TDMQ for CKafka

Getting Started

Product Documentation

©2013-2025 Tencent Cloud. All rights reserved.

Copyright Notice

©2013–2025 Tencent Cloud. All rights reserved.

Copyright in this document is exclusively owned by Tencent Cloud. You must not reproduce, modify, copy or distribute in any way, in whole or in part, the contents of this document without Tencent Cloud's the prior written consent.

Trademark Notice

Tencent Cloud

All trademarks associated with Tencent Cloud and its services are owned by the Tencent corporate group, including its parent, subsidiaries and affiliated companies, as the case may be. Trademarks of third parties referred to in this document are owned by their respective proprietors.

Service Statement

This document is intended to provide users with general information about Tencent Cloud's products and services only and does not form part of Tencent Cloud's terms and conditions. Tencent Cloud's products or services are subject to change. Specific products and services and the standards applicable to them are exclusively provided for in Tencent Cloud's applicable terms and conditions.

Contents

- Getting Started
  - Process Overview
  - Preparations
  - VPC Access
    - Step 1. Create an Instance
    - Step 2. Add a VPC Route
    - Step 3. Create a Topic
    - Step 4. Send/Receive Messages
      - Using SDK to Receive/Send Message (Recommended)
      - Running Kafka Client (Optional)
    - Step 5: Query Messages
    - Step 6: Resource Destruction
  - Public Domain Name Access
    - Step 1: Create an Instance
    - Step 2: Add Public Network Route
    - Step 3: Create Topic
    - Step 4: Configuring ACL Policy
    - Step 5: Send/Receive Messages
      - Using SDK to Send and Receive Messages (Recommended)
      - Running the Kafka Client (Optional)
    - Step 6: Query Messages
    - Step 7: Resource Destruction

©2013-2025 Tencent Cloud. All rights reserved.

Getting Started
Process Overview

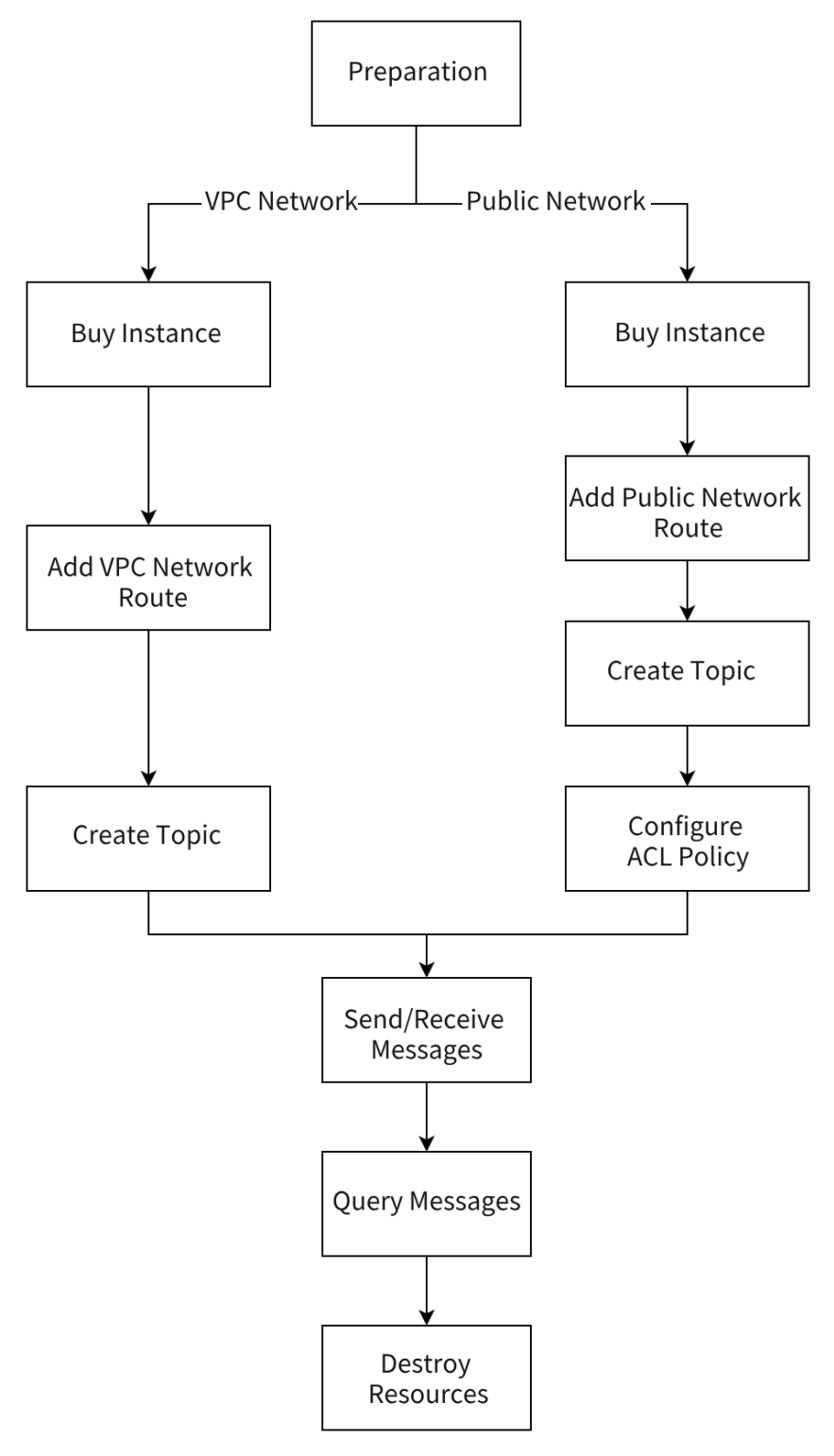
Last updated：2025-08-13 10:57:40

CKafka instances support two connection methods: private network access and public network access. The access process may vary slightly depending on the network type.

- For private network access, if the client and CKafka instance are deployed in the same VPC, the network has default mutual communication with no additional configuration required.
- For public network access, a separate public network route must be enabled, and an ACL policy must be set for the Topic to ensure the user has read/write permissions for the corresponding Topic.

Flowchart

©2013-2025 Tencent Cloud. All rights reserved.



©2013-2025 Tencent Cloud. All rights reserved.

Preparations

Last updated：2025-08-13 14:34:30

This article introduces the preparations before getting started with CKafka, including signing up, obtaining permissions, and preparing VPC resources.

Operation Steps

1. Register a Tencent Cloud account and verify identity.
   - Before creating a CKafka cluster, sign up for a Tencent Cloud account and complete real-name authentication. For specific steps, refer to Register a Tencent Cloud account and Introduction to real-name authentication.
   - If you already have a Tencent Cloud account and completed real-name authentication, please skip this step.
2. Recharge your account.
   - Before creating a CKafka cluster, ensure your account has sufficient amount to pay for the cluster purchase. For specific steps on account recharge, refer to online recharge.
   - If your account has sufficient amount, please skip this step.
3. Configure operation permissions for CKafka in your account.
   - If you use a Tencent Cloud sub-account, refer to Permission Management Overview to grant the sub-account operation permissions for the CKafka cluster.
   - If you use a Tencent Cloud root account, you have operation permissions for all resources by default and can skip this step.
4. Create a VPC and subnet.
   - Before creating a CKafka cluster, please ensure an available VPC and subnet already exist. For specific steps to create a VPC and subnet, see Create VPC and Subnet. The created VPC and CKafka cluster must be in the same region, as the private networks of cloud products in different regions are not interconnected.
   - If you already have a suitable VPC and subnet, skip this step.
5. Set up the environment.
   - VPC

     Prepare a Linux server: This document uses Tencent Cloud Virtual Machine (CVM) as an example. For specific steps to create a CVM, see Purchase Cloud Virtual Machine (CVM). **Please ensure the CVM and CKafka cluster are in the same VPC**. If you have an existing CVM, skip this step.
     Set up environment: Log in to the CVM and install Java JDK version 1.8 or later and Maven version 2.5 or later.
   - public network environment

©2013-2025 Tencent Cloud. All rights reserved.

Can be executed locally. Install Java JDK version 1.8 or later and Maven version 2.5 or later in advance.

©2013-2025 Tencent Cloud. All rights reserved.

VPC Access
Step 1. Create an Instance

Last updated：2025-08-13 14:55:58

Overview

This article introduces the directions to create a CKafka cluster through CKafka console.

Prerequisites

Completed Preparations.

Directions

1. Log in to the CKafka console.
2. Select **Instance List** on the left sidebar, click **Create** to go to the instance purchase page, and enter the purchase information as needs.

Configuration Item	Parameter	Parameter Description
Basic Configuration	product form	Select **Serverful**.
	Billing Mode	Supports **Monthly Subscription** and **Pay-As-You-Go** modes. • **Monthly Subscription:** Payment is required before use resources, mainly suitable for business with relatively stable and long-term usage scenarios. • **Pay-As-You-Go:** Use resources first and pay later, mainly suitable for testing or short-term situations with uncertain peak traffic.
	Cluster type	• Pro Edition is primarily aimed at large-scale production environment customers. • Advanced Edition is primarily aimed at test environment customers in small-scale scenarios.
	Region	Select a region with resources close to those of the client deployment to reduce latency and improve access speed. Cloud products in different regions are not interconnected on a private network and cannot be changed after purchase. Please select carefully.

©2013-2025 Tencent Cloud. All rights reserved.

Cluster Configuration	Name	Optional. Not specified defaults to unnamed.
	Kafka version	Choose the appropriate Kafka version for your business requirements. The recommended choice is version **2.8.1**.
	Peak bandwidth	Choose the appropriate peak bandwidth for your business requirements.
	Disk	Currently supported disk types are SSD Cloud Block Storage and high-performance cloud block storage.
	Partition specification	Partition limit for CKafka instance. The number of partitions included in the package (minimum value) is no charge. Additional partitions are billed per 100 units.
	Message retention	Ranges from 24 to 2160 hours. The default message retention time is 72 hours. Messages will be deleted to preserve sufficient disk space when the retention duration is exceeded.
	Cross-AZ Deployment	The Pro Edition supports up to 4 availability zones for deployment, while the Advanced Edition supports up to 2.
Network Configuration	VPC Network	Bind the domain name of the newly purchased cluster's access point to the prepared private network within the same region.
other configuration	Tag	Option. Tags are used to categorize and manage resources from different dimensions.

3. Confirm the cluster information is correct and click **Purchase Now**. After completing payment on the order payment page, wait 3–5 minutes to see the created cluster on the cluster list page.

©2013-2025 Tencent Cloud. All rights reserved.

Step 2. Add a VPC Route

Last updated：2025-08-13 14:55:58

Overview

After the instance is created, you need to add a VPC network route to the instance. This article introduces the directions to add VPC network routing to the created instance through the CKafka console.

