Distributed Message Service for RabbitMQ

Service Overview

Issue 01

Date 2025-03-24

HUAWEI

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Trademarks and Permissions


and other Huawei trademarks are the property of Huawei Technologies Co., Ltd.
All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Contents

1 What Is DMS for RabbitMQ?..... 1
2 Product Advantages..... 2
3 Application Scenarios..... 4
4 Specifications..... 7
5 Comparing RabbitMQ, Kafka, and RocketMQ..... 10
6 Related Services..... 13
7 Notes and Constraints..... 14
8 Basic Concepts..... 18
9 Exchanges..... 20
10 Permissions Management..... 23
11 Billing..... 26

1 What Is DMS for RabbitMQ?

Based on the open-source RabbitMQ, Distributed Message Service (DMS) for RabbitMQ provides messaging services with rich messaging features, flexible routing, alarms, monitoring, and high availability functions. It is applicable to flash sales, flow control, and system decoupling scenarios.

- Immediate use

  DMS for RabbitMQ provides single-node and cluster instances with a range of specifications for you to choose from. Instances can be created with just a few clicks on the console, without requiring you to prepare servers.

- Rich features

  DMS for RabbitMQ supports Advanced Message Queuing Protocol (AMQP) and a variety of messaging features such as message broadcast, delayed delivery, and dead letter queues.

- Flexible routing

  In RabbitMQ, an exchange receives messages from producers and pushes the messages to queues. RabbitMQ provides direct, topic, headers, and fanout exchanges. You can also bind and customize exchanges.

- High availability

  Cluster RabbitMQ instances provide quorum queues, which can be used to replicate queue data between RabbitMQ nodes, ensuring that queues can still run when a node breaks down.

- Monitoring and alarm

  RabbitMQ cluster metrics are monitored and reported, including broker memory, CPU usage, and network flow. If an exception is detected, an alarm will be triggered.

2 Product Advantages

Huawei Cloud DMS for RabbitMQ provides easy-to-use message queuing based on RabbitMQ. Services can be quickly migrated to the cloud without any change, reducing maintenance and usage costs.

- Rapid deployment

  Simply set instance information on the DMS for RabbitMQ console, submit your order, and a complete RabbitMQ instance will be automatically created and deployed.

- Service migration without modifications

  DMS for RabbitMQ is compatible with open-source RabbitMQ APIs and supports all message processing functions of open-source RabbitMQ.

  If your application services are developed based on open-source RabbitMQ, you can easily migrate them to Huawei Cloud DMS for RabbitMQ after specifying a few authentication configurations.

  NOTE

  RabbitMQ instances are compatible with RabbitMQ 3.8.35.

- Exclusive experience

  RabbitMQ instances are physically isolated from each other and exclusively owned by the tenant.

- High performance

  Each queue can process up to 100,000 transactions per second (with default configurations). Performance can be increased simply by adding queues.

- Data security

  Operations on RabbitMQ instances are recorded and can be audited. Messages can be encrypted before storage.

  In addition to SSL, VPCs and security groups also provide security controls on network access.

- Simple O&M

  Huawei Cloud provides a whole set of monitoring and alarm services, eliminating the need for 24/7 attendance. RabbitMQ instance metrics are monitored and reported, including the number of partitions, topics, and accumulated messages. You can configure alarm rules and receive SMS or email notifications on how your services are running in real time.

- Multi-language support

  RabbitMQ is an open-source service based on AMQP. It is used to store and forward messages in a distributed system. A RabbitMQ server is compiled in Erlang (supporting high concurrency, distribution, and robust fault tolerance), and a RabbitMQ client can be compiled in various programming languages, including Python, Ruby, .NET, Java, JMS, C, PHP, ActionScript, XMPP, STOMP, and AJAX.

3 Application Scenarios

RabbitMQ is popular message-oriented middleware that features highly reliable, asynchronous message delivery. It can be used for transmitting data between different systems in the enterprise application, payment, telecommunications, e-commerce, social networking, instant messaging, video, Internet of Things, and Internet of Vehicle industries.

