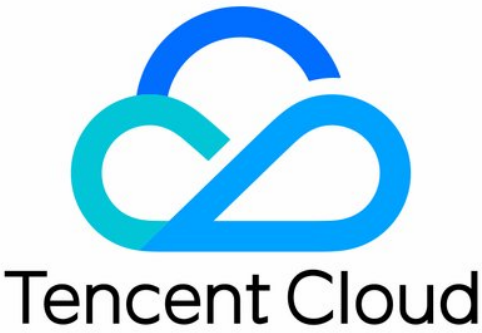
TDMQ for RocketMQ

Product Introduction

Product Documentation



Contents

Product Introduction

Overview

Last updated：2025-07-15 17:42:39

TDMQ for RocketMQ is a distributed high-availability message queue service. Built upon the infrastructure of Apache RocketMQ 4.x and 5.x, it offers various product forms that allow clients of version 4.4.x or later to connect without modifications. It also possesses the underlying capabilities of computing-storage separation and flexible scalability. TDMQ for RocketMQ supports a throughput of up to millions of TPS, making it ideal for various large-scale, low-latency, and reliability-demanding online messaging scenarios.

No-Ops Management

Supports one-click cluster creation, ready to use.
Rapid scaling by TPS specification, offering ultimate elasticity.
5.x TPS supports elastic interval for handling sudden traffic spikes.
5.x storage is Serverless, supporting on-demand use and pay-as-you-go billing.

Feature Rich

Support various message types: transactional, scheduled, delay, sequential messaging.
Support multiple consumption modes: Tag filtering, SQL 92 filtering, cluster consumption, broadcast consumption.

Observability

Provide comprehensive monitoring metrics to quickly identify issues.
Support visualization of message trace, connect downstream business, and better troubleshoot and locate issues.

High Availability

Support cross-AZ deployment to defend against IDC-level faults.
Containerization enables automatic restart in seconds, with no capacity or data damage during node downtime.
Support cluster rate limiting protection to enhance cluster health.
Support cross-cluster message replication to meet disaster recovery requirements.

Security Control

Integrate Tencent Cloud's CAM system with support for operation/resource level permission management.
Visual ACL management page with support for sending and receiving message permission management.
Public network access supports allowlist control.

Smooth Migration

Open-source compatibility, zero business code modification.

Provide a data migration tool to reduce migration costs.

Provide a cluster smooth migration solution with low intrusion and rollback capability to reduce business intrusion.

Differences Between Version 4.x and 5.x

Last updated：2025-07-15 17:42:39

TDMQ for RocketMQ 5.x launched by Tencent Cloud is a next-generation message queue product based on Apache RocketMQ 5.x. It draws on the architectural and technological innovations of the community and practical tutorials of Tencent Cloud's large-scale operations on the message queue products over the years. It is primarily tailored to online business message scenarios, boasting low latency, high availability, high reliability, and scalability. Compared with TDMQ for RocketMQ 4.x of Tencent Cloud and open-source cluster products developed by customers, TDMQ for RocketMQ 5.x has the following advantages:

Item	TDMQ for RocketMQ 5.x	TDMQ for RocketMQ 4.x	Self-built Open-Source RocketMQ 4.x Products of Customers
SDK Compatibility	Compatible with SDKs of Apache RocketMQ 4.x and 5.x.	Compatible with Apache RocketMQ 4.x SDK.	Compatible with Apache RocketMQ 4.x SDK.
Overall Architecture	An architecture that separates compute and storage. Compute resources and storage pools can be automatically or manually scaled swiftly in response to traffic demands.	An architecture that integrates compute with storage.	An architecture that integrates compute with storage for most products.
Storage Elasticity	Storage pools with no quota limit are provided and are charged based on usage.	Storage pools that are charged based on quotas are provided. The storage capacity is limited.	No resource pool.
Computing Elasticity	Automated elasticity. The elastic bandwidth feature is provided for the Pro and Platinum Editions.	Online horizontal or vertical scaling.	Scaling must be performed by O&M personnel manually.
Monitoring Metrics	Over 100 metrics that are not available in the RocketMQ open-source community are provided. More metrics at the topic and group levels are available.	Over 50 metrics that are not available in the RocketMQ open-source community are provided.	External monitoring metrics built by customers are used generally.

