

# 消息队列 RocketMQ 版

## 产品简介

## 产品文档



**【版权声明】**

©2013–2026 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】**

及其他腾讯云服务相关的商标均为腾讯集团下的相关公司主体所有。另外，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默示的承诺或保证。

# 文档目录

## 产品简介

产品概述

什么是消息队列 RocketMQ 版

产品优势

应用场景

开源对比

高可用

使用限制

开服地域

基本概念

# 产品简介

## 产品概述

最近更新时间：2026-01-23 16:23:16

[查看网页](#)

## 产品概述

TDMQ (Tencent Distributed Message Queue) 是腾讯云自主研发的消息中间件产品系列，作为分布式系统中的关键组件，具备**稳定可靠、高弹性、低成本**的特性，提供**异步通信**的基础能力，通过**应用解耦**降低系统复杂度，提升系统可用性和可扩展性。

- **兼容开源主流协议**，包含 CKafka、RocketMQ、RabbitMQ、Pulsar、MQTT 五大子产品。提供迁移方案支持，零业务代码修改，降低迁移成本。
- **覆盖在线场景（电商交易、社交直播等）、离线场景（大数据实时计算、离线分析等）和设备端场景（物联网、车联网等）**，满足泛互、教育、零售、出行、金融、医疗等不同行业和场景的需求。



## 产品优势

### 开箱即用免运维

TDMQ 提供开箱即用的全托管服务，用户一键即可创建集群，彻底告别繁琐的安装部署工作。完善的资源管理界面、全面的监控指标和智能诊断工具，大幅降低运维复杂度和管理成本。

### 跨可用区高可用

TDMQ 通过多重技术手段构建了全方位的容灾体系，采用跨可用区部署架构，有效防范机房级故障风险；通过限流保护策略，动态调节流量压力，确保集群健康运行；同时提供跨集群数据复制能力，全面满足从基础灾备到多活部署等各类高可用场景需求。

## 快速扩缩容高弹性

TDMQ 提供极致弹性伸缩能力，通过一键式操作即可实现资源的快速扩缩容，底层资源无缝变配全程对业务无感知，轻松应对各类突发流量场景。

## Serverless 化低成本

TDMQ 采用存算分离架构，计算层支持秒级弹性伸缩，无需预先资源即可应对突发流量，实现资源利用率最大化；同时存储层支持无限扩展，按需使用按量付费，存储成本降低30%–50%。

## 产品对比

面向不同客户的场景和需求，TDMQ 都可以提供最合适的产品形态和方案。如有任何需求，可以 [联系我们](#) 进行咨询。

产品	产品特性	产品优势	应用场景	适用业务
CKafka	<ul style="list-style-type: none"> <li>高吞吐，大数据生态丰富</li> <li>内核增强，支持版本自动升级</li> <li>智能运维，支持分区均衡、磁盘自动扩容等策略</li> </ul>	高吞吐标杆，性能稳定、应用广泛	对吞吐要求高的离线场景	日志压缩收集、监控数据聚合、流数据集成
RocketMQ	<ul style="list-style-type: none"> <li>低延迟高并发，在线场景广泛落地</li> <li>消息特性丰富，事务/定时/延时/顺序</li> <li>支持平滑迁移，低侵入可回滚</li> </ul>	海量消息堆积、低延迟、高吞吐、高可靠	对可靠性要求高、低延时的在线业务场景	异步解耦、削峰填谷、顺序收发、分布式事务一致性
RabbitMQ	<ul style="list-style-type: none"> <li>社区诞生时间最久，多语言客户端齐全</li> </ul>	100%兼容开源，提供灵活的路由模式	适合中小体量的在线业务场景	秒杀、优先级消息、延迟消息、消息广播
Pulsar	<ul style="list-style-type: none"> <li>存算架构分离，在离线一体化</li> <li>腾讯内部大规模落地</li> </ul>	计算存储分离，灵活扩缩容	兼容在线和离线场景需求	异步解耦、削峰填谷、顺序收发、数据同步
MQTT	<ul style="list-style-type: none"> <li>MQTT 协议，车联网、物联网</li> </ul>	轻量级协议，支持百万客户端同时在线	适用各类移动设备的业务消息投递	车联网、工业互联网、IM 通信、智能家居

# 什么是消息队列 RocketMQ 版

最近更新时间：2026-01-23 16:23:16

消息队列 RocketMQ 版（TDMQ for RocketMQ，简称 TDMQ RocketMQ 版）是一款分布式高可用的消息队列服务，基于 Apache RocketMQ 的 4.x 和 5.x 架构提供不同的产品形态，支持 RocketMQ 4.4.x 及以上版本的客户端零改造接入，同时具备计算存储分离，灵活扩缩容的底层优势。TDMQ RocketMQ 版最多可以支持百万级 TPS 的吞吐量，适用于各类大规模、低延时、对可靠性要求高的在线消息业务场景。

## 产品特性

### 免运维

- 支持一键创建集群，开箱即用。
- 按 TPS 规格快速扩缩容，提供极致弹性。
- 5.x TPS 支持弹性区间，应对突发流量。
- 5.x 存储 Serverless 化，支持按需使用，按量付费。

### 特性丰富

- 支持多种消息类型：事务/定时/延时/顺序消息。
- 支持多种消费模式：Tag 过滤、SQL 92 过滤、集群消费、广播消费。

### 可观测性

- 提供全面的监控指标，快速发现问题。
- 支持可视化消息轨迹，串联上下游业务，更好的排查和定位问题。

### 高可用

- 支持跨可用区部署，抵御机房级故障。
- 容器化秒级自动重启，单节点宕机时容量和数据不受损。
- 支持集群限流保护，提升集群健康度。
- 支持跨集群消息复制，满足灾备需求。

