

腾讯云消息队列 CMQ

消息主题 (Topic) 模型

产品文档



腾讯云

【版权声明】

©2013-2017 腾讯云版权所有

本文档著作权归腾讯云单独所有，未经腾讯云事先书面许可，任何主体不得以任何形式复制、修改、抄袭、传播全部或部分本文档内容。

【商标声明】

及其它腾讯云服务相关的商标均为腾讯云计算（北京）有限责任公司及其关联公司所有。本文档涉及的第三方主体的商标，依法由权利人所有。

【服务声明】

本文档意在向客户介绍腾讯云全部或部分产品、服务的当时的整体概况，部分产品、服务的内容可能有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或模式的承诺或保证。

文档目录

文档声明.....	2
消息主题 (Topic) 模型	4
标签匹配功能说明	4
创建主题	7
修改主题属性.....	8
订阅主题	9
路由键匹配功能说明.....	10
生产者发布消息至主题.....	12
主题向订阅者投递消息.....	13

消息主题 (Topic) 模型

标签匹配功能说明

CMQ 的 TOPIC 模式，近期已增加订阅筛选标签功能，类似于 rabbitMQ 的 direct_routing 模式，topic_pattern 模式在下一个迭代提供。由于筛选标签的使用策略较复杂，具体说明如下：

场景1：Topic : test1 ，在12点01分，调用了订阅发布的API，定义该次操作为『发布test1』，发布后topic投递到订阅者，给A/B/C三个订阅者投递了200条消息，假设结果为：A有100条消息接收失败，B有30条消息接收失败，C的200条消息全部接收成功

场景分析：

- 消息堆积：由于A/B/C三个订阅者中，假定消息投递失败的总集合为110条（A有100条消息失败，B有30条失败，C全部成功，三者之间的失败消息有交错的部分），只要存在任意1个订阅者与某条消息存在关联关系，该消息都不会被立即退订
- 阻塞策略：以A为案例，topic投递给A200条消息，第101条失败时候，则后面的99条的投递会被阻塞。状态为有100条消息接收失败
- 重试策略（退避重试）：Topic-test1会每隔N秒给A重新投递消息，失败的100条里，按顺序，从第一条重新尝试，若失败三次，直接就丢弃，然后投递下一条消息，失败三次后再次丢弃，按顺序排列
- 重试策略（衰退指数重试）：以A为案例，topic会并发投递100条消息（无法保证先后顺序）当订阅者接收的第一条消息失败后，从第一条重新尝试，若失败，继续阻塞。后续新增的投递消息，会继续阻塞
- 消息生命周期不可延长：假定这110条堆积在topic-test1里的消息，中间无论被重试多少次，生命周期都为1天，从生产者推送到topic的时间点，作为起止时间点，到期后删除
- 重新推送：生产者不断往Topic中生产新的消息，客户在12点02分，又调用了订阅发布的API，加上之前失败堆积的110条消息，假设目前topic中共有110（失败堆积）+100（1分钟时间内新增的消息）=210条消息，再次投递。这时，A/B的重试策略为衰退指数重试，处于『失联』状态，则210条消息会继续堆积。对于C，只接收新增的100条消息。

核心：每一条消息的id，作为key，value是关联的订阅者、代表每个订阅者消费成功与否

场景2：Topic : test2 ，在12点01分，调用了订阅发布的API，定义该次操作为『发布test2』，给A/B/C三个订阅者投递了200条消息，假设结果为：A/B/C的200条消息全部接收成功

场景分析：

- 若topic-test2，有且仅有A/B/C三个订阅者，且都确定消费成功后，topic会立即删除200条消息

场景3：

假设我们有4个订阅者A/B/C/D，每个订阅者加上消息标签：A消息标签是apple，B消息标签是xiaomi，C没有设置任何标签，D的消息标签为imac+xiaomi。此时，有生产者publish到topic 100条消息，带的消息标签过滤为：apple，imac，iphone，macbook。此后topic立即投递到A/B/C/D。

场景分析：

- 对于订阅者A而言，由于apple能匹配消息过滤标签中的apple，则能正常收到100条消息
- 对于订阅者B而言，收不到任何消息
- 对于订阅者C，也收不到任何消息
- 对于订阅者D，由于其中一个标签imac能匹配，则能正常收到100条消息

场景4：

假设Topic有且仅有4个订阅者A/B/C/D，四个订阅者都没有设置消息tag。此时，有生产者publish到topic 100条消息，带的消息标签过滤为：apple，imac，iphone，macbook。此后topic立即投递到A/B/C/D。

场景分析：

- 订阅者A/B/C/D，没有tag，则投递时不用匹配消息，A/B/C/D都能收到这100条消息

场景5：假设Topic有且仅有1个订阅者A，A的订阅标签设置为xiaomi。此时，有生产者publish到topic 100条消息，带的消息标签过滤为：apple，imac，iphone，macbook。此后topic立即投递到A

