Distributed Message Service for RabbitMQ

Getting Started

Issue	01
Date	2023-07-20

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Trademarks and Permissions

HUAWEI 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

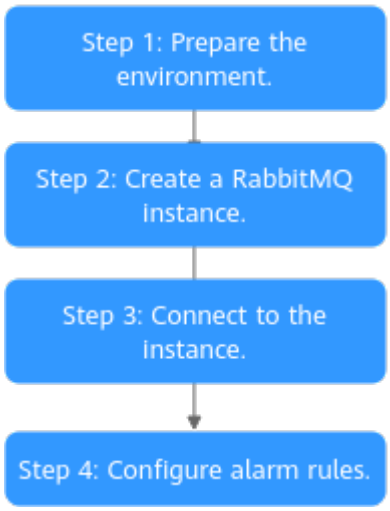
1 Introduction

This document provides instructions for getting started with Distributed Message Service (DMS) for RabbitMQ, including creating a RabbitMQ instance on the console and connecting to a RabbitMQ instance through an Elastic Cloud Server (ECS).

You can also **create a RabbitMQ instance by calling APIs**.

Procedure

Figure 1-1 Procedure for using DMS for RabbitMQ



1. **Prepare the environment**.

   A RabbitMQ instance runs in a Virtual Private Cloud (VPC). Before creating a RabbitMQ instance, ensure that a VPC is available.

2. **Create a RabbitMQ instance**.

   When creating an instance, you can choose whether to enable SSL. If SSL is enabled, data is encrypted for transmission, improving data security. The SSL setting can be configured only when you create an instance. After an instance is created, the SSL setting cannot be changed.

3. Connect to the instance.

A client can connect to an instance **with SSL** or **without SSL**.

4. **Configure alarm rules**.

   Configure alarm rules for a RabbitMQ instance to monitor the service running status.

NOTE

Learn more about the **basic concepts of RabbitMQ**.

2 Step 1: Prepare the Environment

VPC

A VPC provides an isolated virtual network for your RabbitMQ instances. You can configure and manage the network as required.

Step 1 Before creating a RabbitMQ instance, ensure that a VPC and a subnet are available.

For details, see **Creating a VPC**. If you already have an available VPC and subnet, you do not need to create new ones.

Note the following when creating a VPC and subnet:

- The VPC and the RabbitMQ instance must be in the same region.
- Use the default settings when creating a VPC and a subnet.

Step 2 Before creating a RabbitMQ instance, ensure that a security group is available.

For details, see **Creating a Security Group**. If you already have an available security group, you do not need to create a new one.

To use RabbitMQ instances, add the security group rules described in **Table 2-1**. Other rules can be added based on site requirements.

Table 2-1 Security group rules

Direction	Protocol	Port	Source	Description
Inbound	TCP	5672	0.0.0.0/0	Access a RabbitMQ instance (without SSL encryption).
Inbound	TCP	5671	0.0.0.0/0	Access a RabbitMQ instance (with SSL encryption).
Inbound	TCP	15672	0.0.0.0/0	Access the management UI (without SSL encryption).
Inbound	TCP	15671	0.0.0.0/0	Access the management UI (with SSL encryption).

NOTE

After a security group is created, its default inbound rule allows communication among ECSs within the security group and its default outbound rule allows all outbound traffic. In this case, you can access a RabbitMQ instance within a VPC, and do not need to add rules according to **Table 2-1**.

----End

(Optional) EIP

To access a RabbitMQ instance over a public network, prepare an elastic IP address (EIP) in advance.

For details, see **Assigning an EIP**.

The EIP must be created in the region the RabbitMQ instance is in.

ECS

Before connecting to a RabbitMQ instance, ensure that you have purchased an ECS, installed the JDK, and configured environment variables. The following steps describe how to complete these preparations. A Linux ECS is taken as an example. For more information on how to install JDK and configure the environment variables for a Windows ECS, please search the Internet.

Step 1 Log in to the management console, under **Computing**, click **Elastic Cloud Server**, and then create an ECS.