### 安全管控

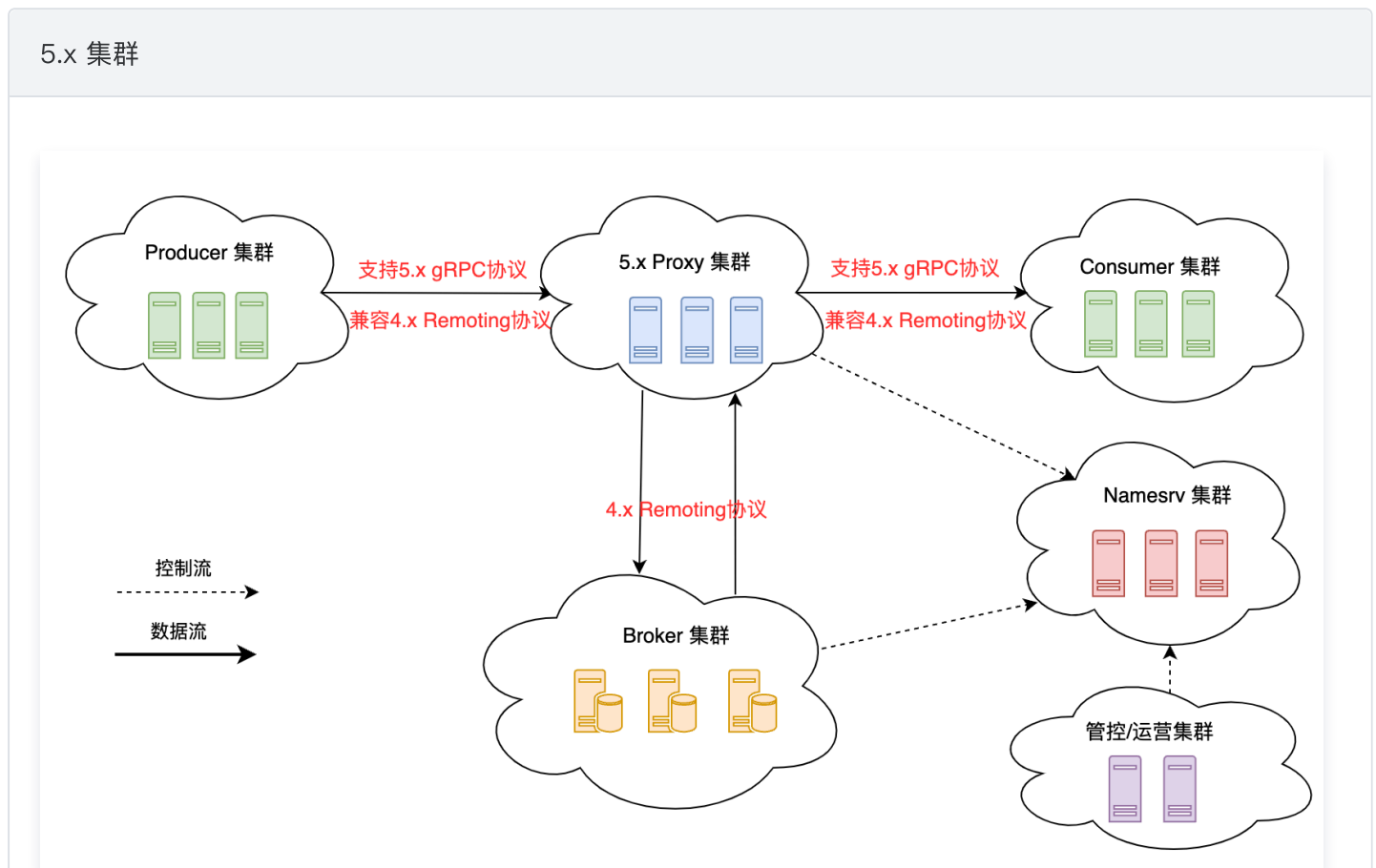
- 集成腾讯云的 CAM 系统，支持操作级/资源级的权限管理能力。
- 可视化的 ACL 管理页面，支持收发消息的权限管理。
- 公网访问支持白名单控制。

### 平滑迁移

- 开源兼容，0业务代码修改。
- 提供元数据迁移工具，降低迁移成本。
- 提供集群平滑迁移方案，低侵入可回滚，减少业务侵入。

## 技术架构

消息队列 RocketMQ 版的系统部署架构图如下：



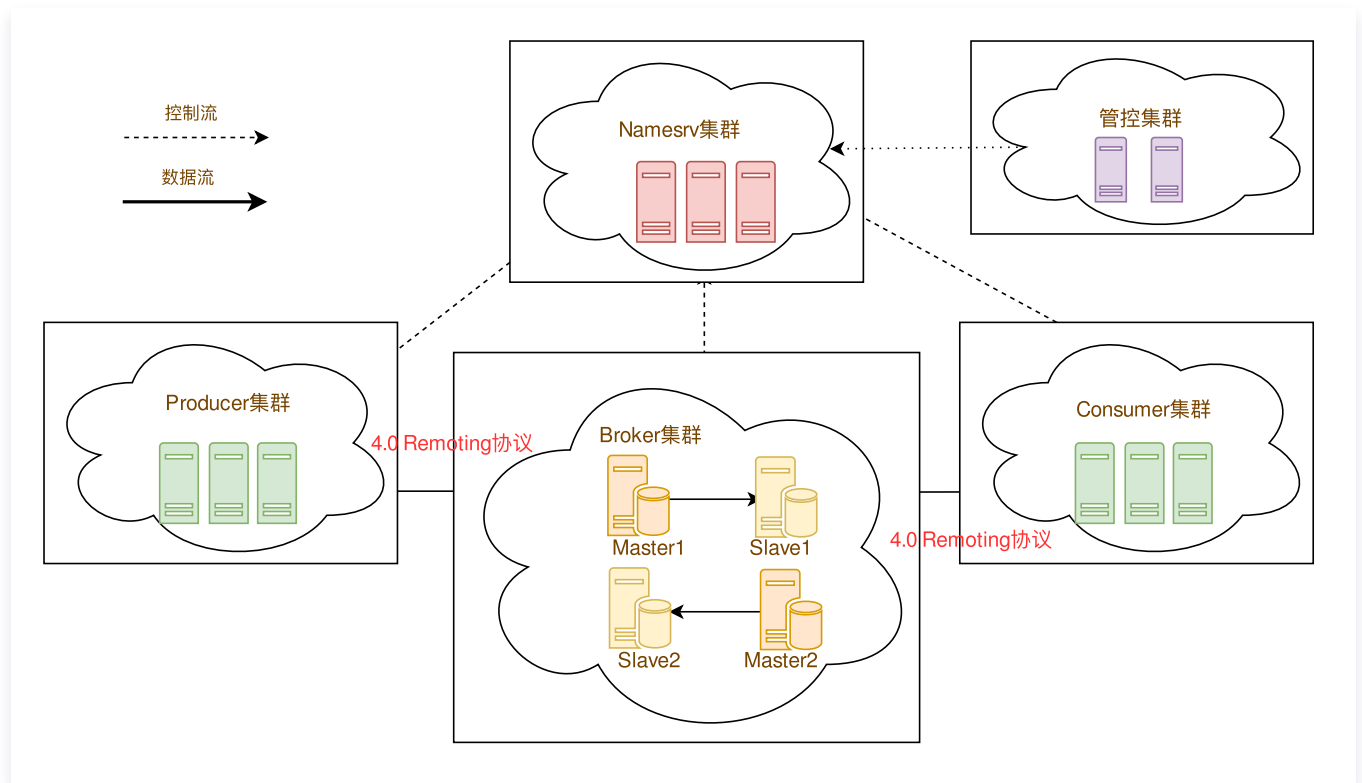
消息队列 TDMQ RocketMQ 版 5.x 系列引入了新的 gRPC 协议和 Proxy 组件，实现了存算分离的架构，对 RocketMQ 的运维和使用都会带来巨大的变化。

其中涉及各个概念如下：

- Producer 集群：客户侧应用，负责生产并发送消息。
- Consumer 集群：客户侧应用，负责订阅和消费处理消息。
- NameServer 集群：服务端应用，负责路由寻址和 Broker 心跳注册。为保证高可用，默认跨可用区部署。
  - 心跳注册：NameServer 相当于注册中心的角色，各个角色的机器都要定时向 NameServer 上报自己的状态，如果超时未上报，NameServer 会认为某个机器出现故障不可用，从而将这个机器从可用列表中删除。
  - 路由寻址：每个 NameServer 中都保存着 Broker 集群的整个路由信息和用于客户端查询的队列信息，生产者和消费者通过 NameServer 去获取整个 Broker 集群的路由信息，从而进行消息的投递和消费。

- Proxy 集群：全新的弹性无状态代理服务，为保证高可用，默认跨可用区部署。将 4.x 中的 Broker 职责进行拆分，对于客户端协议适配、权限管理、消费管理等计算逻辑进行抽离。
- Broker 集群：与 4.x 产品系列相比，在 5.x 系列中，Broker 更专注于存储能力的持续优化。为保证高可用，默认跨可用区部署。

