Message Queue CKafka

Getting Started

©2013–2023 Tencent Cloud. All rights reserved.

Copyright Notice

©2013-2023 Tencent Cloud. All rights reserved.

The complete copyright of this document, including all text, data, images, and other content, is solely and exclusively owned by Tencent Cloud Computing (Beijing) Co., Ltd. ("Tencent Cloud"); Without prior explicit written permission from Tencent Cloud, no entity shall reproduce, modify, use, plagiarize, or disseminate the entire or partial content of this document in any form. Such actions constitute an infringement of Tencent Cloud's copyright, and Tencent Cloud will take legal measures to pursue liability under the applicable laws.

Trademark Notice

This trademark and its related service trademarks are owned by Tencent Cloud Computing (Beijing) Co., Ltd. and its affiliated companies("Tencent Cloud"). The trademarks of third parties mentioned in this document are the property of their respective owners under the applicable laws. Without the written permission of Tencent Cloud and the relevant trademark rights owners, no entity shall use, reproduce, modify, disseminate, or copy the trademarks as mentioned above in any way. Any such actions will constitute an infringement of Tencent Cloud's and the relevant owners' trademark rights, and Tencent Cloud will take legal measures to pursue liability under the applicable laws.

Service Notice

This document provides an overview of the as-is details of Tencent Cloud's products and services in their entirety or part. The descriptions of certain products and services may be subject to adjustments from time to time.

The commercial contract concluded by you and Tencent Cloud will provide the specific types of Tencent Cloud products and services you purchase and the service standards. Unless otherwise agreed upon by both parties, Tencent Cloud does not make any explicit or implied commitments or warranties regarding the content of this document.

Contact Us

We are committed to providing personalized pre-sales consultation and technical after-sale support. Don't hesitate to contact us at 4009100100 or 95716 for any inquiries or concerns.

Contents

©2013-2023 Tencent Cloud. All rights reserved.

Getting Started
Process Overview

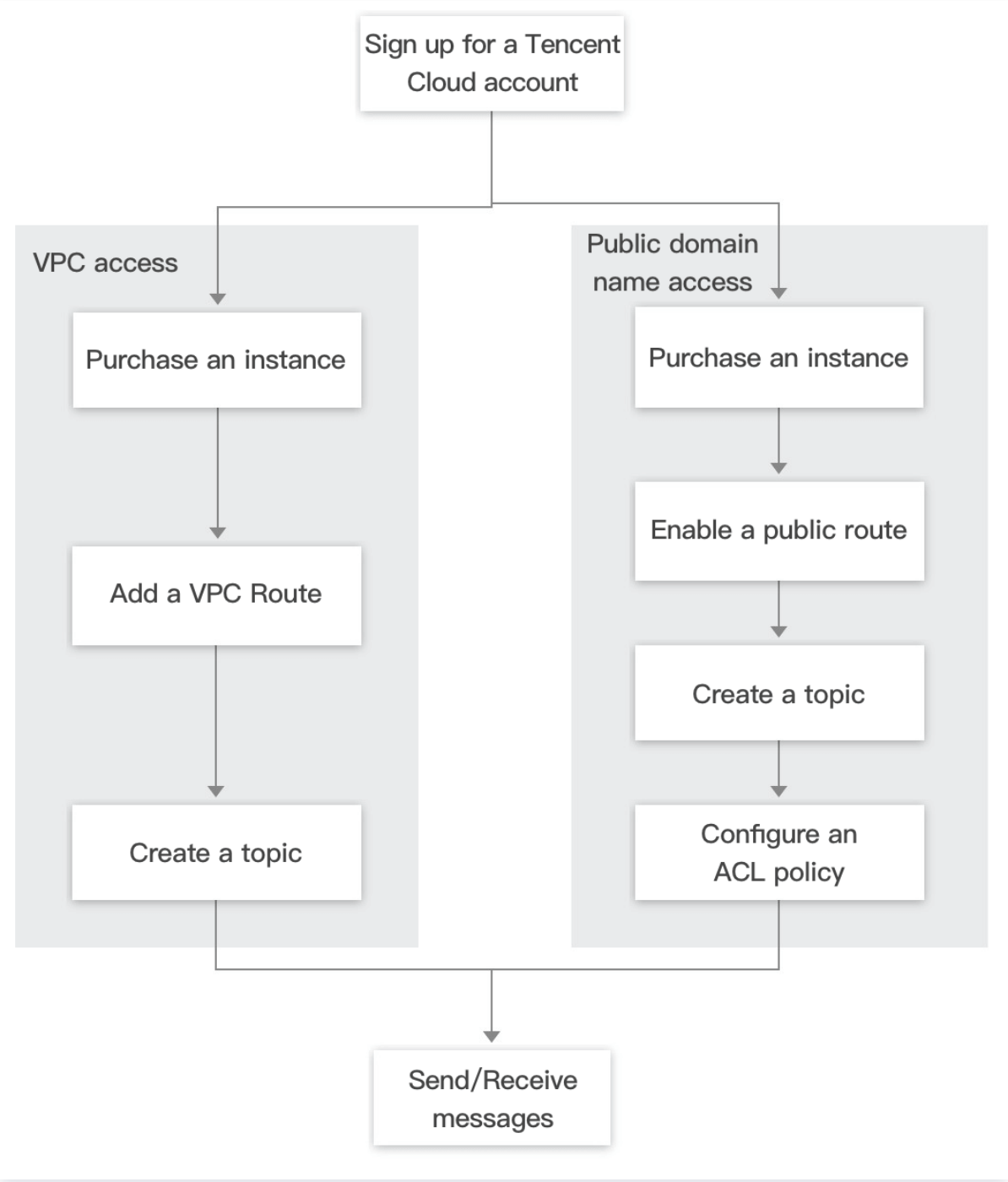
Last updated：2023-09-07 16:04:04

The process of accessing CKafka varies by network type:

- For access via VPC, you can select an appropriate VPC according to your business needs.
- Access via a public network route: you need to enable a separate public route and configure an ACL policy for the topic.

Flowchart

©2013-2023 Tencent Cloud. All rights reserved.



©2013–2023 Tencent Cloud. All rights reserved.

Obtaining Access Permission Getting Access Authorization

Last updated: 2023-09-07 16:06:41

Basic CAM Concepts

The root account authorizes sub-accounts by associating policies. The policy setting can be specific to the level of **[API, Resource, User/User Group, Allow/Deny, and Condition]**.

Account System

- **Root account**: This account owns all Tencent Cloud resources and has unrestricted access to them.
- **Sub-account**: Comprising sub-users and collaborators.
  - **Sub-user**: It is created and fully owned by a root account.
  - **Collaborator**: It has the identity of a root account. After it is added as a collaborator of the current root account, it becomes one of the sub-accounts of the current root account and can switch back to its root account identity.
- **Identity Credentials**: These include two types – login credentials and access certificates. **Login credentials** refer to the username and password, while **access certificates** refer to the cloud API keys (SecretId and SecretKey).

Resource and permission

- **Resource**: An object that is operated in Tencent Cloud services, such as a CVM instance, a VPC instance, etc.
- **Permission**: It is an authorization that allows or forbids users to perform certain operations. By default, the **root account** has full access to all resources under the account, while a **sub-account** does not have access to any resources under its root account.
- **Policy**: It is a syntax rule that defines and describes one or more permissions. The **root account** performs authorization by **associating policies** with users/user groups.

Using CKafka with Sub-Account

A sub-account needs to be authorized in the following two aspects before it can use CKafka:

1. During the use of CKafka, access to other Tencent Cloud resources (such as VPC, CVM) is required, for example, to view the availability zone information of a user's subnet. Therefore, it is necessary to grant sub-accounts permissions to access other cloud

©2013-2023 Tencent Cloud. All rights reserved.

services. For detailed instructions, see Step 1: Grant sub-account permissions to access other Tencent Cloud services.

2. To use CKafka, a sub-account also needs read and write permissions. For detailed instructions, see Step 2: Granting CKafka permissions to the sub-account.

Step 1: Grant the sub-account permissions to access other cloud products.

Create a Custom Policy for Accessing Other Cloud Services

1. Log in to the CAM console as the root account.
2. In the left sidebar, select **Policies** and click **Create Custom Policy**.
3. In the pop-up box for selecting the policy creation method, choose **Create by Policy Syntax** to proceed to the policy syntax creation page.
4. On the Create by Policy Syntax page, select **Blank Template** and click **Next**.
5. You can refer to the API call table and policy syntax below to grant appropriate permissions for other cloud products to the sub-account based on your needs. Create a custom policy, fill in all the required information, and click **Finish**.

   Using CKafka involves calling the following Tencent Cloud services. The root account needs to authorize sub-accounts separately for them to use CKafka features. The custom policies in CKafka include the following calls to Tencent Cloud services:

Tencent Cloud Products	API	API Functionality	Operations Affecting TSE Platform
Cloud Virtual Machine (CVM)	DescribeZones	Querying AZs	Viewing the availability zone of a subnet when creating an instance
Virtual Private Cloud (VPC)	DescribeVpcs	Queries VPC lists	Select the VPC to which the instance access address belongs when creating an instance
Virtual Private Cloud (VPC)	DescribeSubnets	Queries VPC lists	Select the subnet to which the instance access address belongs when creating an instance.
Tencent Cloud Observability	GetMonitorData	Pulls metric	View Monitoring Data in CKafka

©2013-2023 Tencent Cloud. All rights reserved.

Platform (Monitor)		monitoring data	
Tencent Cloud Observability Platform (Monitor)	DescribeDashboard MetricData	Pulls metric monitoring data	View Monitoring Data in CKafka

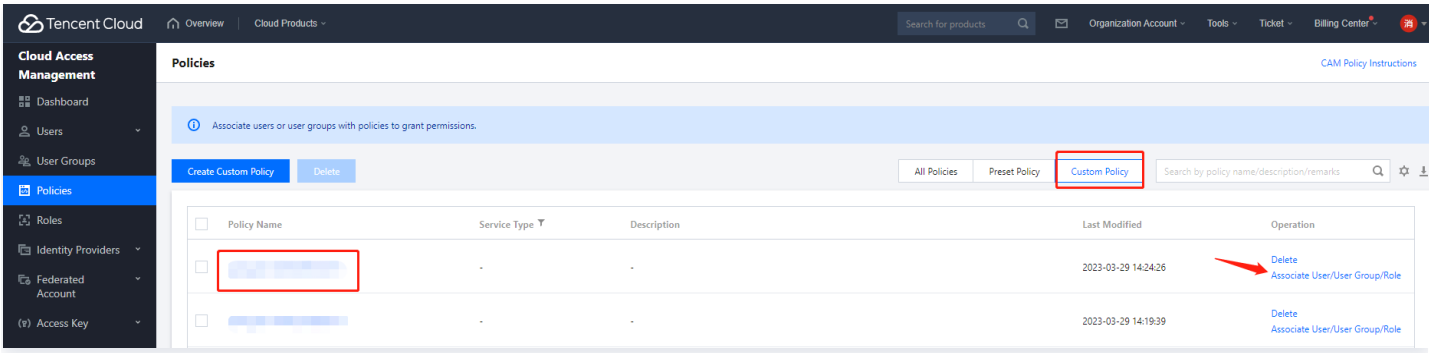
Below is an example of policy syntax:

```
{
 "version": "2.0",
 "statement": [
  {
   "effect": "allow",
   "action": [
    "vpc:DescribeVpcEx",
    "vpc:DescribeSubnetEx",
    "monitor:GetMonitorData",
    "monitor:DescribeDashboardMetricData",
   ],
   "resource": [
    "*"
   ]
  }
 ]
}
```