O&M Complexity	O&M is not required, and automatic scaling is supported.	O&M is not required, and manual scaling is supported.	O&M is conducted via command-line operations, which is complex and costly and requires a high level of professional knowledge.
Billing Mode	Billing is based on specifications, traffic bandwidth, and actual storage usage. The overall costs can be reduced by 30%.	Billing is based on the instance and disk specifications or the number of API calls.	Billing is based on the occupied resources.
Usage Costs	The product is out-of-the-box service and provides support from a professional O&M team.	The product is out-of-the-box service and provides support from a professional O&M team.	O&M is conducted by customers.
Enhanced Features	Out-of-the-box advanced features, including message tracing, grayscale capabilities, and diagnostic capabilities, are provided. The service is undergoing rapid iterations to provide more new features.	Out-of-the-box advanced features, including message tracing, grayscale capabilities, and diagnostic capabilities, are provided.	Customers need to develop advanced features by themselves, which requires a high level of professional knowledge. The code maintenance is challenging due to differences from community standards.
Disaster Recovery Capability	Out-of-the-box disaster recovery capabilities are provided. By default, cross-AZ disaster recovery is adopted. Other schemes, such as remote disaster recovery, multi-site active-active, and united deployment, are also provided for customers to consult about.	Out-of-the-box disaster recovery capabilities are provided. By default, cross-AZ disaster recovery is adopted. Other schemes, such as remote disaster recovery and multi-site active-active, are also provided for customers to consult about.	Customers need to implement disaster recovery capabilities by themselves, which requires a high level of professional knowledge about message queues.
SLA Guarantee	Enterprise-level SLA guarantee provided: service availability of 99.99% and	Enterprise-level SLA guarantee provided: service availability of 99.99% and	Customers need to guarantee the SLA by themselves, which requires a high level of

	storage reliability of 99.9999999%.	storage reliability of 99.9999999%.	professional knowledge about message queues.
Business Support	24/7 support from the expert team of TDMQ for RocketMQ is provided. Training services are provided for customers who purchase the Platinum Edition.	24/7 support from the expert team of TDMQ for RocketMQ is provided.	Customers provide support by themselves.

Strengths

Last updated：2025-07-15 17:42:39

Advantages

Open-Source Compatibility

100% compatible with Apache RocketMQ components and concepts, it supports zero-modification client access for RocketMQ 4.4.x and above versions, with the underlying advantages of compute-storage separation and flexible scaling.

Various Specification Series

Offers multiple sales specifications to cover different business scales. Supports flexible billing methods such as actual message volume, hourly billing, and annual/monthly subscription. Allows flexible scaling between specifications to meet product implementation needs, from small-scale tests to large-scale applications.

Adapting to Cloud Native

The underlying architecture undergoes cloud native transformation and upgrade. The Serverless product form brings ultimate elasticity, supporting elastic computing capability beyond specifications, with storage billed based on actual usage.

Rich Message Types

It supports multiple message types such as general messages, sequential messaging, delayed messages, and distributed transaction messages, along with support for message retry and dead letter mechanisms, meeting various business scenarios.

High performance

A single machine can support up to tens of thousands of production and consumption throughputs. Based on a distributed architecture and stateless services, it can be horizontally scaled out to enhance the throughput of the entire cluster.

Easy to Use and Maintenance-Free

It provides API access APIs and supports open-source SDKs for all languages and versions. It offers a complete set of operation and maintenance services on Tencent Cloud Platform, with real-time monitoring and alerting to help users quickly discover and solve problems, ensuring service availability.

5.X Series Exclusive Advantage

RocketMQ 5.X series not only retains the characteristics of earlier versions such as low delay, high performance, high reliability, trillion-level message capacity, and flexible scalability, but also fully combines cloud native infrastructure and ecological technology to improve resource utilization and elastic capability.
Compared to self-built RocketMQ and Tencent Distributed Message Queue (TDMQ) RocketMQ 4.x series, the 5.x series has the following advantages:

Storage and Computing Elasticity

TDMQ RocketMQ 5.x series adopts a storage and computing separation architecture, significantly improving resource usage rate and elastic capability. Storage is billed on a pay-as-you-go basis, eliminating the need for customers to reserve storage resources in advance for peak demand. Payment is based on actual usage, effectively reducing customer's actual cost. The specification supports elastic TPS, allowing customers to avoid reserving computational resources for unexpected peaks, thus reducing cost.

Lightweight SDK

TDMQ RocketMQ 5.x series is compatible with the open-source community SDK, allowing developers to enjoy the benefits of open-source community iteration. The 5.x series clients are more lightweight, featuring a brand-new minimalist API design for easier integration and use. Meanwhile, the 5.x series provides SDKs in more languages, making more technology stack options available to developers.