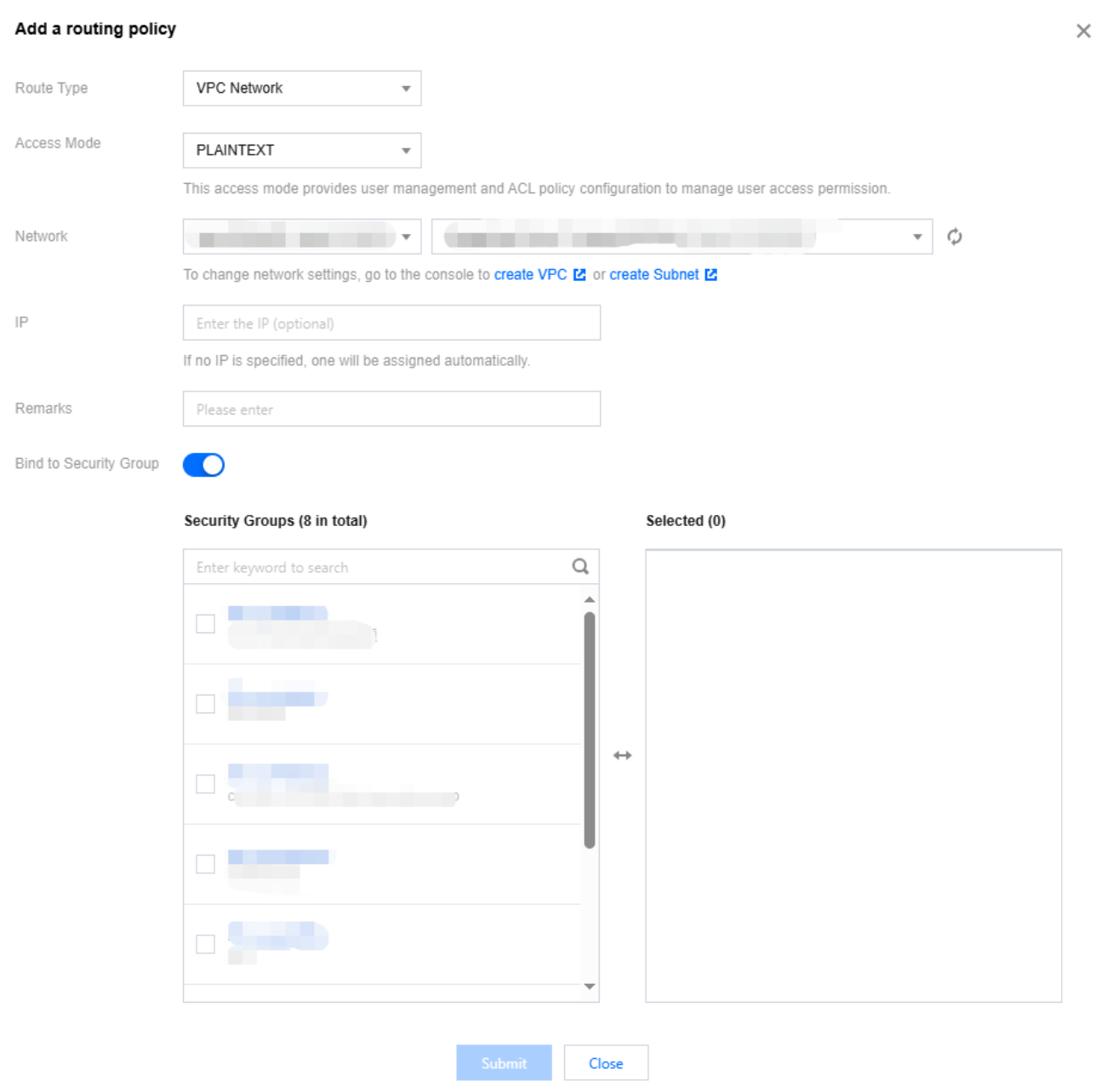
Prerequisites

You have created an instance. For more information, see Step 1. Create an Instance.

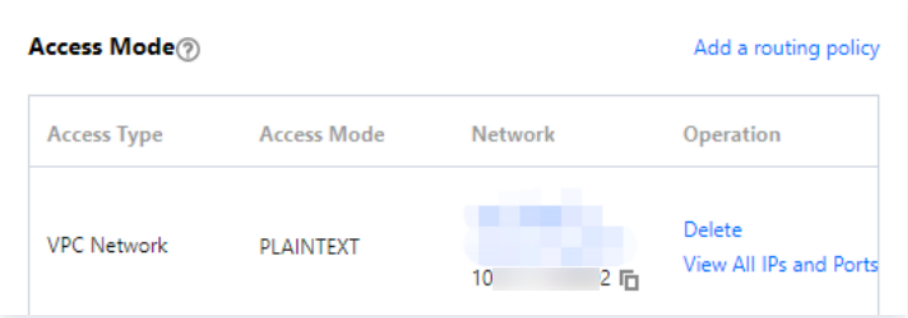
Directions

1. In the Instance list page of the CKafka console, click the "ID/Name" of the created in advance instance to enter the basic information page.
2. On the instance details page, click **Add a routing policy** in the **Access Mode** section to add a VPC route.

©2013-2025 Tencent Cloud. All rights reserved.



Then, you can get the domain name and port for VPC access.



©2013-2025 Tencent Cloud. All rights reserved.

Step 3. Create a Topic

Last updated：2025-08-13 14:55:58

Overview

Before using CKafka for message sending and receiving, you need to create a Topic in the created CKafka instance. The producer sends messages to the specified Topic, and the consumer obtains messages by subscribing to the Topic for consumption. This article introduces the steps to create a Topic in the CKafka console.

Prerequisites

Creating an Instance has been finished.

Directions

1. Log in to the CKafka console.
2. Select **Instance List** in the left sidebar, find the created instance, click the instance "ID", and go to the instance details page.
3. On the instance details page, click **Topic Management** at the top of the page, and click **Create**.
4. In the **Create Topic** dialog box, set parameters as needed replicas.

Parameter	Description
Name	Topic name cannot be changed after input. Name can only contain letters, digits, underscore, "-", and ".". Names starting with double underscore are not supported.
Number of Partitions	A Topic can contain one or more partitions. CKafka uses partition as the allocation unit. The default deployment architecture has at least 3 nodes. It is advisable to start with 3 partitions for more balanced data distribution. You can keep default values here.
Number of Replicas	The number of replicas for a Partition is used to ensure high availability of the Partition. To ensure data reliability, 2 replicas are enabled by default. You can keep default values here.
Tag	Tags are used to manage resources by category from different dimensions. Leave blank here.
retention.ms	Message retention time in the Topic dimension, ranging from 1 minute to 90 days. Keep default values here.
Advance	Advanced Topic parameter configuration. Skip here.

©2013-2025 Tencent Cloud. All rights reserved.

d Configuration	

5. Click **Submit**.

©2013-2025 Tencent Cloud. All rights reserved.

Step 4. Send/Receive Messages Using SDK to Receive/Send Message (Recommended)

Last updated：2025-08-13 14:55:58

Overview

After configuring resources such as instances and topics in the console, you can use the SDK Demo we provide to connect to the cluster and perform message sending and receiving tests. This document uses Java SDK call as an example to introduce the operation process of sending and receiving messages through the open source SDK, helping you better understand the whole process of sending and receiving messages.

Prerequisites

- Completed the initial CKafka resource creation.
- Completed the preparation of Linux server and environment configuration as per Preparations.

Directions

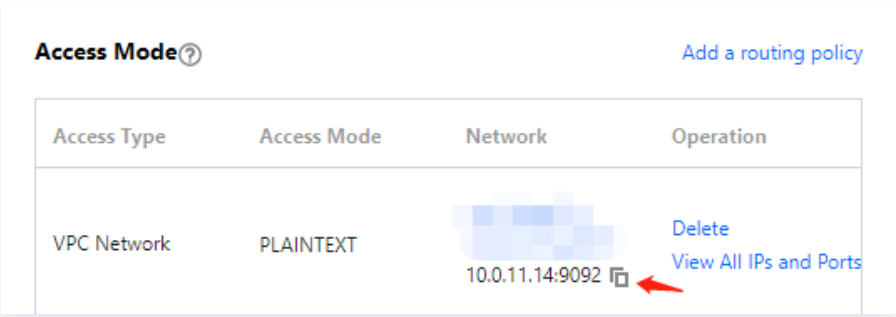
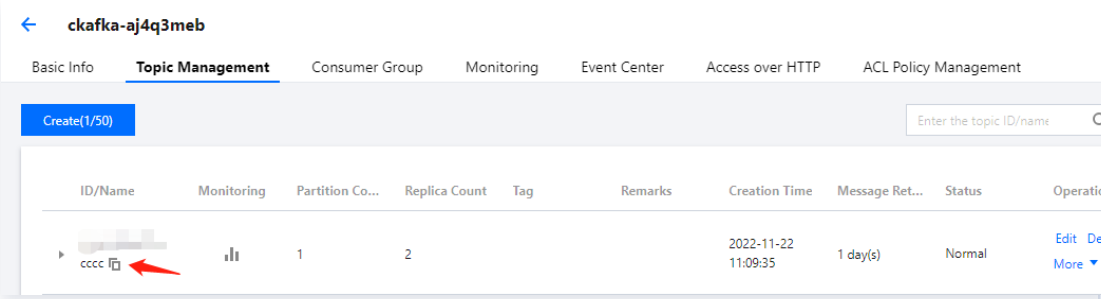
Step 1. Prepare configurations

1. Download Demo.
2. Upload the downloaded demo to the Linux server under the same VPC, log in to the server, and enter the VPC directory under `javakafkademo`.
3. Modify `kafka.properties` in the `resources` directory under the VPC project.

```
## Configure the accessed network by copying the information in the
**Network** column in the **Access Mode** section on the instance
details** page in the console.
bootstrap.servers=xx.xx.xx.xx:xxxx
## Configure the topic by copying the information on the **Topic
Management** page in the console
topic=XXX
## Configure the consumer group as needed
group.id=XXX
```

Parameter	Description

©2013-2025 Tencent Cloud. All rights reserved.

bootstrap.servers	Access network, which can be copied in the **Network** column in the **Access Mode** section on the instance details page in the console 
topic	Topic name, which can be copied on the **Topic Management** page in the console. 
group.id	You can customize it. After the demo runs successfully, you can see the consumer on the **Consumer Group** page.

Step 2. Send messages

1. Compile and run the message production program `CKafkaProducerDemo.java` .

```
public class CKafkaProducerDemo {

    public static void main(String args[]) {
        //Load `kafka.properties`
        Properties kafkaProperties =
CKafkaConfigurer.getCKafkaProperties();

        Properties properties = new Properties();
        //Set the access point. Obtain the access point of the
corresponding topic via the console
        properties.put(ProducerConfig.BOOTSTRAP_SERVERS_CONFIG,
kafkaProperties.getProperty("bootstrap.servers"));
```

©2013-2025 Tencent Cloud. All rights reserved.

```
        //Set the method for serializing Kafka messages.
`StringSerializer` is used in this demo.
        properties.put(ProducerConfig.KEY_SERIALIZER_CLASS_CONFIG,

"org.apache.kafka.common.serialization.StringSerializer");
        properties.put(ProducerConfig.VALUE_SERIALIZER_CLASS_CONFIG,

"org.apache.kafka.common.serialization.StringSerializer");
        //Set the maximum time to wait for a request
        properties.put(ProducerConfig.MAX_BLOCK_MS_CONFIG, 30 *
1000);
        //Set the number of retries for the client
        properties.put(ProducerConfig.RETRIES_CONFIG, 5);
        //Set the interval between retries for the client
        properties.put(ProducerConfig.RECONNECT_BACKOFF_MS_CONFIG,
3000);
        //Construct a producer object
        KafkaProducer<String, String> producer = new KafkaProducer<>
(properties);

        //Construct a Kafka message
        String topic = kafkaProperties.getProperty("topic"); //Topic
of the message. Enter the topic you created in the console
        String value = "this is ckafka msg value"; //Message content.

        try {
            //Batch obtaining future objects can speed up the process,
but the batch size should not be too large.
            List<Future<RecordMetadata>> futureList = new ArrayList<>
(128);
            for (int i = 0; i < 10; i++) {
                //Send the message and obtain a future object
                ProducerRecord<String, String> kafkaMsg = new
ProducerRecord<>(topic,
                        value + ": " + i);
                Future<RecordMetadata> metadataFuture =
producer.send(kafkaMsg);
                futureList.add(metadataFuture);
```