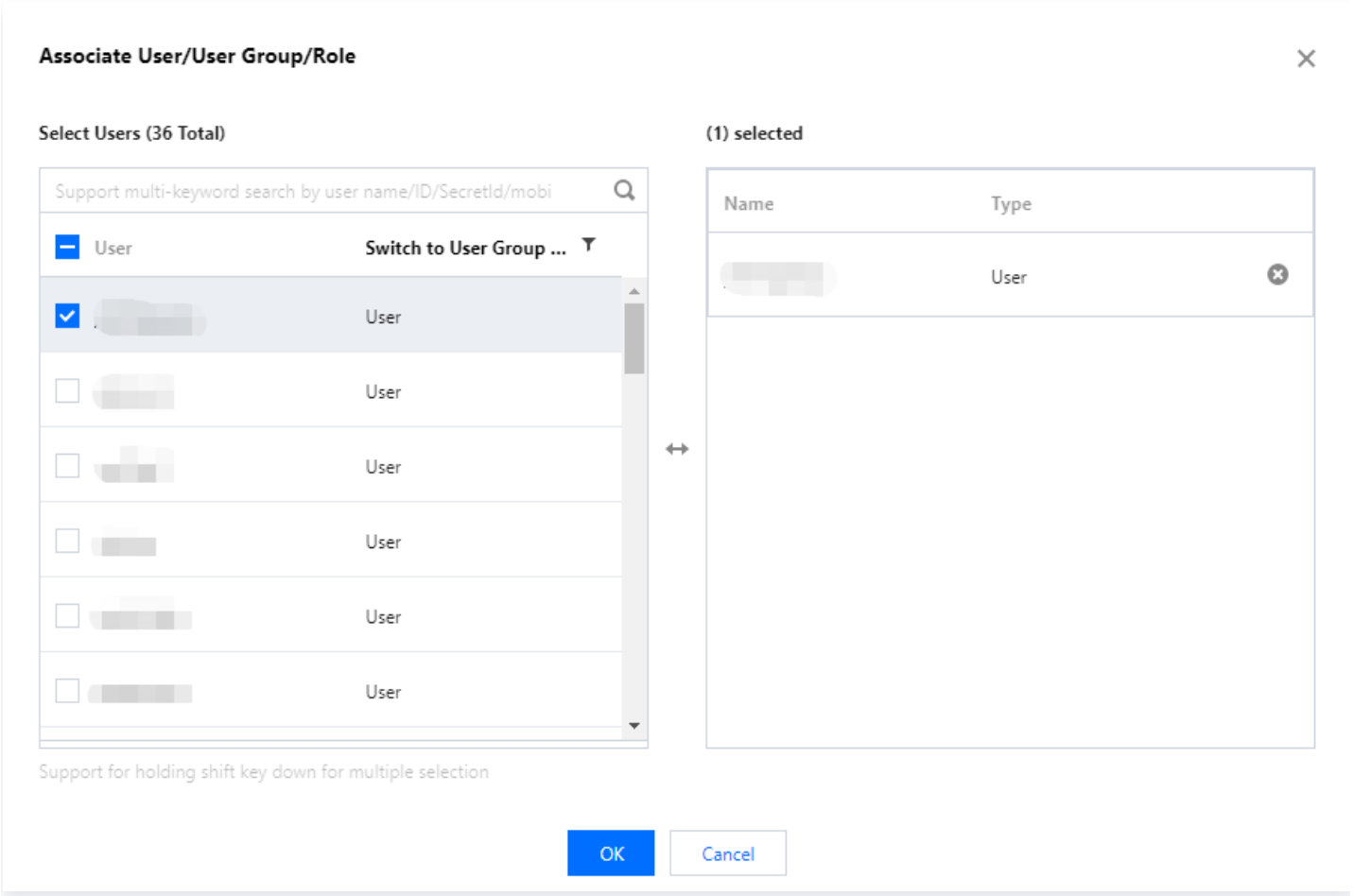
Associate custom policy with sub-account

1. Log in to the CAM console as the root account.
2. In the left sidebar, click **Policies** to go to the policy management page.
3. Click **Custom Policy** on the right to filter, locate the custom policy created in step 1.1, and click **Associate User/Group/Role** in the operation column.

©2013-2023 Tencent Cloud. All rights reserved.

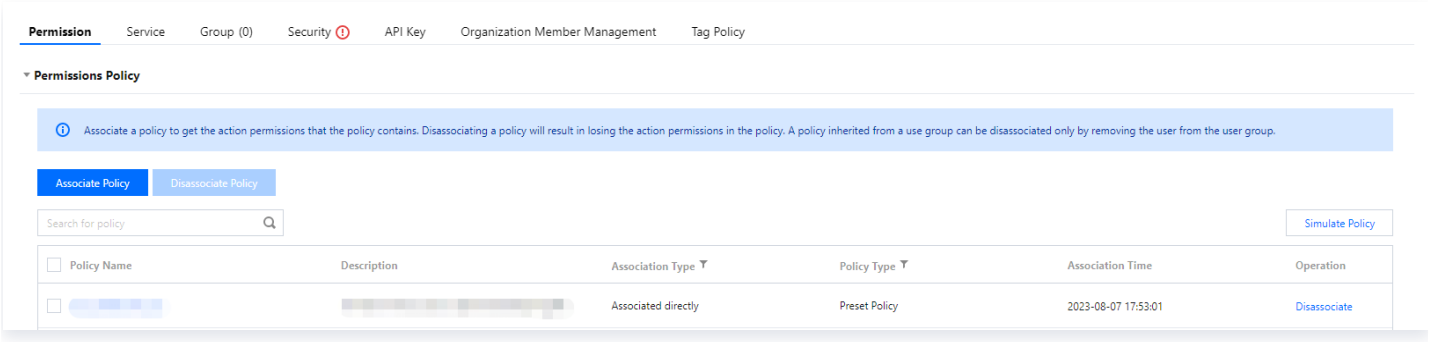


4. Select the sub-account that requires this permission and click **Confirm** to complete the authorization.



5. Click **OK** to complete the authorization. The policy will be displayed in the user's policy list.

©2013-2023 Tencent Cloud. All rights reserved.



Step 2: Grant CKafka Permissions to the Sub-Account

For related operations, refer to the following documents:

- Operational Authorization
- Resource-level Authorization
- Tag-based Authorization

©2013-2023 Tencent Cloud. All rights reserved.

Grant operational-level permissions to sub-accounts

Last updated: 2023-09-07 16:07:26

Scenario

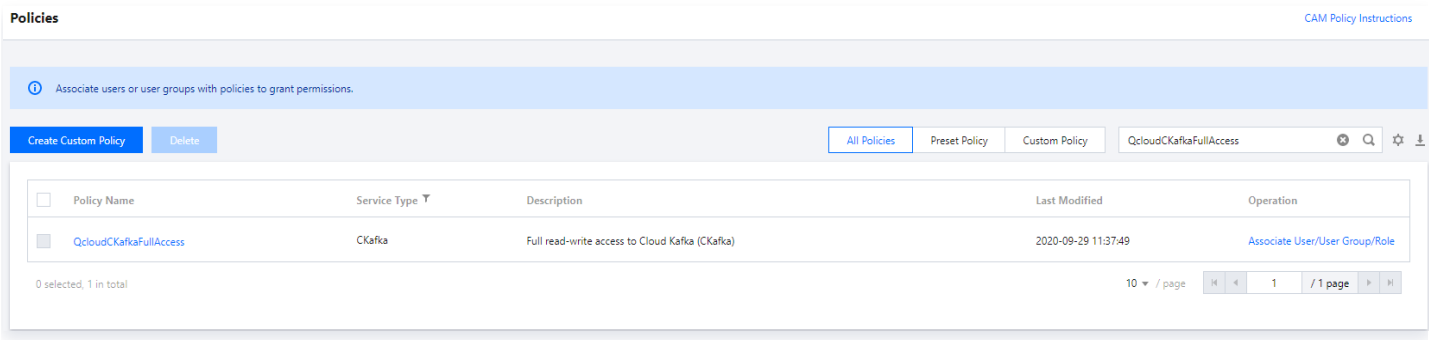
This document guides you through using a Tencent Cloud root account to grant operational-level permissions to sub-accounts. You can grant different read and write permissions to sub-accounts based on your actual needs.

Instructions

Grant full read and write permissions

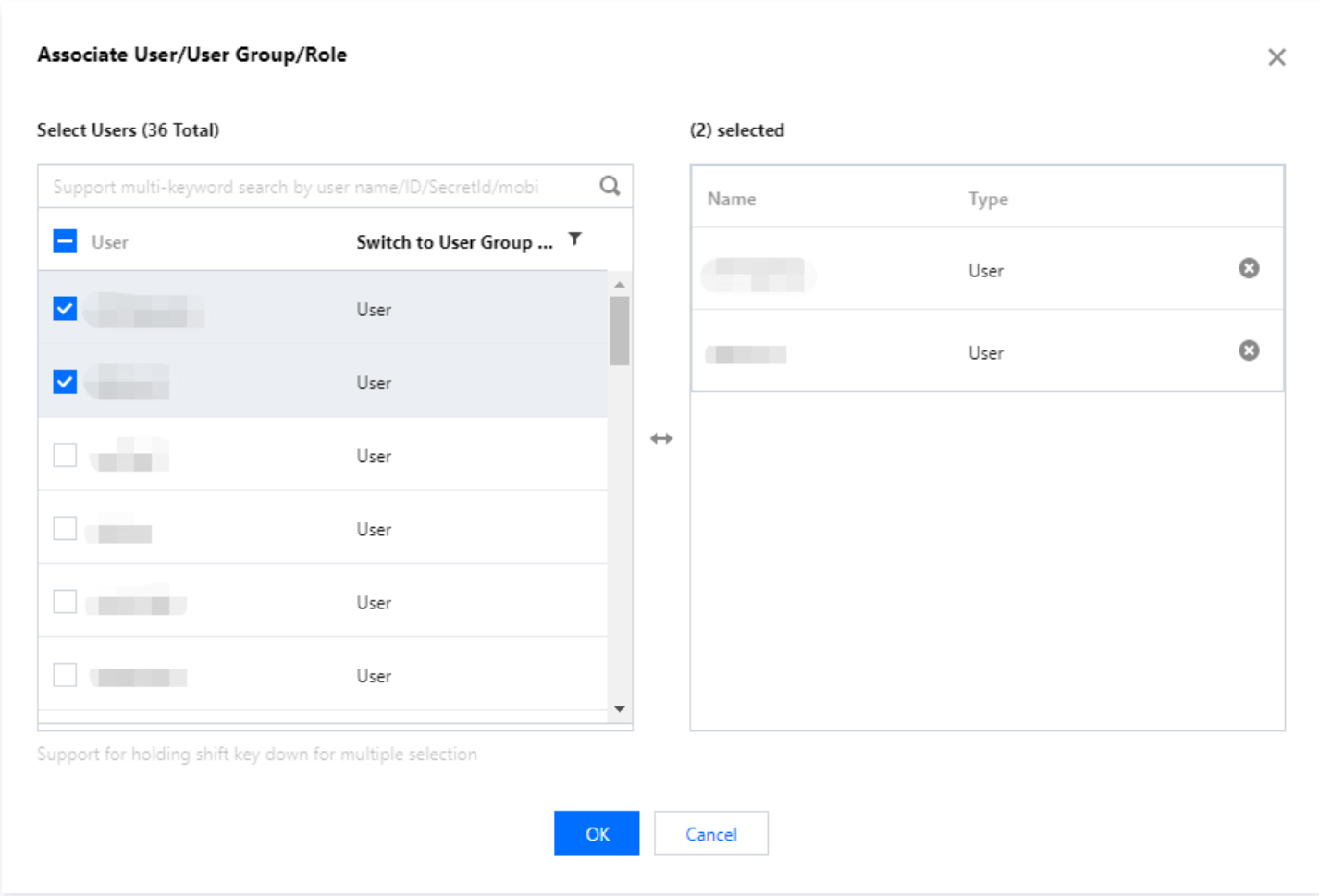
> **Note**
> After granting full read and write permissions to a sub-account, the sub-account will have **complete read and write capabilities** for **all resources** under the root account.

