阿里云
微消息队列MQTT

MQTT 与 MQ 的联系

文档版本：20190913

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。
2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。
3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。
4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的”现状“、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。
5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含”阿里云”、Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。
6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	 **禁止:** 重置操作将丢失用户配置数据。
	该类警示信息可能导致系统重大变更甚至故障，或者导致人身伤害等结果。	 **警告:** 重启操作将导致业务中断，恢复业务所需时间约10分钟。
	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	 **说明:** 您也可以通过按Ctrl + A选中全部文件。
>	多级菜单递进。	设置 > 网络 > 设置网络类型
粗体	表示按键、菜单、页面名称等UI元素。	单击 确定。
`courier` 字体	命令。	执行 `cd /d C:/windows` 命令，进入Windows系统文件夹。
*## *	表示参数、变量。	`bae log list --instanceid` *`Instance_ID`*
[]或者[a|b]	表示可选项，至多选择一个。	`ipconfig` *`[-all|-t]`*
{}或者{a|b}	表示必选项，至多选择一个。	`swich` *`{stand | slave}`*

目录

1 MQTT 与 MQ 的应用场景对比

本文主要在#unique_4的基础上介绍微消息队列 MQTT 和传统消息中间件的关联和区别，并针对实际应用场景下的产品选型给出建议。

背景信息

传统的消息中间件，例如消息队列 MQ、消息队列 Kafka 等都是面向微服务大数据等领域，负责消息的存储和转发，消息的生产者和消费者都是服务端应用。

这种设计很适合服务端技术栈固定、语言平台固定的场景。而移动互联网和 IoT 领域则有所不同，这类场景更侧重于多语言多平台的海量设备接入，消息的生产和消费过程的业务属性很突出，传统的消息中间件并不适合这些领域。

秉承单一职责的原则，微消息队列 MQTT 在设计上是一个面向移动互联网和 IoT 领域的无状态网关，只关心海量移动端设备的接入、管理和消息传输，消息数据的存储则都会路由给后端存储产品，例如传统的消息中间件消息队列 MQ、消息队列 Kafka 等产品。

在这种职责划分下，终端设备将消息发送到微消息队列 MQTT 后，消息会根据微消息队列 MQTT 绑定的存储产品被路由到指定产品，云端应用依然可以维持传统的微服务开发方案，通过对接云端存储产品即可和终端设备进行互动，两者之间通过微消息队列 MQTT 实现了数据互通能力。

适用场景对比

在一个业务场景中，可能包含多种不同类型的应用组件，每个组件承担不同的角色。因此，在方案选型时如需要使用到消息产品，需要先了解微消息队列 MQTT 和传统消息中间件的关联和区别，合理搭配使用，比如组件 A 的消息收发使用微消息队列 MQTT，组件 B 的消息收发使用消息队列 MQ。

下文将根据场景举例描述微消息队列 MQTT 和传统的消息中间件的区别，为方便描述，传统的消息中间件以消息队列 MQ 为例，其他产品例如消息队列 Kafka 和消息队列 AMQP（RabbitMQ）同理。

表 1-1: 适用场景对比

产品名	适用场景
微消息队列 MQTT	面向移动端场景，移动端场景一般都具备海量设备，单设备数据较少的特点。因此，微消息队列 MQTT 适用于拥有大量在线客户端（很多企业设备端过万，甚至上百万），但每个客户端消息较少的场景。

产品名	适用场景
消息队列 MQ	面向服务端的消息引擎，主要用于服务组件之间的解耦、异步通知、削峰填谷等，服务器规模较小（极少企业服务器规模过万），但需要大量的消息处理，吞吐量要求高。因此，消息队列 MQ 适用于服务端进行大批量的数据处理和分析的场景。

组合使用场景示例

- 场景示例一

  在物联网 IoT 场景中，成千上万（甚至数百万）规模的设备传感器可使用微消息队列 MQTT 上传数据，需做数据分析的服务端（即部署在服务器上的应用）则可以通过消息队列 MQ 完成数据的分析与处理。

- 场景示例二

  在车联网场景中，上百万辆车需要上传车辆信息数据到云端（服务端），云端同时也会下发指令到任意车辆或广播到所有的车辆。车辆可以通过 MQTT SDK 连接到微消息队列 MQTT 实现数据上报以及指令接收，监管系统（数据分析系统）可以通过消息队列 MQ 的 SDK 进行消息订阅以及指令下发。如下图所示：

基于以上区别，推荐您在移动端设备上使用微消息队列 MQTT，而在服务端应用中则使用消息队列 MQ（或者其他消息产品）。

功能对比

微消息队列 MQTT 和消息队列 MQ 的具体功能特性的对比如下：

表 1-2: 功能对比

功能特性	微消息队列 MQTT	消息队列 MQ
客户端连接数	客户端规模庞大，百万甚至千万级	一般服务器规模较小，极少数万级
单客户端消息量	单个客户端需要处理的消息少，一般定时收发消息	单个客户端处理消息量大，注重吞吐量
部署场景	移动设备、App 软件、H5 页面等	服务端应用
消费模式	支持广播模式	支持集群消费和广播消费
顺序支持	只支持上行顺序，不支持下行顺序（后续开放）	支持上行和下行顺序

功能特性	微消息队列 MQTT	消息队列 MQ
多语言/系统支持（TCP 协议）	支持 Java、C、C++、.NET、Andriod、iOS、Python、JS、Go 等多种语言和系统	支持 Java、C++、.NET
访问凭证	支持 RAM 的永久访问模式和 MQTT Token 的临时访问模式，详情请参见#unique_5。	支持 RAM 永久访问模式和 STS 临时授权访问

