Message Queue

FAQ

FAQ

FAQ

Quick start

Where does a new group ID start to consume

If this group ID is started for the first time, it ignores the previously sent messages. That is, the group ID ignores the historical messages and starts consumption after its own startup.

If the group ID is started for the second time, the group ID starts consumption from the previous consumption point.

If you want to start consumption from a specific time point, you can use the consumption point reset function in the MQ console, and specify the specific consumption start time. Each reset only affects the specific topic under the specific group ID and will not affect other group IDs.

How should I retry if my message consumption fails

Clustering consumption

If the logic code of the consumption service returns Action.ReconsumerLater or NULL, or throws an exception, the message will undergo the retry process, and will retry at most 16 times. If the message fails after retrying 16 times, it will be discarded. Interval between two retries is as follows:

Retry number	Retry interval
1	10s
2	30s
3	1 min
4	2 min
5	3 min
6	4 min
7	5 min
8	6 min
9	7 min
10	8 min
11	9 min
12	10 min
13	20 min
14	30 min
15	1 h
16	2 h

The message.getReconsumeTimes() method can be called to obtain the number of message retries.

Broadcasting consumption

Broadcasting consumption can ensure that a message will be consumed at least once, but will not retry after a consumption fails.

What should I do if the producer has sent a message but the consumer has not received it

MQ provides several message query methods.

Specify a topic and time range to query all messages received by this topic within the specified time range.

Specify a topic and message ID to perform precise query of messages.

Specify a topic and message key to query messages with the same message key.

The preceding methods can be used to query the specific content and consumption information of messages. To trace the time and location of each node on the entire link from the producer to the consumer, you can use the latest message tracing function provided by MQ. For the specific operation method, see Message tracing.

Is MQ always free of duplicate messages

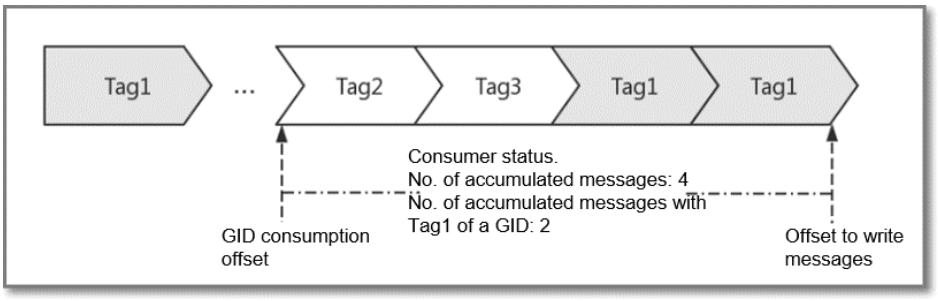
Messages are not duplicated in most cases. As a distributed message middleware, MQ cannot ensure that messages are not duplicated when exceptions (such as network jitter or application processing timeout) occur. However, MQ can ensure that no messages are lost.

Does the value of Accumulation Amount in the console include the number of all tagged messages under topics

Yes. The message producer sends tags of all types to the same topic. Messages are arranged in the queue in order and a message write point is maintained. When a group ID is started, it specifies the tags to be subscribed and obtains the current consumption point from the broker. The broker traverses messages in the queue from the consumption point of the current group ID. If the broker identifies that the tag of a message conforms to the tag subscribed by the group ID, the broker delivers the message to the group ID. Otherwise, the broker skips this message.

As shown in the following figure, the group ID consumption point moves forward. Messages with

Tag2 and Tag3 are filtered out by the broker. The message with Tag1 is required by the group ID and will be delivered to the group ID.



Therefore, the accumulation amount you saw after choosing **Consumer Status > Accumulation Amount** in the console is the unfiltered accumulation amount, which contains messages with all tags.

Configuration

How long can MQ messages be retained on the broker

MQ messages are retained on the broker for up to three days, and the system automatically deletes the unconsumed messages after three days. We recommend that you configure the Monitoring service to keep an eye on the consumption status, and intervene manually upon alarms.

What is the maximum length of the MQ message body

The maximum length of a MQ message varies with the message type, described as follows:

A normal or ordered message: 4 MB

A transactional, scheduled, or delayed message: 64 KB

How do I set the number of consumption threads on an MQ client

To set the number of threads on the MQ client, add a ConsumeThreadNums attribute when starting the consumer. The example is shown as follows:

```
public static void main(String[] args) {
Properties properties = new Properties();
properties.put(PropertyKeyConst.GROUP_ID, "GID_001");
properties.put(PropertyKeyConst.AccessKey, "xxxxxxxxxxx");
properties.put(PropertyKeyConst.SecretKey, "xxxxxxxxxxx");
/**
* Set the number of consumer threads to 20.
*/
properties.put(PropertyKeyConst.ConsumeThreadNums,20);
Consumer consumer = ONSFactory.createConsumer(properties);
consumer.subscribe("TestTopic", "*", new MessageListener() {
public Action consume(Message message, ConsumeContext context) {
System.out.println("Receive: " + message);
return Action.CommitMessage;
}
});
consumer.start();
System.out.println("Consumer Started");
}
```

What should I do if a .dll file loading error or other running error occurs due to incorrect .NET client configuration

See. SDK_GUIDE.pdf in the .NET SDK compressed package to verify that the project configuration is the same as that described in the document.

