阅报文确认
PINGREQ	12	客户端到服务端	心跳请求
PINGRESP	13	服务端到客户端	心跳响应
DISCONNECT	14	客户端到服务端	客户端断开连接
Reserved	15	禁止	保留

- 2.2.2 标志 Flags

2.2.2 标志 Flags

固定报头第1个字节的剩余的4位 [3-0]包含每个MQTT控制报文类型特定的标志，见 表格 2.2 -标志位。表格 2.2中任何标记为“保留”的标志位，都是保留给以后使用的，**必须**设置为表格中列出的值 [MQTT-2.2.2-1]。如果收到非法的标志，接收者**必须**关闭网络连接。有关错误处理的详细信息见 4.8节 [MQTT-2.2.2-2]。

表格 2.2 - 标志位 Flag Bits

控制报文	固定报头标志	Bit 3	Bit 2	Bit 1	Bit 0
CONNECT	Reserved	0	0	0	0
CONNACK	Reserved	0	0	0	0
PUBLISH	Used in MQTT 3.1.1	DUP[1]	QoS[2]	QoS[2]	RETAIN[3]
PUBACK	Reserved	0	0	0	0
PUBREC	Reserved	0	0	0	0
PUBREL	Reserved	0	0	1	0
PUBCOMP	Reserved	0	0	0	0
SUBSCRIBE	Reserved	0	0	1	0
SUBACK	Reserved	0	0	0	0
UNSUBSCRIBE	Reserved	0	0	1	0
UNSUBACK	Reserved	0	0	0	0
PINGREQ	Reserved	0	0	0	0
PINGRESP	Reserved	0	0	0	0
DISCONNECT	Reserved	0	0	0	0

- DUP[1] =控制报文的重复分发标志
- QoS[2] = PUBLISH报文的服务质量等级
- RETAIN[3] = PUBLISH报文的保留标志

PUBLISH控制报文中的DUP, QoS和RETAIN标志的描述见 3.3.1节。

- 控制报文字类型是从值区分为0-15依次对应 Reserved Connect ConnAck Publish PubRec PubComp Subscribe SubAck Unsubscribe PingReq PingResp DisConnect Reserved 相应的Flags 有对应的数据

学习网址

- MQTT百科

- MQTT使用示例代码及第三方相关内容
- MQTT工具地址
- 物联网MQTT文档
- 物联网MQTT社区Wiki
- Mosquito 文档
- MQTT Essentials
- MQTT协议中文版
- MQTT入门篇
- kafka和mqtt的区别是什么？

如有不当之处 敬请指出 如需转载请注明出处 谢谢