## 4.x 集群



其中涉及各个概念如下：

- Producer 集群：客户端应用，负责生产并发送消息。
- Consumer 集群：客户端应用，负责订阅和消费处理消息。
- NameServer 集群：服务端应用，负责路由寻址和 Broker 心跳注册。
  - 心跳注册：NameServer 相当于注册中心的角色，各个角色的机器都要定时向 NameServer 上报自己的状态，如果超时未上报，NameServer 会认为某个机器出现故障不可用，从而将这个机器从可用列表中删除。
  - 路由寻址：每个 NameServer 中都保存着 Broker 集群的整个路由信息和用于客户端查询的队列信息，生产者和消费者通过 NameServer 去获取整个 Broker 集群的路由信息，从而进行消息的投递和消费。
- Broker 集群：服务端应用，负责接收，存储，投递消息，支持主从多副本模式，从节点可选部署，实际现网公有云上数据高可靠直接依赖云盘三副本。
- 管控集群：服务端应用，可视化的管控控制台，负责运维整个集群，例如元数据的管理等。



# 产品优势

最近更新时间：2026-01-23 16:23:16

## 产品优势

### 兼容开源

100% 兼容 Apache RocketMQ 的各个组件与概念，支持 RocketMQ 4.4.x及以上版本的客户端零改造接入，同时具备计算存储分离，灵活扩缩容的底层优势。

### 多种规格系列

提供多种售卖规格，覆盖不同客户的业务规模；支持按实际消息量、按小时和包年包月等多种灵活计费方式。不同规格间可以灵活进行升降配，满足从小规模测试到大范围应用的产品落地规律。

### 适配云原生

底层架构进行云原生改造升级，Serverless 化的产品形态带来极致弹性，支持规格外的弹性计算能力，同时存储按实际使用量收费。

### 丰富的消息类型

支持普通消息、顺序消息、延时消息、分布式事务消息等多种消息类型，支持消息重试和死信机制，满足各类业务场景。

### 高性能

单机最高可支持上万级别的生产消费吞吐，分布式架构，无状态服务，可以横向扩容来增强整个集群的吞吐。

### 易用免运维

提供 API 访问接口，支持开源所有语言和版本的 SDK。提供腾讯云平台整套运维服务，实时监控告警，帮助用户快速发现并解决问题，保证服务的可用性。

## 5.x 系列独有优势

消息队列 RocketMQ 5.x 系列除了兼具低延迟、高性能、高可靠、万亿级消息容量和灵活可扩展等历史版本的特点之外，充分结合云原生大潮下的基础设施和生态技术，提高资源利用和弹性能力。

相比于自建的 RocketMQ 和 消息队列 TDMQ RocketMQ 版 4.x 系列，5.x 系列还具有以下优势：

### 存储和计算弹性

TDMQ RocketMQ 版 5.x 系列使用了存算分离的架构，充分提高了资源利用率和弹性能力。存储使用按量后付费，客户无需为峰值提前存储资源，按实际使用量进行付费，有效降低客户的实际成本。计算规格支持弹性TPS，客户无需为计划外的峰值预留计算资源，有效降本。

## 轻量化 SDK

TDMQ RocketMQ 版 5.x 系列兼容开源社区 SDK，享受开源社区迭代带来的红利。5.x 系列的客户端更加轻量，采用了全新的极简的 API 设计，更易被集成和使用，同时 5.x 系列提供了更多语言的 SDK，使得开发者在使用时有更多的技术栈选择。

## 基础功能增强

TDMQ RocketMQ 版 5.x 系列有了进一步的功能增强，如更加灵活的消息保留时间控制，可以根据集群或者 Topic 粒度设置消息保留时间。消费者组有了更多的白屏化设置，如可以在服务端指定消息重试次数和自由绑定死信队列等功能。

## 可观测性

TDMQ RocketMQ 版 5.x 系列增加了更加丰富的指标，如消息堆积场景相关指标、关键接口的耗时指标、错误分布指标、存储读写流量等指标。这些指标都得以对接腾讯云的监控和告警功能，同时提供完善的云 API，支持集成自助运维系统。

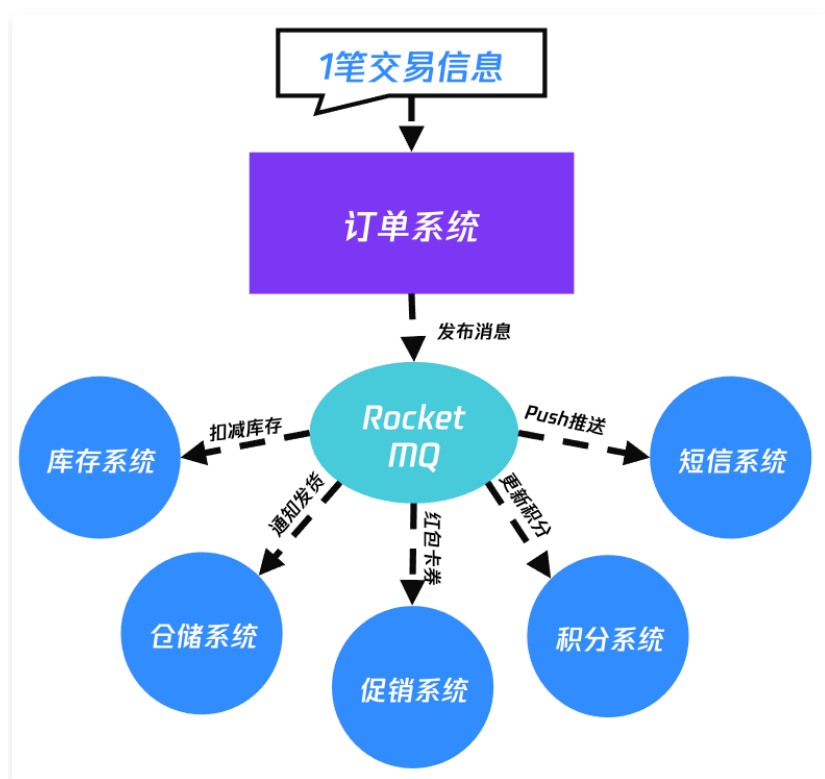
# 应用场景

最近更新时间：2026-01-23 16:23:16

TDMQ RocketMQ 版是基于 Apache RocketMQ 构建的分布式消息中间件，应用于分布式系统或组件之间的消息通讯，具备海量消息堆积、低延迟、高吞吐、高可靠、事务强一致性等特性，满足异步解耦、削峰填谷、顺序收发、分布式事务一致性、日志同步等场景需求。