场景分析：

- 对于订阅者A，订阅标签不匹配，则A无法收到这100条消息。此时这100条消息立即丢弃，不会堆积在CMQ里
- 生产者publish到topic时，消息过滤tag只能在publish前设置一次，跟消息id绑定，不可修改

总结：

1. 对于消息A，没有消息TAG，订阅者有tag，则该订阅者不匹配，收不到消息

2. 对于消息B，消息有TAG，订阅者没有tag，投递时消息不用匹配，订阅者都能收到消息
3. 对于消息C，消息有TAG，订阅者也有tag的时候，两者匹配的，才能收到消息。支持N:M匹配，如消息有10个TAG，订阅者有4个TAG，其中有1个TAG相互能匹配上，则订阅者能收到消息
4. 对于消息D，消息没有TAG，订阅者也没有TAG，投递后，所有订阅者都能收到消息

创建主题

创建主题 (Topic) 时，用户需要指定以下属性值：

- 1) 输入Topic名称，一旦创建后不能再修改Topic名称，只能包含字母，数字，短横线(-)和下划线(_)，名称长度限制在3-64字节之间，长于64字节将被自动截取
- 2) 设定Topic的地域属性
- 3) 设定消息最大长度，Topic的MaximumMessageSize属性，允许发送到该主题的消息体的最大长度，单位为byte，有效值范围为：1024(1KB)-65536(64KB)，默认值为65536。
- 4) 输入备注
- 5) 消息生命周期：消息在本队列中最长的存活时间，从发送到该队列开始经过此参数指定的时间后，不论消息是否被取出过都将被删除；单位为秒，默认值为86400s，不允许修改。
- 6) 消息堆积：默认开启。存在生产者的消息，还未触发推送到订阅者，或订阅者接收消息失败，暂时堆积到OPIC中
- 7) 当前消息堆积数：在topic查看详情处，可查看到当前消息堆积的总数，为近似的堆积数
- 8) 单个topic的Qps性能：单个Topic的生产、投递的QPS峰值为5000

修改主题属性

用户修改主题属性时，有以下限制：

- 1) Topic名称、资源id，两者都不可修改
- 2) 显示Topic的地域属性，不可修改
- 3) 设定消息最大长度：可修改，该项目为允许发送到该主题的消息体的最大长度，单位为byte，有效值范围为：1024(1KB)-65536(64KB)，默认值为65536。
- 4) 创建时间，最后修改时间，不可修改

订阅主题

可以通过指定以下属性订阅一个主题：

1) 输入Topic名称

2) 输入 Topic 资源id

3) 填写订阅名称，一经填写无法修改

4) 填写订阅终端协议，选项有: Queue消息服务、URL地址

5) 订阅地址，填写URL、Queue队列名称，目前只允许Topic发送给同一个账户下的Queue

6) 重试策略：订阅的NotifyStrategy属性，向接收端推送消息出现错误时的重试策略。该策略默认开启。

提供两个选项供选择：a.退避重试：重试3次，间隔时间10-20s之间的一个随机值，超过3次后，该条消息对于该订阅者丢弃，不会再重试； b.

衰退指数重试：重试176次，总计重试时间为1天，间隔时间依次为： 2^0 ， 2^1 ，...，512，512，...

512秒。默认勾选

衰退指数重试

策略。两者必须勾选1个

7) 重试验证：若https，返回码为200，则认为成功

8) 添加订阅者标签：添加订阅者时，可增加FilterTag，增加filtertag后，该订阅者仅能收到带该filtertag的消息。单个tag不超过16个字符的字符串，单个订阅者可最多添加10个tag。只要其中某个tag，能匹配topic的过滤标签，都能收到该次topic投递的消息，若消息不带任何标签，则该订阅者无法收到该类型消息。

9) 单个Topic订阅者上限：单个TOPIC下最多允许关联100个订阅者

10)

该订阅者关联的消息总数量：该数量为近似数值，说明某Topic还有多少消息，等待投递、重试投递到订阅者

路由键匹配功能说明

CMQ的路由键匹配功能类似于 rabbitMQ 的 exchange queue，可以用于消息过滤，根据不同的条件使订阅者获取不同的消息。创建topic时，可开启【路由匹配键】。

使用说明：

- Binding key、Routing key是组合使用的，提供类似于RabbitMQ的消息过滤能力。发消息时配的 Routing key 是客户端发消息带的。创建订阅关系时配的 Binding key 是 topic 和 订阅者的绑定关系。