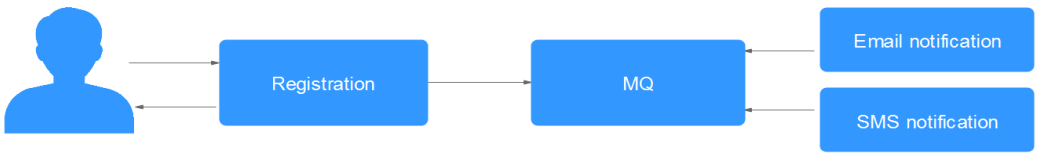
Asynchronous Communication

Non-core or less important messages are sent asynchronously to receiving systems, so that the main service process is not kept waiting for the results of other systems, allowing for faster responses.

For example, RabbitMQ can be used to send a notification email and SMS message after a user has registered with a website, providing fast responses throughout the registration process.

Figure 3-1 Serial registration and notification

Figure 3-2 Asynchronous registration and notification using message queues



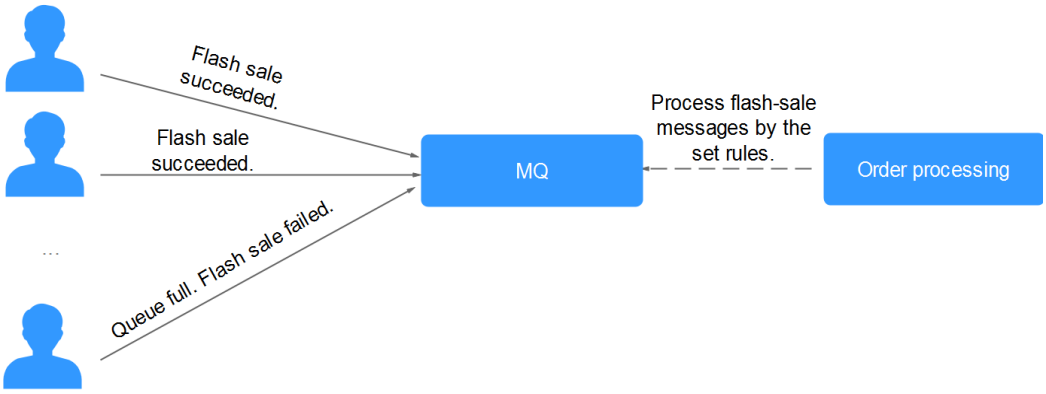
Traffic Control

In e-commerce systems or large-scale websites, there is a processing capability gap between upstream and downstream systems. Traffic bursts from upstream systems with high processing capabilities may have a large impact on downstream systems with lower processing capabilities. For example, online sales promotions involve a huge amount of traffic flooding into e-commerce systems. RabbitMQ

provides a three-day buffer by default for hundreds of millions of messages, such as orders and other information. In this way, message consumption systems can process the messages during off-peak periods.

In addition, flash sale traffic bursts originating from frontend systems can be handled with RabbitMQ, keeping the backend systems from crashing.

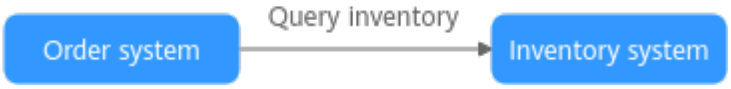
Figure 3-3 Traffic burst handling using RabbitMQ



System Decoupling

Take e-commerce flash sales as an example. Traditionally, an order processing system sends order requests to the inventory system and waits for responses. If the inventory system goes down, the order processing system will not be able to get the data it wants, and the order will fail to be submitted. This means that the order processing system and the inventory system are closely coupled.

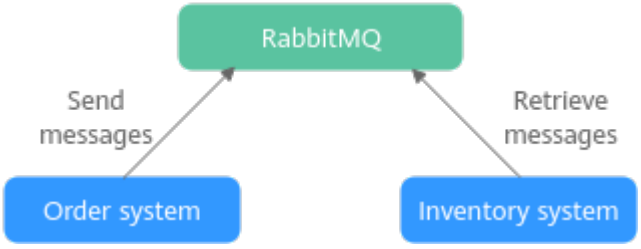
Figure 3-4 Closely coupled systems



With RabbitMQ, order submission data will be stored in queues. Then, a response will be returned indicating that the order has been submitted.