1. Log in to the CAM console as the root account.
2. In the left sidebar, click **Policies** to go to the policy management page.
3. Search for **QcloudCKafkaFullAccess** in the search bar on the right.

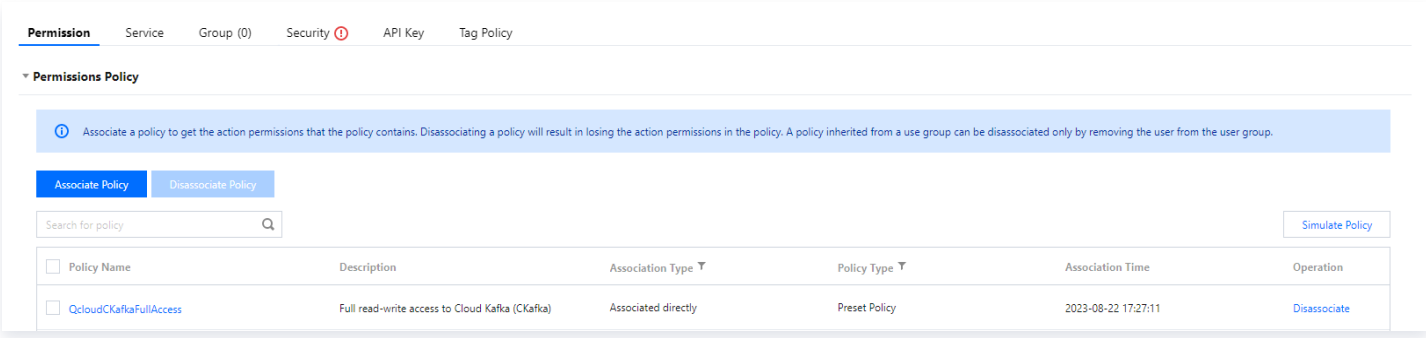


4. In the search results, click the **Associated Users/Groups** of **QcloudCKafkaFullAccess** and select the sub-account to be authorized.

©2013-2023 Tencent Cloud. All rights reserved.



5. Click **OK** to complete the authorization. The policy will be displayed in the user's policy list.



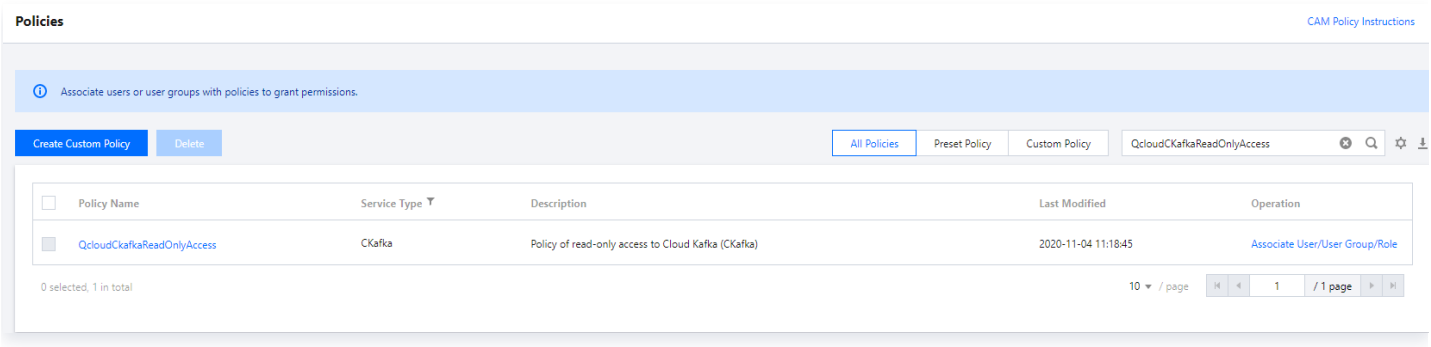
Grant read-only permissions

Note

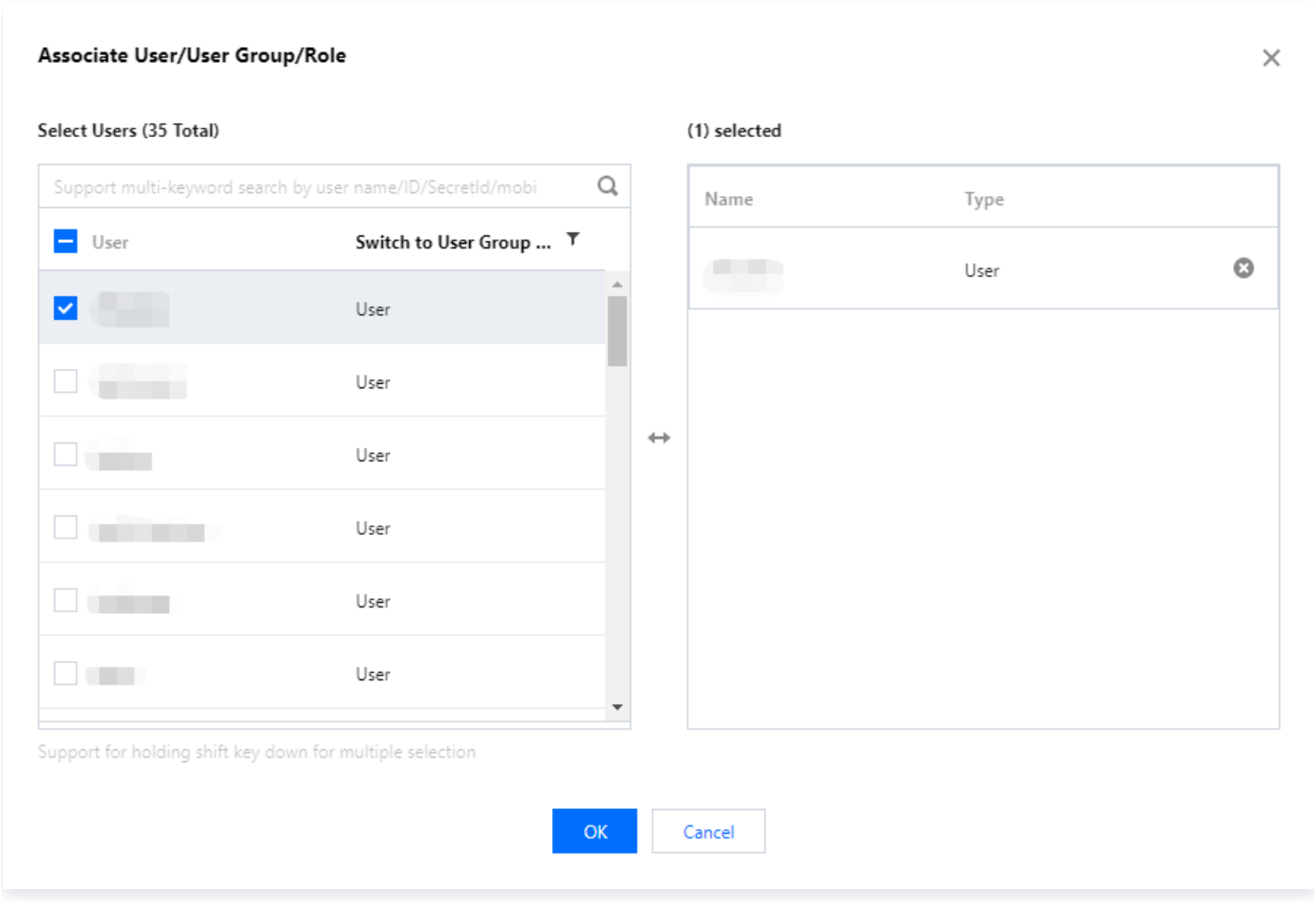
After granting read-only permission to a sub-account, the sub-account will have **read-only access** to **all resources** under the root account.

©2013-2023 Tencent Cloud. All rights reserved.

1. Log in to the CAM console as the root account.
2. In the left sidebar, click **Policies** to go to the policy management page.
3. Search for **QcloudCKafkaReadOnlyAccess** in the right search bar.

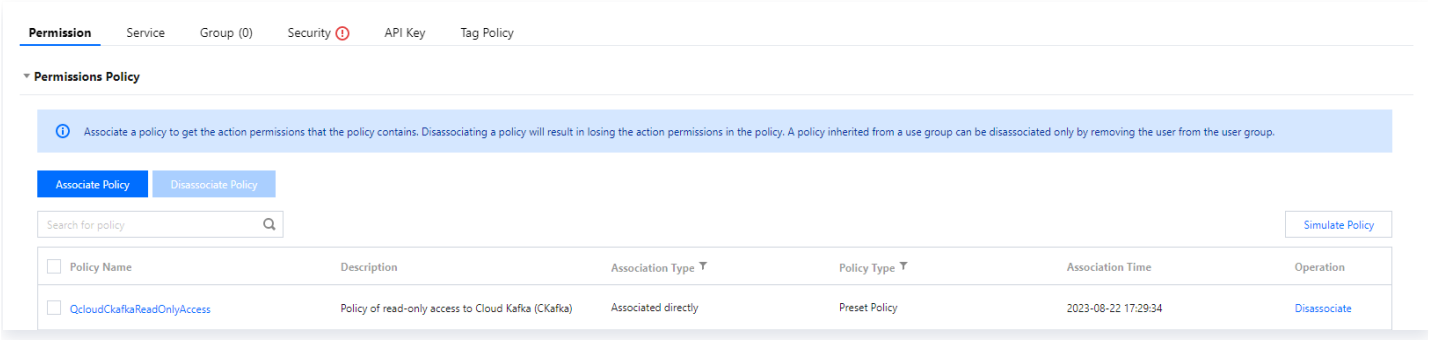


4. In the search results, click the **Associated Users/Groups** of **QcloudCKafkaReadOnlyAccess** and select the sub-account to be authorized.



5. Click **OK** to complete the authorization. The policy will be displayed in the user's policy list.

©2013-2023 Tencent Cloud. All rights reserved.