Basic Feature Enhance

TDMQ RocketMQ 5.x series has further feature enhancements, such as more flexible message retention time control, where retention time can be set based on cluster or Topic granularity. Consumer groups now offer more white-screen settings, such as specifying message retry count and free binding to DLQ on the server side.

Observability

TDMQ RocketMQ 5.x series adds various metrics, such as message backlog scenario metrics, key API time taken metrics, error distribution metrics, and storage read-write traffic metrics. These metrics can interconnect with the cloud platform’s monitoring and alarm features, while providing complete TencentCloud API to support integration with self-service ops systems.

Use Cases

Last updated：2025-07-15 17:42:40

TDMQ for RocketMQ is a distributed message middleware based on Apache RocketMQ. It is applied to message communication between distributed systems or components. It has the characteristics of massive message heap, low latency, high throughput, high reliability, and strong transaction consistency, which meets the requirements of async decoupling, peak shifting, sequential sending and receiving, distributed transaction consistency, and log sync.

Async Decoupling

The transaction engine is the core system of Tencent billing. The data of each transaction order needs to be monitored by dozens of downstream business systems, including inventory system, warehousing system, promotion system, and points system. Such systems use different message processing logic, making it impossible for a single system to adapt to all associated business. In this case, TDMQ for RocketMQ can decouple the coupling between multiple business systems to reduce the impact between systems and improve the response speed and robustness of core business.

Peak Shifting

Companies hold promotional campaigns such as new product launch and festival red packet grabbing from time to time, which often cause temporary traffic spikes and pose huge challenges to each backend application system. In this case, TDMQ for RocketMQ can withstand spikes in traffic. It heaps up messages during peak periods and consumes them in the downstream during off-peak periods, which balances the processing capacities of upstream and downstream systems and improve system availability.

Subscription Notifications

TDMQ for RocketMQ provides scheduled and delayed messages to can meet the ecommerce subscription notification scenarios.

Scheduled message: After a message is sent to the server, the business may want the consumer to receive it at a later time point rather than immediately. This type of message is called "scheduled message".

Delayed message: After a message is sent to the server, the business may want the consumer to receive it after a period of time rather than immediately. This type of message is called "delayed message".

For details about scheduled and delayed messages, see Scheduled Message and Delayed Message.

Consistency of Distributed Transactions

TDMQ for RocketMQ provides distributed transactional messages to loosely couple applications. Reliable transmission and multi-replica technology can ensure that messages are not lost, and the At-Least-Once feature

ensures eventual data consistency.
As a producer, the payment system forms a transaction with the message queue to ensure the consistency of local transactions and message sending.
Downstream business systems (bills, notifications, others) work as consumers to process in parallel.
Messages support reliable retries to ensure eventual data consistency.
The transaction messages of TDMQ for RocketMQ can be used to process transactions, which can greatly improve processing efficiency and performance. A billing system often has a long transaction linkage with a significant chance of error or timeout. TDMQ's automated repush and abundant message retention features can be used to provide transaction compensation, and the eventual consistency of payment tips notifications and transaction pushes can also be achieved through TDMQ for RocketMQ.
For details about transactional messages, see Transactional Message.

Sequential Message Sending/Receiving

Sequential message is an advanced message type provided by TDMQ for RocketMQ. For a specified topic, messages are published and consumed in strict accordance with the principle of First-In-First-Out (FIFO), that is, messages sent first are consumed first, and messages sent later are consumed later. Sequential messages are often used in the following business scenarios:
Order creation: In some ecommerce systems, an order's creation, payment, refund, and logistics messages must be produced or consumed in strict sequence, otherwise the order status will be messed up during consumption, which will affect the normal operation of the business. Therefore, the messages of this order must be produced and consumed in a certain sequence in the client and message queue. At the same time, the messages are sequentially dependent, and the processing of the next message must be dependent on the processing result of the preceding message.
Log sync: In the scenario of sequential event processing or real-time incremental data sync, sequential messages can also play a greater role. For example, it is necessary to ensure that database operations are in sequence when MySQL binlogs are synced.
Financial scenarios: In some matchmaking transaction scenarios like certain securities transactions, the first bidder is given priority in the case of the same biding price, so it is necessary to produce and consume sequential messages in a FIFO manner.
For details about sequential messages, see Sequential Message.