The inventory system consumes the order submission message it has subscribed to. In this way, order submission will not be interrupted even if the inventory system breaks down.

Figure 3-5 System decoupling



High Availability

Normally, there is only one broker. If the broker fails, queues on it will become unavailable.

Queue mirroring is available since RabbitMQ 2.6.0. In a RabbitMQ cluster, queues can be mirrored across brokers. In the event of a broker failure, services are still available because the mirrors will take over.

Quorum queues are available since RabbitMQ 3.8. Quorum queues can be used to replicate queue data, ensuring that queues can still run if a broker breaks down.

4 Specifications

RabbitMQ Instance Specifications

RabbitMQ instances are compatible with RabbitMQ 3.8.35. **Table 4-1** lists the specifications of single-node and cluster RabbitMQ instances.

Table 4-1 Specifications of cluster RabbitMQ instances

Flavor	Brokers	Storage Space (GB)	Reference TPS	Maximum Consumers per Broker	Recommended Queues per Broker	Maximum Connections per Broker
rabbitmq.2u4g.cluster	3	300–90,000	3,000	4,000	100	1,000
	5	500–150,000	5,000	4,000	100	1,000
	7	700–210,000	7,000	4,000	100	1,000
rabbitmq.4u8g.cluster	3	300–90,000	6,000	8,000	200	2,000
	5	500–150,000	10,000	8,000	200	2,000
	7	700–210,000	14,000	8,000	200	2,000
rabbitmq.8u16g.cluster	3	300–90,000	12,000	16,000	400	4,000
	5	500–150,000	20,000	16,000	400	4,000
	7	700–210,000	28,000	16,000	400	4,000
rabbitmq.12u24g.cluster	3	300–90,000	24,000	24,000	600	6,000

Flavor	Brokers	Storage Space (GB)	Reference TPS	Maximum Consumers per Broker	Recommended Queues per Broker	Maximum Connections per Broker
	5	500–150,000	40,000	24,000	600	6,000
	7	700–210,000	56,000	24,000	600	6,000
rabbitmq.16u32g.cluster	3	300–90,000	48,000	32,000	800	8,000
	5	500–150,000	80,000	32,000	800	8,000
	7	700–210,000	112,000	32,000	800	8,000
rabbitmq.24u48g.cluster	3	300–90,000	60,000	40,000	1,000	10,000
	5	500–150,000	100,000	40,000	1,000	10,000
	7	700–210,000	140,000	40,000	1,000	10,000

NOTE

- In the preceding tables, TPS (of production and consumption) is represented by the number of messages (2 KB each) processed per second. In the tests, persistence and queue mirroring were not enabled. Messages were retrieved immediately after creation and were not accumulated in the queues. The data is for reference only and may differ from that in your production environment.
- Performance is related to the queue quantity, message accumulation, number of connections, number of channels, number of consumers, queue mirroring, priority queue, message persistence, and the exchange type. Select instance specifications based on the pressure test result of the service model.
- A maximum of 2047 channels can be opened on a connection.
- Single-node instances can be used for testing. Do not use them for message production. Single-node flavors are not yet available.

Storage Space Selection

In cluster mode, RabbitMQ persists messages to disk. When creating a RabbitMQ instance, select a proper storage space size based on the estimated message size and the number of replicas in a mirrored queue, which can be maximally equal to the number of brokers in the cluster.

For example, if the estimated message size is 100 GB, the disk capacity must be at least: 100 GB x Number of mirrored replicas + 100 GB (reserved).

For single-node instances, select a storage space size based on the estimated message size and the reserved disk space.

You can change the number of brokers in a cluster, but cannot change the specifications of a single-node instance.