使用限制：

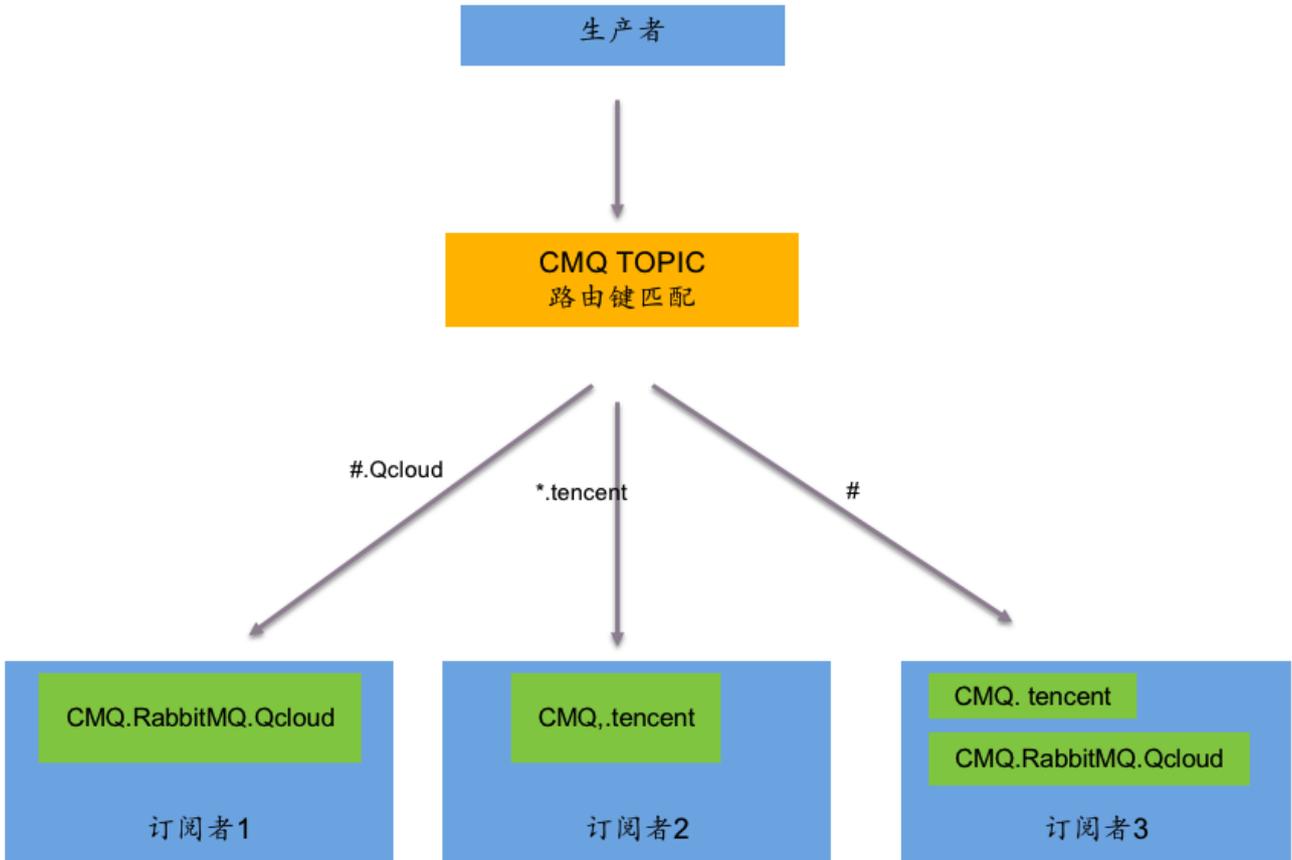
- 1、Binding key的数量不超过5个。单个binding key的长度<=64字节，用于表示发送消息的路由路径，最多含有15个“.”，即最多16个词组
 - 2、Routing key的数量由1个字符串组成。单个Routing key的长度<=64字节，用于表示发送消息的路由路径，最多含有15个“.”，即最多16个词组

通配符说明：

- 1、* (星号)，可以替代一个单词（一串连续的字母串）
- 2、# (井号)：可以匹配一个或多个字符

举例：

- 1、订阅者是『1.*.0』，此时消息为『1.任意字符.0』，则订阅者都能收到消息
- 2、订阅者是『1.#.0』，此时消息为『1.2.3.4.4.2.2.0』，则订阅者都能收到消息（消息中间元素随意）
- 3、订阅者是『#』，则所有消息订阅者都能收到



生产者发布消息至主题

生产者通过指定以下信息向主题推送消息：

- 1) Topic名称
- 2) Topic 资源id
- 3) 发布主题，由用户自定义填写
- 4) 发布内容：为消息的主体内容，客户自定义填写，CMQ不会进行任何编码和修改
- 5) 添加消息过滤标签：Tag，即消息标签、消息类型，用来区分某个CMQ的Topic下的消息分类。允许消费者按照Tag对消息进行过滤，确保消费者最终只消费到他关心的消息类型。该功能默认不开启，未开启时，所有消息向所有订阅者发送，当订阅者设置了tag时，由于不匹配，该订阅者无法收到消息。消息过滤标签描述了该订阅中消息过滤的标签（标签一致的消息才会被推送）。单个tag不超过16个字符的字符串，单条消息可最多添加10个tag

主题向订阅者投递消息

主题向订阅者投递消息时，遵循以下原则：

- 1) Topic会尽最大努力将生产者publish的消息，投递 (notification) 到订阅者
- 2) 当投递重试多次仍失败后，消息会堆积在Topic中，等待下一次投递，若持续失败，将在消息最大生命周期 (1天) 结束后，丢弃该消息