©2013-2025 Tencent Cloud. All rights reserved.

```
            }
            producer.flush();
            for (Future<RecordMetadata> future : futureList) {
                //Sync the future object obtained
                RecordMetadata recordMetadata = future.get();
                System.out.println("produce send ok: " +
recordMetadata.toString());
            }
        } catch (Exception e){
            //If the sending still fails after client internal
retries, the system needs to report and handle the error
            System.out.println("error occurred");
        }
    }
}
```

2. View the execution result.

```
Produce ok:ckafka-topic-demo-0@198
Produce ok:ckafka-topic-demo-0@199
```

Step 3. Consume messages

1. Compile and run the message subscription program `CKafkaConsumerDemo.java` .

```
public class CKafkaConsumerDemo {

    public static void main(String args[]) {
        //Load `kafka.properties`
        Properties kafkaProperties =
CKafkaConfigurer.getCKafkaProperties();

        Properties props = new Properties();
        //Set the access point. Obtain the access point of the topic
via the console.
        props.put(ProducerConfig.BOOTSTRAP_SERVERS_CONFIG,
kafkaProperties.getProperty("bootstrap.servers"));
        //Set the maximum interval between two polls
```

©2013-2025 Tencent Cloud. All rights reserved.

```
        //If the consumer does not return a heartbeat message within
the interval, the broker determines that the consumer is not alive.
the broker removes the consumer from the consumer group and triggers
rebalancing. The default value is 30s.
        props.put(ConsumerConfig.SESSION_TIMEOUT_MS_CONFIG, 30000);
        //Set the maximum number of messages that can be polled at a
time
        //Do not set this parameter to an excessively large value. If
polled messages are not consumed before the next poll, load balancing
is triggered and lagging occurs.
        props.put(ConsumerConfig.MAX_POLL_RECORDS_CONFIG, 30);
        //Set the method for deserializing messages
        props.put(ConsumerConfig.KEY_DESERIALIZER_CLASS_CONFIG,

"org.apache.kafka.common.serialization.StringDeserializer");
        props.put(ConsumerConfig.VALUE_DESERIALIZER_CLASS_CONFIG,

"org.apache.kafka.common.serialization.StringDeserializer");
        //The instances in the same consumer group consume messages in
load balancing mode
        props.put(ConsumerConfig.GROUP_ID_CONFIG,
kafkaProperties.getProperty("group.id"));
        //Construct a consumer object. This generates a consumer
instance
        KafkaConsumer<String, String> consumer = new KafkaConsumer<>
(props);
        //Set one or more topics to which the consumer group
subscribes
        //You are advised to configure consumer instances with the
same `GROUP_ID_CONFIG` value to subscribe to the same topics
        List<String> subscribedTopics = new ArrayList<>();
        //If you want to subscribe to multiple topics, add the topics
here
        //You must create the topics in the console in advance.
        String topicStr = kafkaProperties.getProperty("topic");
        String[] topics = topicStr.split(",");
        for (String topic : topics) {
            subscribedTopics.add(topic.trim());
        }
        consumer.subscribe(subscribedTopics);
```

©2013-2025 Tencent Cloud. All rights reserved.

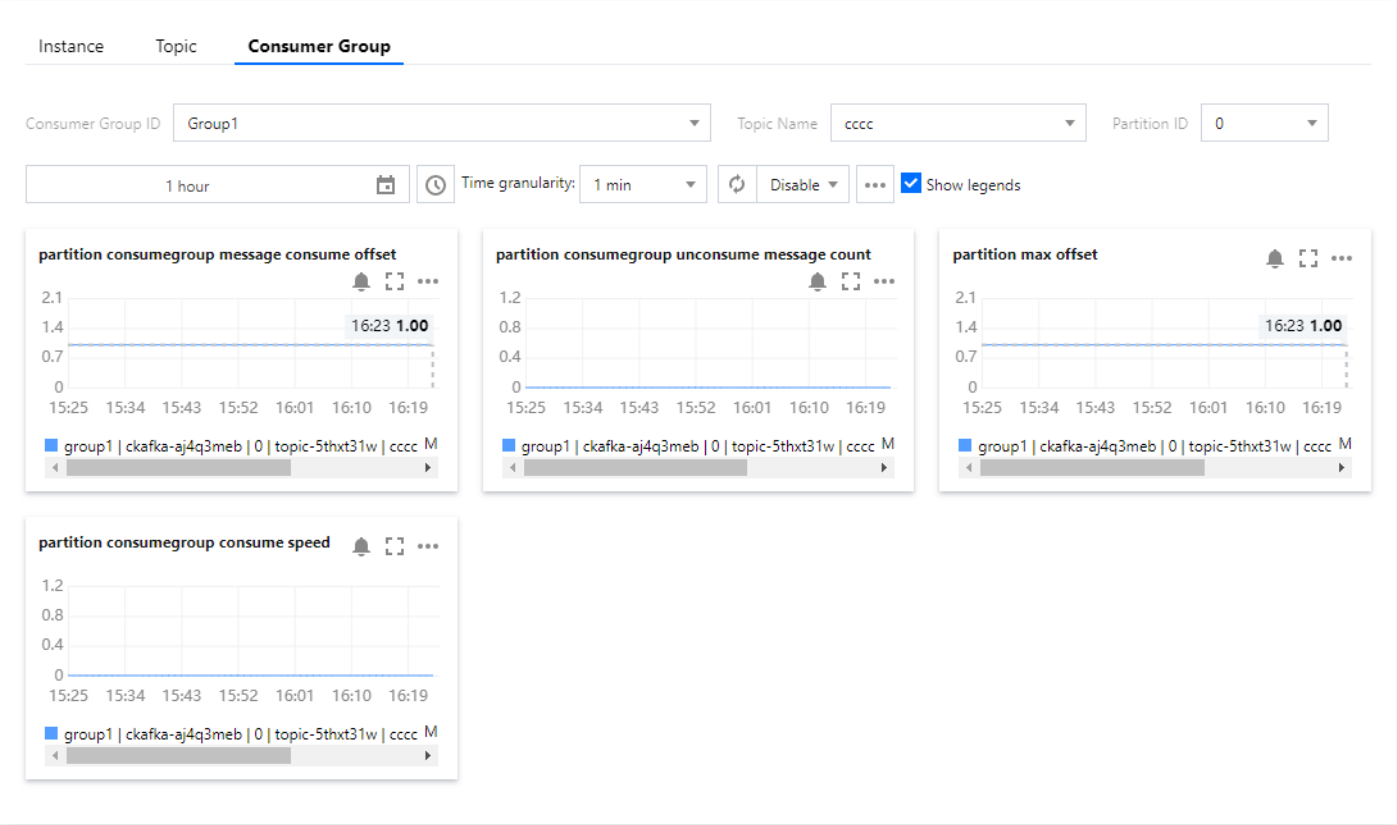
```
        //Consume messages in loop
        while (true){
            try {
                ConsumerRecords<String, String> records =
consumer.poll(1000);
                //All messages must be consumed before the next poll,
and the total duration cannot exceed the timeout interval specified by
`SESSION_TIMEOUT_MS_CONFIG`
                //You are advised to create a separate thread to
consume messages and then return the result in async mode
                for (ConsumerRecord<String, String> record : records)
{
                    System.out.println(
                            String.format("Consume partition:%d
offset:%d", record.partition(), record.offset()));
                }
            } catch (Exception e){
                System.out.println("consumer error!");
            }
        }
    }
}
```

2. View the execution result.

```
Consume partition:0 offset:298
Consume partition:0 offset:299
```

3. On the **Consumer Group** tab page in the CKafka console, select the corresponding consumer group name, enter the topic name, and click **View Details** to view the consumption details.

©2013-2025 Tencent Cloud. All rights reserved.



©2013-2025 Tencent Cloud. All rights reserved.

Running Kafka Client (Optional)

Last updated：2025-08-13 14:55:58

Scenarios

After configuring resources such as instances and topics in the console, you can also download and unzip the Kafka toolkit and perform simple send/receive message tests with the Kafka API.

Prerequisites

- Completed the initial stage of CKafka resource creation.
- Refer to Preparations to complete Linux server preparation and environment configuration.

Operation Steps

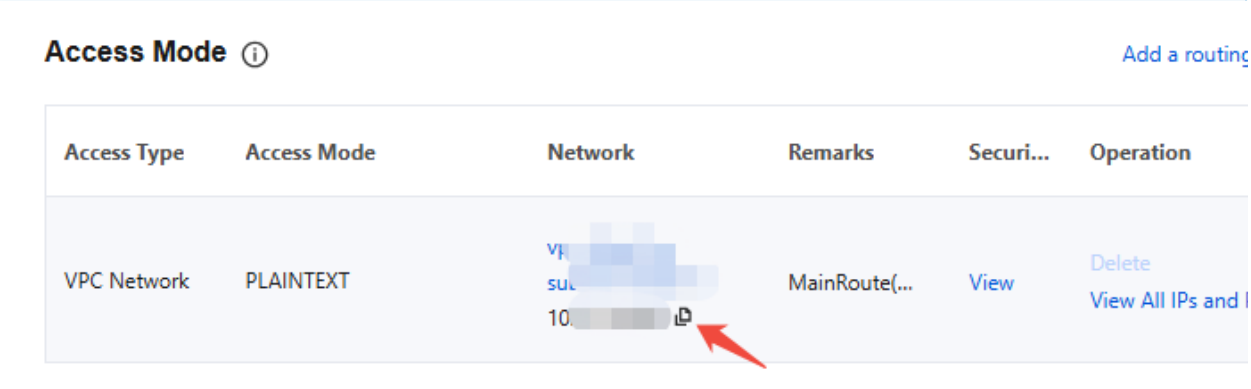
1. Download and decompress the Kafka installation package (Kafka installation package official download address).
2. Upload the downloaded Demo to a Linux server in the same VPC, then log in to the Linux server.
3. To produce and consume messages through CLI commands, go to `./bin` Under contents.

   3.1 Open the terminal to start the consumer.

```
bash kafka-console-consumer.sh --bootstrap-server XXXX:port --
topic XXXX --consumer.config ../config/consumer.properties
```

Note:

- Replace XXXX:port with the domain name and port for VPC network access. Obtain them in the **access method** module on the console instance detail page.



- topic: Replace XXXX with the topic name, and get it on the **topic management** page in the console.

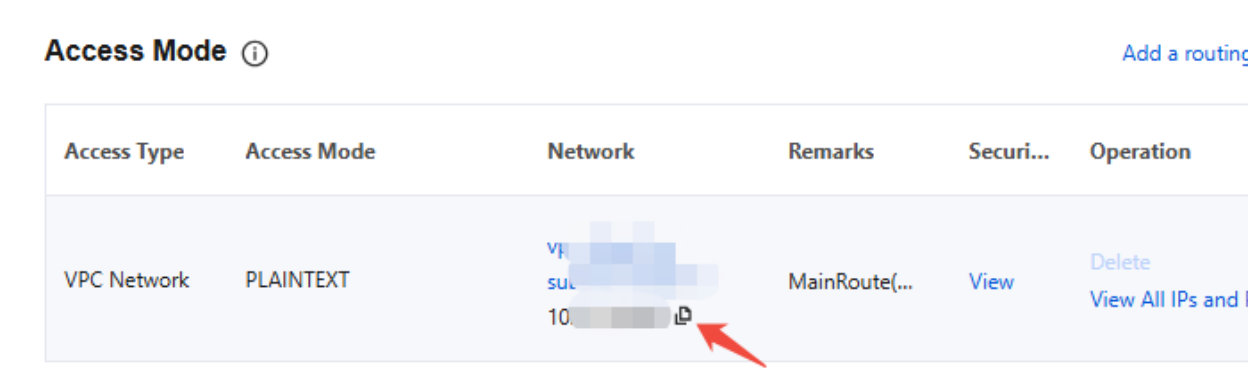
©2013-2025 Tencent Cloud. All rights reserved.