Other authorization methods

- Resource-level Authorization
- Tag-based Authorization

©2013-2023 Tencent Cloud. All rights reserved.

Grant resource-level permissions to a sub-account

Last updated: 2023-09-07 16:08:16

Scenario

This guide demonstrates how to use a root account to grant resource-level permissions to a sub-account, enabling the sub-account to gain control over specific resources.

Prerequisites

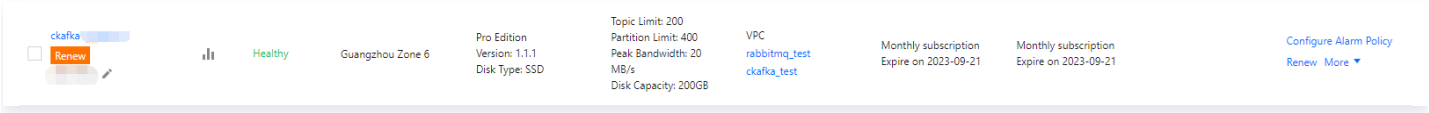
- You possess a Tencent Cloud root account and have already activated the Tencent Cloud Access Management service.
- There is at least one sub-account under the root account, and it has been granted access permissions according to Obtaining Access Authorization for Sub-account.
- You should have at least one CKafka instance.

Instructions

You can use the policy feature in the Access Management console to grant the CKafka resources owned by the root account to a sub-account. Detailed instructions for granting CKafka **resources to a sub-account** are as follows. This example demonstrates granting a cluster resource to a sub-account, and the steps for other resource types are similar.

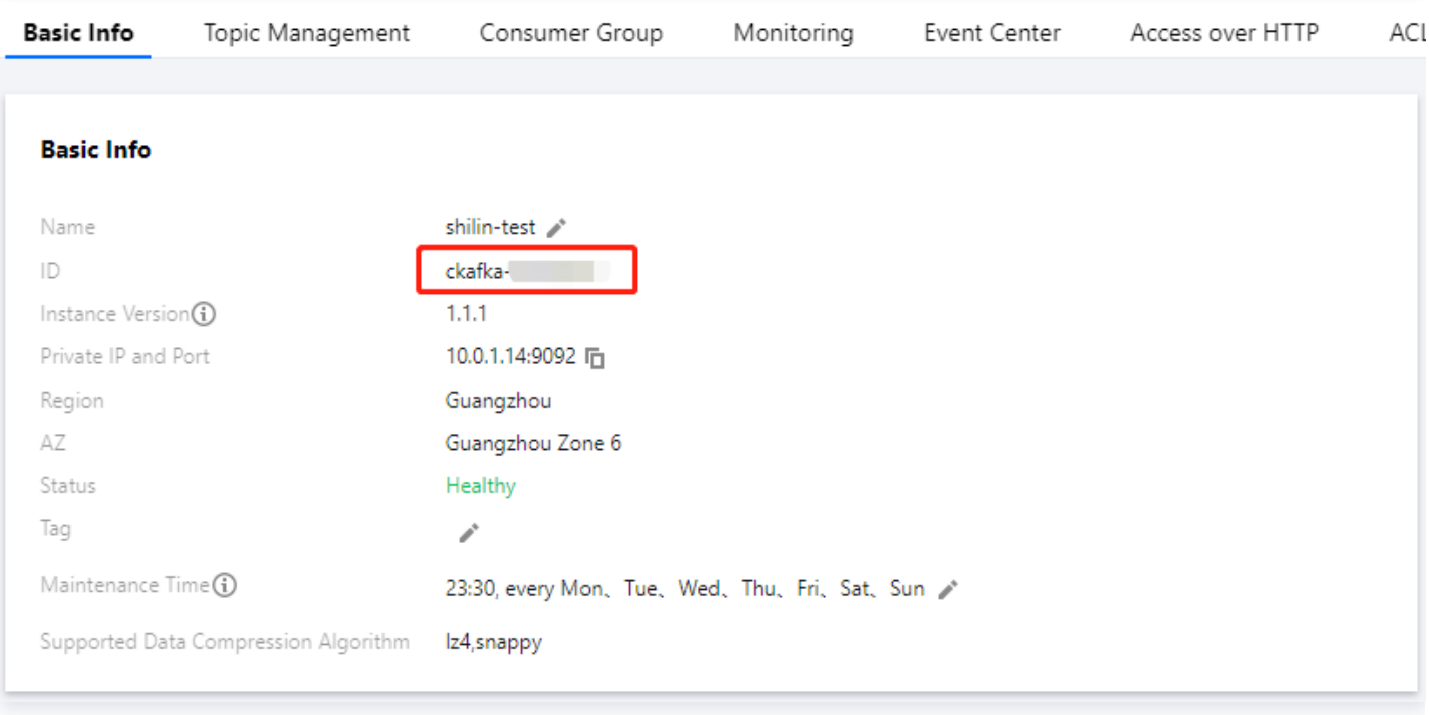
Step 1: Obtain the CKafka Cluster ID

1. Use the **root account** to log in to the CKafka version of the Message Queue console, select an existing cluster instance, and click to enter the details page.



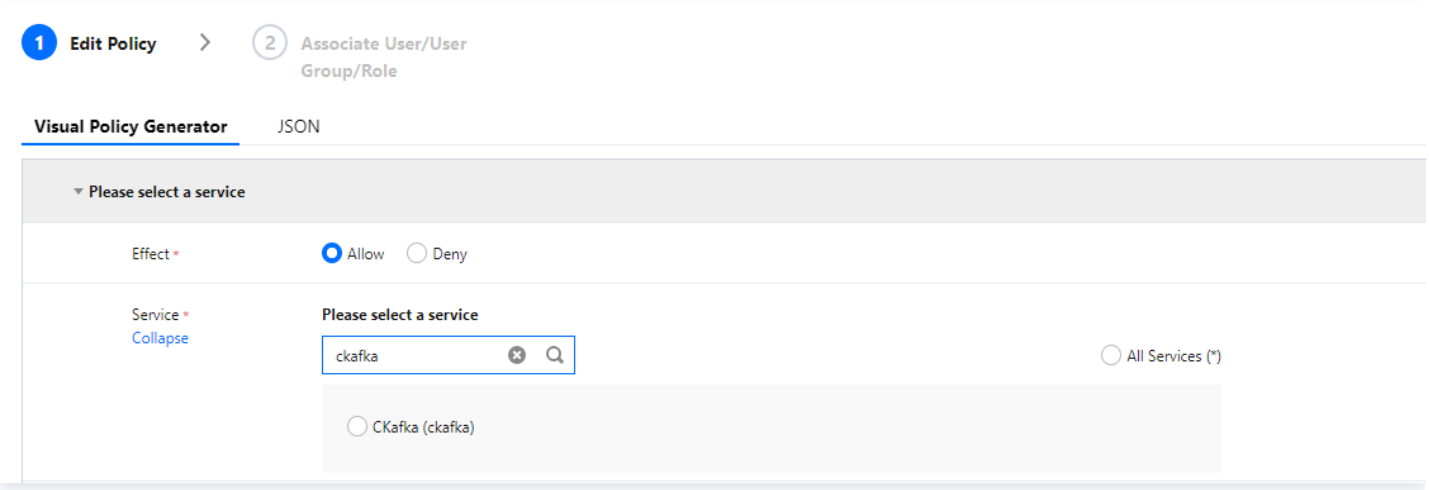
2. In the **Basic Information** section, the **ID** field represents the current CKafka cluster's ID.

©2013-2023 Tencent Cloud. All rights reserved.



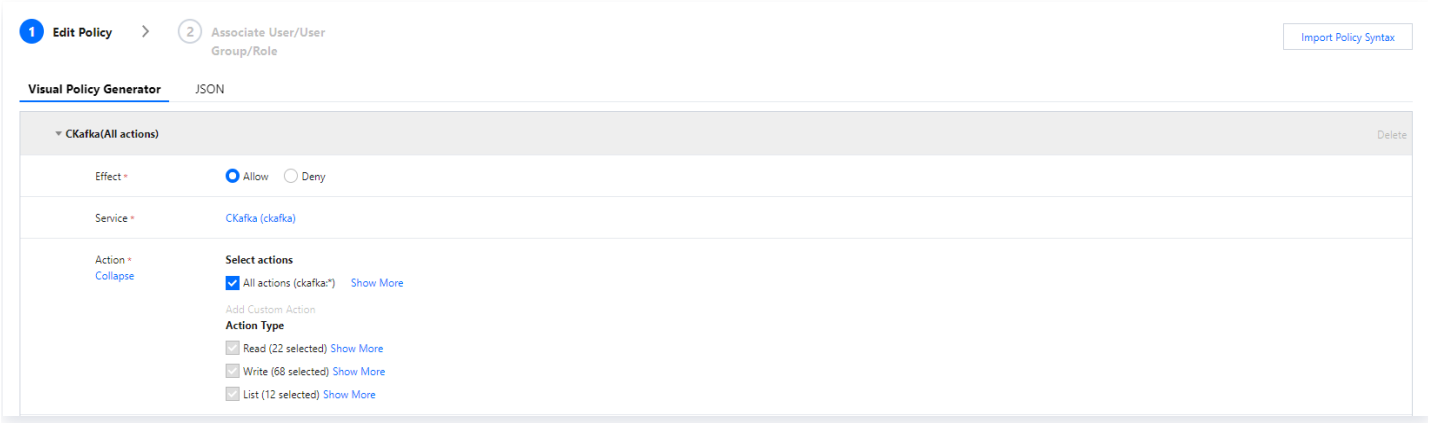
Step 2: Create a new authorization policy

1. Enter the Access Management Console and click **Policies** in the left sidebar.
2. Click **Create Custom Policy** and select **Create by Policy Generator**.
3. In the visual policy generator, keep the **Effect** set to **Allow**. Enter "ckafka" in the **Service** field to filter the results, and select **Message Service (ckafka)** from the list.