5 Comparing RabbitMQ, Kafka, and RocketMQ

Table 5-1 Functions

Feature	RocketMQ	Kafka	RabbitMQ
Priority queue	Not supported	Not supported	Supported. It is recommended that the priority be set to 0–10.
Delayed queue	Supported	Not supported	Supported
Dead letter queue	Supported	Not supported	Supported
Message retry	Supported	Not supported	Not supported.
Retrieval mode	Pull-based and push-based	Pull-based	Pull-based and push-based
Message broadcasting	Supported	Supported	Supported
Message tracking	Supported	Supports offset and timestamp tracking.	Not supported. Once a message retrieval has been acknowledged, RabbitMQ will be notified that the message can be deleted.

Feature	RocketMQ	Kafka	RabbitMQ
Message accumulation	Supported	Supports higher accumulation performance than RabbitMQ thanks to high throughput.	Supported
Persistence	Supported	Supported	Supported
Message tracing	Supported	Not supported	Supported by the firehose feature or the rabbitmq_tracing plugin. However, rabbitmq_tracing reduces performance and should be used only for troubleshooting.
Message filtering	Supported	Supported	Not supported, but can be encapsulated.
Multi-tenancy	Supported	Supported	Supported
Multi-protocol	Compatible with RocketMQ.	Only supports Apache Kafka.	RabbitMQ is based on AMQP.
Multi-language	Supports clients in multiple programming languages.	Kafka is written in Scala and Java and supports clients in multiple programming languages.	Supports clients in multiple programming languages.
Throttling	RocketMQ 5.x supports traffic control based on instance specifications.	Supports throttling on producer or consumer clients, users, and topics.	Supports credit-based throttling on producers, a mechanism that triggers protection from within.
Ordered message delivery	Message order is maintained within a queue.	Supports partition-level FIFO.	Supports FIFO only for single-threaded message queuing without advanced features such as delayed queues or priority queues.

Feature	RocketMQ	Kafka	RabbitMQ
Security	Supports SSL authentication.	Supports SSL and SASL authentication and read/write permissions control.	Supports SSL authentication.
Transactional messages	Supported	Supported	Supported

6 Related Services

- Elastic Cloud Server (ECS)

  An ECS is a basic computing unit that consists of vCPUs, memory, OS, and EVS disks. RabbitMQ instances run on ECSs. A broker corresponds to an ECS.

- Elastic Volume Service (EVS)

  EVS provides block storage services for ECSs. All RabbitMQ data, such as messages, metadata, and logs, is stored in EVS disks.

- Cloud Trace Service (CTS)

  Cloud Trace Service (CTS) generates traces to provide you with a history of operations performed on cloud service resources. The traces include operation requests sent using the management console or open APIs as well as the operation results. You can view all generated traces to query, audit, and backtrack performed operations.

- VPC

  RabbitMQ instances run in VPCs and use the IP addresses and bandwidth of VPC. Security groups of VPCs enhance the security of network access to the RabbitMQ instances.

- Cloud Eye

  Cloud Eye is an open platform that provides monitoring, alarm reporting, and alarm notification for your resources in real time.

  NOTE

  The values of all RabbitMQ instance metrics are reported to Cloud Eye every minute.

- Elastic IP (EIP)

  The EIP service provides independent public IP addresses and bandwidth for Internet access. RabbitMQ instances bound with EIPs can be accessed over public networks.

- Tag Management Service (TMS)

  TMS is a visualized service for fast and unified cross-region tagging and categorization of cloud services.

  Tags facilitate RabbitMQ instance identification and management.

7 Notes and Constraints

This section describes the notes and constraints on Huawei Cloud Distributed Message Service (DMS) for RabbitMQ. Use your RabbitMQ instances as prescribed to avoid program exceptions.

NOTICE

Any instability caused by ignorance of the notes and constraints is not covered by the SLA.

Instance

Table 7-1 Notes and constraints

Item	Constraint
Version	Server version: 3.8.35
Number of connections	The allowed number of connections differs by instance specifications and mode (single-node or cluster). For details, see **Specifications**.
Channels	Number of channels that can be created for a single connection: ≤ 2047
Memory high watermark	≤ 40% If the memory usage exceeds 40%, the high memory watermark may be triggered, blocking publishers.
Disk high watermark	≥ 5 GB If the remaining disk space is less than 5 GB, the high disk watermark is triggered, blocking publishers.