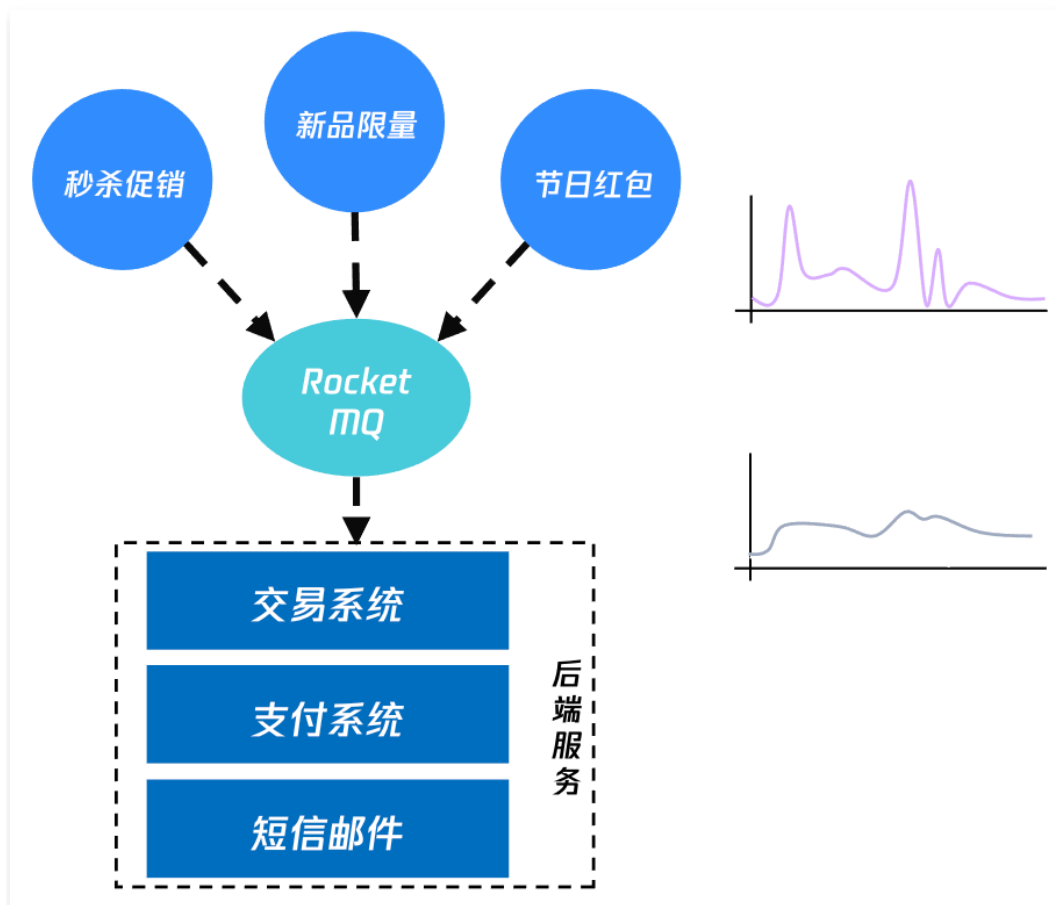
## 异步解耦

交易引擎作为腾讯计费最核心的系统，每笔交易订单数据需要被几十个下游业务系统关注，包括库存系统、仓储系统、促销系统、积分系统等，多个系统对消息的处理逻辑不一致，单个系统不可能去适配每一个关联业务。此时，TDMQ RocketMQ 版可解除多个业务系统之间的耦合度，减少系统之间影响，提升核心业务响应速度和健壮性。



## 削峰填谷

企业不定时举办的一些营销活动，新品发布上线，节日抢红包等，往往都会带来临时性的流量洪峰，这对后端的各个应用系统考验是十分巨大的，如果直接采用扩容方式应对又会带来一定的资源浪费。RocketMQ 可以应对突发性的流量洪峰，在峰值时堆积消息，而在峰值过去后下游系统慢慢消费消息，解决上下游处理能力不匹配，提升系统可用性。



## 订阅通知

消息队列 RocketMQ 版提供的定时、延迟等能力，满足需要订阅通知的电商场景。

- **定时消息**：消息在发送至服务端后，实际业务并不希望消费端马上收到这条消息，而是推迟到某个时间点被消费，这类消息统称为定时消息。
- **延时消息**：消息在发送至服务端后，实际业务并不希望消费端马上收到这条消息，而是推迟一段时间后再被消费，这类消息统称为延时消息。

关于定时与延迟消息的详细内容，请参见 [定时与延迟消息](#)。



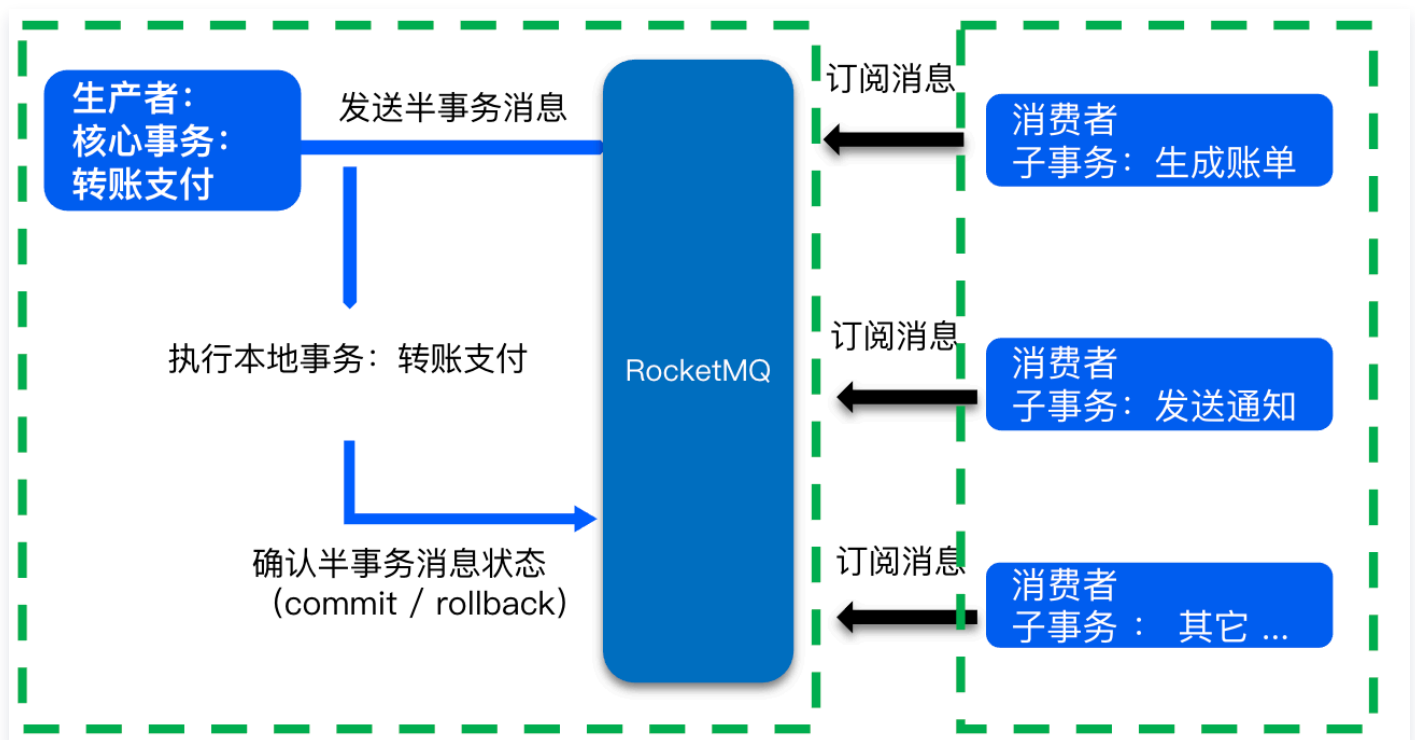
## 分布式事务一致性

RocketMQ 提供分布式事务消息，使应用之间松耦合，可靠传输与多副本技术能确保消息不丢失，At-Least-Once 特性确保数据最终一致性。

- 支付系统作为生产者，与消息队列，组成一个事务，保障本地事务和消息发送的一致性。
- 下游业务系统（账单、通知、其它），作为消费者，并行处理。
- 消息支持可靠重试，保证数据最终一致性。

使用消息队列 RocketMQ 版的事务消息来处理交易事务，可以大大提升处理效率和性能。计费系统的交易链路通常比较长，出错或者超时的概率比较高，这时会借助 TDMQ 的自动重推和海量堆积能力来实现事务补偿，同时支付 Tips 通知和交易流水推送可以通过 RocketMQ 来实现最终一致性。