Message tracing

Why is tracing data not found

When no tracing data is found after you enter the query conditions, check as follows:

1. Currently, the Java (1.2.2 or later) client supports the message tracing function.
2. Check whether the query conditions are set properly, that is, whether Topic, Message ID, and Message Key are set properly.
3. Check whether the query time range is correct. To increase the query speed, you must specify the range of the message sending time. If you still cannot retrieve the data, expand the time range and try again.
4. If the preceding settings are correct but the tracing data is still not found, submit a ticket to get help from Customer Services, and attach the log file. The location of the log file is /home/{user}/logs/ons.log.

What should I do if the consumption information about a consumed message is not displayed in the trace data and the client IP address and producer ID are incorrect

This problem occurs because the client is not upgraded to the version that supports the message tracing function. Therefore, the MQ trace query backend can obtain only incomplete tracing data and the displayed result is abnormal. We recommend that you to update your client at your earliest convenience. For more information, see Message tracing.

Why is my group ID unavailable in the list of consumers

The possible cause is that there are too many downstream subscribers of the messages, and the

space in the tracing map is insufficient to display all of them. Move your pointer to the scroll bar, and scroll it to see complete data.

Why are previous query tasks not displayed

Too many historical query tasks will affect the display result. To avoid this, MQ regularly cleans up historical query tasks, and only saves query tasks within seven days. If you cannot find a historical task, query again.

Alarms

What should I do after receiving a subscription inconsistency alarm

Symptoms

Consumers using the group ID have not received certain messages that they subscribed to. Log on to the MQ console. Choose **Message Query** > **By Message ID**. The queried results show that the message has been consumed at least once, but the message logic shows that the message is not consumed.

Log on to the MQ console. Choose **Consumers** > **Consumer Status**. **Subscription Consistency** is set to **No**.

Analysis

One group ID in MQ indicates a consumer instance group. For most distributed applications, multiple consumer instances are often mounted to the same group ID. Subscription consistency means that the topics and tags of all consumer instances under the same group ID must be exactly the same.

If the consumer instances under the same group ID are configured with different topics, or configured with different tags under the same topic, the subscription relationships will be inconsistent. If the subscription relationships are inconsistent, the message consumption logic will be confusing, and even lead to message loss.

[Cause 1]

Clients using the same group ID have subscribed to messages with different topics.

Example

Clients using the same group ID (GID-MQ-FAQ) subscribed to messages with different topics (MQ-FAQ-TOPIC-1 and MQ-FAQ-TOPIC-2).

Code on JVM-1

```
Properties properties = new Properties();
properties.put(PropertyKeyConst.GROUP_ID, "GID-MQ-FAQ");
Consumer consumer = ONSFactory.createConsumer(properties);
consumer.subscribe("MQ-FAQ-TOPIC-1", "NM-MQ-FAQ", new MessageListener() {
public Action consume(Message message, ConsumeContext context) {
System.out.println("Receive: " + message);
return Action.CommitMessage;
}
});
consumer.start();
```

Code on JVM-2

```
Properties properties = new Properties();
properties.put(PropertyKeyConst.GROUP_ID, "GID-MQ-FAQ");
Consumer consumer = ONSFactory.createConsumer(properties);
consumer.subscribe("MQ-FAQ-TOPIC-2", "NM-MQ-FAQ", new MessageListener() {
public Action consume(Message message, ConsumeContext context) {
System.out.println("Receive: " + message);
return Action.CommitMessage;
}
});

consumer.start();
```

[Cause 2]

Clients using the same group ID subscribed to messages with different tags under the same topic.

Example

Clients using the same group ID (GID-MQ-FAQ) subscribed to messages with different topics (MQ-FAQ-TOPIC-1 and MQ-FAQ-TOPIC-2).

Code on JVM-1

```
Properties properties = new Properties();
properties.put(PropertyKeyConst.GROUP_ID, "GID-MQ-FAQ");
Consumer consumer = ONSFactory.createConsumer(properties);
consumer.subscribe("MQ-FAQ-TOPIC-1", "NM-MQ-FAQ", new MessageListener() {
public Action consume(Message message, ConsumeContext context) {
System.out.println("Receive: " + message);
return Action.CommitMessage;
}
});
consumer.start();
```

Code on JVM-2

```
Properties properties = new Properties();
properties.put(PropertyKeyConst.GROUP_ID, "GID-MQ-FAQ");
Consumer consumer = ONSFactory.createConsumer(properties);
consumer.subscribe("MQ-FAQ-TOPIC-2", "NM-MQ-FAQ", new MessageListener() {
public Action consume(Message message, ConsumeContext context) {
System.out.println("Receive: " + message);
return Action.CommitMessage;
}
});

consumer.start();
```

Solution

Perform the following steps:

Check the subscription codes of different clients. Ensure that the subscription relationships (including the subscribed topics and tags) of all clients with the same group ID are consistent.

