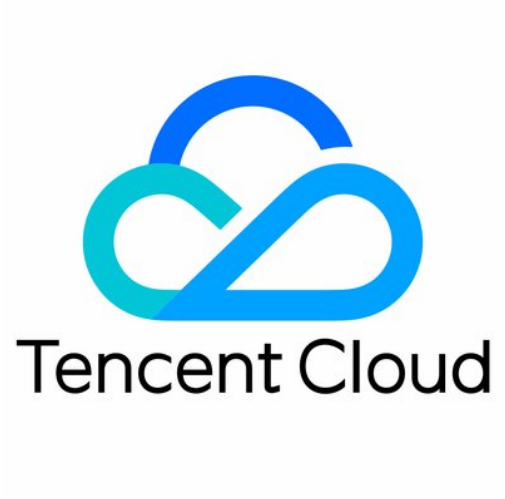
Cloud Message Queue

Message Topic Model

Product Documentation



Contents

Message Topic Model
Tag Matching

Last updated：2021-06-01 14:32:44

The topic mode of CMQ supports subscription filtering by tag, which is similar to the "direct_routing" mode of RabbitMQ. The "topic_pattern" mode will be provided in the next iteration. As the usage policies of filter tags are complex, this document uses different scenarios as examples to describe them.

Scenario Description

Scenario 1

There are four subscribers A, B, C, and D and the message tags are as follows: `apple` for A, `xiaomi` for B, `imac+xiaomi` for C, and none for D.
A producer publishes 100 messages to the topic with message filter tags `apple` , `imac` , `iphone` , and `macbook` , and the topic is delivered to A, B, C, and D immediately.

Scenario analysis:

Subscriber	Message Tag	Message Receipt Description
A	apple	As `apple` can match the `apple` message filter tag, the 100 messages can be received properly.
B	xiaomi	No messages can be received.
C	imac+xiaomi	As `imac` can match the `imac` message filter tag, the 100 messages can be received properly.
D	-	All messages can be received.

Scenario 2

A topic has only four subscribers A, B, C, and D, none of which set any message tags.
A producer publishes 100 messages to the topic with message filter tags `apple` , `imac` , `iphone` , and `macbook` , and the topic is delivered to A, B, C, and D immediately.

Scenario analysis:
As none of subscribers A, B, C, and D have tags, messages do not need to be matched during delivery, and all of

them can receive the 100 messages.

Scenario 3

A topic has only one subscriber A whose subscription tag is set to `xiaomi`.
A producer publishes 100 messages to the topic with message filter tags `apple`, `imac`, `iphone`, and `macbook`, and the topic is delivered to A immediately.

Scenario analysis:

- As the subscription tag of subscriber A does not match, A cannot receive the 100 messages. In this case, the 100 messages will be discarded immediately and will not be retained in CMQ.
- When the producer publishes to the topic, message filter tags can be set once only before publishing. They will be bound to message IDs and cannot be modified.

Scenario 4

A topic is named `test1`, and the subscription publishing API is called at 12:01 (this operation is named "publishing test1"). After the publishing, the topic is delivered to subscribers A, B, and C so as to deliver 200 messages to them. Suppose the result is as follows: A fails to receive 100 messages, B fails to receive 30 messages, and C successfully receives all the 200 messages.

Scenario analysis:

- **Message retention:** among subscribers A, B, and C, suppose the total number of messages failed to be delivered is 110 (100 ones fail to be delivered to A, 30 to B, and none to C, and the failed ones may be repeated). As long as any subscriber is associated with a message, the message will not be unsubscribed from immediately.
- **Blocking policy:** taking A as an example, the topic delivers 200 messages to A, and if the 101st message fails to be delivered, the delivery of subsequent 99 messages will be blocked. In this case, 100 messages will fail to be delivered.
- **Retry policy (backoff retry):** the `test1` topic will deliver the messages again to A once every N seconds. Specifically, the failed 100 messages will be delivered to A again starting from the first one in sequence. If a message fails to be delivered for three consecutive times, it will be directly discarded, and the next message will be delivered, which may also be discarded after three failures, and so on.
- **Retry policy (exponential decay retry):** taking A as an example, the topic delivers 100 messages concurrently to A (the sequence cannot be guaranteed). When a message fails to be delivered to A, the very message will be retried. If it fails again, subsequent messages will be blocked.
- **Inextensible message lifecycle:** suppose the 110 messages are retained in the `test1` topic. No matter how many times they are retried for delivery, their lifecycle still remains 1 day. The time point when a message is pushed by the producer to the topic will be the lifecycle start time point, and the message will be deleted once the lifecycle expires.