3.2 Open another terminal window to start the producer.

```
bash kafka-console-producer.sh --broker-list XXXX:port --topic
XXXX --producer.config ../config/producer.properties
```

Note:

- Replace XXXX:port with the domain name and port for VPC network access. Obtain them in the **access method** module on the console instance detail page.



- topic: Replace XXXX with the topic name, and get it on the **topic management** page in the console.

3.3 After entering the message content and pressing Enter, you can see that the consumer end also receives the message almost at the same time.

- Production message:

```
[ bin % bash kafka-console-producer.sh --broker-list ckafka]
.ap-guangzhou.ckafka.tencentcloudmq.com:6014 --topic test --producer.co
nfig ../config/producer.properties
>hello world
>this is a message
>this is another message
>
```

- Consumption message:

```
[ bin % bash kafka-console-consumer.sh --bootstrap-server c]
kafka- .ap-guangzhou.ckafka.tencentcloudmq.com:6014 --topic test --consum
er.config ../config/consumer.properties
hello world
this is a message
this is another message
```

©2013-2025 Tencent Cloud. All rights reserved.

©2013-2025 Tencent Cloud. All rights reserved.

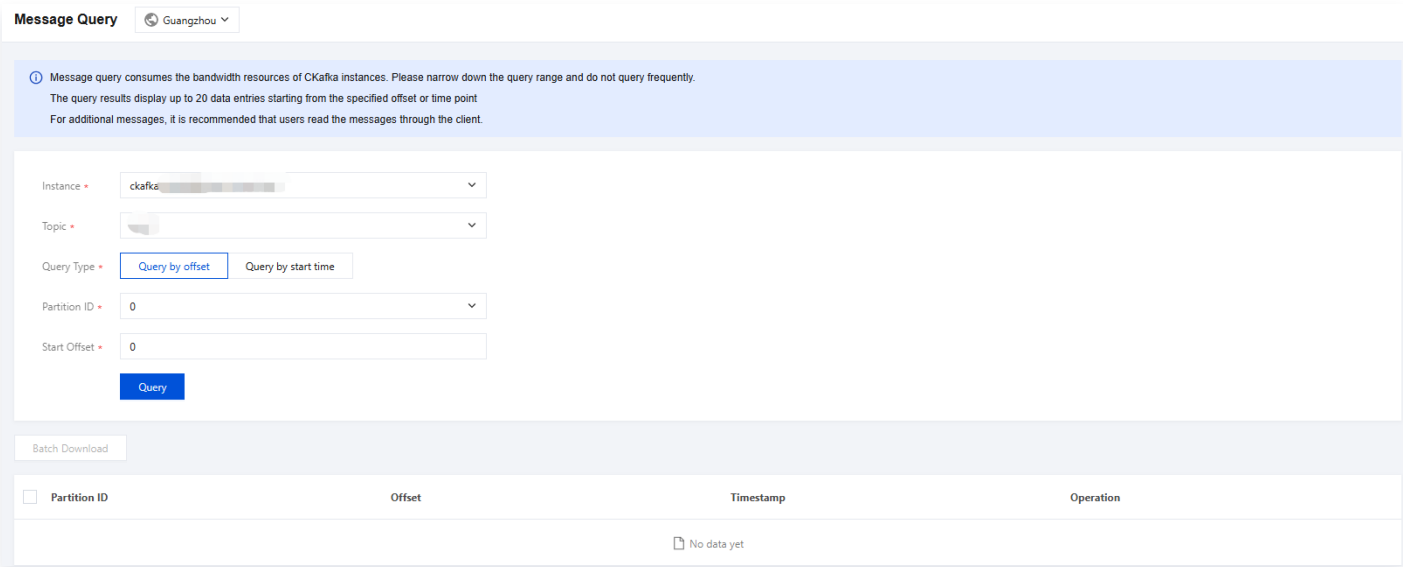
Step 5: Query Messages

Last updated：2025-08-13 14:55:58

After sending and receiving messages, you can use the message query feature in the console to view message details for the specified Topic.

Operation Steps

1. Log in to the CKafka console.
2. Select **Instance List** in the left sidebar, after selecting the region, click the ID/Name of the target instance to enter the instance details page.
3. On the instance details page, select **Topic Management**, then click **More** > **Message Query** in the right operation column of the target topic.
4. On the message query page, you can choose two methods to query messages published to that Topic: **query by offset** or **query by starting time**.
5. Click **view detail** in the Action column to view the specific message content.



©2013-2025 Tencent Cloud. All rights reserved.

Step 6: Resource Destruction

Last updated：2025-08-13 14:55:58

After getting started, you can use the purchased cluster for live production environment. If the cluster specification does not meet business needs, you can adjust cluster specification in the console. For the directions, see Adjusting Cluster Specification. If the cluster is no longer needed, you can terminate cluster resources. We will complete the refund based on your billing mode and usage duration, and clean up related data.

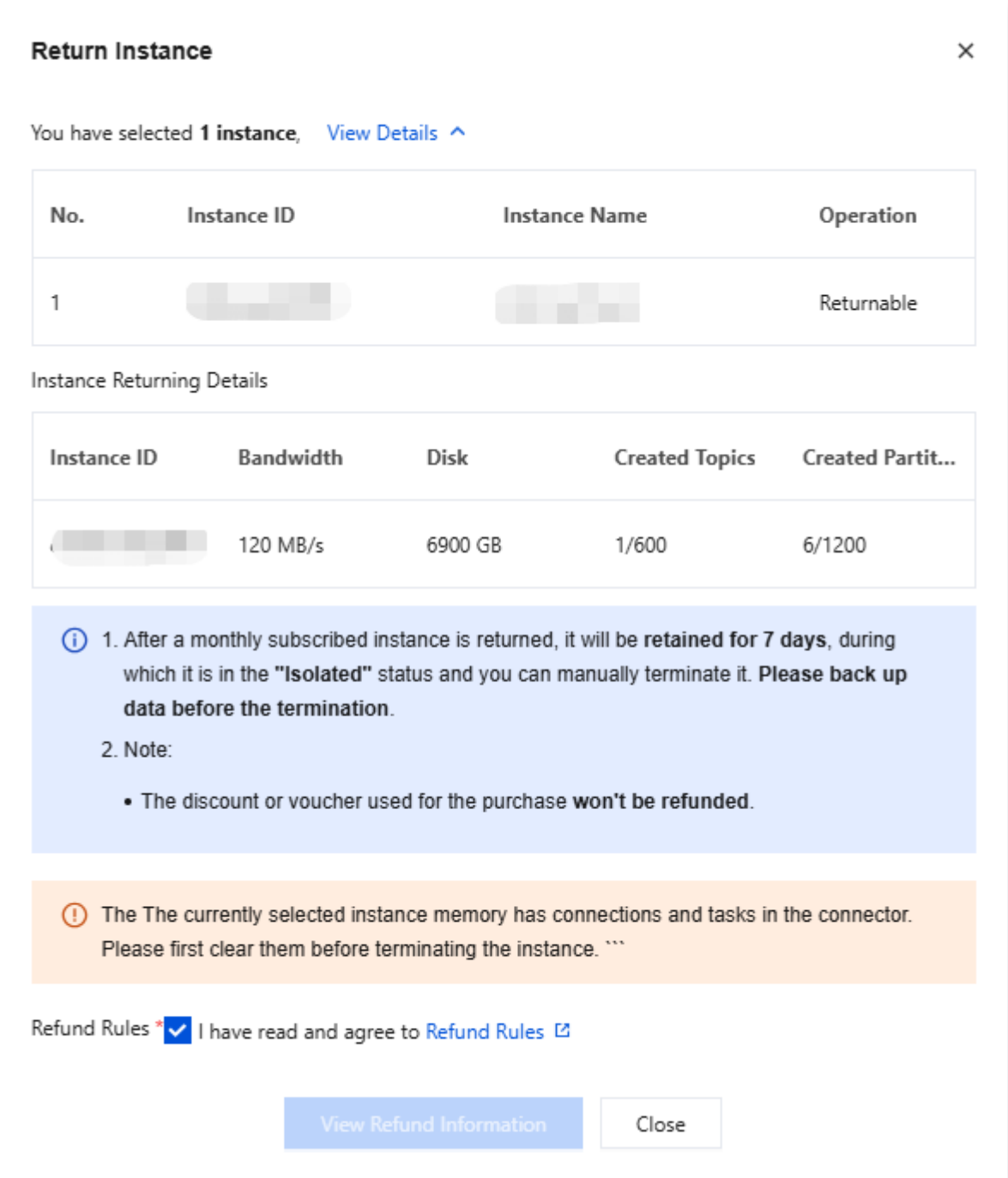
Operation Steps

Note:

- Danger: Completely terminating the instance will immediately remove it from the account instance list. After destruction, all data will be erased and unrecoverable. Back up data in advance.
- This step uses a pay-as-you-go instance as an example. The termination process for annual and monthly subscription instances may vary slightly. For details, see Terminating/Returning Instances.

1. Log in to the CKafka console.
2. Select **Instance List** in the left sidebar, find the target instance after selecting a region, and click **More** > **Terminate/Return** in the action bar.
3. After confirming the termination details, click **Next**, then **Terminate** in the pop-up window for confirming termination to terminate the instance.

©2013-2025 Tencent Cloud. All rights reserved.



©2013-2025 Tencent Cloud. All rights reserved.

Public Domain Name Access
Step 1: Create an Instance

Last updated：2025-08-13 14:48:02

Scenarios

This article introduces the directions to create a CKafka cluster through CKafka console.

Prerequisites

Preparations completed.

Operation Steps

Operation Steps

1. Log in to the CKafka console.
2. Click **Instance List** in the left sidebar, then click **Create** to enter the instance purchase page and select purchase information based on business needs.

Configuration Item	Parameter	Parameter Description
Basic Configuration	product form	Select **Serverful**.
	Billing Mode	Supports **Monthly Subscription** and **Pay-As-You-Go** modes. • **Monthly Subscription:** Payment is required to use resources, mainly suitable for relatively stable business scenarios for long-term use. • **Pay-As-You-Go:** Use resources first and pay later, mainly suitable for test or short-term situations with uncertain peak traffic.
	Cluster type	• Pro Edition is primarily aimed at large-scale production environment customers. • Advanced Edition is primarily aimed at test environment customers in small-scale scenarios.
	Region	Select a region with resources close to those of the client deployment to reduce network delay and improve access speed. Cloud products in different regions are not interconnected on a private network and cannot be changed after purchase. Please select carefully.

©2013-2025 Tencent Cloud. All rights reserved.

Cluster Configuration	Name	Optional. Not specified defaults to unnamed.
	Kafka version	Choose the appropriate Kafka version for your business needs. The recommended choice is version **2.8.1**.
	Peak bandwidth	Choose the appropriate peak bandwidth for your business needs.
	Disk	Currently supported disk types are SSD Cloud Block Storage and high-performance cloud block storage.
	Partition specification	Partition limit for CKafka instance. The package includes a minimum value of partitions at no charge. Additional partitions are charged per 100-unit increment.
	Message retention	Ranges from 24 to 2160 hours. The default message retention time is 72 hours. Messages will be deleted to preserve sufficient disk space when the retention duration is exceeded.
	Cross-AZ Deployment	The Professional Edition supports deployment in up to 4 availability zones, while the Advanced Edition supports up to 2 availability zones.
Network Configuration	VPC Network	Bind the domain name of the newly purchased cluster's access point to a prepared private network within the same region.
other configuration	Tag	Optional. Tags are used to categorize and manage resources from different dimensions.