Restart all client applications.

Verification

Consumers can receive messages as expected.

Log on to the MQ console. Choose **Consumers** > **Consumer Status**. **Subscription Consistency** is set to **Yes**.

How do I add a message accumulation alarm on MQ

Log on to theMQ console.

In the left-side navigation pane, choose **Monitoring**.

On the page that is displayed, click **Add Monitoring Item** in the upper-right corner. Select the group ID and topic for monitoring, and set the alarm threshold and alarm recipient information.

For more information, see Monitoring.

What should I do if I receive a message accumulation alarm

Symptoms

Log on to the MQ console. Choose **Consumers** > **Consumer Status**. **Accumulation Amount** is set to a larger value.

Log on to the MQ console. Choose **Message Tracing** > **Create Query Task** > **By Message ID**. It is found that certain messages have been sent to the broker, but have not been delivered to the downstream consumers.

Analysis

After MQ messages are sent to the broker, the client that is configured with the group ID pulls certain messages from the broker to the local machine for consumption based on the current consumption

point. In the consumption process, it may take a long time to consume a single message due to various reasons, such as access to the locked shared resources, competition for the I/O and network resources, and no timer configured for HTTP calls. As a result, messages start to accumulate on the broker.

Solution

Troubleshoot the problem as follows:

Log on to the MQ console. Choose **Statistics** > **Message Consumption** to query historical consumption records. If message write is faster than message consumption, adjust the code or expand the capacity of the consumer.

Print the Jstack information "jstack -l {PID} | grep ConsumeMessageThread" in the application. If messages are blocked, print the Jstack information consecutively five times, and identify the place where the consumption thread is stuck. Rectify the fault, restart the application, and check whether message consumption has recovered.

If messages are not accumulating any more, check whether the threshold value is too small and causes message accumulation. Choose **Monitoring** > **Monitoring Items**, and click **Edit** to increase the alarm threshold for message accumulation.

Verification

Print the Jstack information "jstack -l {PID} | grep ConsumeMessageThread" in the application. The consumption thread is not blocked.

Log on to the MQ console. Choose **Consumers** > **Consumer Status**. The value of **TPS** increases and the value of **Accumulation Amount** decreases.

Billing

Why am I charged RMB 2 on a daily basis?

MQ fees. MQ fees equal the total value of the API calling fee and the topic resource

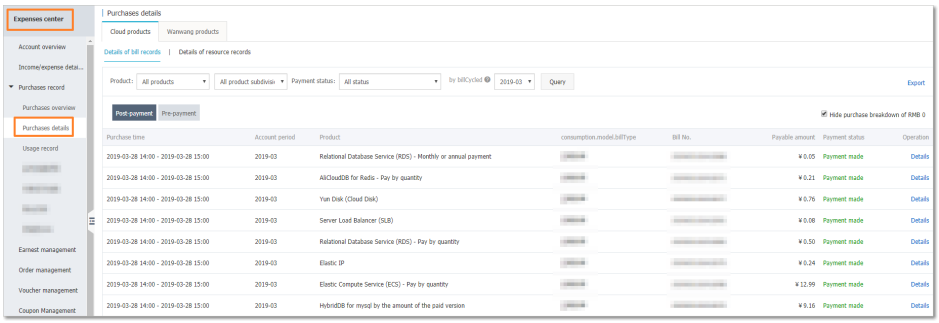
occupation fee. The resource occupation fee for each topic is RMB 2/day.

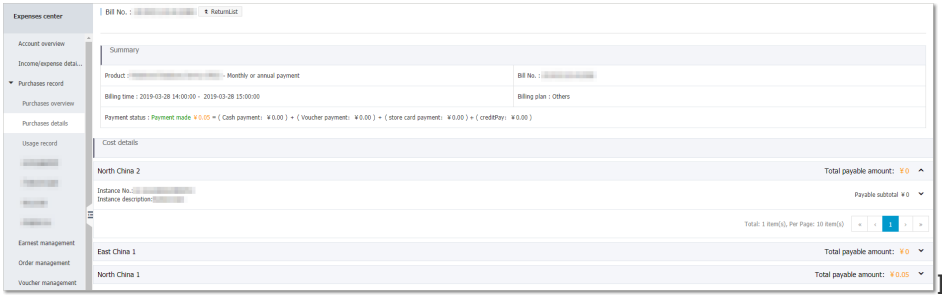
Why did I receive a bill and get charged today even though I already deleted the topic yesterday?

The topic resource occupation fee is calculated from 00:00 to 23:59:59 each day, and is billed on the next day. Therefore, the topic that you deleted yesterday had already been counted in the billing system, so you received a bill today. You will not receive a bill tomorrow.

Why did I get charged even though I did not use the MQ service?