For details, see **Purchasing an ECS**. If you already have an available ECS, skip this step.

Step 2 Log in to the ECS.

Step 3 Install JDK or JRE, and add the following contents to **.bash_profile** in the home directory to configure the environment variables **JAVA_HOME** and **PATH**: In this command, **/opt/java/jdk1.8.0_151** is the JDK installation path. Change it to the path where you install JDK or JRE.

```
export JAVA_HOME=/opt/java/jdk1.8.0_151
export PATH=$JAVA_HOME/bin:$PATH
```

Run the **source .bash_profile** command for the modification to take effect.

NOTE

Use Oracle JDK instead of ECS's default JDK (for example, OpenJDK), because ECS's default JDK may not be suitable. Obtain Oracle JDK 1.8.111 or later from **Oracle's official website**.

----End

3 Step 2: Create a RabbitMQ Instance

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.

Advanced Message Queuing Protocol (AMQP) is an advanced message queue protocol that provides an open standard of application layer protocols.

Prerequisites

Ensure that a VPC is available. For details on how to create a VPC, see **Virtual Private Cloud User Guide**.

If you already have an available VPC, you do not need to create a new one.

Procedure

Step 1 Log in to the RabbitMQ console, and click **Buy RabbitMQ Instance** in the upper right corner.

Step 2 Specify **Billing Mode**, **Region**, **Project**, and **AZ**.

Step 3 Specify the instance name and the enterprise project.

Step 4 Configure the following instance parameters:

1. **Version**: RabbitMQ version. Currently, only 3.8.35 is supported.
2. **Instance Type**: Select **Single-node** or **Cluster**.
   - **Single-node**: There is only one RabbitMQ broker.
   - **Cluster**: There are multiple RabbitMQ brokers, achieving highly reliable message storage.
3. **CPU Architecture**: Currently, only x86 architecture is supported.
4. **Broker Flavor**: Select a flavor as required.

   NOTE

   To ensure service stability and reliability, DMS for RabbitMQ sets the default memory high watermark to 40%. Publishers will be blocked if the memory usage exceeds 40%. To avoid reaching the high watermark, retrieve messages stacked in the queue in time.

5. **Brokers**: Select the required number of brokers.
6. **Storage Space**: Indicates the disk type and total storage space of the RabbitMQ instance.

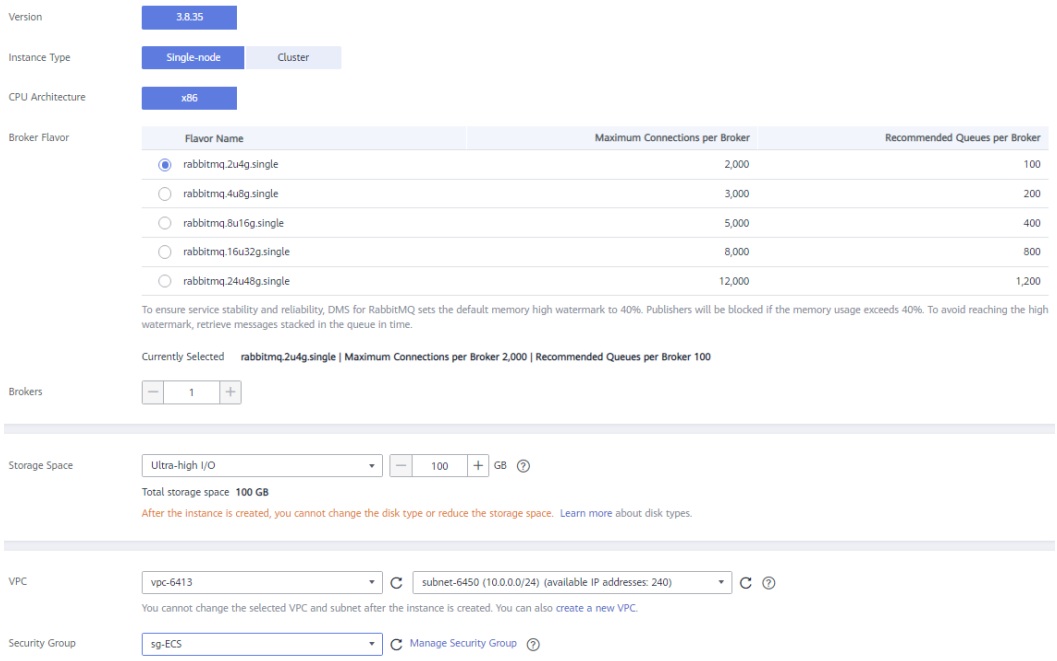
   For details about how to select a disk type, see **Disk Types and Performance**.