Distributed Cache Sync

During sales and promotions, there are a wide variety of products with frequent price changes. When users query item prices multiple times, the cache server's network interface may be fully loaded, which makes page opening slower. After the broadcast consumption mode of TDMQ for RocketMQ is adopted, a message will be consumed by all nodes once, which is equivalent to syncing the price information to each server as needed in place of the cache.

Architecture

Last updated：2025-07-15 17:42:40

This document describes the deployment architecture of TDMQ for RocketMQ to help you better understand its architectural principles.

Deployment Architecture

The system deployment architecture of TDMQ for RocketMQ is shown in the following diagram:
5.x cluster
4.x cluster

The TDMQ for RocketMQ 5.x introduces the new gRPC protocol and Proxy components, implementing an architecture featuring separation of computation and storage separation. This significantly changes both the Ops and usage of RocketMQ.
The concepts involved are as follows:
Producer cluster: A client-side application responsible for producing and sending messages.
Consumer cluster: A client-side application responsible for message subscription and consumption.
NameServer cluster: A server-side application responsible for routing address location and Broker heartbeat registration. To ensure high availability, it is deployed across AZs by default.
Heartbeat registration: The NameServer acts as a registration center. Machines of each role must report its status to the NameServer regularly. If a machine does not report within a certain time window, the NameServer will presume it to be faulty and remove it from the availability list.
Route addressing: Every NameServer stores both the complete routing information of the Broker cluster and the queue information for client queries. Producers and consumers use the NameServer to acquire route information of the entire Broker cluster, which then allows for message delivery and consumption.
Proxy cluster: A brand-new resilient stateless proxy service deployed across AZs by default to ensure high availability. It splits the responsibilities of the 4.x Broker, extracting computation logic such as client protocol adaptation, permission management, and consumption managemt.
Broker cluster: Compared with the 4.x series, the Broker in the 5.x series is more focused on the continuous enhancements of storage capabilities. To ensure high availability, it is deployed across AZs by default.

The core concepts are as follows:
Producer cluster: Client-side application, which is responsible for producing and sending messages.
Consumer cluster: Client-side application, which is responsible for subscribing to and consuming messages.

Nameserver cluster: server-side application, which is responsible for address routing and broker heartbeat registration.
Heartbeat registration: Nameserver acts as the registration center. Machines in each role must regularly report their status to Nameserver. If a machine fails to report beyond the timeout period, Nameserver will consider it faulty and unavailable and remove it from the available list.
Address routing: Each Nameserver stores the entire routing information of the Broker cluster and the queue information used for client queries. Producers and consumers obtain the routing information of the entire Broker cluster through Nameserver to deliver and consume messages.
Broker cluster: Server application, which is responsible for receiving, storing, and delivering messages. It supports primary-secondary multi-copy mode where the deployment of secondary nodes is optional. The actual high reliability of data in the production environment on the public cloud directly depends on the three copies of the cloud disk.
Management cluster: Server application that is a visual management and control console. It is responsible for operating the entire cluster, such as source data sending/receiving and management.

Comparison with Apache RocketMQ

Last updated：2025-07-15 17:42:40

The performance comparison between TDMQ for RocketMQ and Apache RocketMQ is detailed below:

5.x cluster

4.x cluster

Feature Category	Feature Item	Tencent Cloud TDMQ RocketMQ 5.X	Open-source RocketMQ
Security Control	Message sending/receiving ACL management	Support fine-grained role refinement for producing and sending messages, with more granular permission refinement.	Default ACL authentication method with single authentication.
	root account and sub-account	Fully support Tencent Cloud root and sub-accounts, enabling CAM root and sub-accounts, cross-account authorization, and other services within the enterprise.	Not supported
	Scaling	Customers do not need to care about the size of underlying machines or perform scaling. They only need to purchase appropriate specifications based on business volume and freely adjust specifications as needed. Underlying resources achieve elastic scaling through containerization and automated intelligent ops.	Dependent on a self-built O&M team, with a low level of automation and white-screen issues.
	High Availability	Tencent Cloud ensures high availability at the underlying cloud level. Customers do not need to worry about deploying disaster recovery architecture.	Self-deployment of a high availability architecture increases operation and maintenance difficulty.
Migration Tools	Migration Tools	Simultaneously support two migration methods: migrate by importing self-built metadata and changing the access point, and support Seamless Migration with white-screen operation, performing grayscale migration by different Topics in phases.	Migration is not supported.