Check the bill to see if you have used any MQ service.





If you do not need to use the MQ service any more, delete all resources in the MQ console to avoid unnecessary expenditures.

Why did I get charged even though I did not activate MQ in the console?

When your MQ instance has been in arrears for more than 72 hours, Alibaba Cloud will suspend the service. Your access to the MQ console and MQ API will be denied. However, you still have to pay the arrears incurred before you release the MQ service.

How do I disable the MQ service?

Delete the topic resources in all regions, and disable all producers and consumers.

The total number of messages in one day was 631,238, but according to the bill, the

number of API calls was 126,315,056. Where did these API calls come from?

Number of API calls = Number of API calls to transmit messages + Number of API calls to subscribe to messages + Number of long polling API calls. Note: A long polling request is an API call generated by the MQ consumer to push a message in real time. Each queue generates a long polling request every 15 seconds. If a message is generated within these 15 seconds, the long polling request will not be counted.

Ordered messages

Do ordered messages support clustering consumption and broadcasting consumption

Ordered messages support clustering consumption other than broadcasting consumption.

Can a message be an ordered message, a scheduled message, and a transactional message at the same time

No. Ordered messages, scheduled messages, and transactional messages are different and mutually exclusive message types.

Which regions are supported by ordered messages

Ordered messages support all public cloud regions and AntCloud regions of MQ.

What is the application scope of ordered messages

For the usage scope of ordered messages, including applicable scenarios and restrictions, see Ordered messages.

Why is the performance of globally ordered messages unsatisfactory

Globally ordered messages are processed in strict compliance with the FIFO message blocking principle. If a message is not consumed successfully, the next message will be stored in the topic queue forever. To improve the TPS of globally ordered messages, upgrade the instance configuration and reduce the time required by the application on the message client to process the local service logic as far as possible.

What methods are supported for sending ordered messages

Ordered messages only support reliable synchronous transmission methods.

Troubleshooting

Usage exception

The broker connection failed

Possible causes

Your Alibaba Cloud ECS instance and the MQ broker are not in the same region.

You may have accessed the MQ service on a non-Alibaba Cloud ECS instance, and the topic you created does not support access by non-Alibaba Cloud ECS instances.

Recommended solution

Perform the following steps:

Ensure that the ECS instance and the topic are in the same region.

If you access MQ on non-Alibaba Cloud ECS instances, make sure that the region of the topic is **Internet**.

The producer and consumer startup failed and the group IDs are duplicates

Possible causes

Multiple producer or consumer instances are started in the same JVM process, and these instances are configured with the same group ID, resulting in client startup failure.

Recommended solution

Perform the following steps:

Make sure that only one producer or consumer instance under the same group ID is started in a JVM process. In other words, one producer instance and one consumer instance under the same group ID can be started simultaneously in a JVM process, but multiple producer or consumer instances are not allowed to be started simultaneously.

Restart the application.

In broadcasting consumption mode, an error occurred while loading the JSON file for consumer starting

Possible causes

The Fastjson version is outdated. Consequently, the consumer in the broadcasting consumption mode fails to download the local offsets.json file and Fastjson cannot be started.

Recommended solution

Upgrade Fastjson to the version supported by ons-client to ensure that the local offsets.json file can be normally downloaded. By default, offsets.json is located in the /home/{user}/.rocketmq_offsets/ directory.

Failed to obtain the queue list during active message subscription

Possible causes

You did not create this topic in the console. As a result, the subscriber failed to retrieve the topic queue information when starting.

Recommended solution

Perform the following steps:

Log on to the MQ console and select **Topics** from the left nagivation pane. Click **Create**

Topic and create a topic as prompted.

select **Groups** from the left nagivation pane. Click **Create Group ID** and create a group ID as prompted.

Restart the application.

Cannot find name server is displayed

In the event of onsaddr configuration error, the following error will be returned in the log:

"Exception in thread "main" com.aliyun.openservices.ons.api.exception.ONSClientException: Cannot find name server. Please check your network connection."

In this case, perform the following steps:

Check if the deployment restriction has been violated. For more information, see the Note in the Create Topic in **Step 2: Create Resources** in Quick start for primary accounts.

Check the connection between the local environment and the endpoint.