   – For a single-node instance, the value range is 100 GB to 30,000 GB.
   – For a cluster instance, the value range is 100 GB x Number of brokers to 300,000 GB x Number of brokers.
7. **VPC**: Select a VPC and a subnet.

   A VPC provides an isolated virtual network for your RabbitMQ instances. You can configure and manage the network as required.
8. **Security Group**: Select a security group.

   A security group is a set of rules for accessing a RabbitMQ instance. Click **Manage Security Group**. On the displayed console, view or create security groups.

Figure 3-1 Configuring the instance parameters



Step 5 Enter the username and password used for connecting to the RabbitMQ instance.

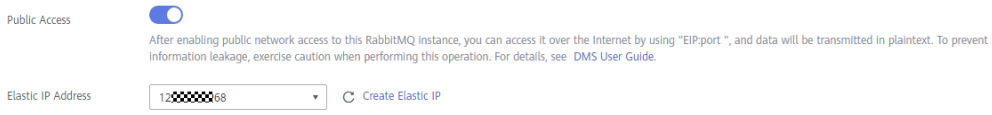
Step 6 Click **More Settings** to configure more parameters.

1. Configure **Public Access**.

   Public access can be enabled or disabled.

   A RabbitMQ instance with public access enabled can be accessed by using an EIP. After you enable public access, **Elastic IP Address** is displayed. Select an EIP or click **Create Elastic IP** to view or buy EIPs.

   Figure 3-2 Configuring public access for a RabbitMQ instance



NOTE

- In comparison with intra-VPC access, enabling public access might lead to packet loss and jitter. Therefore, you are advised to enable public access only during the service development and testing phases.
- If you manually unbind or delete an EIP on the VPC console, the public access function of the corresponding RabbitMQ instance is automatically disabled.

2. Configure **SSL**.

   This parameter indicates whether SSL authentication is enabled when a client is accessing an instance. If **SSL** is enabled, data will be encrypted before transmission for enhanced security.

   Once the instance is created, SSL cannot be enabled or disabled.

3. Specify tags.

   Tags are used to identify cloud resources. When you have many cloud resources of the same type, you can use tags to classify cloud resources by dimension (for example, by usage, owner, or environment).

   - If you have created predefined tags, select a predefined pair of tag key and value. You can click **View predefined tags** to go to the Tag Management Service (TMS) console and view or create tags.
   - You can also create new tags by entering **Tag key** and **Tag value**.

   Up to 20 tags can be added to each RabbitMQ instance.

4. Enter a description of the instance.

Step 7 Click **Buy**.

Step 8 Confirm the instance information, read and agree to the *HUAWEI CLOUD Customer Agreement*, and then submit the request.

Step 9 Return to the instance list and check whether the instance has been created.

It takes 3 to 15 minutes to create an instance. During this period, the instance status is **Creating**.

- If the instance is created successfully, its status changes to **Running**.
- If the instance fails to be created, view **Instance Creation Failures**. Delete the instance and create another instance. If the instance creation fails again, contact customer service.

----End

4 Step 3: Connect to an Instance to Create and Retrieve Messages

4.1 Connecting to an Instance Without SSL

The following demo shows how to access and use a RabbitMQ instance in a VPC, assuming that the RabbitMQ client is deployed in an ECS.