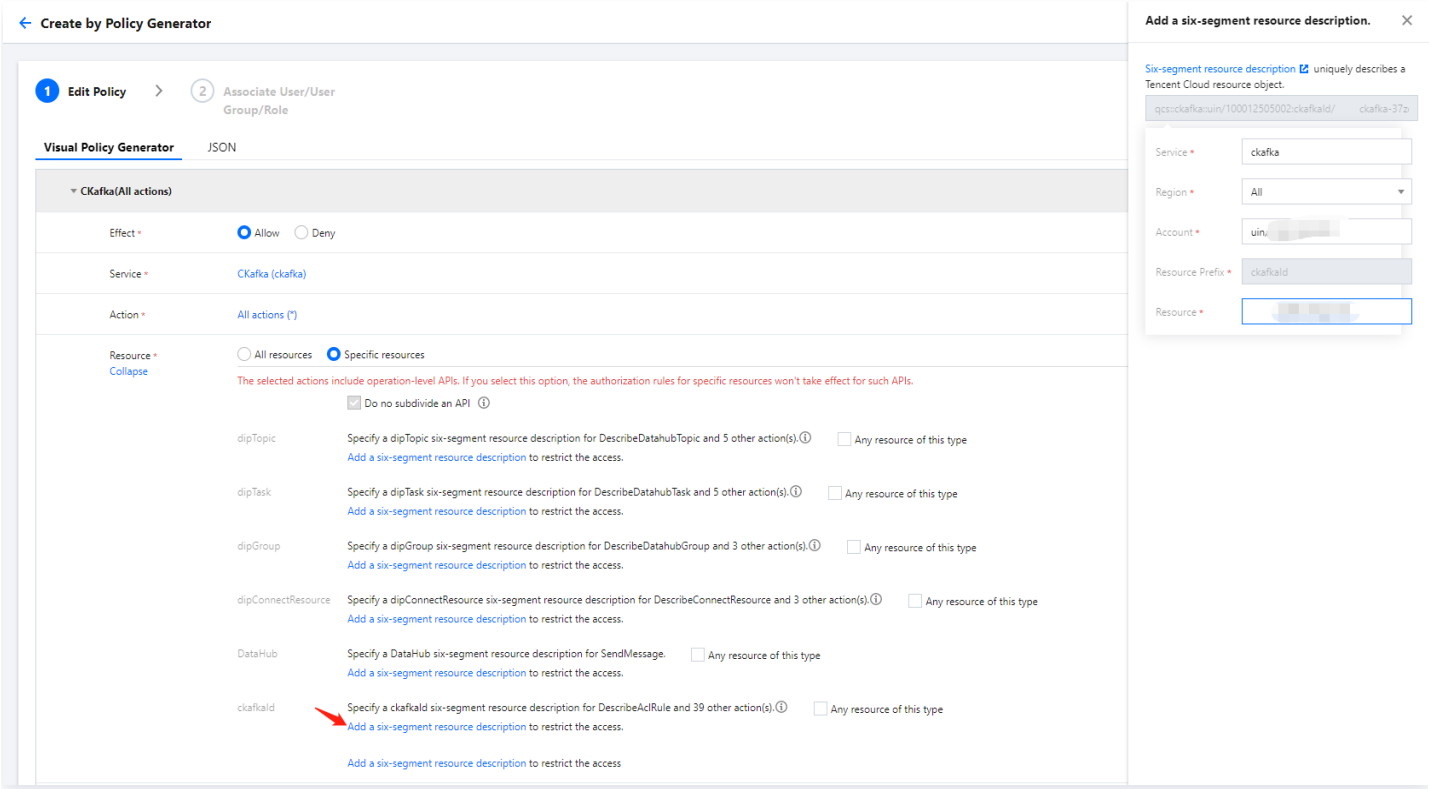


4. In **Operations**, select **All Operations**. You can also choose the operation type according to your needs.

©2013–2023 Tencent Cloud. All rights reserved.

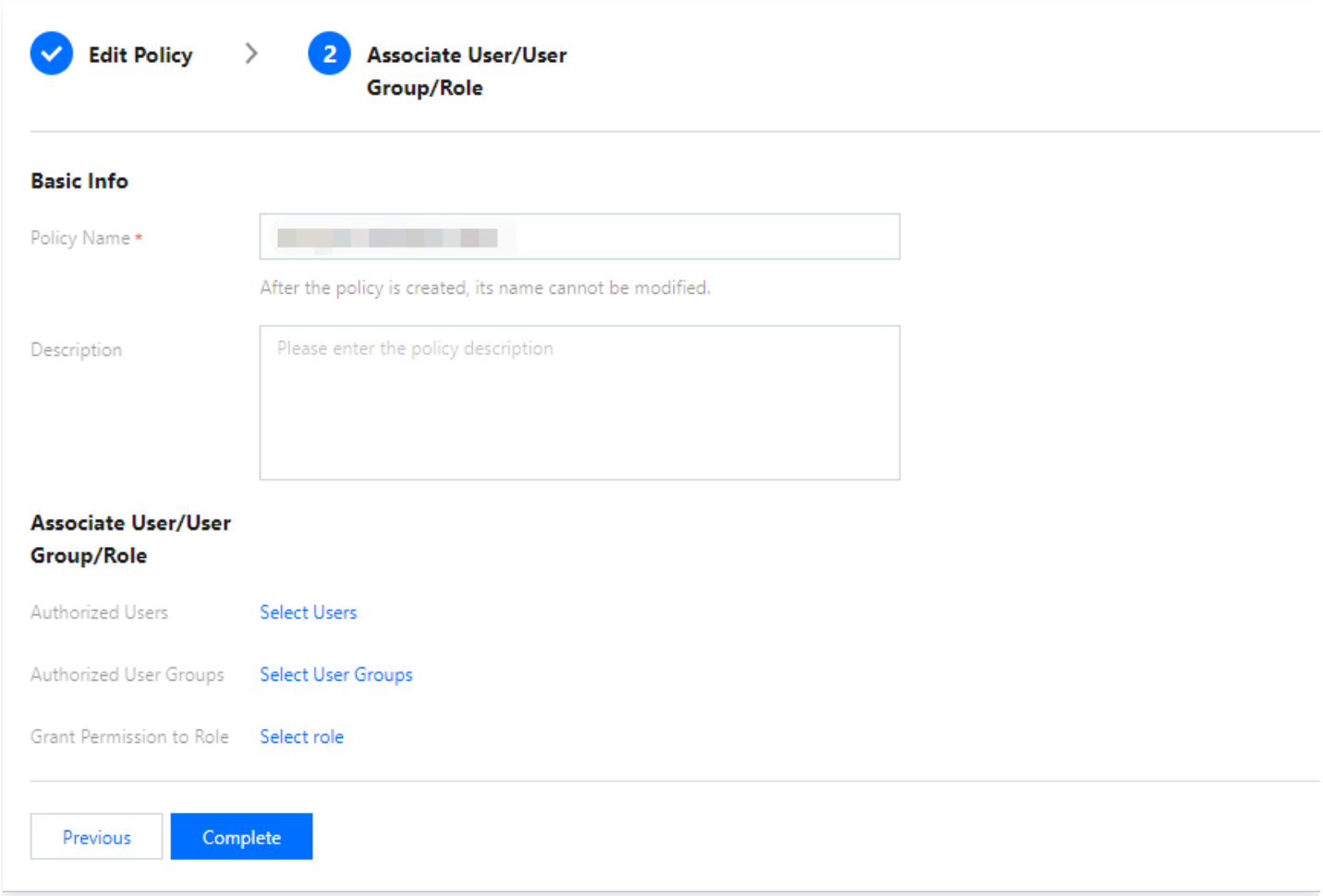


5. In **Resources**, select **Specific resources**, find the **ckafkaId** resource type, and you can either check the box for **Any resource of this type (grant access to all cluster resources)** or click **Add six-segment resource description (grant access to specific cluster resources)**.
6. In the pop-up sidebar, under **Resource**, enter the **Cluster ID**. To obtain the Cluster ID, refer to Step One.



7. Click Next and enter the policy name as needed.
8. Click **Select User** or **Select User Group** to choose the user or user group that requires resource permissions.

©2013-2023 Tencent Cloud. All rights reserved.



9. Click **Finish** to grant the sub-account the ability to access the related resources with the assigned permissions.

Other authorization methods

- Operational Authorization
- Tag-based Authorization

©2013-2023 Tencent Cloud. All rights reserved.

Grant tag-level permissions to sub-accounts

Last updated: 2023-12-06 16:32:48

Scenario

This guide demonstrates how to use tag-based authentication for a root account to grant a sub-account access to resources under a specific tag. The sub-account with granted permissions will gain control over resources associated with the corresponding tag.

Prerequisites

- You possess a Tencent Cloud root account and have already activated the Tencent Cloud Access Management service.
- The root account must have at least one sub-account, and the sub-account must have been granted access permissions according to Obtaining Access Authorization for Sub-account.
- You must have at least one CKafka cluster resource instance.
- You must have at least one **tag**. If you don't, you can create one by going to the Tag Console > **Tag List**.

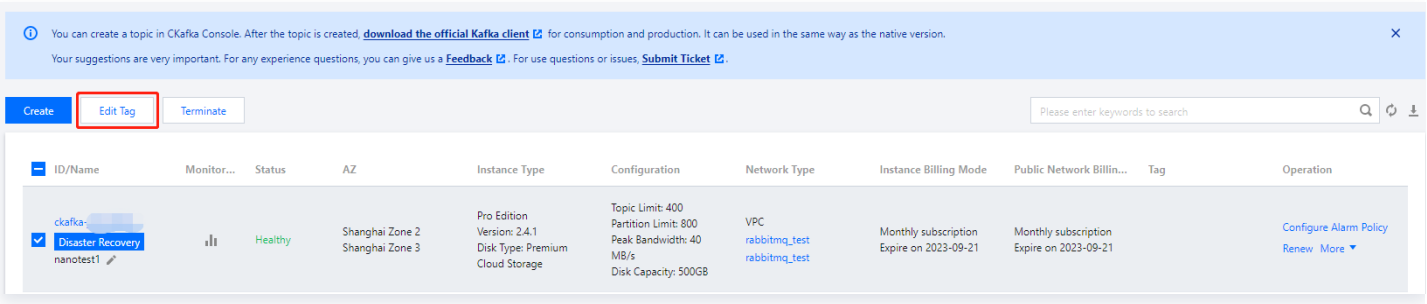
Instructions

You can use the policy feature in the Access Management console to grant read and write permissions for CKafka resources with bound tags owned by the root account to sub-accounts through the **tag-based authorization** method. Detailed steps for **granting resource permissions to sub-accounts based on tags** are as follows.

Step 1: Bind a tag to the resource

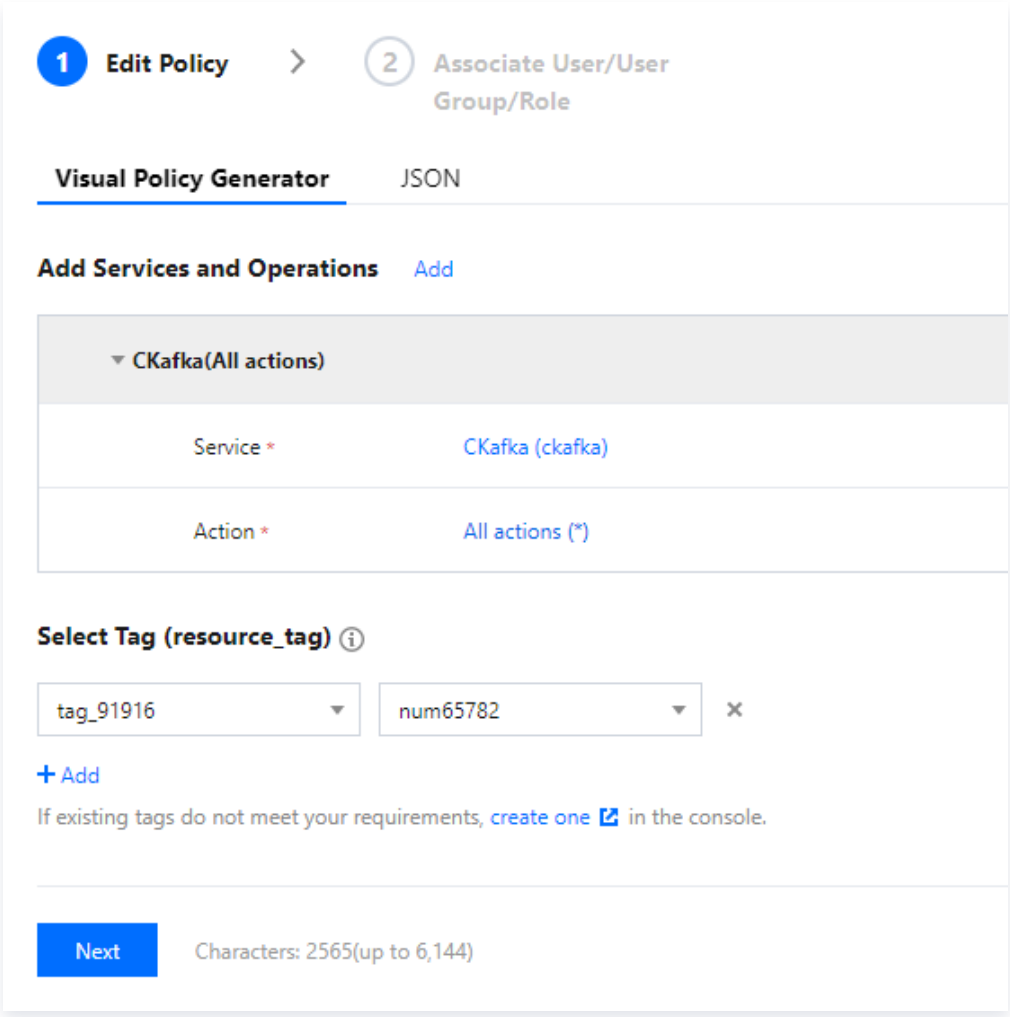
1. Log in to the **TDMQ for CKafka console** using the root account and navigate to the Instance List page.
2. Select the target instance and click on the **Edit Resource Tag** in the upper left corner to bind the resource tag to the instance.