3. Confirm the cluster information is correct and click **Purchase Now**. After completing payment on the order payment page, wait 3-5 minutes to see the created cluster on the cluster list page.

©2013-2025 Tencent Cloud. All rights reserved.

Step 2: Add Public Network Route

Last updated：2025-08-13 14:48:02

Scenarios

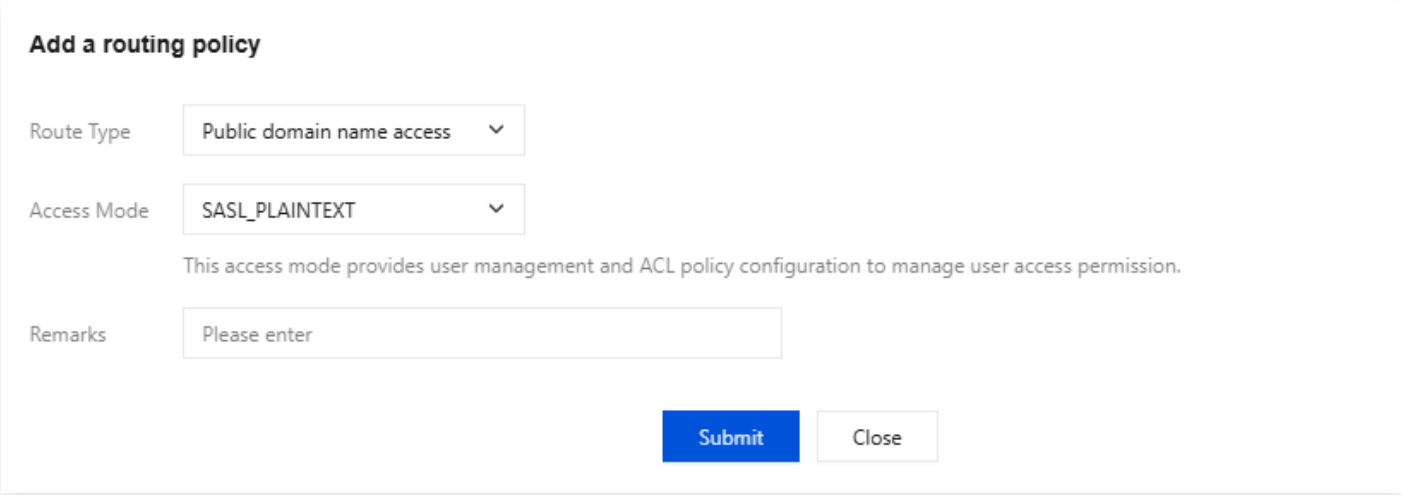
In the public network access scenario, you need to add a public network route to the instance. This article introduces the directions to add public network routing to the created instance through the CKafka console.

Prerequisites

Creating an Instance has been finished.

Operation Steps

1. In the Instance list page of the CKafka console, click the "ID/Name" of the created instance to enter the basic information page.
2. On the instance details page, select **access method** module and click **add routing policy** in the top-right corner to add a public network route.



3. Add to obtain the domain name and API for public network access.

©2013-2025 Tencent Cloud. All rights reserved.

Access Mode

Add a routing policy

Access Type	Access Mode	Network	Remarks	Securi...	Operation
Public domain name access	SASL_PLAINTEXT	ck 1	-	-	Delete View All IPs and Ports
VPC Network	PLAINTEXT	s	MainRoute(实...	View	Delete View All IPs and Ports
Supporting Environment	PLAINTEXT		ConnectorSu...	-	Delete View All IPs and Ports

©2013-2025 Tencent Cloud. All rights reserved.

Step 3: Create Topic

Last updated：2025-08-13 14:48:02

Scenarios

Before using CKafka for message sending and receiving, you need to create a Topic resource in the created CKafka instance. Producers send messages to the specified Topic, and consumers obtain messages by subscribing to the Topic for consumption. This article introduces the directions to create a Topic in the CKafka console.

Prerequisites

Creating an Instance has been finished.

Operation Steps

1. Log in to the CKafka console.
2. Select **Instance List** in the left sidebar, find the created instance, click the instance's "ID", and go to the instance details page.
3. Select the **Topic management** tab at the top of the instance details page, then click **create**.
4. In the create Topic window, select information such as the number of partitions and replicas.

Parameter	Description
Name	Topic name cannot be changed after input. Name can only contain letters, digits, underscore, "-", and ".". Names starting with double underscore are not supported.
Number of Partitions	A Topic can contain one or more partitions. CKafka uses partitions as the allocation unit. The default deployment architecture requires at least 3 nodes. A starting number of 3 partitions is recommended for more balanced data distribution. You can keep default values here.
Number of Replicas	The number of replicas for a Partition ensures high availability of the Partition. To ensure data reliability, 2 replicas are enabled by default. You can keep default values here.
Tag	Tags manage resources by category from different dimensions. Leave blank here.
retention.ms	Message retention time in the Topic dimension, ranging from 1 minute to 90 days. Keep default values here.
Advanced	Advanced Topic parameter configuration. Skip here.

©2013-2025 Tencent Cloud. All rights reserved.

Configuration	

5. Click **Submit** to complete Topic creation.

©2013-2025 Tencent Cloud. All rights reserved.

Step 4: Configuring ACL Policy

Last updated：2025-08-13 14:48:02

Scenarios

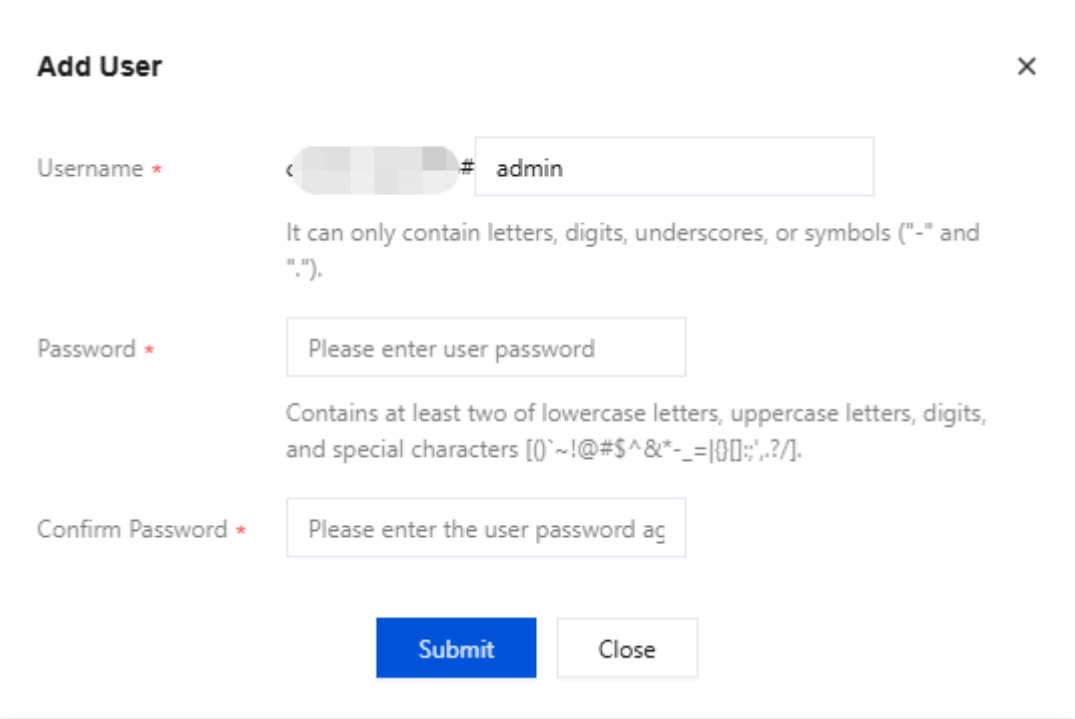
In a public network access scenario, you must perform ACL policy settings on the Topic to underwrite that the user has production/consumption permissions for the Topic resource.

Prerequisites

Have been Created a Topic.

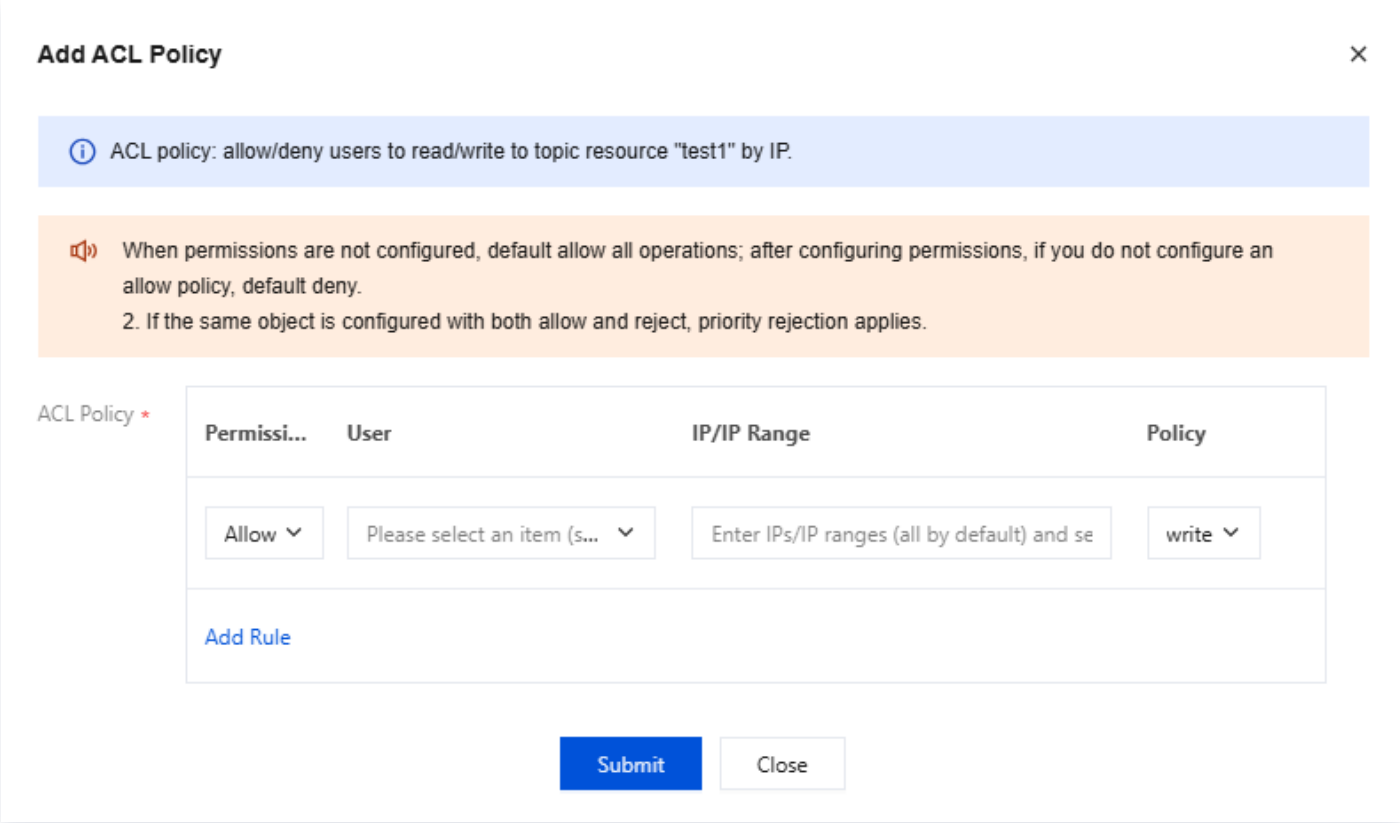
Operation Steps

1. Log in to the CKafka console.
2. Select **Instance List** in the left sidebar, find the instance created in advance, click the instance's "ID", and enter the instance details page.
3. On the instance details page, select **ACL Policy Management** > **user management**, click **create**, add user, and set the username and password.