Monitoring Alarm	Resource Dashboard	Core metric observation, production and consumption reports, and fine-grained monitoring support drilldown at public network, cluster, topic, group, and topic&group levels; resource Top ranking is provided by cluster granularity.	Simple support, with some monitoring metrics missing.
	Alarm Management	Message backlog, delay and multiple metric alarms, Cloud Monitor linkage.	Not supported
elasticity	elastic TPS	Pro Edition and Platinum Version support enabling temporary elastic traffic beyond normal traffic specification.	Not supported. On-demand scaling of machines is required, raising operational complexity.
	Storage without quota limit	Storage is charged by actual usage.	Not supported. Specified disk size is required at deployment time. Oversized disks may lead to resource waste, and cloud-purchased disks do not support scale-down. Undersized disks require frequent scaling, increasing operation and maintenance difficulty.
Stability	Distributed throttling	Supports various rate limiting methods and policies (global/local, downgrade policy, multiple rule priorities), effectively avoiding downtime caused by excessively large traffic.	Unsupported. Existence of traffic overload may cause downtime.
	Message sending and receiving ratio adjustment	Support adjusting the message sending and receiving ratio at the cluster level and perform traffic throttling separately for more reasonable resource utilization.	Not supported
Message Lifecycle	Topic Level save time setting	Support setting message retention time based on Topic to further save on storage costs.	Not supported
Console feature	Client stack view	Support, and support code search within the stack.	Not supported

optimization	Query messages by Tag	Supported	Not supported
	Query retry/dead letter/delayed message.	Query specific type of messages based on specified conditions.	Not supported
Compatibility	S3-compatible history for legacy SDK	Fully compatible with ALL legacy SDKs after 4.1.	Not fully compatible, an error occurred in certain scenarios.

Feature Type	Feature	TDMQ for RocketMQ	Apache RocketMQ
Basic features	Scheduled message	The scheduled time is accurate down to the second and can be customized.	You can only specify the delay level.
	Visual management	Visual management for clusters, topics, and groups is supported. You can view the details of subscriptions and consumer status.	Visual management is supported but is less user-friendly. The console doesn't distinguish between topic types.
Availability	Elastic scaling	You don't need to manually deploy, configure, or scale up underlying computing resources because operations such as node registration are automatically performed in a visual manner. You can expand the number of nodes horizontally, increase the disk capacity, and upgrade the configurations of a single node vertically as needed at any time.	A self-built Ops team is required, and operations are performed in a less automatic or visualized manner.
	High reliability	With three data replicas, the server can be automatically restarted in seconds after the downtime, without affecting the message capacity and data.	Data can be replicated in sync or async mode. You need to design the deployment scheme and related parameters. The primary sync schemes won't be automatically used after the failover.
	Cross-AZ high-availability deployment	This feature is supported to avoid losses caused by data center-level failures.	This feature is supported, but it is time-consuming for you to

			design the deployment schemes and parameters.
Observability	Resource dashboard	You can monitor core metrics at a fine granularity and view production and consumption details.	This feature is supported but with fewer monitoring metrics.
	Alarming	With the capabilities provided by Cloud Monitor, alarms will be triggered in case of message heap or delayed message sending/receiving.	Not supported
Security management and control	Tenant namespace isolation	You can implement this feature in the console in a visual manner.	This feature is not supported. Namespaces cannot be truly isolated due to bugs.
	Root account and sub-account management	Supports authorization between Tencent Cloud CAM root accounts and sub-accounts and between enterprise accounts.	Not supported
Migration tool	Tool for migrating from Apache RocketMQ	You can easily migrate from Apache RocketMQ to TDMQ for RocketMQ by using scripts.	-

Use Limits

Last updated：2025-07-15 17:42:40

This document lists the limits of certain metrics and performance in TDMQ for RocketMQ. Be careful not to exceed the limits during use so as to avoid exceptions.