RabbitMQ instances are compatible with the open-source RabbitMQ protocol. To access a RabbitMQ instance in your service code, see the tutorials for different languages at **https://www.rabbitmq.com/getstarted.html**.

Prerequisites

- A RabbitMQ instance has been created following the instructions in **Step 2: Create a RabbitMQ Instance**, and the username and password used to create the instance have been obtained.
- The **Instance Address (Private Network)** or **Instance Address (Public Network)** of the instance has been recorded from the instance details.
- An ECS has been created, and its VPC, subnet, and security group configurations are the same as those of the RabbitMQ instance.
- You have installed the JDK and configured the environment variables. For details, see **Step 1: Prepare the Environment**.

Accessing the Instance Using CLI

Step 1 Run the following command to download **RabbitMQ-Tutorial.zip**:

```
$ wget https://dms-demo.obs.cn-north-1.myhuaweicloud.com/RabbitMQ-Tutorial.zip
```

Step 2 Run the following command to decompress **RabbitMQ-Tutorial.zip**:

```
$ unzip RabbitMQ-Tutorial.zip
```

Step 3 Run the following command to navigate to the **RabbitMQ-Tutorial** directory, which contains the precompiled JAR file:

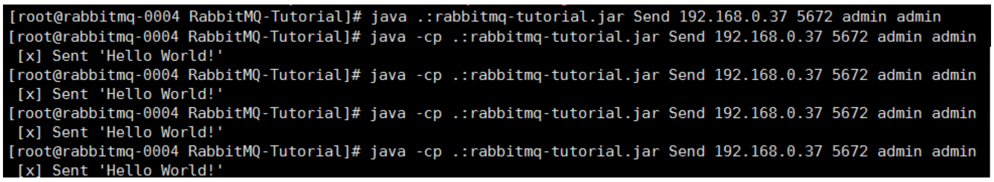
```
$ cd RabbitMQ-Tutorial
```


Step 4 Create messages using the sample project.

```
$ java -cp .:rabbitmq-tutorial.jar Send host port user password
```

host indicates the connection address for accessing the instance. *port* is the listening port of the instance, which is **5672** by default. *user* and *password* indicate the username and password used for accessing the instance.

Figure 4-1 Sample project for message creation



```
[root@rabbitmq-0004 RabbitMQ-Tutorial]# java .:rabbitmq-tutorial.jar Send 192.168.0.37 5672 admin admin
[root@rabbitmq-0004 RabbitMQ-Tutorial]# java -cp .:rabbitmq-tutorial.jar Send 192.168.0.37 5672 admin admin
 [x] Sent 'Hello World!'
[root@rabbitmq-0004 RabbitMQ-Tutorial]# java -cp .:rabbitmq-tutorial.jar Send 192.168.0.37 5672 admin admin
 [x] Sent 'Hello World!'
[root@rabbitmq-0004 RabbitMQ-Tutorial]# java -cp .:rabbitmq-tutorial.jar Send 192.168.0.37 5672 admin admin
 [x] Sent 'Hello World!'
[root@rabbitmq-0004 RabbitMQ-Tutorial]# java -cp .:rabbitmq-tutorial.jar Send 192.168.0.37 5672 admin admin
 [x] Sent 'Hello World!'
```

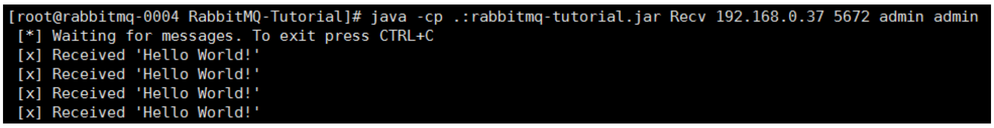
Press **Ctrl**+**C** to exit.

Step 5 Retrieve messages using the sample project.

```
$ java -cp .:rabbitmq-tutorial.jar Recv host port user password
```

host indicates the connection address for accessing the instance. *port* is the listening port of the instance, which is **5672** by default. *user* and *password* indicate the username and password used for accessing the instance.