Item	Constraint
cluster_partition_handling	pause_minority When a network partition occurs in a cluster, cluster brokers will determine whether they are in a minority, that is, fewer than or equal to the total number of brokers. Minority brokers pause when a partition starts, detect the network status periodically, and start again when the partition ends. If queue mirroring is not enabled, queue replicas in the minority will no longer be available for message creation and retrieval. This strategy sacrifices availability for data consistency.
rabbitmq_delayed_message_exchange	There may be an error of about 1%. The actual delivery time may be earlier or later than the scheduled delivery time.
RabbitMQ plug-ins	RabbitMQ plug-ins can be used for testing and service migration. Do not use them for production. Reliability issues caused from using plug-ins are not within commitments on SLAs.
VPC, subnet, and AZ	After an instance is created, its VPC, subnet, and AZ cannot be modified.
Storage space per broker	The storage space can be expanded but cannot be reduced.
Broker quantity	• The broker quantity can be increased but cannot be decreased for a cluster instance. • Services may temporarily stutter during the broker increase. Ensure that your client can auto-reconnect. Modify specifications during off-peak hours. • This function is not available for single-node instances.
Broker Flavor	• The broker flavor can be increased or decreased. • For single-node instances and cluster ones without mirrored/quorum queues configured, services may stutter for several minutes during the modification. Ensure that your client can auto-reconnect. Modify specifications during off-peak hours. • For cluster instances configured with mirrored/quorum queues, services may stutter for several seconds during the modification. Ensure that your client can auto-reconnect. Modify specifications during off-peak hours.

Virtual Host

Table 7-2 Constraint

Item	Constraint
Deleting a virtual host	The default virtual host created in instance creation cannot be deleted.

Exchange

Table 7-3 Notes and exchanges

Item	Constraint
Default exchange	For RabbitMQ 3.8.35 instances, seven exchanges are created by default after virtual host creation. These exchanges include (AMQP default), amq.direct, amq.fanout, amq.headers, amq.match, amq.rabbitmq.trace, and amq.topic.
Binding an exchange	• In RabbitMQ 3.8.35, the exchange **(AMQP default)** cannot be bound with any exchange. • **Internal** exchanges can only be bound with exchanges and not queues.
Deleting an exchange	In RabbitMQ 3.8.35, the default exchange cannot be deleted.

Queue

Table 7-4 Notes and constraints

Item	Constraint
Binding a queue	• In RabbitMQ 3.8.35, the exchange **(AMQP default)** cannot be bound with any queue. • **Internal** exchanges can only be bound with exchanges and not queues.
Lazy queues	Available only for RabbitMQ 3.8.35.
Quorum queues	Available only for RabbitMQ 3.8.35.
Single active consumer	Only available in RabbitMQ 3.8.35.

Message

Table 7-5 Notes and constraints

Item	Constraint
Message size	≤ 50 MB per message Do not send a message larger than 50 MB. Otherwise, the message will fail to be created.

8 Basic Concepts

Huawei Cloud DMS for RabbitMQ uses RabbitMQ as the messaging engine. In RabbitMQ, messages are sent by producers, stored in queues, and received by consumers. The following explains basic concepts of RabbitMQ.

Message

A message has a message body and a label. The message body, in JSON or other formats, contains the content of the message. The label only describes the message.

Messages are sent by producers and retrieved by consumers, but a producer and a consumer are not directly linked to each other.

Producer

A producer is an application that sends messages to queues. The messages are then delivered to other systems or modules for processing as agreed.

Consumer

A consumer is an application that receives messages. Consumers subscribe to queues. During routing, only the message body will be stored in the queue, so only the message body will be consumed by consumers.

Queue

A queue stores messages that are sent from producers and await retrievals by consumers. If different consumers subscribe to the same queue, the messages in that queue will be distributed across the consumers.

Broker

Nodes that provide message middleware services.

Virtual Host

Virtual hosts provide logical separation of exchanges, queues, and bindings. Different applications run on different virtual hosts without interfering with each other.