关于事务消息的详细内容，请参见 [事务消息](#)。

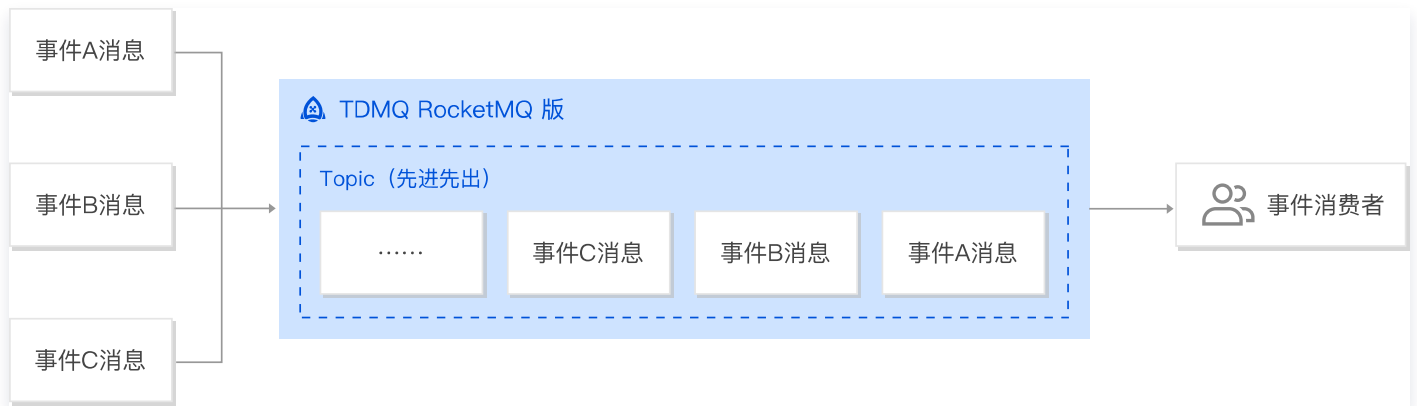


## 顺序收发

顺序消息是消息队列 RocketMQ 提供的一种高级消息类型，对于一个指定的Topic，消息严格按照先进先出（FIFO）的原则进行消息发布和消费，即先发送的消息先消费，后发送的消息后消费。顺序消息常用于以下业务场景：

- **订单创建场景：** 在一些电商系统中，同一个订单相关的创建订单消息、订单支付消息、订单退款消息、订单物流消息必须严格按照先后顺序来进行生产或者消费，否则消费中传递订单状态会发生紊乱，影响业务的正常进行。因此，该订单的消息必须按照一定的顺序在客户端和消息队列中进行生产和消费，同时消息之间有先后的依赖关系，后一条消息需要依赖于前一条消息的处理结果。
- **日志同步场景：** 在有序事件处理或者数据实时增量同步的场景中，顺序消息也能发挥较大的作用，如同步 mysql 的 binlog 日志时，需要保证数据库的操作是有顺序的。
- **金融场景：** 在一些撮合交易的场景下，比如某些证券交易，在价格相同的情况下，先出价者优先处理，则需要按照FIFO的方式生产和消费顺序消息。

关于顺序消息的详细介绍，请参见 [顺序消息](#)。



## 分布式模缓存同步

企业举办打折促销活动时，产品种类繁多并且价格频繁变动，用户访问较多次商品价格查询，缓存服务器网卡满载，影响页面的打开速度。使用 TDMQ RocketMQ 版的广播消费模式，那么这条消息会被所有节点消费一次，相当于把价格信息同步到需要的每台机器上，取代缓存的作用。

# 开源对比

最近更新时间：2026-01-23 16:23:17

TDMQ RocketMQ 版和开源版 RocketMQ 的对比如下：

5.x 集群			
功能大类	功能项	腾讯云 TDMQ RocketMQ 5.x	开源 RocketMQ
安全管控	收发消息 ACL 管理	支持生产消息和发送消息的角色细化，支持更加细粒度的权限细化。	默认 ACL 鉴权方式，鉴权方式单一。
	主子账号	全面支持腾讯云主子账号，实现 CAM 主子账号及企业内跨账号和授权等服务。	不支持。
	扩缩容	客户无需关注底层机器大小，也无需扩缩容机器，只需要根据业务量购买相应的规格，并支持在不同规格间进行自由升降配。底层资源通过容器化实现了弹性扩缩容，自动化智能运维。	依赖自建运维团队，自动化、白屏化程度低。
	高可用	腾讯云底层实现高可用方案，客户无需关心部署容灾架构。	需要自行部署高可用架构，增加运维难度。
迁移工具	迁移工具	同时支持两种迁移方式：自建元数据导入后改接入点迁移 和 支持无感迁移，白屏化操作，按照不同的 Topic 分阶段进行灰度迁移。	不支持迁移。
监控告警	资源大盘	核心指标观测、生产消费报表和细粒度监控，支持公网、集群、topic、group 以及 topic&group 粒度的指标下钻；集群粒度提供资源 Top 排行。	简单支持，部分监控指标缺失。
	报警管理	消息积压、延迟等多项指标告警，云监控联动。	不支持。
弹性能力	弹性 TPS	专业版和铂金版支持，在正常的流量规格外，支持开启临时的弹性流量支持。	不支持，需要按需扩缩容机器，提高运维复杂度。

	存储免配额限制	存储按实际使用量进行收费。	不支持，部署时需要指定磁盘大小，磁盘买大容易导致资源浪费，且云上购买的磁盘不支持扩容；磁盘买小需要频繁进行扩缩容，提高运维难度。
稳定性	分布式限流	支持多种限流方式和策略（全局/本地，降级策略，多规则优先级等），能有效避免流量过大引起宕机。	不支持，存在因为流量过载导致宕机问题。
	收发消息配比调整	支持在集群维度调整调整整个集群的消息收发占比，并进行分别限流，资源利用更加合理。	不支持。
消息生命周期	Topic 级别保存时间设置	支持，根据 Topic 设置消息保留时间，进一步节约存储成本。	不支持。
控制台功能优化	客户端堆栈查看	支持，并支持堆栈内指定代码搜索。	不支持。
	按 Tag 查询消息	支持。	不支持。
	查询重试/死信/延时消息。	单独根据指定条件查询特定类型的消息。	不支持。
兼容性	兼容历史低版本 SDK	完全兼容 4.1之后的所有低版本 SDK。	不完全兼容，部分场景会出现报错。

#### 4.x 集群

TDMQ RocketMQ 版与开源 RocketMQ 的性能对比详情如下：

功能大类	功能项	腾讯云 TDMQ RocketMQ 4.x	开源 RocketMQ
基础功能	定时消息	优化（精准秒级），支持任意延时刻度	有限支持（指定延迟级别）
	可视化管理能力	支持对集群、topic、group 等进行可视化管理和详细信息浏览，包括订阅关系、消	简单支持，易用性一般，控制台不区分

		费者状态等	topic 类型
可用性	弹性伸缩	客户无需关注部署和扩容，无需人工配置，节点注册等操作完全自动化和白屏化；用户可以根据需要随时横向扩容节点数量、增加存储磁盘大小、垂直升级单节点配置	依赖自建运维团队，自动化、白屏化程度低
	高可靠	数据三副本，容器化秒级自动重启，保证宕机时容量和数据都不受损	支持同步复制和异步复制，需要自行设计部署方案和参数，主从同步方案不支持自动切主
	跨 AZ 高可用部署	支持跨 AZ 高可用部署，避免机房级故障	支持但较为繁琐，需要自行设计部署方案和参数
可观测性	资源大盘	核心指标观测、生产消费报表和细粒度监控	简单支持，部分监控指标缺失
	报警管理	消息积压、延迟告警，腾讯云可观测平台联动	不支持
安全管控	租户命名空间隔离	控制台可视化支持	不支持，命名空间 bug 较多，无法实现真正隔离
	主子账号管理	全面支持腾讯云主子账号，实现 CAM 主子账号及企业间跨账号的授权服务	不支持
迁移工具	开源迁移工具	脚本化一键迁移，可以无缝从开源 RocketMQ 迁移	-