- **If the topic is in the Internet:**

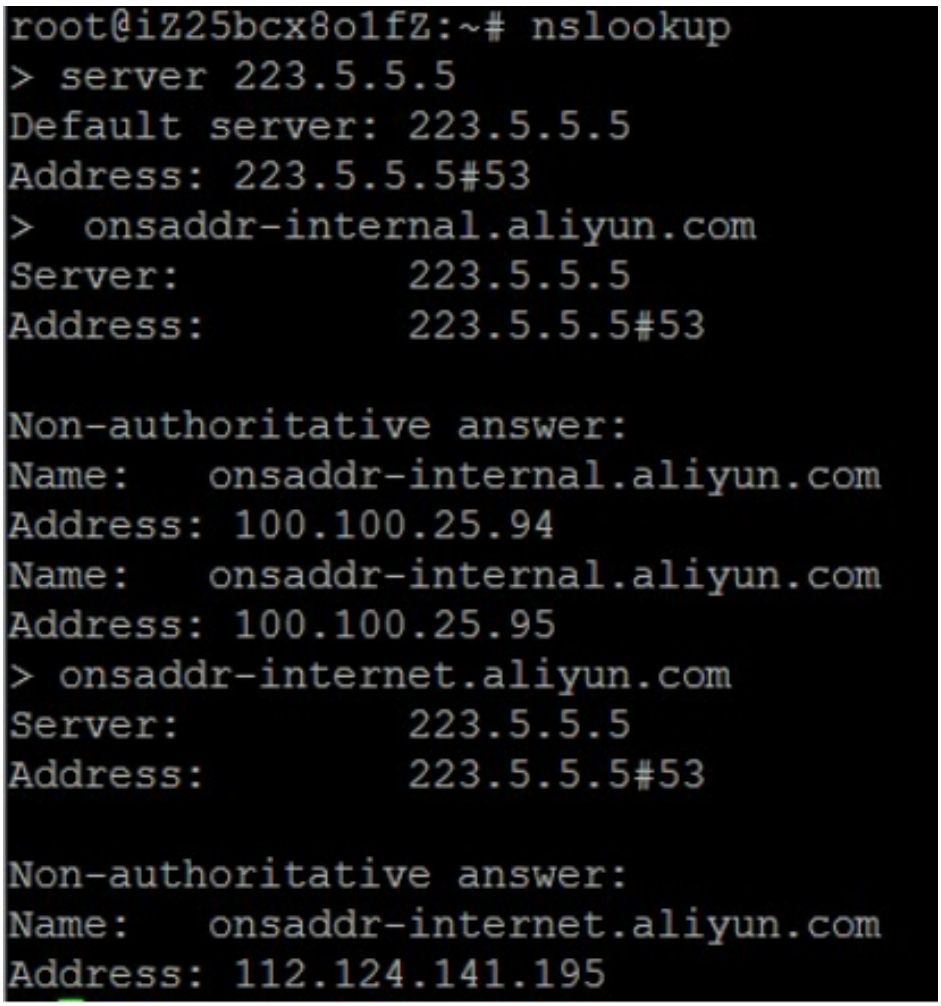
Action: ping **onsaddr-internet.aliyun.com.**

Normally this is resolved to 112.124.141.195.

If the topic is in a production environment:

Action: ping **onsaddr-internal.aliyun.com**

Normally this is resolved to 100.100.25.94/95. For example, if the IP address of the endpoint cannot be resolved, add DNS 223.5.5.5 on the local machine, after which, you can see:



You cannot set proxy for MQ. If you are using the Internet, you must add the following addresses (ports 80 and 8080) to the whitelist when applying for the security policy:

112.124.141.191

112.124.141.195

115.28.250.94

115.28.250.95

1. Use cURL to obtain the metadata information about the name server from the endpoint.

If the topic is in a production environment:

curl http://onsaddr-internal.aliyun.com:8080/rocketmq/nsaddr4client-internal

It is normal if 100.100.26.1:8080;100.100.26.2:8080;100.100.25.96:8080 is returned.

If the topic is in the Internet:

curl http://onsaddr-internet.aliyun.com/rocketmq/nsaddr4client-internet

It is normal if 112.124.141.191:80 is returned.

If the problem persists, submit a ticket.

The displayed message status is _Consumed_, but the consumer cannot view this status

The message status is “Consumed”, but the service log at the consumer shows that the message has not been received. The possible causes are as follows:

After receiving the message, the service code does not immediately print the message.

Assume that the service logic is directly executed after a message is received, and the code misses a certain logic branch. In this case, the message information will not be recorded in the log, which leads to the false symptom that the message is not received.

After receiving a message, we recommend that you print the message information immediately to keep the messageId, timestamp, and reconsumeTime.

Multiple consumer instances are deployed.

It is unavoidable that the consumer may be restarted multiple times, especially during the debugging stage. Once multiple consumption processes exist simultaneously (the previous process is not terminated before the next is started), multiple consumer instances will share the message information, similar to the clustering consumption mode. A message that fails to be received by a consumer is actually received by another consumer.

Log on to the MQ console. Choose **Consumers > Consumer Status > Connection Status**. The consumer instance deployment information is displayed, such as the number of instances and connection IP address of each instance. Check the displayed information and handle the fault accordingly.

In the consumption process, messages are not caught and therefore are re-delivered.

public class MessageListenerImpl implements MessageListener {@Overridepublic Action consume(Message message, ConsumeContext context) {//The message processing logic throws an exception. The message will retry.doConsumeMessage(message);//If an exception that has not been caught occurred in doConsumeMessage(), log of this row will not be printed.log.info(“Receive Message, messageId:” , message.getMsgID());

```
 return Action.CommitMessage;
}
```

}

If the problem persists, contact Alibaba Cloud After-Sales Technical Support and provide the local SDK log.