5.x Cluster

4.x Cluster

Cluster

Limit Type	Limit
Cluster name length	Enter 3 to 64 characters.
Cluster TPS cap	Different clusters are limited by specification, and exceeding the limit will result in rate-limiting. TPS is converted based on message type and message size. For conversion specifications, see Billing Instructions.
public network security policy	50

Topic

Limit Type	Trial Version Cluster Restrictions	Basic Version Cluster Restrictions	Professional Edition Cluster Limit	Platinum Edition Cluster Limit
Single-cluster topic cap	The corresponding Topic for DLQ and retry queue does not occupy quota.	100-999. The limit can be self-service increased in the console based on different cluster specifications. For details, see Billing Instructions. The	300-999. The limit can be self-service increased in the console based on different cluster specifications. For details, see Billing Instructions. The corresponding Topic for DLQ and retry queue does not occupy quota.	1000-10000. The limit can be self-service increased in the console based on different cluster specifications. For details, see Billing Instructions. The corresponding Topic for DLQ and retry queue does not occupy quota.

		corresponding Topic for DLQ and retry queue does not occupy quota.		
Topic name length	Enter 3 to 100 characters.	Enter 3 to 100 characters.	Enter 3 to 100 characters.	Enter 3 to 100 characters.
Single-topic producer limit	1000	1000	1000	1000
Single-topic consumer limit	500	500	500	500

Group

Limit Type	Trial Version Cluster Restrictions	Basic Version Cluster Restrictions	Professional Edition Cluster Limit	Platinum Edition Cluster Limit
Single-cluster group cap	500	1000-2000. The limit can be self-service increased in the console based on different cluster specifications. For details, see Billing Instructions.	3000-10000. The limit can be self-service increased in the console based on different cluster specifications. For details, see Billing Instructions.	10000-100000. The limit can be self-service increased in the console based on different cluster specifications. For details, see Billing Instructions.
Group name length	Enter 3 to 64 characters.	Enter 3 to 64 characters.	Enter 3 to 64 characters.	Enter 3 to 64 characters.

Message

Limit Type	Trial Version Cluster Restrictions	Basic Version Cluster Restrictions	Professional Edition Cluster Limit	Platinum Edition Cluster Limit
Maximum	1-3 days, defaults to 3 days, can be		1-7 days, defaults to 3 days, can be	

message retention time	adjusted at the cluster level		adjusted at the Topic level	
Maximum message delay	7 days	40 days. For special requirements, submit a ticket.	40 days. For special requirements, submit a ticket.	Customizable, up to 1 year
Message Size	4 MB	4 MB	Maximum 4 MB. For special requirements, submit a ticket.	customizable
Consumer offset reset	Default 3 days, same as message retention time			

Cluster

Limit Type	Virtual Cluster Restrictions	Dedicated Cluster Restrictions	Common Cluster Restrictions
Single-region cluster cap	10	No limit	No limit
Cluster name length	Enter 3 to 64 characters.	Enter 3 to 64 characters.	Enter 3 to 64 characters.
Cluster TPS cap	4000	Over 2000, calculated by node specification	Over 8000, calculated by TPS specification
public network security policy	Unsupported	50	50

Namespace

Limit Type	Virtual Cluster Restrictions	Dedicated Cluster Restrictions	Common Cluster Restrictions
Single-cluster namespace cap	10	10	Not involved.
Namespace name length	Enter 3 to 32 characters.	Enter 3 to 32 characters.	Not involved.

Topic

Limit Type	Virtual Cluster Restrictions	Dedicated Cluster Restrictions	Common Cluster Restrictions
Single-cluster topic cap	The corresponding Topic for DLQ and retry queue does not occupy quota.	200-500, calculated by cluster specification. The corresponding Topic for DLQ and retry queue does not occupy quota.	400 by default, calculated by cluster specification. The corresponding Topic for DLQ and retry queue does not occupy quota.
Topic name length	Enter 3 to 64 characters.	Enter 3 to 64 characters.	Enter 3 to 64 characters.
Single-topic producer limit	1000	1000	1000
Single-topic consumer limit	500	500	500

Group

Limit Type	Virtual Cluster Restrictions	Dedicated Cluster Restrictions	Common Cluster Restrictions
Single-cluster group cap	1500	2000-5000, calculated by cluster specification	Defaults to 4000, calculated by cluster specification
Group name length	Enter 3 to 64 characters.	Enter 3 to 64 characters.	Enter 3 to 64 characters.

Message

Limit Type	Virtual Cluster Restrictions	Dedicated Cluster Restrictions	Common Cluster Restrictions
Maximum message retention time	3 days	Defaults to 3 days, adjustable at the cluster level	Defaults to 3 days, supports cluster-level adjustment
Maximum message delay	40 days	40 days. For special requirements, submit a ticket.	40 days. For special requirements, submit a ticket.
Message Size	4 MB	Maximum 4 MB. For	Maximum 4 MB. For special

		special requirements, submit a ticket.	requirements, submit a ticket.
Consumer offset reset	3 days	Default 3 days, same as message retention time	Default 3 days, same as message retention time

Service Regions

Last updated：2025-07-15 17:42:40

A Region refers to the geographical location of a physical data center. An AZ (Availability Zone) is a physical data center with independent power and network within the same region in Tencent Cloud. For more details, see Cloud Virtual Machine - Regions and AZs.