Exchange

Exchanges receive and assign messages. Producers send messages to exchanges first, rather than directly to queues. Exchanges route messages to one or more queues based on routing keys. If there are no matching queues, the messages are discarded. To learn about exchange types, see **Exchanges**.

9 Exchanges

Direct, fanout, topic, and headers exchanges are available.

Direct Exchange

How It Works

1. A queue is bound to a direct exchange with a routing key.
2. The direct exchange routes a received message to the bound queue whose routing key is matched.

Routing

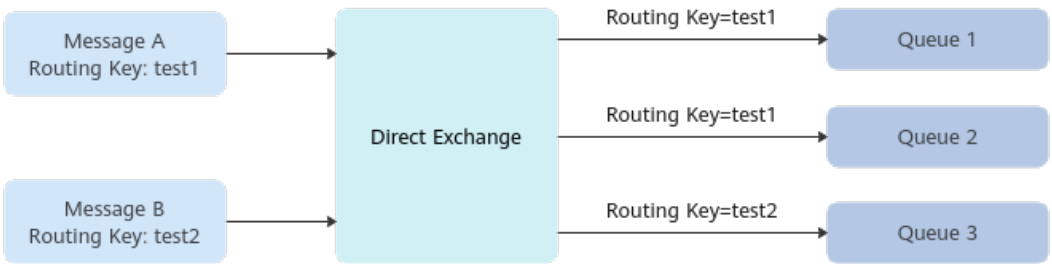
Based on a routing key matching

Scenario

Unicast routing

Example

Figure 9-1 Example direct exchange



As shown in this chart, Message A will be sent to Queue 1 and Queue 2. Message B will be sent to Queue 3.

Fanout Exchange

How It Works

A fanout exchange bound with multiple queues routes received messages to each queue. Fanout exchanges forward messages faster than other exchanges.

Routing

The fanout exchange delivers messages to all bound queues.

Scenario

Broadcast routing

Example

Figure 9-2 Example fanout exchange

As shown in this chart, Message A will be sent to Queue 1 and Queue 2.

Topic Exchange

How It Works

1. A queue is bound to a topic exchange with a routing key that includes a wildcard.
2. The topic exchange routes a received message to the queue if the message's routing key wildcard is matched.

Supported wildcards are stars (*****) and hashes (**#**). Separate wildcards and words by periods (.), for example, **test.#**.

- ***** matches one word.
- **#** matches zero or more words.

Routing

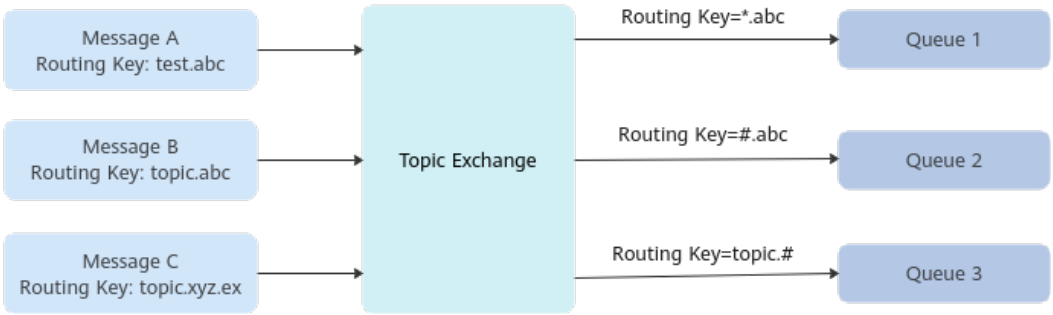
Based on a routing key wildcard matching

Scenario

Multicast routing

Example

Figure 9-3 Example topic exchange



As shown in this chart, Message A will be sent to Queue 1 and Queue 2, Message B to Queue 1, Queue 2, and Queue 3, and Message C to Queue 3.

Headers Exchange

How It Works

1. A queue is bound to a headers exchange with a binding expressed in a key-value pair.
2. The headers exchange routes a message to the queue if the binding matches the message's key-value header.