- **Repush**: the producer continuously produces new messages to the topic, and the subscription publishing API is called again at 12:02. Supposed that the topic now has 210 messages (110 retained messages that fail to be delivered plus 100 messages that are produced in one minute) and delivers them again, in this case, as the retry policy for A and B is exponential decay retry , A and B cannot "respond", and the 210 messages will continue to be retained. C will receive only the 100 new messages.

> Each message ID is used as the key, and the value is the associated subscribers, indicating whether consumption of each subscriber is successful.

Scenario 5

A topic named `test2` calls the subscription publishing API at 12:01, and this operation is named "publishing test2", which delivers 200 messages to subscribers A, B, and C.
Assume that A, B, and C successfully receive all the 200 messages.

Scenario analysis:
If the topic `test2` has only three subscribers A, B, and C, and the 200 messages are successfully consumed, it will immediately delete all these messages.

Rule Summary

You can summarize the following tag match rules from the aforementioned scenarios:

Subscriber	Whether Subscriber Has Tag	Whether Message Tag Exists	Message Receipt Description
A	Yes	No	The subscriber does not match and cannot receive messages.
B	No	Yes	The delivered messages do not need to be matched and all subscribers can receive messages.
C	Yes	Yes	Messages can be received only when the tags match. N:M match is supported; for example, if a message has 10 tags, and a subscriber has 4 tags, the subscriber can receive the message as long as there is one matching tag.
D	No	No	After a message is delivered, all subscribers can receive it.

Creating Topics

Last updated：2021-06-01 14:33:02

Overview

This document describes how to create a topic in the console.

Directions

Note：
You need to use the root account to create a topic subscription; otherwise, message delivery may fail.

1. Log in to the CMQ Console and click **Topic Subscription** on the left sidebar.
2. In the top-left corner of the topic subscription page, select a region.
3. On the topic subscription page, click **Create** and enter the information as prompted.
4. Click **Submit** to create a topic in the specified region.

You need to specify the following attributes when creating a topic:

Attribute	Description
Region	It sets the topic region and can be selected at the top of the topic subscription page.
Topic name	You need to enter the topic name, which cannot be modified once the topic is created. It can contain 3–64 bytes of letters, digits, hyphens (-), and underscores (_). Excessive bytes will be automatically truncated. The name is case-sensitive. To avoid confusion, topics that have the same name in the same letter case cannot be created.
Maximum message size	`MaximumMessageSize` attribute of topic. It specifies the maximum length of the message body that can be sent to the topic. It is measured in bytes and ranges from 1 to 1,024 KB. The default value is 64 KB.
Message lifecycle	It specifies the maximum period of time during which a message can be retained in the topic. After the period specified by this parameter has elapsed since a message is sent to the queue, the message will be deleted no matter whether it has been fetched. It is measured in seconds and defaulted to 86,400 seconds, which cannot be modified.

Attribute	Description
Message retention	It is enabled by default. Messages that are produced by a producer but have not triggered push to subscribers or failed to be received by subscribers will be retained in the topic temporarily. In the topic list, you can view the approximate total number of currently retained messages.
Message filter type	Tag: for more information, please see Tag Matching Feature Description. Routing matching: for more information, please see Routing Key Matching Feature Description.

Modifying Topic Properties

Last updated：2020-09-04 16:26:26

Operation Scenarios

This document describes how to modify the attributes of a created topic in the console.

Prerequisites

You have created a topic.

Directions

1. Log in to the CMQ Console and click **Topic Subscription** on the left sidebar.
2. In the topic subscription list, click **Change Settings** in the "Operation" column to modify the maximum message size.
3. Click **Submit**.

- The topic region cannot be modified.
- The topic name and ID cannot be modified.
- The maximum message size can be modified and specifies the maximum length of the message body that can be sent to the topic. It ranges from 1 to 1,024 KB, and the default value is 64 KB.
- The creation time and modification time cannot be modified.