4. On the user list page, select the **policy list** > **resource** tab, click **edit ACL policy** in the Operation column of the created Topic, and add read and write permissions for the user.

©2013-2025 Tencent Cloud. All rights reserved.



©2013-2025 Tencent Cloud. All rights reserved.

Step 5: Send/Receive Messages

Using SDK to Send and Receive Messages (Recommended)

Last updated：2025-08-13 14:48:03

Background

After configuring resources such as instances and topics in the console, you can use the SDK Demo we provide to connect to the cluster and perform a message sending and receiving test. This document uses Java SDK call as an example to introduce the operation process of sending and receiving messages through the open source SDK, helping you better understand the whole process of sending and receiving messages.

Prerequisites

- Completed the initial stage of CKafka resource creation.
- Refer to Preparations to complete environment configuration.

Operation Steps

Step 1: Prepare for Configuration

1. Download Demo.
2. Decompress the downloaded Demo and enter the PUBLIC_SASL directory under javakafkademo.
3. Modify the JAAS configuration file ckafka_client_jaas.conf.

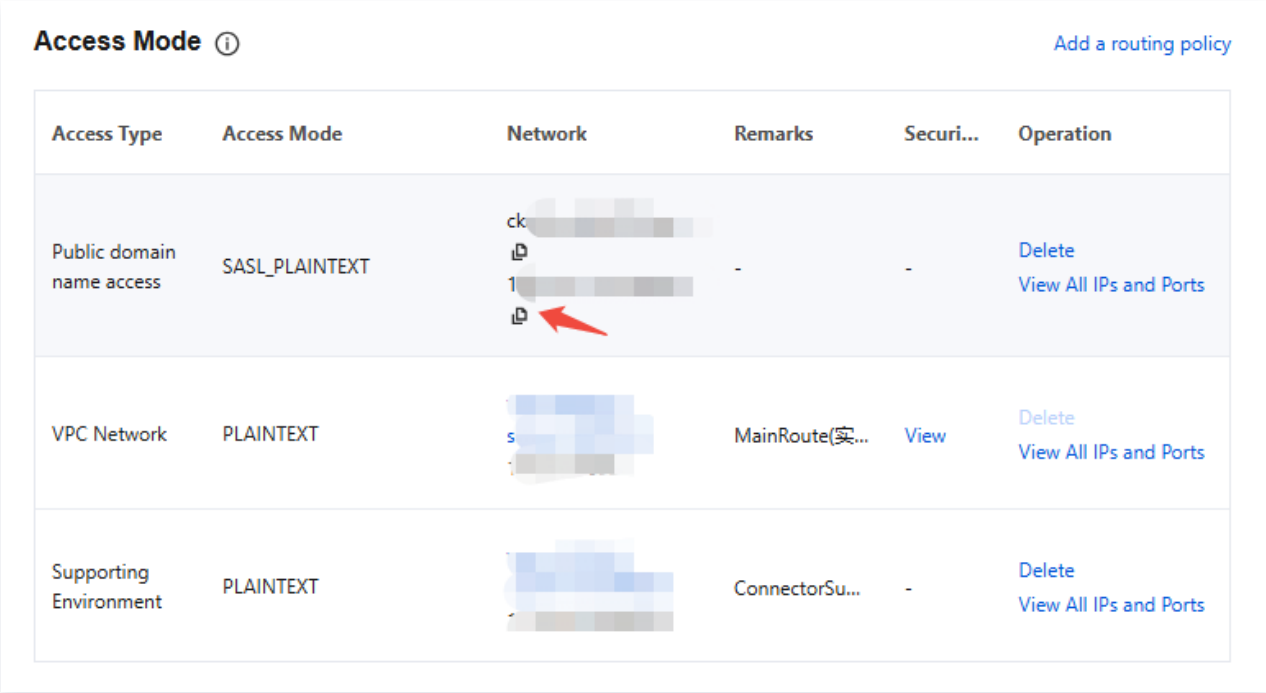
```
KafkaClient {
org.apache.kafka.common.security.plain.PlainLoginModule required
username="yourinstance#yourusername"
password="yourpassword";
};
```

> **Note:**
> Username is `instance ID` + `#` + `Configured username`, password is the configured user password.

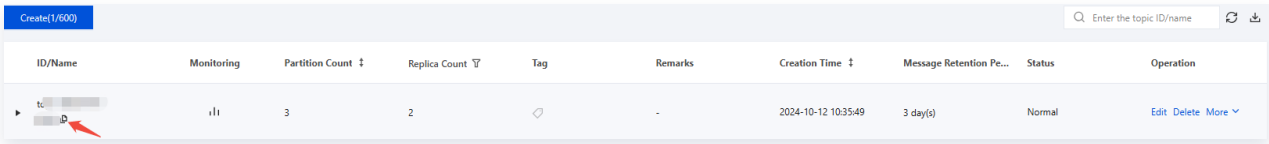
4. Modify the configuration file kafka.properties.

©2013-2025 Tencent Cloud. All rights reserved.

```
## Configure the access network and copy it from the network column of
the Access Mode module on the instance details page of the console.
bootstrap.servers=ckafka-xxxxxxx
## Configure Topic and copy it on the topic management page on the
console.
topic=XXX
## Configure Consumer Group and customize the settings
group.id=XXX
##Path to the JAAS configuration file ckafka_client_jaas.conf.
java.security.auth.login.config.plain=/xxxx/ckafka_client_jaas.conf
```

Parameter	Description
bootstrap.servers	Access the network and copy it from the network column of the Access Mode module on the instance details page in the console. 

©2013-2025 Tencent Cloud. All rights reserved.

topic	Topic name, you can copy it on the topic management page in the console. 
group.id	You can customize the settings. After the Demo runs successfully, you can see the Consumer Group on the Consumer Group tab in the console.
java.security.auth.login.confi	Enter the path to the JAAS configuration file ckafka_client_jaas.conf.

©2013-2025 Tencent Cloud. All rights reserved.

g
.
p
l
a
i
n

Step 2: Sending Messages

1. Compile and run the sendMessage program CKafkaSaslProducerDemo.java.

```
public class CKafkaSaslProducerDemo {

     public static void main(String args[]) {
             //Set the path to the JAAS configuration file.
             CKafkaConfigurer.configureSaslPlain();

             //Load kafka.properties.
             Properties kafkaProperties =
CKafkaConfigurer.getCKafkaProperties();

             Properties props = new Properties();
             //Set the access point. Obtain the access point of the
corresponding topic through the console.
             props.put(ProducerConfig.BOOTSTRAP_SERVERS_CONFIG,
kafkaProperties.getProperty("bootstrap.servers"));

             //Access protocol.
             props.put(CommonClientConfigs.SECURITY_PROTOCOL_CONFIG,
"SASL_PLAINTEXT");
             //Plain method.
             props.put(SaslConfigs.SASL_MECHANISM, "PLAIN");

             //Message serialization method for TDMQ for CKafka.
             props.put(ProducerConfig.KEY_SERIALIZER_CLASS_CONFIG,
"org.apache.kafka.common.serialization.StringSerializer");
             props.put(ProducerConfig.VALUE_SERIALIZER_CLASS_CONFIG,
"org.apache.kafka.common.serialization.StringSerializer");
             //The maximum waiting time for the request.
```

©2013-2025 Tencent Cloud. All rights reserved.

```
            props.put(ProducerConfig.MAX_BLOCK_MS_CONFIG, 30 *
1000);
            //Set the number of internal retries on the client.
            props.put(ProducerConfig.RETRIES_CONFIG, 5);
            //Set the interval of internal retries on the client.
            props.put(ProducerConfig.RECONNECT_BACKOFF_MS_CONFIG,
3000);
            //Construct a Producer object. Note that this object is
thread-safe. Generally speaking, only one Producer object is required
in one process.
            KafkaProducer<String, String> producer = new
KafkaProducer<>(props);

            //Construct a CKafka message.
            String topic = kafkaProperties.getProperty("topic");
//The topic to which the message belongs. Fill it in here after
applying in the console.
            String value = "this is ckafka msg value"; //The content
of the message.

            try {
                    // Batch getting future objects can speed up the
process. Note that the batch size should not be too large.
                    List<Future<RecordMetadata>> futures = new
ArrayList<>(128);
                    for (int i =0; i < 100; i++) {
                            //Send a message and get a Future object.
                            ProducerRecord<String, String>
kafkaMessage = new ProducerRecord<>(topic, value + ": " + i);
                            Future<RecordMetadata> metadataFuture =
producer.send(kafkaMessage);
                            futures.add(metadataFuture);

                    }
                    producer.flush();
                    for (Future<RecordMetadata> future: futures) {
                            //Synchronously obtain the result of the
Future object.
                                    RecordMetadata recordMetadata =
future.get();
```

©2013-2025 Tencent Cloud. All rights reserved.

```
                                    System.out.println("Produce ok:"
+ recordMetadata.toString());
                        }
                } catch (Exception e) {
                        //After the client retries internally, the
sending still fails. The business needs to deal with this type of
error.
                        System.out.println("error occurred");
                }
        }
}
```

2. Operation results (output).

```
Produce ok:ckafka-topic-demo-0@198
Produce ok:ckafka-topic-demo-0@199
```

Step 3: Consuming Messages

1. Compile and run the Consumer subscription message program CKafkaSaslConsumerDemo.java.

```
public class CKafkaSaslConsumerDemo {
    public static void main(String args[]) {
        //Set the path to the JAAS configuration file.
        CKafkaConfigurer.configureSaslPlain();

        //Load kafka.properties.
        Properties kafkaProperties =
CKafkaConfigurer.getCKafkaProperties();

        Properties props = new Properties();
        //Set the access point. Obtain the access point of the
corresponding topic through the console.
        props.put(ProducerConfig.BOOTSTRAP_SERVERS_CONFIG,
kafkaProperties.getProperty("bootstrap.servers"));

        //Access protocol.
        props.put(CommonClientConfigs.SECURITY_PROTOCOL_CONFIG,
"SASL_PLAINTEXT");
```

©2013-2025 Tencent Cloud. All rights reserved.

```
        //Plain method.
        props.put(SaslConfigs.SASL_MECHANISM, "PLAIN");
        //The maximum allowed interval between two polls.
        //If the consumer exceeds this value without returning a
heartbeat, the server will determine that the consumer is not alive,
remove the consumer from the Consumer Group and trigger Rebalance. The
default value is 30s.
        props.put(ConsumerConfig.SESSION_TIMEOUT_MS_CONFIG, 30000);
        //The maximum number of polls per time.
        //Be careful not to change this value too large. If too much
data is polled and cannot be consumed before the next poll, a load
balancing will be triggered, resulting in lag.
        props.put(ConsumerConfig.MAX_POLL_RECORDS_CONFIG, 30);
        //How to deserialize the message.
        props.put(ConsumerConfig.KEY_DESERIALIZER_CLASS_CONFIG,
"org.apache.kafka.common.serialization.StringDeserializer");
        props.put(ConsumerConfig.VALUE_DESERIALIZER_CLASS_CONFIG,
"org.apache.kafka.common.serialization.StringDeserializer");
        //The consumer group to which the current consumption instance
belongs. Fill it in after applying in the console.
        //Consumption instances belonging to the same group will carry
consumption messages.
        props.put(ConsumerConfig.GROUP_ID_CONFIG,
kafkaProperties.getProperty("group.id"));
        //Construct a consumption object, that is, generate a consumer
instance.
        KafkaConsumer<String, String> consumer = new
KafkaConsumer<String, String>(props);
        //Set the topic that the consumer group subscribes to. You can
subscribe to multiple topics.
        //If GROUP_ID_CONFIG is the same, it is recommended to set the
subscribed topic to the same as well.
        List<String> subscribedTopics =  new ArrayList<String>();
        //If you need to subscribe to multiple topics, just add them
here.
        //Each topic needs to be created in the console first.
        String topicStr = kafkaProperties.getProperty("topic");
        String[] topics = topicStr.split(",");
        for (String topic: topics) {
            subscribedTopics.add(topic.trim());
```

©2013-2025 Tencent Cloud. All rights reserved.