# 高可用

最近更新时间：2026-01-23 16:23:17

腾讯云 TDMQ RocketMQ 版采用“多主架构 (Multi-Master) + 跨可用区部署”保障服务高可用，同时利用“云盘三副本”机制保障数据高可用。这种架构摒弃了传统的 Master-Slave 模式，极大地简化了运维复杂性，并提供了金融级的可靠性。

## 集群高可用

集群高可用的核心目标是：确保即使部分组件发生故障，整个消息服务依然能够对外提供不间断的读写服务。腾讯云 TDMQ RocketMQ 版通过“NameServer 集群 + 无状态 Proxy + Broker 多主 + 跨可用区部署”的组合，构建了无单点故障、具备区域级容灾能力的高可用服务集群。

## NameServer 集群化与跨可用区部署

NameServer 是 RocketMQ 的“大脑”，负责服务发现和路由管理。它本身必须是高可用的。在腾讯云方案中：

- 集群化部署：至少部署 2 个 NameServer 节点，组成一个无状态的集群。
- 跨可用区 (AZ) 部署：将这些 NameServer 节点分散部署在同一地域下的多个不同可用区。可用区是物理隔离的数据中心，一个可用区的故障不会影响其他可用区。

这样，任何一个 NameServer 节点甚至整个可用区的故障，都不会影响路由信息的获取，Producer 和 Consumer 依然可以从其他健康的 NameServer 节点获取到 Broker 的地址列表。

## Broker 多主 (Multi-Master) 架构

与传统的 Master-Slave 架构不同，多主架构下，没有主从之分，所有 Broker 节点都是 Master，都可以随时接收生产者的消息写入请求。

## 读写负载均衡

Producer 在发送消息时，会从 NameServer 获取所有可用的 Master Broker 列表，并采用轮询等策略将消息写入不同的 Broker，天然实现了写入流量的负载均衡。

## 无缝故障转移

- 单节点故障：当某个 Broker 节点因宕机或网络问题与 NameServer 心跳中断时，NameServer 会立即将其从路由表中剔除。
- 客户端自动重试：Producer 和 Consumer 会定时从 NameServer 更新 Broker 列表。当它们发现某个 Broker 不可用时，会自动跳过该节点，将请求（如消息发送、消息拉取）无缝切换到集群中其他健康的 Broker 节点上。
- 整个过程对业务应用完全透明，无需人工干预，实现了秒级的故障转移。

## 跨可用区 (Cross-AZ) 部署 Broker

为了应对数据中心级别的灾难，我们将多个 Master Broker 节点分散部署在不同的可用区。根据用户的可用区选择，强制跨可用区分布。

## 故障场景模拟

假设集群部署在广州地域的三个可用区（AZ1、AZ2、AZ3），每个可用区都有 Broker 节点。

### 场景一：AZ1 的某台 Broker 服务器宕机。

- NameServer 侦测到心跳超时，将该 Broker 从路由信息中移除。
- 新的生产和消费请求会自动路由到 AZ1 的其他 Broker 以及 AZ2、AZ3 的 Broker 上，服务不中断。

### 场景二：AZ1 整个可用区因电力或网络故障完全不可用。

- 所有位于 AZ1 的 NameServer 和 Broker 节点都将离线。
- 由于 NameServer 和 Broker 在 AZ2 和 AZ3 仍有健康的节点，客户端会连接到这些节点上。
- 整个集群的服务能力会暂时下降，但核心的收发消息功能依然可用，保证了业务的连续性。

## 跨可用区（Cross-AZ）部署 Proxy

针对 5.x 的产品形态，我们也和 Broker 一样，根据用户的可用区选择，强制跨可用区分布。基于 Proxy 的“无状态”特性。这意味着：

- 不存储业务数据：消息数据、消费位点（Offset）等持久化状态信息依然存储在 Broker 的云盘上。
- 不维持长会话状态：Proxy 自身不保存关键的、不可恢复的会话信息。任何一个 Proxy 节点都可以处理任何一个客户端的请求。

同时在部署层面，采用多节点集群 + 前置负载均衡（Load Balancer）。

- 部署多个 Proxy 实例：我们会部署至少 3 个或更多的 Proxy 实例，并将它们像 NameServer 和 Broker 一样，分散在不同的可用区。
- 配置前置负载均衡器（LB）：在所有 Proxy 实例的前方，我们会配置一个负载均衡器（例如腾讯云的 CLB）。这个 LB 对外提供一个唯一的虚拟 IP（VIP）地址。
- 客户端连接 VIP：所有的 Producer 和 Consumer 客户端，不再直接连接 NameServer 或 Broker，而是只配置并连接这个统一的 LB VIP 地址。

## 数据高可用

数据高可用是消息队列的生命线，其目标是确保已成功写入的消息数据不会因任何硬件故障而丢失。传统 RocketMQ 依赖 Master-Slave 同步复制（SYNC\_FLUSH）来保证数据不丢，但这带来了性能开销大、主从切换复杂等问题。

腾讯云利用了 IaaS 层的能力，通过云硬盘（CBS）的三副本机制，从根本上解决了数据持久化的高可用问题。

“Broker + 云盘三副本”的模式，是一种典型的计算存储分离架构。它将 RocketMQ Broker 变成了一个“无状态”的计算节点，而将“有状态”的数据存储交给了专业、高可用的分布式块存储服务，从而实现了数据的高可用。

## 什么是云盘三副本？

云硬盘（Cloud Block Storage, CBS）是腾讯云提供了一种高可用、高可靠、低时延的网络块存储设备。其核心特性之一就是，数据三副本机制。

- 当 RocketMQ Broker 向其挂载的云盘写入一条消息数据（CommitLog/ConsumeQueue）时，云盘的底层存储系统会自动、同步地将这份数据写入到同一可用区内三个不同物理机架上的三份物理拷贝。
- 只有当三份拷贝都成功写入后，写操作才会向上层应用（即 RocketMQ Broker）返回成功。
- 这个过程对上层应用是完全透明的，Broker 只是在进行一次普通的本地磁盘写入。

## 云盘三副本如何保障数据高可用？

这种架构将数据可靠性的保证，从应用层（RocketMQ Master-Slave 复制）下沉到了更可靠、更高效的基础设施层（分布式存储）。

### 无惧单盘/单机故障

任何一个数据副本所在的物理磁盘或服务器发生故障，系统会自动从另外两个健康的副本中恢复数据，并选择一个新的位置创建新的副本，始终保持三副本状态。整个过程对业务无感知，数据零丢失。

### 简化架构，告别主从复制

由于数据在存储层已经实现了高可用，我们不再需要部署 Slave 节点，也无需配置复杂的 Master-Slave 同步复制。这带来了巨大优势：

- 无复制延迟：不存在主从之间的数据同步延迟问题。
- 运维极简：无需处理主从切换、数据补齐等复杂的运维场景。
- 快速恢复：当 Broker 节点（虚拟机）故障后，我们只需启动一个新的 Broker 实例，并重新挂载原来的云盘。由于所有消息数据都完好无损地保存在云盘上，Broker 实例可以立即恢复服务，恢复时间（RTO）大大缩短。

## 容器化部署

在上述高可用架构的基础上，如果我们将整个 RocketMQ 集群（包括 NameServer 和 Broker）进行容器化部署，并运行在以腾讯云 TKE（Tencent Kubernetes Engine）为代表的容器编排平台上，可以实现标准化交付、快速的扩容、以及异常场景下的自动恢复。