Subscribing Topics

Last updated：2020-08-04 15:52:57

You can subscribe to a topic by specifying the following attributes:

- Topic name.
- Topic resource ID.
- Subscription name, which cannot be modified once entered.
- Subscription terminal protocol, which can be a queue message service or URL address.
- Subscription address, which can be a URL or queue name. Currently, a topic can send messages to a queue under the same account.
- Retry policy: it is the `NotifyStrategy` attribute of the subscription, i.e., the retry policy used when an error occurs during message push to recipients. This policy is enabled by default, and you need to select one of the following two options:
  - Backoff retry: an attempt will be retried three times at random intervals between 10 and 20 seconds. After three retries, the message will be discarded for the subscriber and will not be retried again.
  - Exponential decay retry (checked by default): an attempt will be retried 176 times at exponentially increasing intervals: 2^0 seconds, 2^1 seconds, ..., 512 seconds, 512 seconds, ..., 512 seconds. The total retry duration is 1 day.
- Retry verification: if the HTTP return code is 200, it is successful.
- Subscriber tag: when adding a subscriber, you can add filter tags (`FilterTag`), so that the subscriber can receive only messages with the specified tags. One tag can contain up to 16 characters, and up to 5 tags can be added for one subscriber. As long as a tag matches a topic filter tag, the subscriber can receive messages delivered by the topic. If a message does not have any tag, the subscriber cannot receive it.
- Maximum number of subscribers to topic: one topic can be associated with up to 500 subscribers.
- Total number of messages associated with subscriber: it is an approximate value and indicates the number of messages that are waiting for or being retried for delivery to the subscriber in a topic.

Publishing Messages to Topics

Last updated：2021-06-29 18:09:33

A producer can push a message to a topic by specifying the following information:

- Topic name
- Topic resource ID
- Target topic, which can be customized
- Published content: it is the body of the message and can be customized. CMQ will not encode or modify it.
- Message filter tag: a tag, i.e., message tag or message type, is used to identify a message category under a topic in CMQ. A consumer can filter messages by tag so that it can consume only message types of interest to it. This feature is disabled by default. If it is disabled, all messages will be sent to all subscribers. If a tag is set by a subscriber, unmatched messages will not be received by the subscriber. A message filter tag describes the tag used for message filtering in the subscription (only messages with the same tag can be pushed). One tag can contain up to 16 characters, and up to 5 tags can be added to one message.

Routing Key Matching

Last updated：2020-05-12 21:46:27

CMQ routing key match is similar to exchange queue in RabbitMQ and can be used to filter messages so as to enable subscribers to get different messages by condition. When creating a topic, you can enable **Routing Matching Key**.

Instructions

The binding key and routing key are used together and function in a way similar to the message filtering capability of RabbitMQ. The routing key carried when a message is sent is added by the client, and the binding key carried when a subscription relationship is created is the binding relationship between the topic and the subscriber.

Use Limits

- There can be up to 5 binding keys, and each of them can contain up to 64 bytes to represent the route for message sending, which can have up to 15 `.` , i.e., up to 16 phrases.
- All routing keys are contained in a string, and each of them can contain up to 64 bytes to represent the route for message sending, which can have up to 15 `.` , i.e., up to 16 phrases.

Wildcard Description

- ◦ (asterisk) represents a word (a letter string) and cannot be empty.

· (hashtag) matches zero or multiple characters.