The matching algorithm uses a specific binding key-value pair, which is **x-match**. Values:

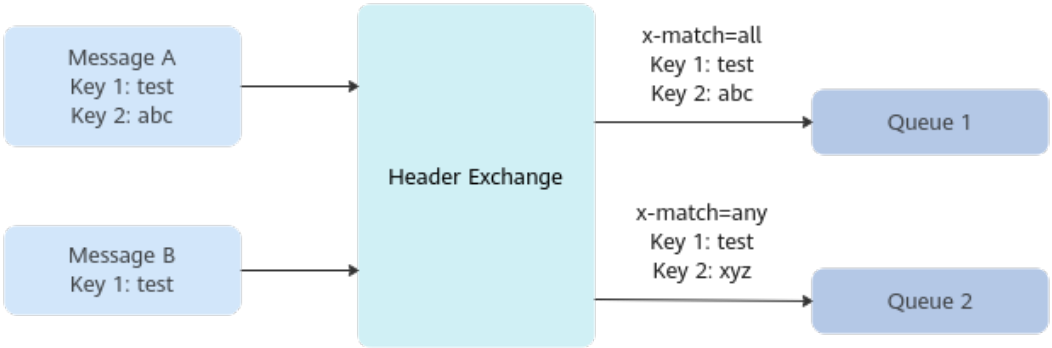
- **all**: Messages are routed only when all header pairs match.
- **any**: Messages are routed when any header pair matches.

Routing

Based on matching between key-value pairs in the message headers and the binding (a key-value pair)

Example

Figure 9-4 Example headers exchange



As shown in this chart, Message A will be sent to Queue 1 and Queue 2, and Message B to Queue 2.

10 Permissions Management

You can use Identity and Access Management (IAM) to manage DMS for RabbitMQ permissions and control access to your resources. IAM provides identity authentication, permissions management, and access control.

You can create IAM users for your employees, and assign permissions to these users on a principle of least privilege (PoLP) basis to control their access to specific resource types. For example, you can create IAM users for software developers and assign specific permissions to allow them to use DMS for RabbitMQ resources but prevent them from being able to delete resources or perform any high-risk operations.

If your HUAWEI ID does not require individual IAM users for permissions management, skip this section.

IAM is free of charge. You pay only for the resources you use. For more information, see **IAM Service Overview**.

DMS for RabbitMQ Permissions

By default, new IAM users do not have any permissions assigned. To assign permissions to these new users, add them to one or more groups, and attach permissions policies or roles to these groups.

DMS for RabbitMQ is a project-level service deployed and accessed in specific physical regions. When assigning DMS for RabbitMQ permissions to a user group, specify region-specific projects where the permissions will take effect. If you select **All projects**, the permissions will be granted for all region-specific projects. When accessing DMS for RabbitMQ, the users need to switch to a region where they have been authorized to use this service.

You can grant permissions by using roles and policies.

- Roles: A type of coarse-grained authorization mechanism that provides only a limited number of service-level roles. When using roles to grant permissions, you also need to assign dependency roles. However, roles are not an ideal choice for fine-grained authorization and secure access control.
- Policies: A fine-grained authorization strategy that defines permissions required to perform operations on specific cloud resources under certain conditions. This mechanism allows for more flexible policy-based authorization for securer access control. For example, you can grant DMS for

RabbitMQ users only the permissions for managing a certain type of DMS for RabbitMQ instances. Most policies define permissions based on APIs. For the API actions supported by DMS for RabbitMQ, see **Permissions Policies and Supported Actions**.

NOTE

Permissions policies of DMS for RabbitMQ are based on DMS. Therefore, when assigning permissions, select DMS permissions policies.

Table 10-1 lists all the system-defined roles and policies supported by DMS for RabbitMQ.

Table 10-1 System-defined roles and policies supported by DMS for RabbitMQ

Role/Policy Name	Description	Type	Dependency
DMS FullAccess	Administrator permissions for DMS. Users granted these permissions can perform all operations on DMS.	System-defined policy	None
DMS UserAccess	Common user permissions for DMS, excluding permissions for creating, modifying, deleting, and scaling up instances.	System-defined policy	None
DMS ReadOnlyAccess	Read-only permissions for DMS. Users granted these permissions can only view DMS data.	System-defined policy	None
DMS VPCAccess	VPC operation permissions to assign to DMS agencies.	System-defined policies	None
DMS KMSAccess	KMS operation permissions to assign to DMS agencies.	System-defined policies	None
DMS Administrator	Administrator permissions for DMS.	System-defined role	This role depends on the **Tenant Guest** and **VPC Administrator** roles.