Figure 4-2 Sample project for message retrieval



```
[root@rabbitmq-0004 RabbitMQ-Tutorial]# java -cp .:rabbitmq-tutorial.jar Recv 192.168.0.37 5672 admin admin
 [*] Waiting for messages. To exit press CTRL+C
 [x] Received 'Hello World!'
 [x] Received 'Hello World!'
 [x] Received 'Hello World!'
 [x] Received 'Hello World!'
```

To stop retrieving messages, press **Ctrl**+**C** to exit.

----End

Java Sample Code

Accessing an instance and creating messages

```
ConnectionFactory factory = new ConnectionFactory();
 factory.setHost(host);
 factory.setPort(port);

 factory.setUsername(user);
 factory.setPassword(password);
 Connection connection = factory.newConnection();
 Channel channel = connection.createChannel();

 channel.queueDeclare(QUEUE_NAME, false, false, false, null);

 String message = "Hello World!";
 channel.basicPublish("", QUEUE_NAME, null, message.getBytes("UTF-8"));
 System.out.println(" [x] Sent '" + message + "'");

 channel.close();
 connection.close();
```

Accessing an instance and retrieving messages

```
ConnectionFactory factory = new ConnectionFactory();
 factory.setHost(host);
```

```
factory.setPort(port);
factory.setUsername(user);
factory.setPassword(password);
Connection connection = factory.newConnection();
Channel channel = connection.createChannel();

channel.queueDeclare(QUEUE_NAME, false, false, false, null);
System.out.println(" [*] Waiting for messages. To exit press CTRL+C");

Consumer consumer = new DefaultConsumer(channel)
{
    @Override
    public void handleDelivery(String consumerTag, Envelope envelope, AMQP.BasicProperties properties,
            byte[] body)
            throws IOException
    {
        String message = new String(body, "UTF-8");
        System.out.println(" [x] Received '" + message + "'");
    }
};
channel.basicConsume(QUEUE_NAME, true, consumer);
```

4.2 Connecting to an Instance with SSL

If SSL is enabled, data will be encrypted before transmission for enhanced security.

The following demo shows how to access and use a RabbitMQ instance in a VPC, assuming that the RabbitMQ client is deployed in an ECS.

RabbitMQ instances are compatible with the open-source RabbitMQ protocol. To access a RabbitMQ instance in your service code, see the tutorials for different languages at **https://www.rabbitmq.com/getstarted.html**.

Prerequisites

- A RabbitMQ instance has been created following the instructions in **Step 2: Create a RabbitMQ Instance**, and the username and password used to create the instance have been obtained.
- The **Instance Address (Private Network)** or **Instance Address (Public Network)** of the instance has been recorded from the instance details.
- An ECS has been created, and its VPC, subnet, and security group configurations are the same as those of the RabbitMQ instance.
- You have installed the JDK and configured the environment variables. For details, see **Step 1: Prepare the Environment**.

Accessing the Instance Using CLI

Step 1 Run the following command to download **RabbitMQ-Tutorial-SSL.zip**:

```
$ wget https://dms-demo.obs.cn-north-1.myhuaweicloud.com/RabbitMQ-Tutorial.zip
```

Step 2 Run the following command to decompress **RabbitMQ-Tutorial-SSL.zip**:

```
$ unzip RabbitMQ-Tutorial-SSL.zip
```

Step 3 Run the following command to navigate to the **RabbitMQ-Tutorial-SSL** directory, which contains the precompiled JAR file:

```
$ cd RabbitMQ-Tutorial-SSL
```


Step 4 Create messages using the sample project.

```
$ java -cp .:rabbitmq-tutorial-sll.jar Send host port user password
```

host indicates the connection address for accessing the instance. *port* is the listening port of the instance, which is **5671** by default. *user* and *password* indicate the username and password used for accessing the instance.