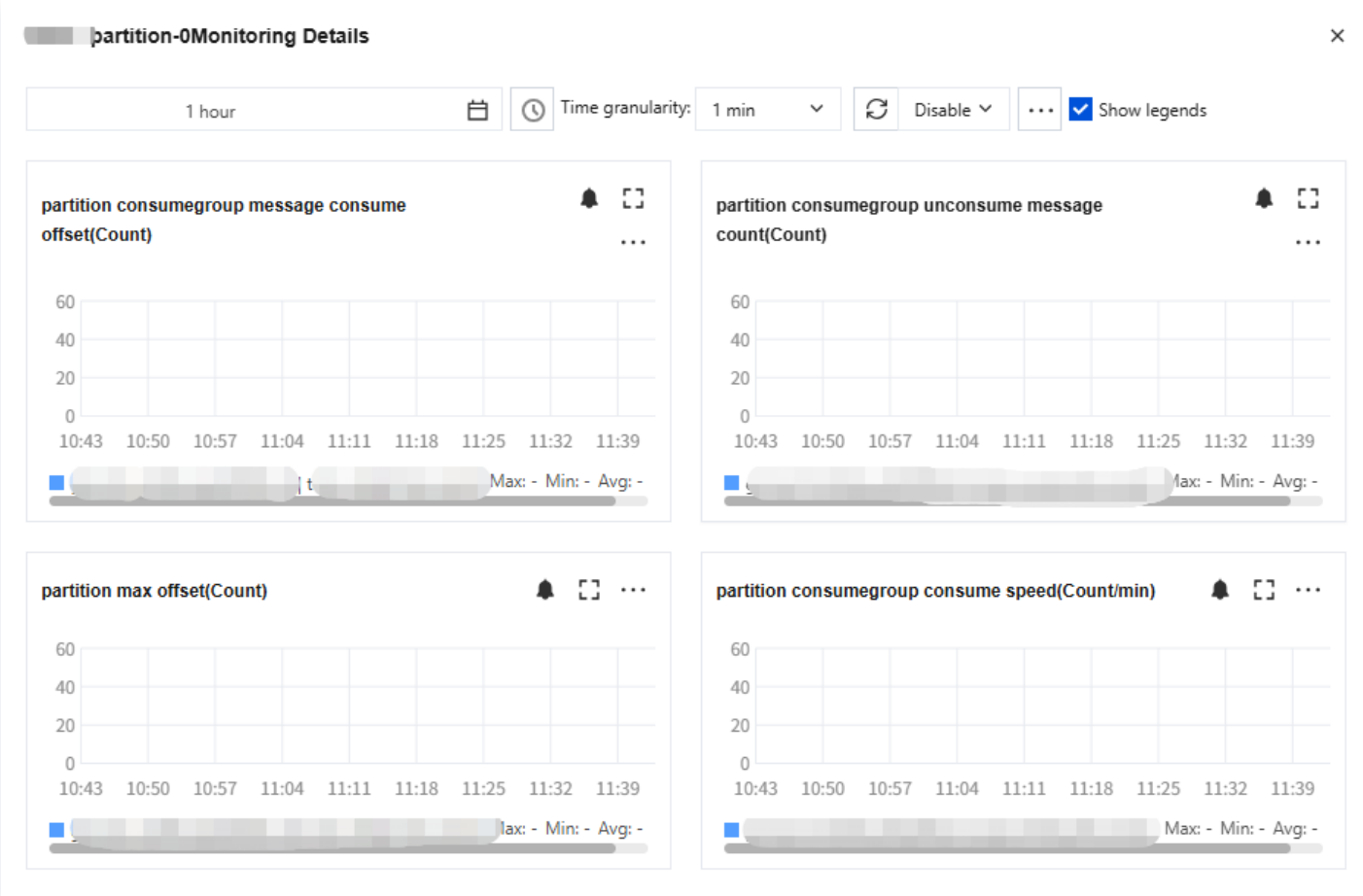
```
        }
        consumer.subscribe(subscribedTopics);

        //Loop consumption messages.
        while (true){
            try {
                ConsumerRecords<String, String> records =
consumer.poll(1000);
                //This data must be consumed before the next poll, and
the total time must not exceed SESSION_TIMEOUT_MS_CONFIG.
                for (ConsumerRecord<String, String> record : records)
{
                    System.out.println(String.format("Consume
partition:%d offset:%d", record.partition(), record.offset()));
                }
            } catch (Exception e) {
                System.out.println("consumer error!");
            }
        }
    }
}
```

2. Execution Result.

```
Consume partition:0 offset:298
Consume partition:0 offset:299
```

3. On the **Consumer Group** page in the CKafka console, select the corresponding consumer group name, enter the topic name in the topic name field, click **Query Details** to view consumption details.

©2013-2025 Tencent Cloud. All rights reserved.



©2013-2025 Tencent Cloud. All rights reserved.

Running the Kafka Client (Optional)

Last updated：2025-08-13 14:48:02

Scenarios

After configuring resources such as instances and topics in the console, you can also download and unzip the Kafka toolkit to perform simple message sending and receiving tests with the Kafka API.

Prerequisites

- Completed the initial stage of CKafka resource creation.
- Refer to Preparations to configure the environment.

Operation Steps

1. Download and decompress the Kafka installation package (Kafka installation package official download link).
2. Configure ACL policies locally.
   2.1 In toolkit `./config` Directory, add the following content at the end of producer.properties and consumer.properties files:

```
security.protocol=SASL_PLAINTEXT
sasl.mechanism=PLAIN
```

   2.2 Create a file named ckafka_client_jaas.conf with the following content:

```
KafkaClient {
            org.apache.kafka.common.security.plain.PlainLoginModule
required
            username="yourinstance#yourusername"
            password="yourpassword";
};
```

> **Note:**
> Note: The username is `instance ID` + `#` + `newly configured username`, and the password is the newly configured user password.

   2.3 In the toolkit `./bin` In the directory, add the declaration of the path of the JAAS file to the

©2013-2025 Tencent Cloud. All rights reserved.

beginning of the kafka-console-producer.sh and kafka-console-consumer.sh files (it must be a full path):

```
export KAFKA_OPTS="-Djava.security.auth.login.config=****/config/ckafka_client_jaas.conf"
```

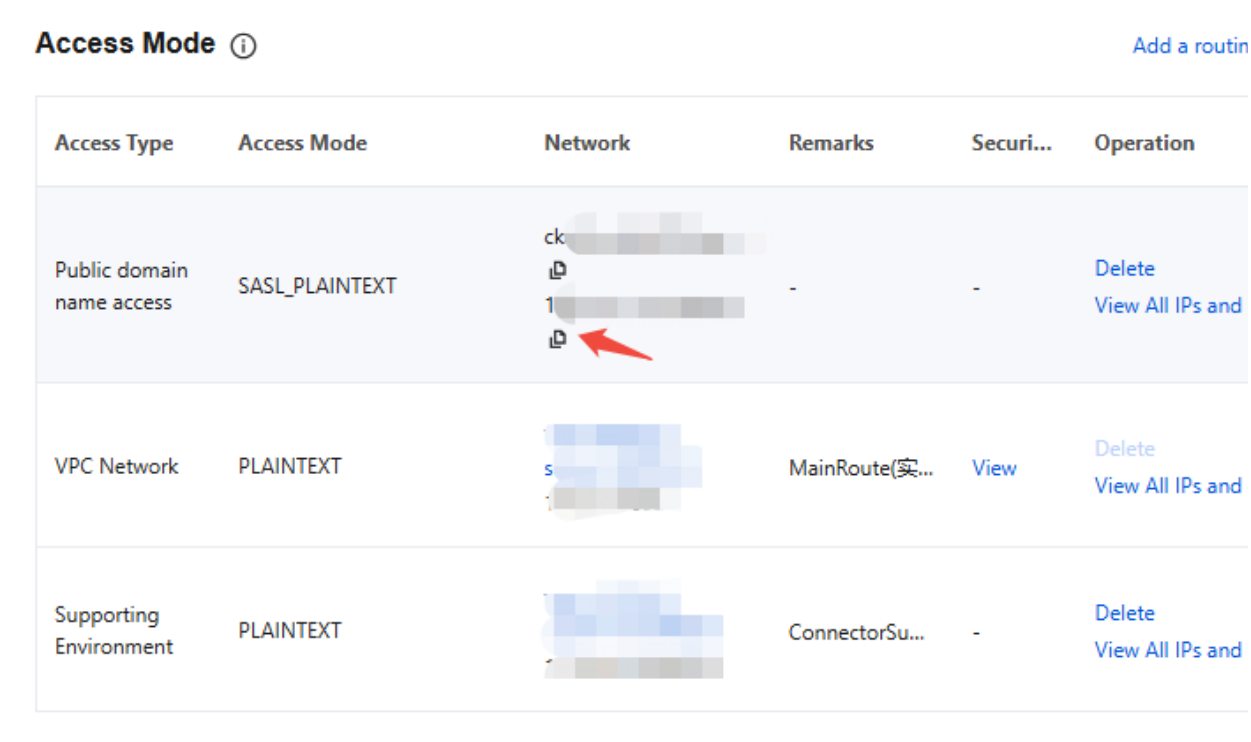
3. Produce and consume messages through CLI commands.

3.1 Open the terminal to start the consumer.

```
bash kafka-console-consumer.sh --bootstrap-server XXXX:port --topic XXXX --consumer.config ../config/consumer.properties
```

Note:

- broker-list: Replace XXXX:port with the domain name and port for public network access. Obtain them from the **access method** module on the console instance detail page.



- topic: Replace XXXX with the topic name. Get it on the **topic management** page in the console.