# 使用限制

最近更新时间：2026-01-23 16:23:17

本文列举了 TDMQ RocketMQ 版集群中对一些指标和性能的限制，请您在使用中注意不要超出对应的限制值，避免出现异常。

## 5.x 集群

### 集群

限制类型	限制
集群名称长度	3-64个字符
集群 TPS 上限	不同集群按照规格进行限制，超出会被限流。TPS 按照消息类型和消息大小进行折算，折算规格见 <a href="#">5.x 集群</a>
公网安全策略	50 条

### Topic

限制类型	体验版集群限制	基础版集群限制	专业版集群限制	铂金版集群限制
单集群内 Topic 数量上限	50，死信队列和重试队列对应的 Topic 不占用配额	100-999，按不同的集群规格计算，控制台可以自助提升额度，详细参见 <a href="#">5.x 集群</a> ，死信队列和重试队列对应的 Topic 不占用配额	300-999，按不同的集群规格计算，控制台可以自助提升额度，详细参见 <a href="#">5.x 集群</a> ，死信队列和重试队列对应的 Topic 不占用配额	1000-10000，按不同的集群规格计算，控制台可以自助提升额度，详细参见 <a href="#">5.x 集群</a> ，死信队列和重试队列对应的 Topic 不占用配额
Topic 名称长度	3-100个字符	3-100个字符	3-100个字符	3-100个字符
单 Topic 最大生产者客户端数量	1000	1000	1000	1000
单 Topic 最大消	500	500	500	500

费者客户端数量

## Group

限制类型	体验版集群限制	基础版集群限制	专业版集群限制	铂金版集群限制
单集群内 Group 数量上限	500	1000–2000，按不同的集群规格计算，控制台可以自助提升额度，详细参见 <a href="#">5.x 集群</a>	3000–10000，按不同的集群规格计算，控制台可以自助提升额度，详细参见 <a href="#">5.x 集群</a>	10000–100000，按不同的集群规格计算，控制台可以自助提升额度，详细参见 <a href="#">5.x 集群</a>
Group 名称长度	3–64个字符	3–64个字符	3–64个字符	3–64个字符

## 消息

限制类型	体验版集群限制	基础版集群限制	专业版集群限制	铂金版集群限制
消息最大保留时间	1–3天，默认为 3 天，可在集群维度调整		1–7天，默认为 3 天，可在 Topic 维度调整	
消息最大延时	7天	40天，特殊需求可以通过 <a href="#">工单咨询</a>	40天，特殊需求可以通过 <a href="#">工单咨询</a>	可定制，最长1年
消息大小	4 MB	4 MB	最大 4 MB，特殊需求可以通过 <a href="#">工单咨询</a>	可定制
消费位点重置	默认 3天，和消息保留时间一致			

## 4.x 集群

### 集群

限制类型	虚拟集群限制	专享集群限制	通用集群限制

单地域内集群数量上限	10个	不限制	不限制
集群名称长度	3-64个字符	3-64个字符	3-64个字符
集群 TPS 上限	4000	2000 以上，按不同的节点规格计算	8000 以上，按不同的 TPS 规格计算
公网安全策略	不支持	50 条	50 条

## 命名空间

限制类型	虚拟集群限制	专享集群限制	通用集群限制
单集群内命名空间数量上限	10个	10个	不涉及
命名空间名称长度	3-32个字符	3-32个字符	不涉及

## Topic

限制类型	虚拟集群限制	专享集群限制	通用集群限制
单集群内 Topic 数量上限	150，死信队列和重试队列对应的 Topic 不占用配额	200-500，按不同的集群规格计算，死信队列和重试队列对应的 Topic 不占用配额	默认为 400，按不同的集群规格计算，死信队列和重试队列对应的 Topic 不占用配额
Topic 名称长度	3-64个字符	3-64个字符	3-64个字符
单 Topic 最大生产者数量	1000	1000	1000
单 Topic 最大消费者数量	500	500	500

## Group

限制类型	虚拟集群限制	专享集群限制	通用集群限制
单集群内 Group 数量上限	1500	2000-5000，按不同的集群规格计算	默认为 4000，按不同的集群规格计算
Group 名称长度	3-64个字符	3-64个字符	3-64个字符

## 消息

限制类型	虚拟集群限制	专享集群限制	通用集群限制
消息最大保留时间	3天	默认 3天，支持集群粒度调整	默认 3天，支持集群粒度调整
消息最大延时	40天	40天，特殊需求可以通过 <a href="#">工单咨询</a>	40天，特殊需求可以通过 <a href="#">工单咨询</a>
消息大小	4 MB	最大 4 MB，特殊需求可以通过 <a href="#">工单咨询</a>	最大 4 MB，特殊需求可以通过 <a href="#">工单咨询</a>
消费位点重置	3天	默认 3天，和消息保留时间一致	默认 3天，和消息保留时间一致

# 开服地域

最近更新时间：2026-01-23 16:23:17

地域（Region）是指物理的数据中心的地理区域。可用区（Zone）是指腾讯云在同一地域内电力和网络互相独立的物理数据中心。详细介绍请参见 [云服务器-地域和可用区](#)。

## 支持的地域

### 中国地区

地域	取值	5.x 集群	4.x 专享集群	4.x 通用集群	4.x 虚拟集群 (停止新购)	
华南地区	广州	ap-guangzhou	✓	✓	✓	✓
	清远	ap-qingyuan	×	✓	✓	×
	深圳金融	ap-shenzhen-fsi	✓	×	×	×
华东地区	南京	ap-nanjing	✓	✓	✓	✓
	上海	ap-shanghai	✓	✓	✓	✓
	上海金融	ap-shanghai-fsi	✓	✓	✓	✓
	上海自动驾驶云	ap-shanghai-adc	✓	✓	✓	✓
港澳台地区	中国香港	ap-hongkong	✓	✓	✓	✓
华北地区	北京	ap-beijing	✓	✓	✓	✓

	北京金融	ap-beijing-fsi	×	×	×	✓
西南地区	成都	ap-chengdu	✓	×	×	×
	重庆	ap-chongqing	✓	✓	×	×

## 其他国家和地区

地域	取值	5.x 集群	4.x 专享集群	4.x 通用集群	4.x 虚拟集群 (停止新购)	
亚太东南	新加坡	ap-singapore	✓	✓	✓	✓
	曼谷	ap-bangkok	✓	×	×	×
	雅加达	ap-jakarta	✓	×	✓	✓
亚太东北	首尔	ap-seoul	✓	×	×	✓
	东京	ap-tokyo	✓	×	×	✓
美国西部	硅谷	na-siliconvalley	✓	✓	✓	✓
美国东部	弗吉尼亚	na-ashburn	✓	✓	✓	✓
欧洲地区	法兰克福	eu-frankfurt	✓	×	×	✓
南美地区	圣保罗	sa-saopaulo	✓	×	×	×

如以上地域不满足您的需求，您可以 [提交工单](#) 申请开通新的地域和可用区。

## 如何选择地域和可用区

关于选择地域和可用区时，您需要考虑以下几个因素：

- 消息队列 RocketMQ 集群所在的地域、您以及您的目标用户所在的地理位置。建议您在购买消息队列 RocketMQ 集群时，选择最靠近您的客户地域，以降低访问时延、提高访问速度。
- 消息队列 RocketMQ 和其他云产品的关系。建议您在选择其他云产品时，尽量都在同个地域同个可用区，以便各云产品之间可通过内网进行通信，降低访问时延、提高访问速度。
- 业务高可用和容灾考虑。即使只有一个私有网络的场景下，建议您将业务至少部署在不同的可用区，以保证可用区间的故障隔离，实现跨可用区容灾。
- 不同可用区间可能会有网络的通信延迟，需要结合业务的实际需求进行评估，在高可用和低延迟之间找到最佳平衡点。