Figure 4-3 Sample project for message creation

```
root@ecs-3b6f RabbitMQ-Tutorial-SSL]# java -cp .:rabbitmq-tutorial-sll.jar Send 192.168.1.35 5671 root adm[redacted]n13
LF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
LF4J: Defaulting to no-operation (NOP) logger implementation
LF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
[x] Sent 'Hello World!'
root@ecs-3b6f RabbitMQ-Tutorial-SSL]# java -cp .:rabbitmq-tutorial-sll.jar Send 192.168.1.35 5671 root a[redacted]n13
LF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
LF4J: Defaulting to no-operation (NOP) logger implementation
LF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
[x] Sent 'Hello World!'
```

Press **Ctrl**+**C** to exit.

Step 5 Retrieve messages using the sample project.

```
$ java -cp .:rabbitmq-tutorial-sll.jar Recv host port user password
```

host indicates the connection address for accessing the instance. *port* is the listening port of the instance, which is **5671** by default. *user* and *password* indicate the username and password used for accessing the instance.

Figure 4-4 Sample project for message retrieval

```
root@ecs-3b6f RabbitMQ-Tutorial-SSL]# java -cp .:rabbitmq-tutorial-sll.jar Recv 192.168.1.35 5671 root ad[redacted]n13
LF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
LF4J: Defaulting to no-operation (NOP) logger implementation
LF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
[*] Waiting for messages. To exit press CTRL+C
[x] Received 'Hello World!'
[x] Received 'Hello World!'
C[root@ecs-3b6f RabbitMQ-Tutorial-SSL]#
```

To stop retrieving messages, press **Ctrl**+**C** to exit.

----End

Java Sample Code

Accessing an instance and creating messages

```
ConnectionFactory factory = new ConnectionFactory();
 factory.setHost(host);
 factory.setPort(port);

 factory.setUsername(user);
 factory.setPassword(password);
 factory.useSslProtocol();
 Connection connection = factory.newConnection();
 Channel channel = connection.createChannel();

 channel.queueDeclare(QUEUE_NAME, false, false, false, null);

 String message = "Hello World!";
 channel.basicPublish("", QUEUE_NAME, null, message.getBytes("UTF-8"));
 System.out.println(" [x] Sent '" + message + "'");

 channel.close();
 connection.close();
```

Accessing an instance and retrieving messages

```
ConnectionFactory factory = new ConnectionFactory();
 factory.setHost(host);
```



```
factory.setPort(port);
factory.setUsername(user);
factory.setPassword(password);
factory.useSslProtocol();
Connection connection = factory.newConnection();
Channel channel = connection.createChannel();

channel.queueDeclare(QUEUE_NAME, false, false, false, null);
System.out.println(" [*] Waiting for messages. To exit press CTRL+C");

Consumer consumer = new DefaultConsumer(channel)
{
    @Override
    public void handleDelivery(String consumerTag, Envelope envelope, AMQP.BasicProperties properties,
            byte[] body)
            throws IOException
    {
        String message = new String(body, "UTF-8");
        System.out.println(" [x] Received '" + message + "'");
    }
};
channel.basicConsume(QUEUE_NAME, true, consumer);
```


5 Step 4: Configure Alarm Rules

This section describes the alarm rules of some metrics and how to configure the rules. In actual scenarios, you are advised to configure alarm rules for metrics by referring to the following alarm policies.

Table 5-1 Alarm rules for RabbitMQ instances

Metric	Alarm Policy	Description	Solution
Memory High Watermark	Alarm threshold: Raw data ≥ 1 Number of consecutive periods: 1 Alarm severity: Critical	A threshold of 1 indicates that the memory high watermark is reached, blocking message publishing.	• Accelerate message retrieval. • Use publisher confirms and monitor the publishing rate and duration on the publishing end. When the duration increases significantly, apply flow control.
Disk High Watermark	Alarm threshold: Raw data ≥ 1 Number of consecutive periods: 1 Alarm severity: Critical	A threshold of 1 indicates that the disk high watermark is reached, blocking message publishing.	• Reduce the number of messages accumulated in lazy queues. • Reduce the number of messages accumulated in durable queues. • Delete queues.