Resource not found

The group ID does not exist

Possible causes

The group ID has not been created in the console. As a result, when this group ID is used to connect to the broker, verification fails on the broker.

Recommended solution

Perform the following steps:

1. Log on to the MQ console and choose **Groups** and then create a group ID.
2. Restart the application.

The host name does not exist

Possible causes

A possible cause is that the host name or host IP address cannot be retrieved correctly. To verify this assumption, run the hostname command.

If the host name cannot be displayed normally, the assumption is correct. Otherwise, this issue may have been caused by another reason. Submit a ticket to seek help from Customer Services.

Recommended solution

Perform the following steps:

On the machine for which the error is reported, run the following command to check the host name.

```
[root@iZ231wxgt6mZ ~]# hostnameiZ231wxgt6mZ
```

If an error is reported, check if an alias has been defined for the host name, for example, alias xxx= 'hostname' in .bash_profile or .bashrc. It is also possible that the command path is not under $PATH.

2. Ping the host.

```
[root@iZ231wxgt6mZ ~]# ping iZ231wxgt6mZ
```

If the host name cannot be pinged, bind the local IP address to the /etc/hosts file. Each ECS instance has a binding relationship between the local IP address and the host name by default. Do not remove it manually.

Check the system configuration.

Check if the host name in /etc/sysconfig/network is consistent with the host name bound in /etc/hosts. If it is inconsistent, modify the host name. After the content in /etc/sysconfig/network has been modified, you must restart the instance so that the modification can take effect. Exercise caution when you modify configurations in a system file, as doing so may cause other exceptions.

After these three steps have been performed, the error UnknownHostException will not be returned again when the client starts.

Inconsistent statuses

An unexpected client is connected

Symptoms

According to the feedback from the user side, the consumer has not received several messages that have been sent. Log on to the MQ console. Choose **Message Tracing** > **Create Query Task** > **By Message ID**. It is found that certain messages have been sent to the broker, but they have not been delivered to the downstream consumers.

Log on to the MQ console. Choose **Consumers** > **Consumer Status**. Client IP addresses not within the expected range are displayed under **Connection Status** , and certain messages are accumulated on clients that are only within the unexpected range.

Analysis

Log on to the MQ console. Choose **Consumers** > **Consumer Status**. The connection status of all clients under the current group ID is displayed under **Connection Status**. The IP addresses and process IDs of unexpected clients can be located. Verify that the configuration loaded by the corresponding process is correct (including AccessKeyId, AccessKeySecret, Topic, and Group ID). Otherwise, the client process may occupy certain queues, and messages cannot be consumed correctly.

Cause

In the same environment, if a client under a group ID is started by using the incorrect configuration (the incorrect AccessKeyId, AccessKeySecret, or Topic), this client process may occupy certain queues under the topic and messages cannot be consumed properly. As a result, messages are accumulated on the broker and cannot be delivered to the correct downstream consumers in time.

Verification method

Locate the faulty process based on the connection status, and then check the configuration of AccessKeyId, AccessKeySecret, or Topic of the process by using /{user.home}/logs/ons.log or program code.

Fixing method

Close the faulty process. Then, accumulated messages will be immediately rebalanced and delivered to the correct clients. After the fault is rectified, restart the faulty process. (Fast recovery)

Verification

Log on to the MQ console.

Choose **Consumers** > **Consumer Status**. **All connected clients displayed under Connection Status** are as expected and can consume messages normally. In addition, **Subscription Consistency** is set to **Yes**.

The message is invalid

Possible causes

The message attribute or content is invalid, including:

- The message is null.
- The message content is empty.
- The message content length is 0.
- The message content length exceeds the limit.

Recommended solution

Check whether the preceding exceptions exist in the message and handle the exceptions as prompted.

The parameter is invalid

Possible causes

The following table lists the cases in which the parameters are invalid:

Nested exception	Exception description
consumeThreadMin Out of range [1, 1000]	The number of threads set on the consumer is

	inappropriate.
consumeThreadMax Out of range [1, 1000]	The number of threads set on the consumer is inappropriate.
messageListener is null	messageListener is not set.
consumerGroup is null	No group ID is set.
msg delay time more than 40 day	The delay for a scheduled message cannot exceed 40 days.

Recommended solution

Perform the following steps:

Modify the parameter settings on the client according to the exception prompt and ensure that parameter settings are in the valid ranges.

Restart the application.

The client is abnormal

Possible causes

- After the consumer and producer instances are created, the start() method is not called to start the client.
- After the consumer and producer instances are created, startup of the client fails due to an exception in the start() process.
- After creating the consumer and producer instances, start() is successfully called, but the client is shut down by calling shutdown().

Recommended solution

Perform the following steps:

1. Check that the start() method is called after the consumer and producer instances are created. Ensure that the client is started.
2. Check ons.log for exceptions that occur in the startup process of the client.

