分布式消息 MQTT 产品
用户使用指南

天翼云科技有限公司

一、 产品简介

1.1 产品定义

分布式消息服务 MQTT 是面向移动互联网以及物联网领域的轻量级消息中间件，可以在有限的资源条件下，为连接远程设备提供实时可靠的消息服务并支持数据高效分类存储、再处理，实现终端设备与云端应用互通。

1.2 产品优势

分布式消息服务 MQTT 的产品优势主要包括以下几个方面：

- 兼容多协议：协议丰富、兼容支持原生多种标准协议，如 MQTT、MQTT-SN、CoAP、LwM2M，兼容任何支持 MQTT 的协议，SDK 覆盖绝大多数移动端开发平台及开发语言。
- 安全可靠：数据安全，支持 SSL 加密通信，提供身份认证，数据传输更安全可靠。
- 低成本高性能：稳定承载大规模终端设备连接，资源占用少 消息路由快速低延时，单集群支持百万规模的路由。
- 消息互通：支持队列与 Kafka 消息互通，实现终端设备与云端应用互通。

1.3 功能特性

分布式消息服务 MQTT 的功能特性主要体现在以下几个方面：

- 消息模型：支持发布/订阅消息模型，提供一对多的消息分发方式。
- Qos 支持：提供三种级别 QoS 支持：
  - Qos0：At-Most-Once 允许消息少量丢失，最多传输一次
  - Qos1：At-Least-Once 确保消息一定到达，可少量重复
  - Qos2：Exactly-Once 有且仅有一次
- 离线消息：根据业务场景需求对符合条件的终端端断线重连提供离线消息的获取
- 双向互通：终端消息按第一级主题分类存储至消息队列 kafka，云端应用消费处理并进行指令下发。
- 连接查询：设备连接信息、订阅关系、分组在线设备数的实时查询；
- 轨迹查询：提供设备维度的上下线轨迹的查询
- 运维完善：在线连接、主题、订购关系、会话以及消息收发统计历史查询，并提供指标的阈值设置，超过阈值进行告警提示

1.4 应用场景

广泛应用于移动互联网以及物联网领域，车联网、工业设备、智能家居、即时聊天等多种应用场景。

场景一

智能抄表。电表、水表、燃气表已是每个家庭必备，集成消息队列 MQTT 系统后，再无需进行人工抄表，所有住户的用电、用水、燃气、供暖数据都可由数据后台一键抄取，省时省力避免上门催收的尴尬，助力能源管理部门提供更优质服务

场景二

智能家居，通过门监测住户出入行为，同时结合实时气温与时间，控制空调、灯具、电视机、音箱等的启动和关闭

1.5 产品规格

公测期免费，计费类型是包周期。

产品规格	连接数	消息 TPS	订阅关系数
MQTT -高级版	30w	5w/s	50w
MQTT-基础版	10w	2w/s	10w

1.6 名词解释

实例	创建购买消息队列 MQTT 服务的实体单元，包含 MQTT Broker 和 kafka 集群。
MQTT Broker	MQTT 协议交互的服务端节点，承担 MQTT 客户端承连接接入、管理、数据转发、消息数据存储至 kafka 消息队列。
MQTT 客户端	和 MQTT Broker 交互的终端节点
父 Topic	MQTT 协议，topic 存在层级性，定义第一层级 topic 为父 topic，
子 Topic	除第一层父 topic 外，MQTT 协议 topic 后续层级统称子 topic
ClientID	每个 MQTT 客户端唯一标识
GroupID	若终端设备需要分类，可创建 GroupId；对应 ClientID 则以 GroupID@@开头，可按 GroupId 提供在线数查询。非强制要求，用

	户自行决定是否使用。
消息队列 kafka	MQTT Broker 对接收的终端消息数据按父 topic 分类存储至 kafka 集群队列；云端应用系统通过 kafka 集中处理分析和指令下发
云端应用服务	用户后端消费数据系统