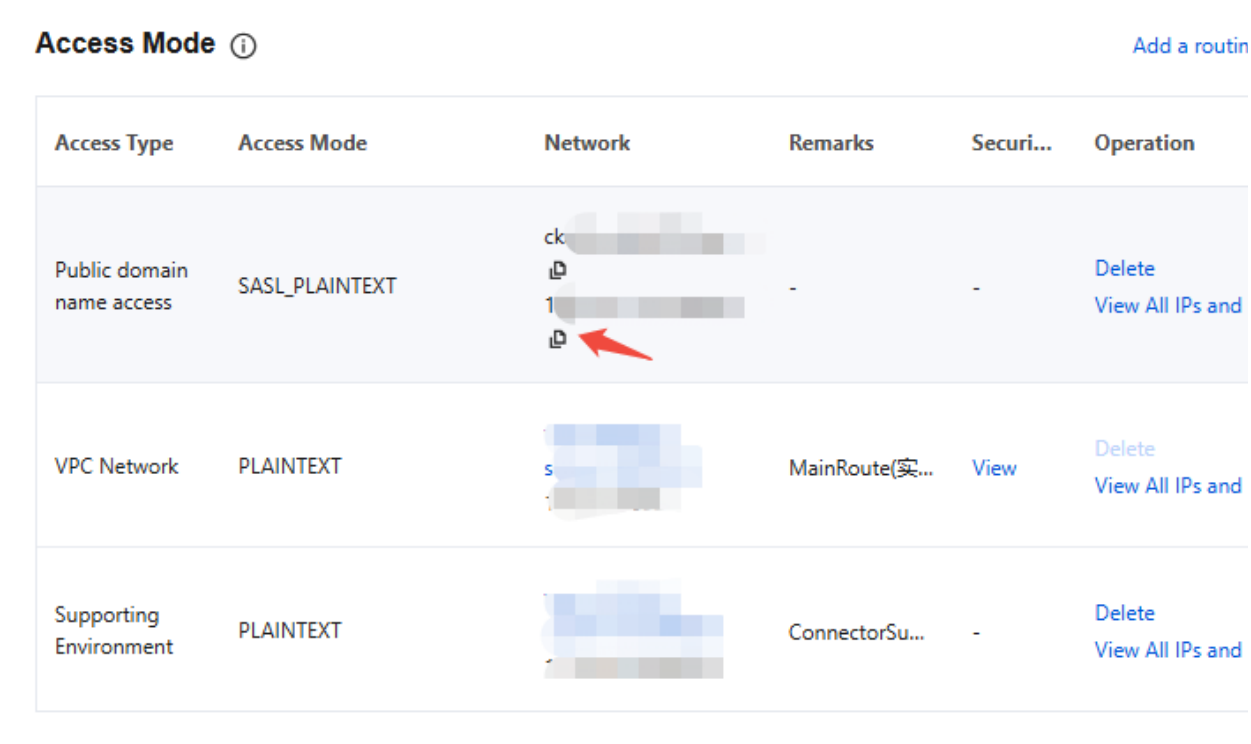
3.2 Open another terminal window to start the producer.

©2013-2025 Tencent Cloud. All rights reserved.

```
bash kafka-console-producer.sh --broker-list XXXX:port --topic
XXXX --producer.config ../config/producer.properties
```

Note:

- broker–list: Replace XXXX:port with the domain name and port for public network access. Obtain them from the **access method** module on the console instance detail page.



- topic: Replace XXXX with the topic name. Get it on the **topic management** page in the console.

3.3 After entering the message content and pressing Enter, you can see that the consumer end also receives the message almost at the same time.

- Production message:

```
bin % bash kafka-console-producer.sh --broker-list ckaf
.ap-guangzhou.ckafka.tencentcloudmq.com:6014 --topic test --producer.
nfig ../config/producer.properties
>hello world
>this is a message
>this is another message
>
```

©2013-2025 Tencent Cloud. All rights reserved.

- Consumption message:

```
[                    bin % bash kafka-console-consumer.sh --bootstrap-server
kafka-    .ap-guangzhou.ckafka.tencentcloudmq.com:6014 --topic test --cons
er.config ../config/consumer.properties
hello world
this is a message
this is another message
```

©2013-2025 Tencent Cloud. All rights reserved.

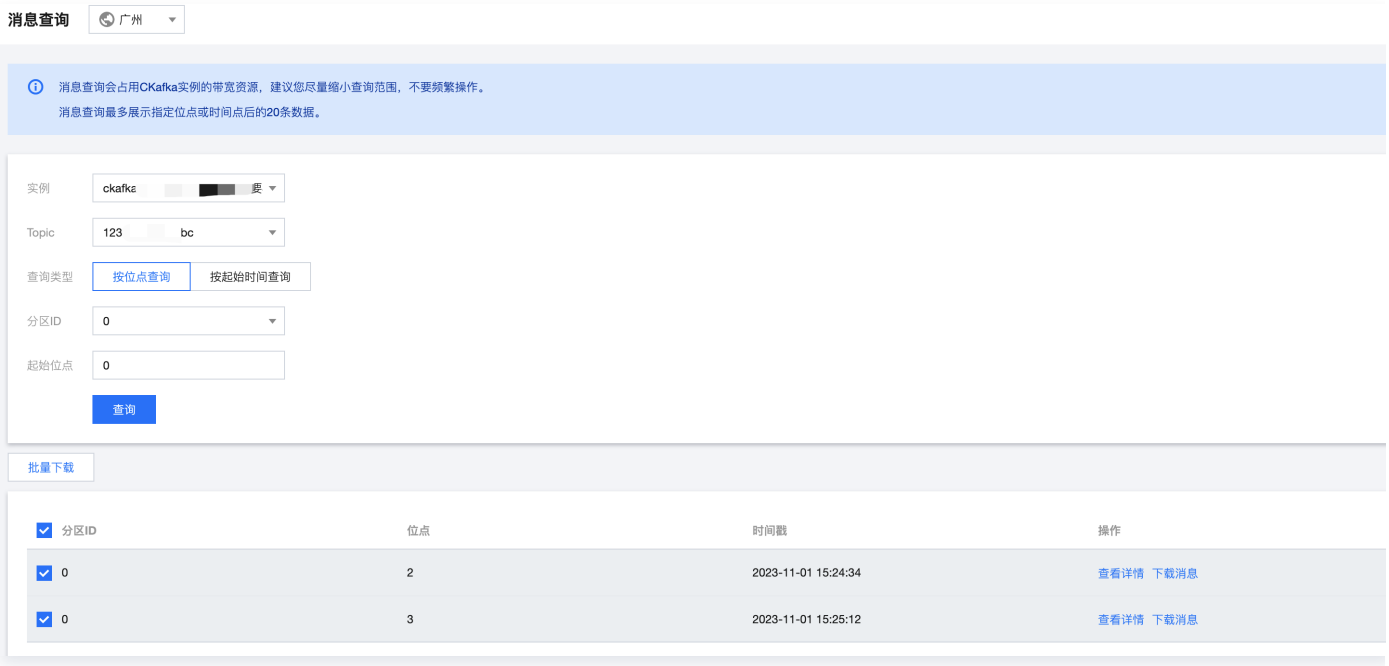
Step 6: Query Messages

Last updated：2025-08-13 14:48:02

After sending and receiving messages, you can use the message query feature in the console to view message details for the specified Topic.

Operation Steps

1. Log in to the CKafka console.
2. Select **instance list** in the left sidebar, after selecting the region, click the ID/Name of the target instance to enter the instance details page.
3. On the instance details page, select **topic management**, then click **more** > **message query** in the right operation column of the target Topic.
4. On the message query page, you can choose two methods to query messages published to that Topic: **query by offset** or **query by start time**.
5. Click **view detail** in the Action column to view the specific message content.



©2013-2025 Tencent Cloud. All rights reserved.

Step 7: Resource Destruction

Last updated：2025-08-13 14:48:02

After getting started, you can use the purchased cluster for a live production environment. If the cluster specification does not meet business needs, you can adjust the cluster specification in the console. For the directions, see Adjusting Cluster Configuration. If the cluster is no longer needed, you can terminate the cluster. We will process the refund based on your billing mode and usage duration, then clean up related data.

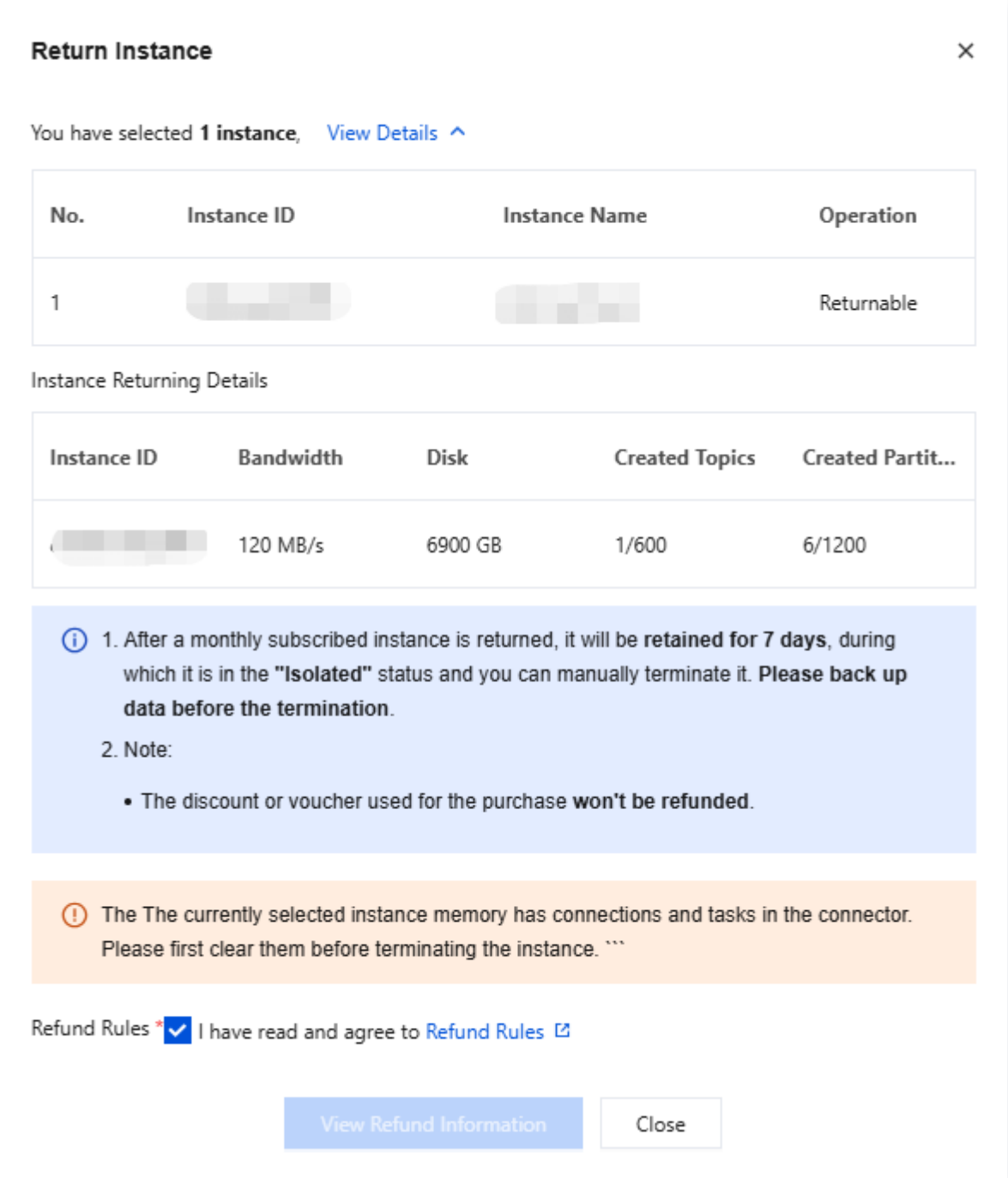
Operation Steps

Note:

- Terminate the instance immediately removes it from the account instance list. After destruction, ALL data will be erased and unrecoverable. Back up data in advance.
- This step uses a pay-as-you-go instance as an example. The termination process for annual and monthly subscription instances varies slightly. For details, refer to Terminating/Returning Instances.

1. Log in to the CKafka console.
2. Select **Instance List** in the left sidebar, find the target instance after selecting a region, and click **More** > **Terminate/Return** in the action bar.
3. After confirming the termination details, click **Next**, then **Terminate** in the pop-up window for confirming termination to terminate the instance.

©2013-2025 Tencent Cloud. All rights reserved.



©2013-2025 Tencent Cloud. All rights reserved.