The subscriptions are inconsistent

Symptoms

Multiple consumers are started on different JVMs. For consumer instances under the same group ID, different topics are configured, or topics are the same but tags are different. As a result, the subscription relationships are inconsistent, and messages do not meet the expectations.

Sample of the incorrect code

Sample 1: Clients using the same group ID (GID-MQ-FAQ) subscribed to messages with different topics (MQ-FAQ-TOPIC-1 and MQ-FAQ-TOPIC-2).

Code on JVM-1:

```
Properties properties = new Properties();
properties.put(PropertyKeyConst.GROUP_ID, "GID-MQ-FAQ");
Consumer consumer = ONSFactory.createConsumer(properties);
consumer.subscribe("MQ-FAQ-TOPIC-1", "NM-MQ-FAQ", new MessageListener() {
public Action consume(Message message, ConsumeContext context) {
System.out.println("Receive: " + message);
return Action.CommitMessage;
}
});
consumer.start();
```

Code on JVM-2

```
Properties properties = new Properties();
properties.put(PropertyKeyConst.GROUP_ID, "GID-MQ-FAQ");
Consumer consumer = ONSFactory.createConsumer(properties);
consumer.subscribe("MQ-FAQ-TOPIC-2", "NM-MQ-FAQ", new MessageListener() {
public Action consume(Message message, ConsumeContext context) {
System.out.println("Receive: " + message);
return Action.CommitMessage;
}
});

consumer.start();
```

Example 2: Consumer instances using the same group ID (GID-MQ-FAQ) subscribed to messages with different tags (NM-MQ-FAQ-1 and NM-MQ-FAQ-2) of the same topic.

Code on JVM-1:

```
Properties properties = new Properties();
properties.put(PropertyKeyConst.GROUP_ID, "GID-MQ-FAQ");
Consumer consumer = ONSFactory.createConsumer(properties);
consumer.subscribe("MQ-FAQ-TOPIC-1", "NM-MQ-FAQ-1", new MessageListener() {
public Action consume(Message message, ConsumeContext context) {
System.out.println("Receive: " + message);
return Action.CommitMessage;
```

```
}
});
consumer.start();
```

Code on JVM-2:

```
Properties properties = new Properties();
properties.put(PropertyKeyConst.GROUP_ID, "GID-MQ-FAQ");
Consumer consumer = ONSFactory.createConsumer(properties);
consumer.subscribe("MQ-FAQ-TOPIC-1", "NM-MQ-FAQ-2", new MessageListener() {
public Action consume(Message message, ConsumeContext context) {
System.out.println("Receive: " + message);
return Action.CommitMessage;
}
});

consumer.start();
```

Recommended solution

When the same group ID is used to start multiple consumer instances on different JVMs, ensure that the topics and tags configured for these consumer instances are consistent.

Log reference

How can I determine the current status based on client logs

The client log file of MQ is ons.log, including INFO, WARN, and ERROR logs.

This topic describes common client log information, which helps you obtain information from the printed logs and determine the current status in order to troubleshoot faults.

The following table lists log information in the ons.log file (continuously updated).

Log level	Printed information	Description	Solution
INFO	[persistAll] Group: CID_XXXX ClientId: 10.31.40.100@171374#14159XXX#-2036649XXX#20931314294957XXX	This indicates that the message has been consumed and the consumption progress has been made persistent on	N/A

	updateConsumeOffsetToBroker MessageQueue [topic=XXXX, brokerName=qdinternetorder-XX, queueId=X] 1013XXX	the MQ broker. MessageQueue includes the message topic, corresponding broker name, and consumption queue ID.	
INFO	[PULL_TPS] [CID_XXXX@CID_XXXX] Stats In One Minute, SUM: 0 TPS: 0.00 AVGPT: 0.00</br>[PULL_RT] [%RETRY%CID_XXXX@CID_XXXX] Stats In One Minute, SUM: 0 TPS: 0.00 AVGPT: 0.00	This printed information displays the TPS when messages are pulled by calling consumeQueue.	N/A
WARN	[TIMEOUT_CLEAN_QUEUE]broker busy, start flow control for a while, period in queue: 905ms, size of queue: 1164	The MQ broker is under high pressure and cannot process so many requests. The broker writes pageCache first and then refreshes the disk when storing data. Therefore, the expired requests are cleared every 10 seconds (this process can determine whether the cache page is busy).	1. Scale out by increasing brokers to share the pressure, and 2. increase the value of osPageCacheBusyTimeOutMills.
WARN	execute the pull request exception com.aliyun.openservices.shade.com.alibaba.rocketmq.client.exception.MQBrokerException: CODE: 25 DESC: the consumer's subscription not latest	The broker reports its router information to NameServer at a preset interval. If the network jitters and the latest subscription information cannot be obtained during this process, this warning will appear when consumers need to consume messages.	N/A
WARN	[WRONG]mq is consuming, so can not unlock it,	During load balancing, the message queue is	N/A