©2013-2023 Tencent Cloud. All rights reserved.



Step 2: Authorize by Tag

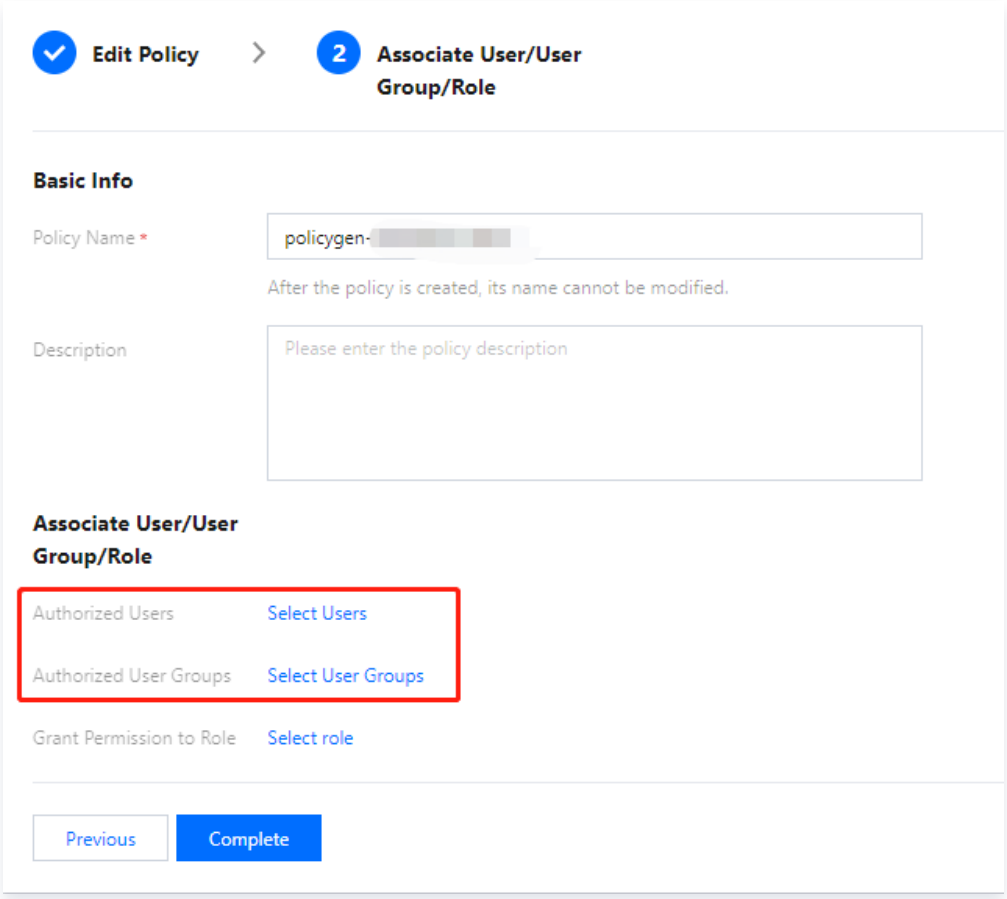
1. Enter the Access Management Console and click **Policies** in the left sidebar.
2. Click **Create Custom Policy** and select **Authorize by Tag**.
3. In the visual policy generator, enter Ckafka in the **Service** field to filter the results. Select **Message Service (ckafka)** from the results. In the **Action** field, select **All Actions**. You can also select the appropriate actions as per your requirements.



4. Click Next and enter the policy name as needed.

©2013–2023 Tencent Cloud. All rights reserved.

5. Click **Select User** or **Select User Group** to choose the user or user group that requires resource permissions.



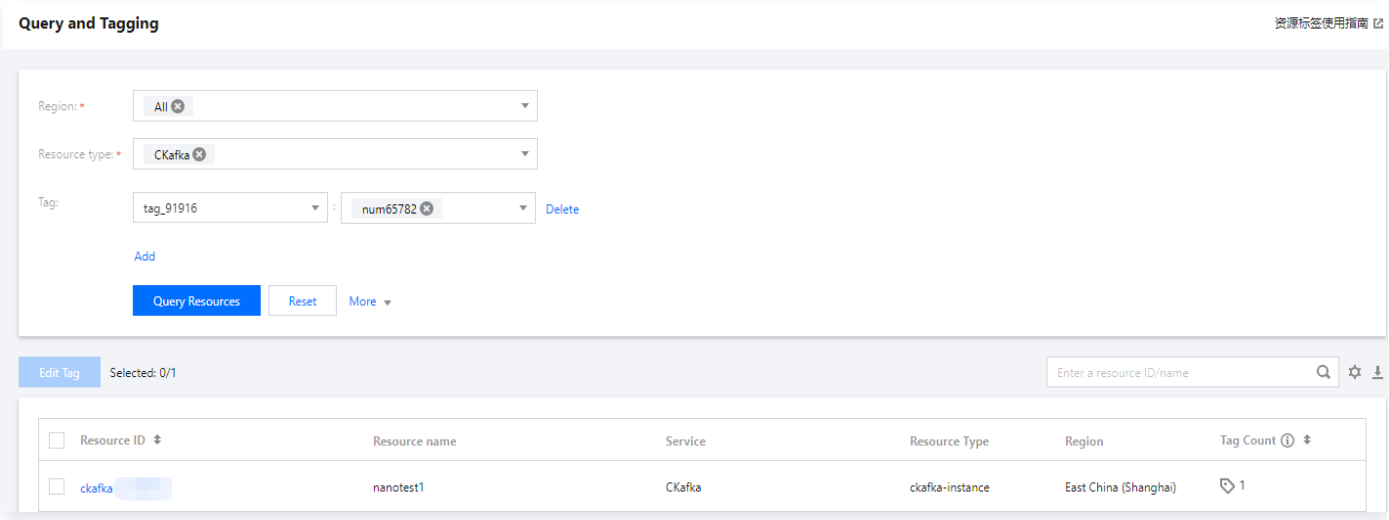
6. Click **Complete**, and the relevant sub-account will be able to control resources under the specified tag according to the policy.

Centralized management of resource tags

You can also manage resource tags in a unified manner at the Tag Control Console. The detailed steps are as follows:

1. Log in to the Tencent Cloud Tag Console.
2. In the left navigation bar, select Resource Tags, choose the query conditions as needed, and in **Resource Type**, select **TDMQ for CKafka** > **CKafka Instance**.
3. Click **Query Resources**.
4. In the results, select the required resources and click **Edit Tags** to perform batch binding or unbinding of tags.

©2013-2023 Tencent Cloud. All rights reserved.



Other authorization methods

- Operational Authorization
- Resource-level Authorization

©2013-2023 Tencent Cloud. All rights reserved.

Access via Public Domain Name Step 1. Creating an Instance

Last updated: 2023-09-07 16:25:04

Scenario

This document describes how to create an instance in the CKafka console.

Preparations

- You have signed up for a Tencent Cloud account as instructed in Signing Up.
- You have created a VPC. For more information, see Creating VPC.

Instructions

1. Log in to the CKafka console.
2. Click **Instance List** on the left sidebar, then click **Create** to access the instance purchase page, and select the purchase information based on your business requirements.
   - Billing Mode: Pro Edition instances support both monthly subscription and pay-as-you-go, and Standard Edition instances support monthly subscription only.
   - Specs Type: CKafka instances are divided into Standard Edition and Pro Edition based on their specifications. For the comparison between the two editions, see Product Specifications.
   - Kafka Version: Select a Kafka version based on your business needs. For more information, see Suggestions for CKafka Version Selection.
   - Region: select a region close to the resource for client deployment.
   - Availability Zone:
     - Standard Edition: it does not support multi-AZ deployment.
     - Pro Edition: If the current region supports multi-AZ deployment, you can select up to four AZs for deployment. For more information on cross-AZ deployment, see Cross-AZ Deployment.
   - Product Specs: select an appropriate model based on the peak bandwidth and disk capacity.
   - Message Retention Period: The value range is 24 hours (default) to 2,160 hours. Expired messages will be deleted to free up disk space.
     When the disk capacity is insufficient (i.e., the disk utilization reaches 90%), old messages will be deleted in advance to ensure the service availability.

©2013-2023 Tencent Cloud. All rights reserved.

- VPC: Select an appropriate VPC based on your business needs.
- Public Network Bandwidth: The Standard Edition offers 1 Mbps public network bandwidth by default, while the Pro Edition provides 3 Mbps. If required, you can pay to upgrade the public network bandwidth for your Pro Edition instance. For more information, refer to Public Network Bandwidth Management.
- Instance Name: When purchasing multiple instances, you can batch create instances by its numeric suffix (which is numbered in an ascending order) or its designated pattern string. For detailed directions, see Naming with Consecutive Numeric Suffixes or Designated Pattern String.

3. Click **Buy Now**. The created instance will be displayed in the instance list in about 3-5 minutes.

©2013-2023 Tencent Cloud. All rights reserved.