选型指导

基本原则总结如下：

- 对于部署在服务器上的应用，推荐使用消息队列 MQ 接入；
- 对于部署在移动终端、App 或浏览器页面等平台上的应用，推荐使用微消息队列 MQTT 接入。

针对常见的应用场景，建议的微消息队列 MQTT 和消息队列 MQ 选型如下：

表 1-3: 选型推荐

场景	部署端	微消息队列 MQTT	消息队列 MQ
设备端上报状态数据	移动终端	√	×
接收并处理分析设备的上报数据	移动终端	×	√
对多个设备下发控制指令	服务器	×	√
直播、弹幕、聊天 App 收发消息	应用	√	×
服务端接收并分析聊天消息	服务器	×	√

说明：

√ 表示建议使用该消息队列产品；× 表示不建议使用该消息队列产品。

2 MQTT 与 MQ 的消息结构映射

本文针对使用消息队列 MQ SDK 与微消息队列 MQTT 交互的场景，提供交互中所涉及的消息结构和属性字段的映射关系，方便您更好的理解和组合使用这两个产品。

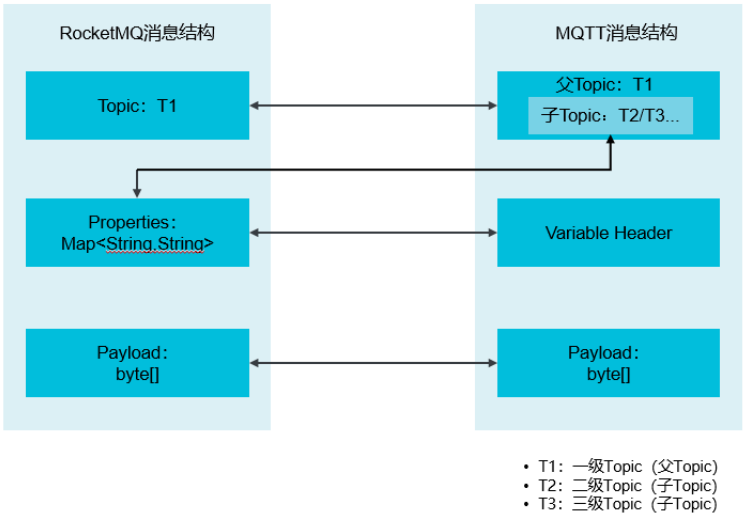
微消息队列 MQTT 是一款面向移动端的网关产品，在实际消息收发中，可单独使用，也可搭配其他存储产品，例如消息队列 MQ 作为消息存储使用。

如果单独使用微消息队列 MQTT，则无需关注本文提供的映射关系，一切遵循标准 MQTT 协议规范即可。

关于微消息队列 MQTT 的详细介绍，请参见#unique_7和#unique_8。

消息结构映射

微消息队列 MQTT 和消息队列 MQ 都是基于发布/订阅（Pub/Sub）模型的消息系统，两者概念上存在很多相似之处，下图列举了关键概念的区别和映射关系。



如上图所示，在微消息队列 MQTT 中 Topic 是多级结构，而消息队列 MQ 的 Topic 仅有一级，因此，微消息队列 MQTT 中的一级 Topic 映射到消息队列 MQ 的 Topic，而二级和三级 Topic 则映射到消息队列 MQ 的消息属性（Properties）中。

消息队列 MQ 协议中的消息（Message）可以拥有自定义属性（Properties），而 MQTT 协议目前的版本不支持属性，但为了方便溯源 MQTT 协议中的 Header 信息和设备信息，微消息队列 MQTT 的部分信息将被映射到消息队列 MQ 的消息属性中，方便使用消息队列 MQ 的 SDK 接入的用户获取。

> **说明:**
> 关于具体如何在消息队列 MQ 中设置属性字段来映射微消息队列 MQTT 信息，请参见下文属性字段映射的表格。

消息队列 MQ 和微消息队列 MQTT 的消息负载（Payload）均是您的业务消息的数据序列化结果，消息队列 MQ 和微消息队列 MQTT 不会对业务消息再做进一步的编解码处理。

属性字段映射

目前，微消息队列 MQTT 和消息队列 MQ 支持的属性字段映射关系如下表所示。使用消息队列 MQ 和微消息队列 MQTT 的 SDK 的应用交互时，可以通过读写这些属性字段来设置或获取信息。

关于 QoS、cleanSession、Topic 以及 Client ID 的详细解释，请参见#unique_8。

属性 Key	属性可选值	说明
qoslevel	0，1，2	消息队列 MQ 发给微消息队列 MQTT 消息时可以设置，如果不设置，默认为“1”； 微消息队列 MQTT 发给消息队列 MQ 的消息可以直接读取。
cleansessionflag	true，false	消息队列 MQ 发给微消息队列 MQTT 客户端 P2P 消息时设置，如不设置，默认为“true”； 其他消息不可以设置，微消息队列 MQTT 发给消息队列 MQ 的消息可以直接读取。
mqttSecondTopic	具体的子级 Topic 字符串	消息队列 MQ 发给微消息队列 MQTT 客户端消息时如果需要子级 Topic 来做过滤，则设置，如不设置，默认为空； 微消息队列 MQTT 发给消息队列 MQ 的消息可以直接读取。

属性 Key	属性可选值	说明
mqttRealTopic	业务上希望客户端收到消息时显示的子级字符串	消息队列 MQ 发给微消息队列 MQTT 客户端消息时如果希望客户端收到消息后显示成指定的子级 Topic 名称，则可以设置；一般用于 P2P 消息，若不设置，P2P 消息默认使用自己固定的 Topic； 微消息队列 MQTT 发给消息队列 MQ 的消息时无该属性。
clientId	具体的 clientId 字符串	不可设置，微消息队列 MQTT 发给消息队列 MQ 消息时，用于追踪该消息的发送源的 clientId。