Supported Regions

China

Region		Value	5.X Cluster	4.X Dedicated Cluster	4.X Generic Cluster	4.X Virtual Cluster Stop New Purchases
South China region	Guangzhou	ap-guangzhou	✓	✓	✓	✓
	Qingyuan	ap-qingyuan	×	✓	✓	×
	Shenzhen Finance	ap-shenzhen-fsi	✓	×	×	×
East China	Nanjing	ap-nanjing	✓	✓	✓	✓
	Shanghai	ap-shanghai	✓	✓	✓	✓
	Shanghai Finance	ap-shanghai-fsi	✓	✓	×	✓
	Shanghai Autonomous Driving Cloud	ap-shanghai-adc	✓	✓	✓	✓
Hong Kong (China), Macao	Hong Kong (China)	ap-hongkong	✓	✓	✓	✓

(China), and Taiwan (China)						
North China region	Beijing	ap-beijing	✓	✓	✓	✓
	Beijing Finance	ap-beijing-fsi	×	×	×	✓
Southwest China	Chengdu	ap-chengdu	✓	×	×	×
	Chongqing	ap-chongqing	✓	✓	×	×

Other countries and regions

Region		Value	5.X Cluster	4.X Dedicated Cluster	4.X Generic Cluster	4.X Virtual Cluster Stop New Purchases
Southeast Asia Pacific	Singapore	ap-singapore	✓	✓	✓	✓
	Bangkok	ap-bangkok	✓	×	×	×
	Jakarta	ap-jakarta	✓	×	✓	✓
Northeast Asia	Seoul	ap-seoul	✓	×	×	✓
	Tokyo	ap-tokyo	✓	×	×	✓
West US	Silicon Valley	na-siliconvalley	✓	✓	✓	✓
East US	Virginia	na-ashburn	✓	✓	✓	✓
Europe	Frankfurt	eu-frankfurt	✓	×	×	✓
South America	São Paulo	sa-saopaulo	✓	×	×	×

If the above regions do not meet your requirements, you can submit a ticket to apply for new regions and availability zones.

How to Choose Regions and AZs?

When choosing regions and availability zones, you need to consider the following factors:
Message Queue RocketMQ cluster region, geographical location of you and your target users. It is advisable to choose the region closest to your customers when purchasing a Message Queue RocketMQ cluster to reduce access delay and improve access speed.
Relationship between RocketMQ and other cloud products. It is recommended that you choose other cloud services in the same region and availability zone to communicate through the private network, reducing access latency and improving access speed.
High business availability and disaster recovery considerations. Even if there is only one VPC, it is recommended to deploy services across different AZs to ensure fault isolation between availability zones and achieve cross-AZ disaster recovery.
Communications between different availability zones may have network communication delays. This needs to be evaluated in combination with the actual needs of services to find the best balance between high availability and low latency.

Concepts

Last updated：2025-07-15 17:42:40

This document lists the common concepts and their definitions in TDMQ for RocketMQ.

Message (Message)

A message is the physical carrier of information transmitted by the messaging system. It is the smallest unit of the produced or consumed data. A producer encapsulates the load and extended attributes of business data into messages and sends the messages to a TDMQ for RocketMQ broker. Then, the broker delivers the messages to the consumer based on the relevant semantics.

Topic (Topic)

A topic is the collection of a type of messages. It is the basic unit for message subscription in TDMQ for RocketMQ. Each topic contains several messages.

Message Tag (MessageTag)

Tags are used to categorize different types of messages in the same topic. Topic and tag are basically the first-level and second-level classifications of messages, respectively.

Message Queue (MessageQueue)

A message queue (also known as a message partition) is a physical entity for message storage, and a topic can contain multiple queues. Messages in a queue can only be consumed by one consumer rather than multiple consumers in one consumer group.

Message Offset (MessageQueueOffset)

Messages are stored in multiple queues of a specified topic based on the order in which they arrive at the TDMQ for RocketMQ broker. Each message has a unique coordinate of type `Long` in the queue, which is defined as the message offset.

Consumption Offset (ConsumerOffset)

A message is not removed from the queue immediately after it has been consumed by a consumer. TDMQ for RocketMQ will record the offset of the last consumed message based on each consumer group. Such an offset is defined as the consumption offset.