Step 2. Add a public route

Last updated: 2023-09-07 16:25:16

Scenario

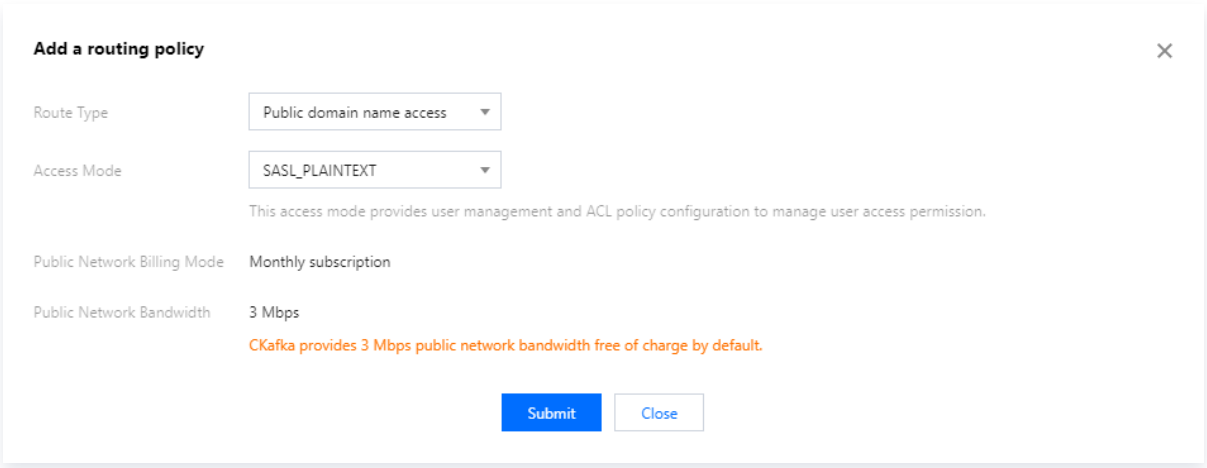
To enable public network access, you need to add a public route for the instance. This document describes how to add a public route for a created instance.

Preparations

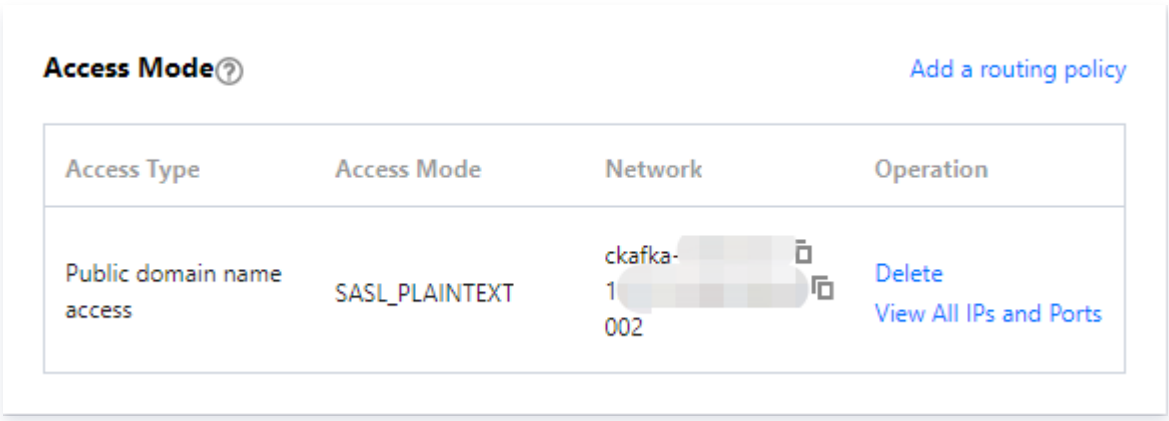
You have created an instance. For more information, see Step 1. Create an Instance.

Instructions

1. On the Instance List page, click the ID/name of the instance created in Step 1. Create an Instance.
2. On the instance details page, click **Add a routing policy** in the **Access Mode** section to add a public network route.



3. Then, you can get the domain name and port for public network access.



©2013-2023 Tencent Cloud. All rights reserved.

Step 3. Create a Topic

Last updated: 2023-09-07 16:25:26

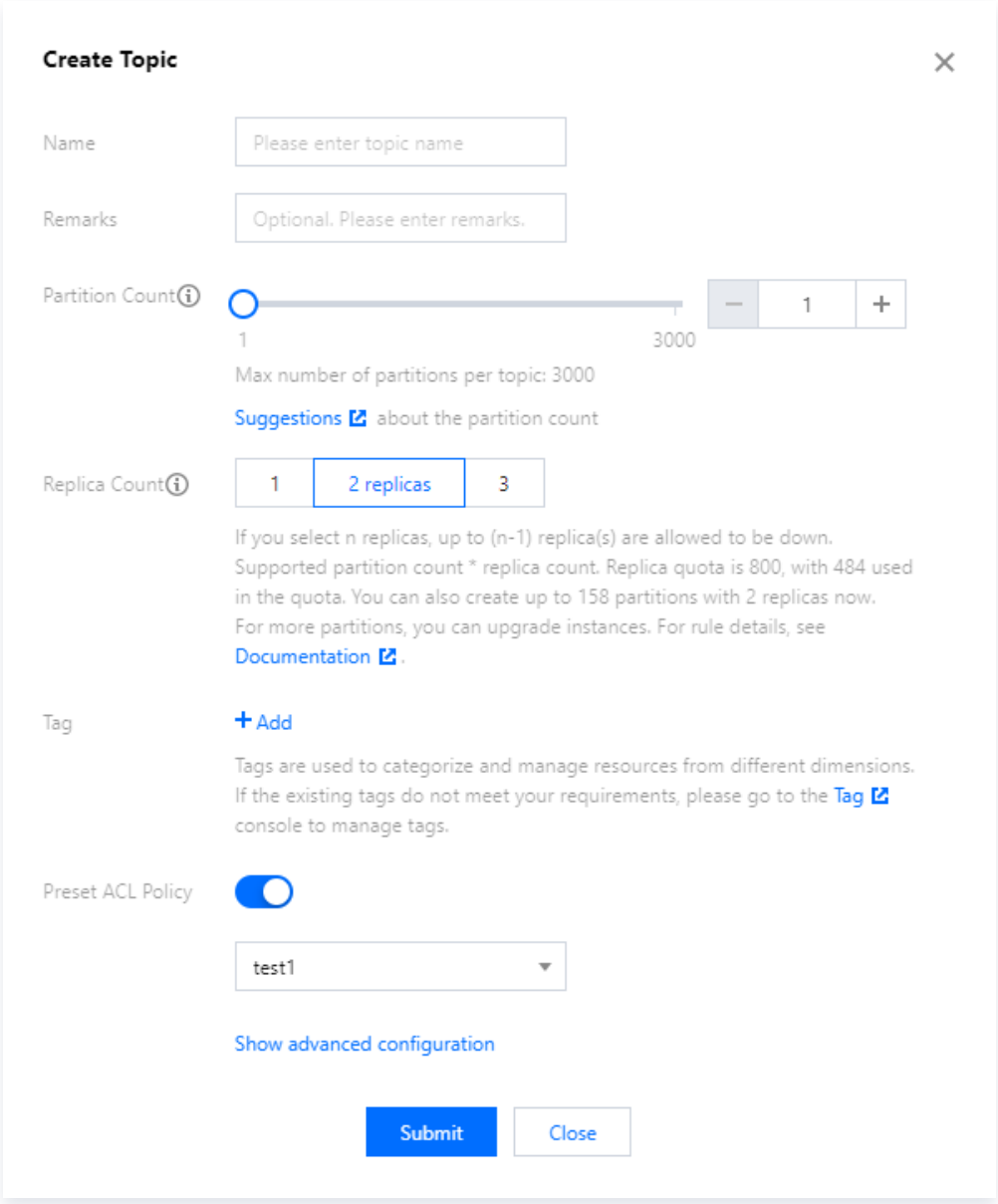
Scenario

This document describes how to create a topic in a created instance in the CKafka console.

Instructions

1. Log in to the CKafka console.
2. On the **Instance List** page, click on the "ID/Name" of the instance created in Step 1 to access 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** window, set the number of partitions and replicas and other parameters.

©2013-2023 Tencent Cloud. All rights reserved.



- Name: the topic name, which cannot be changed once entered and can contain only letters, digits, "_", "-", and ".".
- Partition Count: It is a concept in physical partition, where one topic can contain one or more partitions. CKafka uses partition as an allocation unit.
- Replica Count: the number of partition replicas, which ensure the availability of partitions. For data reliability concerns, CKafka does not support single-replica topics currently. 2 replicas are created by default.
  Replicas are also counted into the number of partitions. For example, if you create 1 topic with 6 partitions, and 2 replicas for each partition, then you have a total of 12 partitions (1 x 6 x 2).
- Tags: Set resource tags. For a detailed introduction to tags, please refer to Tag Management.
- Preset ACL Policy: Select the preset ACL policy. For more information on ACL policy,

©2013-2023 Tencent Cloud. All rights reserved.

see Configuring ACL Policy.

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

©2013–2023 Tencent Cloud. All rights reserved.

Step 4. Configure an ACL Policy

Last updated: 2023-09-07 16:25:36

Scenario

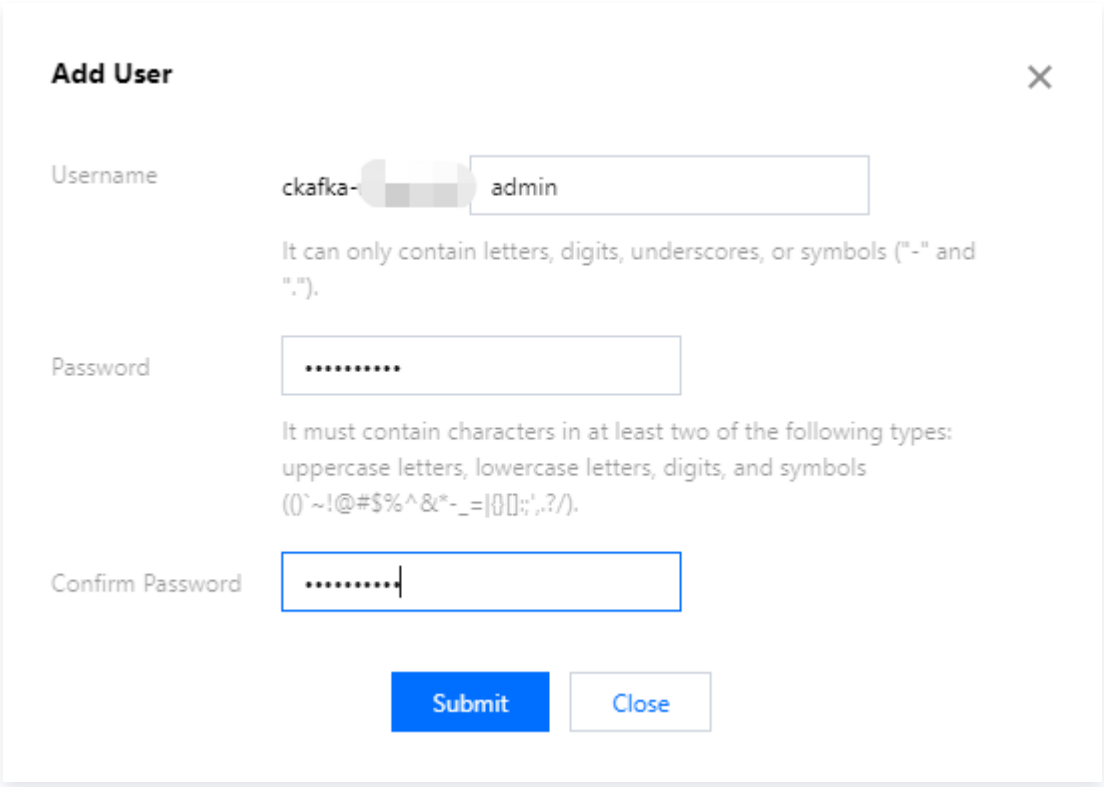
To enable public network access, you need to configure an ACL policy for a topic. This document describes how to configure an ACL policy for a created topic in the CKafka console.