	MessageQueue [topic=XX, brokerName=szorder2-02, queueId=1]. maybe hanged for a while, 2	attempted to be locked. If the lock is not successful within one second, the current message queue has been accessed by other consumers and cannot be unlocked.	
WARN	doRebalance, XXX-CID, add a new mq failed, MessageQueue [topic=XXXX, brokerName=szorder2-XX, queueId=X], because lock failed	An ordered topic is used currently. To ensure the ordered consumption of messages in a single shard, a lock mechanism is enabled. If this log is printed for a client, consumers have started to consume messages in a shard.	N/A
WARN	get Topic [XXXXXX] RouteInfoFromNameServer is not exist value com.aliyun.openservices.shade.com.alibaba.rocketmq.client.exception.MQClientException: CODE: 17 DESC: No topic route info in name server for the topic: TOPIC_XXXXX</br> See http://rocketmq.apache.org/docs/faq/ for further details.	- Incorrect AccessKey (containing AccessKeyId and AccessKeySecret) - No group ID created under the current instance in the console - Incorrect NameServer Addr in instantiation code	- Configure the correct AccessKey - . Create a group ID under the current instance - . We recommend that you configure NameServer Addr for Java SDK 1.8.0 and later versions. The parameter can be obtained from the MQ console, which is

			inconsistent with the ONSAddr configured in earlier versions.
WARN	com.aliyun.openservices.ons.api.impl.authority.exception.AuthenticationException: signature validate by dauth failed	Incorrect AccessKey (containing AccessKeyId and AccessKeySecret)	Configure the AccessKey used to create the group ID.
WARN	NettyClientPublicExecutor_3 - execute the pull request exception com.aliyun.openservices.shade.com.alibaba.rocketmq.client.exception.MQBrokerException: CODE: 26 DESC: subscription group [CID_XXX] does not exist, See http://rocketmq.apache.org/docs/faq/ for further details.	The subscription relationship is not pushed to the RocketMQ broker.	Add the group ID information to the subscription.json file.
WARN	execute the pull request exception com.aliyun.openservices.shade.com.alibaba.rocketmq.client.exception.MQBrokerException: CODE: 24 DESC: the consumer's subscription not exist	The subscription relationship is missing.	N/A

Insufficient application memory

Symptoms

The memory is exhausted on the machine where the application is deployed.

The keyword **OutOfMemory** can be found in /{user.home}/logs/ons.log.

According to the user-side feedback, in the MQ console, choose **Consumers** > **Consumer Status**.**Accumulation Amount** shows that many messages have accumulated, and **Connection Status** shows that message accumulation has occurred on all the connected clients. Troubleshooting through Jstack shows that the ConsumeMessageThread _ thread is not blocked.

Analysis

A MQ consumer actively pulls messages from the broker, caches them to the client, and then delivers the messages to the business consumption logic.

In versions earlier than 1.7.0. Final, the client caches a maximum of 1000 messages to each queue of each topic by default. Assume that each topic has 16 queues (two primary clusters and two standby clusters, and eight queues on each broker) and that the average size of messages ordered by this topic is 64 KB. The size of the messages cached by this topic on the client is 16 * 1000 * 64 KB = 1 GB. If the user subscribes to eight topics at the same time and caches messages in the client memory, the memory usage will eventually exceed the user’ s JVM configuration, resulting in OOM.

Cause 1

Depending on ons-client versions earlier than 1.7.0. Final, the average size of messages for each topic exceeds 4 KB. In addition, message consumption is slow, which easily leads to cache of messages in the client memory.

Verification method

Check whether the keyword **OutOfMemory** can be found in /{user.home}/logs/ons.log, or run the command jmap -dump:live,format=b,file=heap.bin <pid> to verify the objects that occupy a large amount of memory.

Recovery solution

Upgrade ons-client to 1.7.0. Final or later, configure the com.aliyun.openservices.ons.api.PropertyKeyConst#MaxCachedMess ageSizeInMiB parameter for the corresponding ConsumerBean. Then, restart the application.

Cause 2

Depending on ons-client 1.7.0. Final or later, the maximum memory consumption is 512 MB by default (that is, the total cache size of all topics subscribed by a group ID). If the application still has

the OOM problem, configure the com.aliyun.openservices.ons.api.PropertyKeyConst#MaxCachedMessageSizeInMiB parameter to customize the maximum memory consumption (ranging from 16 MB~2,048 MB) upon startup of ConsumerBean.

Verification method

Check the ons-client version used by the application and use the JVM to confirm the size of memory allocated to the process.

Recovery solution

Based on the memory usage of the machine where the application runs, configure the com.aliyun.openservices.ons.api.PropertyKeyConst#MaxCachedMess ageSizeInMiB parameter for the corresponding ConsumerBean. Then, restart the application.

Verification

The keyword **OutOfMemory** cannot be found in /{user.home}/logs/ons.log.

Log on to the MQ console. Choose **Consumers** > **Consumer Status**. The value of **TPS** increases and the value of **Accumulation Amount** decreases.