Table 10-2 lists the common operations supported by each DMS for RabbitMQ system policy or role. Select the policies or roles as required.

Table 10-2 Common operations supported by system-defined policies

Operation	DMS FullAccess	DMS UserAccess	DMS ReadOnlyAccess	DMS VPCAccess	DMS KMSAccess
Creating instances	√	×	×	×	×
Modifying instances	√	×	×	×	×
Deleting instances	√	×	×	×	×
Modifying instance specifications	√	×	×	×	×
Querying instance information	√	√	√	×	×

Helpful Links

- **What Is IAM?**
- **Creating a User and Granting DMS for RabbitMQ Permissions**
- **Permissions Policies and Supported Actions**

11 Billing

Huawei Cloud DMS for RabbitMQ supports pay-per-use. For details, see **Product Pricing Details**.

Billing Items

Huawei Cloud DMS for RabbitMQ is billed based on RabbitMQ instance specifications and storage space.

Table 11-1 Billing of DMS for RabbitMQ

Billing Item	Description
Instance	• RabbitMQ instances are billed based on the specifications described in **Table 11-2**. • RabbitMQ instances can be billed on a pay-per-use (hourly) basis.

Billing Item	Description
Storage	• RabbitMQ instances are also billed based on the storage space. For each type of instance specification, you can choose the high I/O or ultra-high I/O disk type to meet your service requirements. You can specify the number of replicas when creating a topic. For example, if the required disk size to store the message data is 100 GB and there are three replicas, the disk capacity should be at least: 100 GB x 3 = 300 GB. • Storage space can be specified with increments of 100 GB. For details about the storage space range, see **Table 11-2**. • Storage space can be billed on a pay-per-use (hourly) basis.

Table 11-2 Specifications of cluster RabbitMQ instances

Flavor	Brokers	Storage Space (GB)	Reference TPS	Maximum Consumers per Broker	Recommended Queues per Broker	Maximum Connections per Broker
rabbitmq.2u4g.cluster	3	300–90,000	3,000	4,000	100	1,000
	5	500–150,000	5,000	4,000	100	1,000
	7	700–210,000	7,000	4,000	100	1,000
rabbitmq.4u8g.cluster	3	300–90,000	6,000	8,000	200	2,000
	5	500–150,000	10,000	8,000	200	2,000
	7	700–210,000	14,000	8,000	200	2,000
rabbitmq.8u16g.cluster	3	300–90,000	12,000	16,000	400	4,000
	5	500–150,000	20,000	16,000	400	4,000
	7	700–210,000	28,000	16,000	400	4,000

Flavor	Brokers	Storage Space (GB)	Reference TPS	Maximum Consumers per Broker	Recommended Queues per Broker	Maximum Connections per Broker
rabbitmq.12u24g.cluster	3	300–90,000	24,000	24,000	600	6,000
	5	500–150,000	40,000	24,000	600	6,000
	7	700–210,000	56,000	24,000	600	6,000
rabbitmq.16u32g.cluster	3	300–90,000	48,000	32,000	800	8,000
	5	500–150,000	80,000	32,000	800	8,000
	7	700–210,000	112,000	32,000	800	8,000
rabbitmq.24u48g.cluster	3	300–90,000	60,000	40,000	1,000	10,000
	5	500–150,000	100,000	40,000	1,000	10,000
	7	700–210,000	140,000	40,000	1,000	10,000

Billing Modes

Pay-per-use (hourly): This billing mode is flexible, enabling you to start and stop services anytime. You pay only for the actual usage duration. The minimum time unit is one hour. Less than an hour is recorded as an hour.

Configuration Changes

You can increase the number of brokers (for cluster instances only) and the disk size (for both cluster and single-node instances).