Metric	Alarm Policy	Description	Solution
Memory Usage	Alarm threshold: Raw data > Expected usage (30% is recommended) Number of consecutive periods: 3–5 Alarm severity: Major	To prevent high memory watermarks from blocking publishing, configure an alarm for this metric on each node.	• Accelerate message retrieval. • Use publisher confirms and monitor the publishing rate and duration on the publishing end. When the duration increases significantly, apply flow control.
CPU Usage	Alarm threshold: Raw data > Expected usage (70% is recommended) Number of consecutive periods: 3–5 Alarm severity: Major	A high CPU usage may slow down publishing rate. Configure an alarm for this metric on each node.	• Reduce the number of mirrored queues. • For a cluster instance, add nodes and rebalance queues between all nodes.
Available Messages	Alarm threshold: Raw data > Expected number of available messages Number of consecutive periods: 1 Alarm severity: Major	If the number of available messages is too large, messages are accumulated.	See **the solution to preventing message accumulation**.
Unacked Messages	Alarm threshold: Raw data > Expected number of unacknowledged messages Number of consecutive periods: 1 Alarm severity: Major	If the number of unacknowledged messages is too large, messages may be accumulated.	• Check whether the consumer is abnormal. • Check whether the consumer logic is time-consuming.

Metric	Alarm Policy	Description	Solution
Connections	Alarm threshold: Raw data > Expected number of connections Number of consecutive periods: 1 Alarm severity: Major	A sharp increase in the number of connections may be a warning of a traffic increase.	The services may be abnormal. Check whether other alarms exist.
Channels	Alarm threshold: Raw data > Expected number of channels Number of consecutive periods: 1 Alarm severity: Major	A sharp increase in the number of channels may be a warning of a traffic increase.	The services may be abnormal. Check whether other alarms exist.
Erlang Processes	Alarm threshold: Raw data > Expected number of processes Number of consecutive periods: 1 Alarm severity: Major	A sharp increase in the number of processes may be a warning of a traffic increase.	The services may be abnormal. Check whether other alarms exist.

NOTE

- Set the alarm threshold based on the service expectations. For example, if the expected usage is 35%, set the alarm threshold to 35%.
- The number of consecutive periods and alarm severity can be adjusted based on the service logic.

Procedure

Step 1 Log in to the management console.

Step 2 In the upper left corner, click and select a region.

NOTE

Select the region where your RabbitMQ instance is.

Step 3 Click and choose **Application** > **Distributed Message Service for RabbitMQ** to open the console of DMS for RabbitMQ.

Step 4 View the instance metrics using either of the following methods:

- Click  next to a RabbitMQ instance name. On the Cloud Eye console, view the metrics of the instance, nodes, and queues. Metric data is reported to Cloud Eye every minute.
- Click the desired RabbitMQ instance to view its details. In the navigation pane, choose **Monitoring** view. On the displayed page, view the metrics of the instance, nodes, and queues. Metric data is reported to Cloud Eye every minute.

Step 5 Hover the mouse pointer over a metric and click  to create an alarm rule for the metric.

Step 6 Specify the alarm rule details.

For more information about creating alarm rules, see **Creating an Alarm Rule**.

1. Enter the alarm name and description.
2. Specify the alarm policy and alarm severity.

   For example, an alarm can be triggered and notifications can be sent once every day if the raw value of connections exceeds the preset value for three consecutive periods and no actions are taken to handle the exception.
3. Set **Alarm Notification** configurations. If you enable **Alarm Notification**, set the validity period, notification object, and trigger condition.
4. Click **Create**.

----End

6 Common Practices

You can use the common practices provided by DMS for RabbitMQ to meet your service requirements.

Table 6-1 Common practices

Practice	Description
Migrating RabbitMQ services	Migrate RabbitMQ services from an off-cloud, single-node or cluster instance to a RabbitMQ instance on Huawei Cloud.
Migrating queues	Configure queue load balancing to handle uneven queue distribution across nodes in a RabbitMQ cluster due to node scale-out or queue deletion.