1.7 产品架构

MQTT 客户端

MQTT Broker 交互的 pub 和 sub 设备。

MQTT Broker

承担移动端连接接入、连接管理、数据转发至订阅的设备和 kafka 消息队列。

消息队列 kafka

存储 MQTT 消息，接收后端应用服务系统指令转发至 MQTT broker。

应用服务

分析、处理 kafka 的数据和向其下发指令。

消息分类存储

主题是多级形式的，消息按第一级即父 topic 分类存储至 kafka 队列。

二、计费说明（公测阶段免费）

三、快速入门

3.1、创建实例

1、 登入天翼云门户

2、 选择相应的资源池

3、 进入服务列表->云产品->消息中间件，选择分布式消息服务 MQTT，进入控制台实例列表页点击“订购实例”进入开通页进行实例创建。

3.2、连接实例

1、创建用户名和密码，需要数据存储创建父主题，需要分组管理创建 GroupId；

2、对用户进行主题授权

3、公网接入若未绑定弹性公网 ip 需先进行购买弹性 ip 并进行绑定

4、终端设备使用 MQTT 客户端通过终端连接地址接入；

5、云端应用使用 KAFKA 客户端通过服务端连接地址接入。

6、支持 SSL

3.3、使用须知

3.3.1、公网接入

公网访问需购买弹性 ip，提供以下两种方式实现：

1、 开通时，开通页面选择已购买未使用的弹性 ip，自动绑定

2、 先开通 MQTT 实例，后通过控制台提供的公网绑定入口进行绑定操作

弹性 ip 带宽大小计算规则可参照：带宽= 报文大小*TPS*120%,建议按 120%购买，应对突发流程。

如基础版规格，报文大小 1KB，TPS 2W/s,则带宽=20000*1000*8=160Mb/s,建议 200Mb/s

3.3.2、消息数据存储

终端消息数据按父 topic 存储至 kafka 队列，需先在控制台创建父 topic；对未创建父 topic 的消息可正常收发，但不会存储至 kafka 队列。

Kafka 存储内容格式：{

"clientId": 设备 clientId,

"topic": 主题,

"payload": 消息内容,

"ts": 发送的时间戳

}

3.3.3、会话机制

终端 clean session=true,断线后会话信息清除，再次上线后之前所有的订阅关系以及离线消息丢失。 clean session=false 断开连接的情况下，MQTT Broker 也会为断连客户端保存一个会话，默认 2 小时，超期未重连订购关系清除；对于 clean session=false 的客户端断线重连后可接收 Qos>0 的离线消息。对于客户端因网络等各种原因断线，需要加上重连和订购关系重新订购机制。

3.3.4、离线消息

对于 clean session=false 的客户端在未超出会话失效期断线重连后可接收 Qos>0 的离线消息

3.3.5、系统主题

系统主题	说明
mq2mqtt	用于云端服务向终端发送消息。发往该主题消息会转发至 MQTT Broker 实现云端与移动端互通
mqtt-device-connect	设备上线主题，内容 {"clientid":客户端 ID,"ts":上线时间戳 }
mqtt-device-disconnect	设备下线主题，内容 {"clientid":客户端 ID,"ts":下线时间戳 }

3.3.6、名称解释

名词	说明
终端连接地址	即 MQTT Broker 端接入地址，设备端使用；
服务端连接地址	即 kakfa 集群连接地址，云端应用服务使用
订阅关系	终端设备每订阅一个主题即一个订阅关系

3.3.7、使用限制

限制项	限制值	说明
最大报文	128KB	允许的 MQTT 报文最大长度
Topic 和 Client ID 可用字符	仅限数字 0-9，字母 a-z 或 A-Z，和符号 “_”	Client ID 和 MQTT 的 Topic 不允许使用其他非常规字符，否则可能导致无法连接和收发消息
Client ID 长度	128 个字符	Client ID 长度不得超过该限制，否则会导致连接被断开。
用户消息收发 TPS		根据实例规格不同进行限制
用户连接数		根据实例规格不同进行限制
最大 QoS 等级	2	请合理使用 Qos>0 等级，以免避免内存超额。
单个设备订阅 Topic 数量	无	
每个持久化会话离线消息存储数量	10 条	服务端会限制每个持久化会话的离线消息数量。超过该限制后，服务端会从最早的消息开始清理。因此，请合理使用持久化订阅模式，以免避免内存超额。
顺序消息		指每个客户端发送消息的顺序性

限制项	限制值	说明
Kafka 分区数	3	

四、用户指南

4.1、SDK 支持

分布式消息服务 MQTT 支持标准的 MQTT 协议，理论上适配所有的 MQTT 客户端 SDK。

推荐对应的第三方 SDK 如下表：

语言/平台	推荐的第三方 SDK
Java	Eclipse Paho SDK
iOS	MQTT-Client-Framework
Android	Eclipse Paho SDK
JavaScript	Eclipse Paho JavaScript
Python	Eclipse Paho Python SDK
C	Eclipse Paho C SDK
C#	Eclipse Paho C# SDK
Golang	Eclipse Paho Golang SDK