# 基本概念

最近更新时间：2026-01-23 16:23:17

本文列举了 TDMQ RocketMQ 版中常见的概念与定义。

## 广播消费 (Broadcasting Consumption)

当使用广播消费模式时，每条消息会被推送到集群内所有注册过的消费者，保证消息至少被每个消费者消费一次。适用于每条消息需要被集群下每一个消费者处理的场景。

## 集群 (Cluster)

集群是 TDMQ RocketMQ 版中的一个资源维度，不同集群的 Topic、Group 等资源完全隔离。每个集群会有集群的资源限制例如 Topic 总数、消息保留时长等

## 集群消费 (Clustering Consumption)

当使用集群消费模式时，任意一条消息只需要被集群内的任意一个消费者处理即可。适用于每条消息只需要被处理一次的场景。

## 消费者 (Consumer)

消费者是 RocketMQ 中用来接收并处理消息的运行实体。消费者通常被集成在业务系统中，从服务端获取消息，并将消息转化成业务可理解的信息，供业务逻辑处理。

## 消费位点 (ConsumerOffset)

一条消息被某个消费者消费完成后不会立即从队列中删除，RocketMQ 会基于每个消费者分组记录消费过的最新一条消息的位点，即消费位点。

## 死信队列 (Dead-Letter Queue)

死信队列是一种特殊的消息队列，用于集中处理无法被正常消费的消息的队列。当消息在重试队列中达到一定重试次数后仍未能被正常消费，TDMQ 会判定这条消息在当前情况下无法被消费，将其投递至死信队列。

实际场景中，消息可能会由于持续一段时间的服务宕机，网络断连而无法被消费。这种场景下，消息不会被立刻丢弃，死信队列会对这种消息进行较为长期的持久化，用户可以在找到对应解决方案后，创建消费者订阅死信队列来完成对当时无法处理消息的处理。

## 消息过滤 (Message Filtering)

消费者可以通过订阅指定消息标签 (Tag) 对消息进行过滤，确保最终只接收被过滤后的消息合集。过滤规则的计算和匹配在 RocketMQ 的服务端完成。

## 分组 (Group)

可分为生产者组和消费者组：

- 生产者组：同一类 Producer 的集合，这类 Producer 发送同一类消息且发送逻辑一致。如果发送的是事务消息，且生产者发送后崩溃，则 Broker 服务器会联系同一个生产者组的其他生产者实例以提交或者回溯消费。
- 消费者组：同一类 Consumer 的集合，这类 Consumer 通常消费同一类消息且消费逻辑一致。消费者组使得在消息消费方面实现了负载均衡和容错。消费者组的消费者实例必须订阅完全相同的 Topic。

## 消息索引 (MessageKey)

消息索引是 RocketMQ 提供的面向消息的索引属性。通过设置的消息索引可以快速查找到对应的消息内容。

## 消息 (Message)

消息系统所传输信息的物理载体，生产和消费数据的最小单位。生产者将业务数据的负载和拓展属性包装成消息发送到服务端，服务端按照相关语义将消息投递到消费端进行消费。

## 消息堆积 (Message Backlog)

生产者已经将消息发送到 RocketMQ 的服务端，但由于消费者的消费能力有限，未能在短时间内将所有消息正确消费掉，此时在服务端保存着未被消费的消息，该状态即消息堆积。一分钟统计一次，如果消息已经过了保留时间（默认3天），那么这部分消息是不会再计算在堆积里面了（因为服务端已经进行了删除）。

## 消息队列 (MessageQueue)

存储消息的物理实体，一个 Topic 可以包含多个 Queue，Queue 也叫消息分区，一个 Queue 中的消息只能被一个消费者组中的一个消费者消费，一个 Queue 中的消息不允许同一个消费者组中的多个消费者同时消费。

## 消息位点 (MessageQueueOffset)

消息是按到达 RocketMQ 服务端的先后顺序存储在指定主题的多个队列中，每条消息在队列中都有一个唯一的 Long 类型坐标，这个坐标被定义为消息位点。

## 消息保留策略 (Message Retention Policy)

定义了消息在 RocketMQ 服务端的持久化时间，超过保留时间的消息将被自动清理。

## 消息标签 (MessageTag)

为消息设置的标签，用于同一个 Topic 下区分不同类型的消息，可以理解为 Topic 是消息的一级分类，Tag 是消息的二级分类。

## 消息轨迹 (Message Trace)

在一条消息从生产者发出到消费者接收并处理过程中，由各个相关节点的时间、地点等数据汇聚而成的完整链路信息。通过消息轨迹，您能清晰定位消息从生产者发出，经由 RocketMQ 服务端，投递给消费者的完整链路，方便定位排查问题。

## 消息类型 (MessageType)

按照消息传输特性的不同而定义的分类，用于类型管理和安全校验。RocketMQ 支持的消息类型有普通消息、顺序消息、事务消息和定时/延时消息。

## 消息查询 (Message Query)

腾讯云 TDMQ RocketMQ 版控制台和 SDK 提供了强大的消息查询功能，支持通过 Message ID、Message Key 或时间范围来查询消息，对于业务排查和问题定位至关重要。

## 命名空间 (Namespace)

用于对消息主题 (Topic)、资源分组等进行逻辑隔离的单元。在腾讯云 TDMQ RocketMQ 版中，命名空间是资源管理和权限控制的重要边界。

## 普通消息 (Normal Message)

普通消息是一种基础的消息类型，由生产者投递到指定 Topic 后，被订阅了该 Topic 的消费者所消费。普通消息的 Topic 中无顺序的概念，可以使用多个分区数来提升消息的生产和消费效率，在吞吐量巨大时其性能最好。

## 顺序消息 (Ordered Message)

顺序消息是消息队列 RocketMQ 提供的一种高级消息类型，对于一个指定的 Topic，消息严格按照先进先出 (FIFO) 的原则进行消息发布和消费，即先发送的消息先消费，后发送的消息后消费。

## 生产者 (Producer)

生产者是 RocketMQ 系统中用来构建并传输消息到服务端的运行实体。生产者通常被集成在业务系统中，将业务消息按照要求封装成消息并发送至服务端。

## 重试队列 (Retry Queue)

重试队列是一种为了确保消息被正常消费而设计的队列。当某些消息第一次被消费者消费后，没有得到正常的回应，则会进入重试队列，当重试达到一定次数后，停止重试，投递到死信队列中。

由于实际场景中，可能会存在的一些临时短暂的问题（如网络抖动，服务重启等）导致消息无法及时被处理，但短暂时间过后又恢复正常。这种场景下，重试队列的重试机制就可以很好解决此类问题。

## 重置消费位点 (Reset Consumer Offset)

以时间轴为坐标，在消息持久化存储的时间范围内，重新设置消费者分组对已订阅主题的消费进度。设置完成后消费者将接收设定时间点之后的消息。

## 定时/延时消息 (Scheduled/Delayed Message)

生产者发送消息后，消息不会立即被消费者获取，而是会在特定的时间点（或延迟特定时间后）才会被投递给消费者。

## 主题 (Topic)

Topic 表示一类消息的集合，每个主题包含若干消息，是 RocketMQ 进行消息订阅的基本单位。

## 事务消息 (Transactional Message)

支持分布式事务的一种消息类型。TDMQ RocketMQ 版提供了完整的事务消息解决方案，确保分布式系统下的数据最终一致性。

## VPC 网络接入 (VPC Access)

指您的生产者/消费者客户端与 TDMQ RocketMQ 服务端之间的网络连接方式。建议客户端与 TDMQ 实例位于同一地域的同一 VPC 内，以获得最佳的网络性能和安全性。