Message Index (MessageKey)

A message index is a message-oriented index property in TDMQ for RocketMQ. By setting the message index, you can quickly find the corresponding message content.

Producer (Producer)

A producer in TDMQ for RocketMQ is a functional messaging entity that creates messages and sends them to the broker. It is typically integrated into the business system to encapsulate data as messages and send them to the broker.

Consumer (Consumer)

A consumer is an entity that receives and processes messages in TDMQ for RocketMQ. It is usually integrated into the business system to obtain messages from TDMQ for RocketMQ brokers and convert the messages into information that can be perceived and processed by business logic.

Group (Group)

Groups include producer groups and consumer groups.
Producer group: It is the collection of the same type of producers that send the same type of messages with the same sending logic. If a producer sends transactional messages and crashes afterward, the broker will contact other producer instances in the producer group to commit or cancel the transaction.
Consumer group: It is the collection of the same type of consumers that consume the same type of messages with the same consumption logic. It can ensure load balancing and fault tolerance in the message consumption process. Consumer instances in a consumer group must subscribe to the same topics.

Message Type (MessageType)

Messages are classified by message transmission characteristic for message type management and security verification. TDMQ for RocketMQ has four message types: general message, sequential message, transactional message, and scheduled/delayed message.

General Message

The general message is a basic message type. After the produced general messages are delivered to a specified topic, they will be consumed by consumers that subscribe to this topic. A topic with general messages is sequence-insensitive. Therefore, you can use multiple topic partitions to improve message production and consumption efficiency. This approach performs best when dealing with high throughput.

Sequential Message

In TDMQ for RocketMQ, the sequential message is an advanced message type. Sequential messages in a specified topic are published and consumed in a First In First Out (FIFO) manner, that is, the first produced messages are first consumed.

Retry Letter Queue

A retry letter queue is designed to ensure that messages are consumed normally. When a message is consumed for the first time by a consumer but is not acknowledged, it will be placed in the retry letter queue and will be retried there until the maximum number of retries is reached. It will then be delivered to the dead letter queue.
In actual scenarios, messages may not be processed promptly due to temporary issues such as network jitter and service restart. The retry mechanism of the retry letter queue can be a good solution in this case.

Dead Letter Queue

A dead letter queue is a special type of message queue used to centrally process messages that cannot be consumed normally. If a message cannot be consumed after a specified number of retries in the retry letter queue, TDMQ will determine that the message cannot be consumed under the current situation and deliver it to the dead letter queue.
In actual scenarios, messages may not be consumed due to service downtime or network disconnection. In this case, they will not be discarded immediately; instead, they will be persistently stored in the dead letter queue. After fixing the problem, you can create a consumer to subscribe to the dead letter queue to process such messages.

Clustering Consumption

Clustering consumption: If the clustering consumption mode is used, each message only needs to be processed by any of the consumers in the cluster. This mode is suitable for scenarios where each message only needs to be processed once.

Broadcasting Consumption

Broadcasting consumption: If the broadcasting consumption mode is used, each message will be pushed to all registered consumers in the cluster to ensure that the message is consumed by each consumer at least once. This mode is suitable for scenarios where each message needs to be processed by each consumer in the cluster.

Message Filtering

A consumer can filter messages by subscribing to specified message tags to ensure that it only receives the filtered messages. The whole filtering process is completed in the TDMQ for RocketMQ broker.

Consumption Offset Reset

Resetting the consumption offset means resetting a consumer group's consumption offset for subscribed topics within the persistent message storage period based on the time axis. After the offset is reset, the consumers will receive the messages that the producer sends to the TDMQ for RocketMQ broker after the set time point.

Message Trace

The message trace records the entire lifecycle of a message from the time it is sent by the producer to the time it is received and processed by the consumer. With this feature, you can track the entire trace of a message, starting from its production by a producer, its storage and distribution within the TDMQ for RocketMQ broker, and finally its consumption by one or more consumers. This helps you troubleshoot any problems that may occur during message processing.

Message Heap

Message heap occurs in scenarios where the producer has sent messages to the TDMQ for RocketMQ broker but the consumer fails to normally consume all these messages promptly due to its consumption capability limit. In this case, the unconsumed messages will be heaped in the broker. The message heap data is collected once every minute. After the message retention period (3 days by default) elapses, the unconsumed messages will no longer be heaped in the broker because they have been deleted by the broker.