语言/平台	推荐的第三方 SDK
Node.js	MQTT-JS
PHP	Mosquitto-PHP

4.2、主题规则

主题形式：**父 topic/子级 topic1/子级 topic2…。（父 topic 需要先创建）**

使用 MQTT 消息队列发消息，会把消息以父 Topic 主题分类保存在 kafka 上，应用服务可通过 kafka 客户端以父 Topic 为主题消费消息。

云端应用服务统一发送到 kafka topic 为 **mq2mqtt** 的主题队列上，**移动端 topic**、会话属性 **Qos** 和 **cleansession** 保存在 **Record Header** 中，MQTT 设备通过订阅 **移动端 topic**，实现云端到移动端通讯。

Kakfa header 与 mqtt 属性映射如下表

Kafka Header	MQTT 属性
qoslevel	Qos
cleanSession	cleanSession
mqttTopic	主题

4.3、消息收发

4.3.1、MQTT 客户端收发

使用 MQTT SDK 接入终端连接地址进行消息生产消费。

4.3.2、MQTT 发送 kafka 接收

终端设备使用 MQTT SDK 接入终端连接地址进行消息发布，云端应用服务使用 kafka sdk 接入
服务端连接地址按父主题进行数据消费。

4.3.3、MQTT 发送顺序消息 kafka 接收顺序消息

创建父主题，类型分区顺序，父主题以 orderMsg2mq-开头。

终端设备使用 MQTT SDK 接入终端连接地址进行消息发布，云端应用服务使用 kafka sdk 接入

服务端连接地址按父主题进行分区顺序数据消费

4.3.4、Kafka 发送 MQTT 接收

云端应用服务使用 kafka sdk 接入服务端连接地址往系统主题 mq2mqtt 下发指令，终端设备使用 MQTT SDK 接入终端连接地址接收;

示例：

```
public static void main(String[] args) {
    Properties properties = new Properties();
    properties.put("bootstrap.servers", "xxx.xx.xxx:port");
    properties.put("key.serializer", "org.apache.kafka.common.serialization.StringSerializer");
    properties.put("value.serializer", "org.apache.kafka.common.serialization.StringSerializer");
    Producer<String, String> producer = null;
    try {
        producer = new KafkaProducer<String, String>(properties);
        for (int i = 0; i < 100000; i++) {
            String msg = "Message 1" ;
            Headers headers=new RecordHeaders();
            headers.add(new RecordHeader("qoslevel",String.valueOf(1).getBytes()));
            headers.add(new RecordHeader("cleanSession","false".getBytes()));
            headers.add(new RecordHeader("mqttTopic","/test3/xxx".getBytes()));
            producer.send(new ProducerRecord<String, String>( topic: "mq2mqtt",  partition: null, key: null,msg,headers));
        }
    } catch (Exception e) {
        e.printStackTrace();

    } finally {
        producer.close();
    }
}
```

五、常见问题

5.1.1、终端设备一直处于断线状态

检查终端客户端端使用是否进行了重连机制。

5.1.2、终端断线重连后一直未收到消息

对于 cleanSession=true 的客户端，断线重连后需要重新主题订阅；

对于 cleanSession=false 的客户端，断线在 2h 时间范围内重新成功无需进行主题订购，超过也需主题重新订阅。

5.1.3、客户端连接异常断开

1、 不同的客户端使用相同的 Client ID

2、 报文是否超过最大限制

3、 权限不匹配

5.1.4、MQTT Broker 连接数为 2

1、默认每个 Broker 有两个系统连接；

2、broker 进程异常完成重启前，客户端重连只其他节点

5.1.5、离线消息缺失

对于 clean session=false 的客户端在未超出会话失效期断线重连后可接收 Qos>0 的离线消息，离线消息限制不超过 10 条。

5.1.6、Kafka 消息保留时长

72 小时

5.1.7、消息未存储至 kafka

1、 创建父主题

2、 若已创建父主题，可删除再进行创建

5.1.8、应用通过 kafka 发送指令设备未收到

1、 发往 kafka 消息主题是 mq2mqtt

2、 附带头部属性

Kafka Header	MQTT 属性
qoslevel	Qos
cleanSession	cleanSession
mqttTopic	主题

5.1.9、终端发送消息成功订阅者未收到

检查是否终端使用的用户发送消息的主题授权