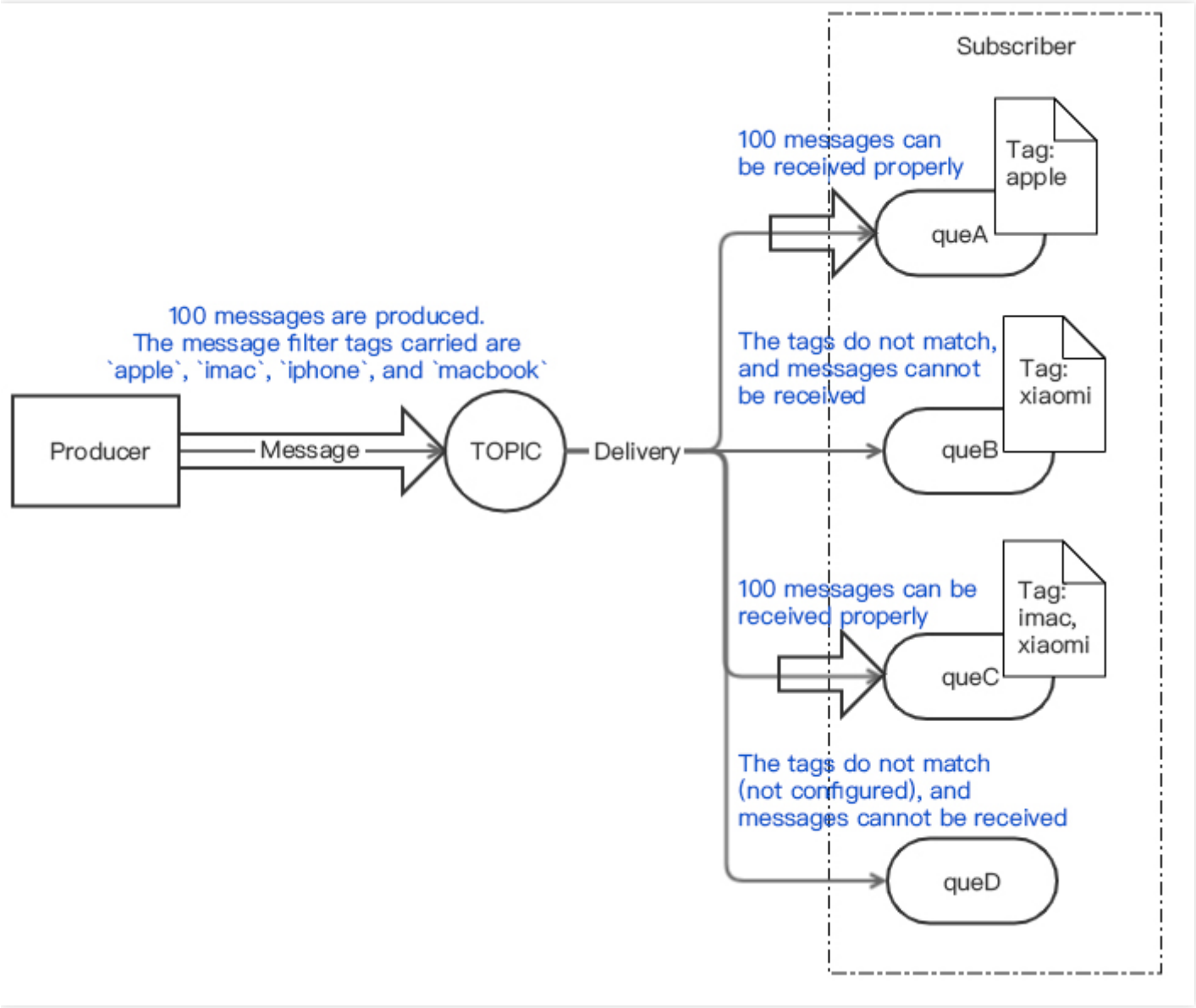
Example:

- If the subscriber is **1.*.0**, and the message is **1.any character.0**, then it can be received by the subscriber.
- If the subscriber is **1.#.0**, and the messages are **1.2.3.4.4.2.2.0** and **1.0**, then both of them can be received by the subscriber (the elements in the middle of the messages can be arbitrary).
- If the subscriber is **#**, then the subscriber can receive all messages.

Delivering Messages to Subscriber by Topic

Last updated：2021-06-01 14:33:34

The model where a topic delivers a message to a subscriber is as shown below:



The topic follows the rules below when delivering the message to the subscriber:

- The topic will try its best to deliver the message published by the producer to the subscriber.
- If the delivery fails after multiple retries, the message will be retained in the topic and wait for the next delivery. If the next delivery still fails, the message will be discarded after the maximum lifecycle (1 day).

If the topic fails to deliver the message to the subscriber, you can troubleshoot as follows:

- The `SecretId` and `SecretKey` need to be provided for message delivery (persistent key is required). You can get them in API Key Management.
- You need to use a root account rather than sub-account to create a subscription.
- If the queue name does not exist, please check whether the queue exists in the CMQ Console.