Preparations

You have created a topic. For more information, see Step 3. Create a Topic.

Instructions

1. On the instance details page, select **ACL Policy Management** > **User Management** and click **Create** to add a user and set the username and password.



2. Select the **Policy List** tab, click the **Resource** tab, and choose Edit ACL Policy in the topic operation column created in **Step 3. Create a Topic** to grant read and write permissions to the user.

©2013-2023 Tencent Cloud. All rights reserved.

Step 5. Send/Receive Messages Using SDK to Receive/Send Message (Recommended)

Last updated: 2023-09-07 16:26:04

Overview

This document provides a step-by-step guide on how to connect to CKafka in a public network environment and send/receive messages using a Java client as an example. For other language clients, please refer to SDK Documentation.

Preparations

- Ensure JDK 1.8 or later is installed
- You have installed Maven 2.5 or later.
- Download the demo

Instructions

Step 1. Prepare the configuration

1. Decompress the downloaded demo and enter the `PUBLIC_SASL` directory under `javakafkademo`.
2. Modify the JAAS configuration file `ckafka_client_jaas.conf`.

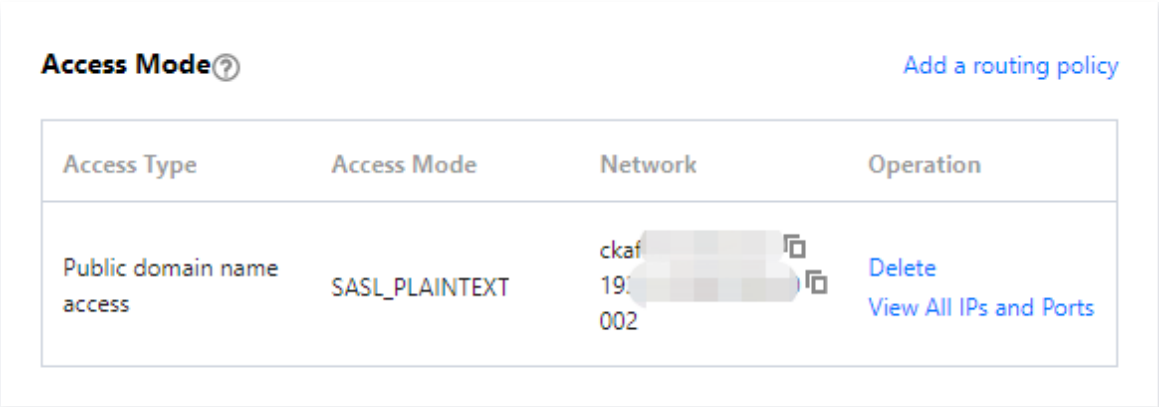
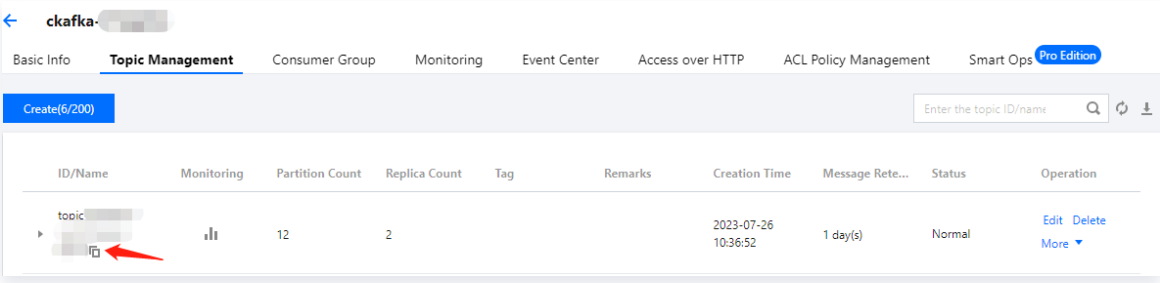
```
KafkaClient {
org.apache.kafka.common.security.plain.PlainLoginModule required
username="yourinstance#yourusername"
password="yourpassword";
};
```

> **Note**
> The username is composed of the `Instance ID`, followed by a `#`, and then the `configured username`. The password is the configured user password.

3. Modify the Kafka configuration file `kafka.properties`.

©2013-2023 Tencent Cloud. All rights reserved.

```
## 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=ckafka-xxxxxxx
## 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
The path of the JAAS configuration file named ckafka_client_jaas.conf.
java.security.auth.login.config.plain=/xxxx/ckafka_client_jaas.conf
```

Category	Note
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 from 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.
java.security.aut	Enter the path of the JAAS configuration file `ckafka_client_jaas.conf`.

©2013–2023 Tencent Cloud. All rights reserved.

h.logi n.con fig.pl ain

Step 2. Send messages

1. Compile and execute the message sending program, CKafkaSaslProducerDemo.java.

```
public class CKafkaSaslProducerDemo {

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

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

        Properties props = new Properties();
        //Set the access point. Obtain the corresponding topic access point from
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");

        //Set the serialization method for Kafka version messages in the message
queue.
        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 a request.
        props.put(ProducerConfig.MAX_BLOCK_MS_CONFIG, 30 * 1000);
        //Set the number of retries for the client.
        props.put(ProducerConfig.RETRIES_CONFIG, 5);
        //Set the interval between retries for the client.
        props.put(ProducerConfig.RECONNECT_BACKOFF_MS_CONFIG, 3000);
        //Construct a Producer object. Note: a Producer object is thread-safe, and
generally one Producer object is sufficient for a process.
        KafkaProducer<String, String> producer = new KafkaProducer<>(props);
```

©2013–2023 Tencent Cloud. All rights reserved.

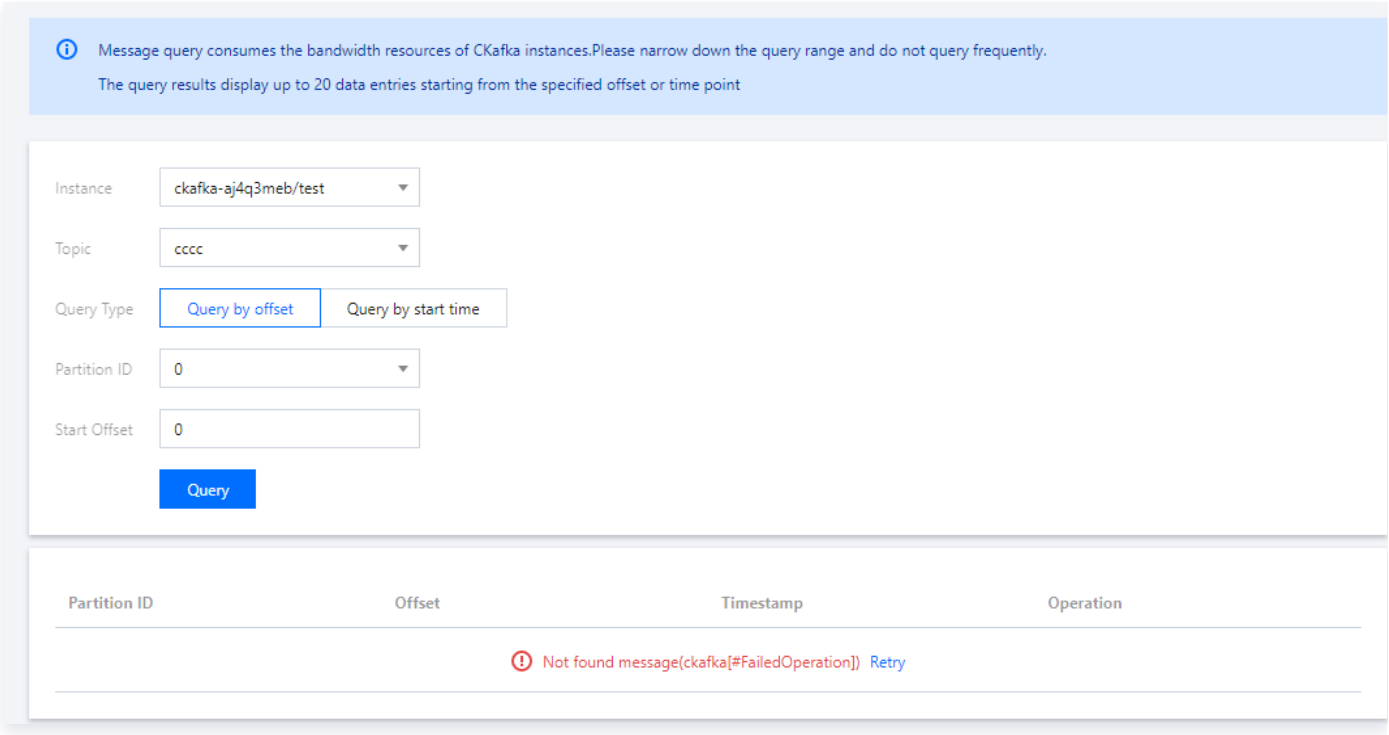
```
        //Construct a Kafka version message for the message queue.
        String topic = kafkaProperties.getProperty("topic"); //Topic of the message.
Enter the topic you created in the console.
        String value = "This is the CKafka message content"; //Content of the
message.

        try {
            //Batch retrieval of Future objects can enhance performance.
However, be cautious not to make the batch size excessively large.
            List<Future<RecordMetadata>> futures = new ArrayList<>(128);
            for (int i =0; i < 100; i++) {
                //Dispatch a message and obtain 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();
                    System.out.println("Produce 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 running result (output).

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

3. In the CKafka console **Topic Management** page, select the corresponding topic, click **More** > **Message Query** to view the recently sent message.

©2013–2023 Tencent Cloud. All rights reserved.



Step 3. Consume messages

1. Compile and run the CKafkaSaslConsumerDemo.java program to subscribe to messages.

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

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

        Properties props = new Properties();
        //Set the access point. Obtain the corresponding topic access point from 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");
        //The maximum allowed interval between two polls.
```

©2013–2023 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.
        //Be cautious not to set this value too high. If too much data is polled and not
consumed before the next poll, a load balancing will be triggered, causing a delay.
        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");
        //Specify the consumer group to which the current consumer instance
belongs. Enter the group you created in the console.
        //Instances within the same consumer group consume messages in a load-
balanced manner.
        props.put(ConsumerConfig.GROUP_ID_CONFIG,
kafkaProperties.getProperty("group.id"));
        //Construct a consumer object, which essentially generates a consumer
instance.
        KafkaConsumer<String, String> consumer = new KafkaConsumer<String,
String>(props);
        //Set the topics for the consumer group to subscribe to, multiple topics can be
subscribed.
        //If the GROUP_ID_CONFIG is the same, it is recommended to set the
subscribed topics to be the same as well.
        List<String> subscribedTopics =  new ArrayList<String>();
        //To subscribe to multiple topics, simply add them 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);

        //Consume messages in a loop.
        while (true){
            try {
                ConsumerRecords<String, String> records = consumer.poll(1000);
                // The data must be consumed before the next poll, and the total time
taken should not exceed SESSION_TIMEOUT_MS_CONFIG.
                for (ConsumerRecord<String, String> record : records) {
```

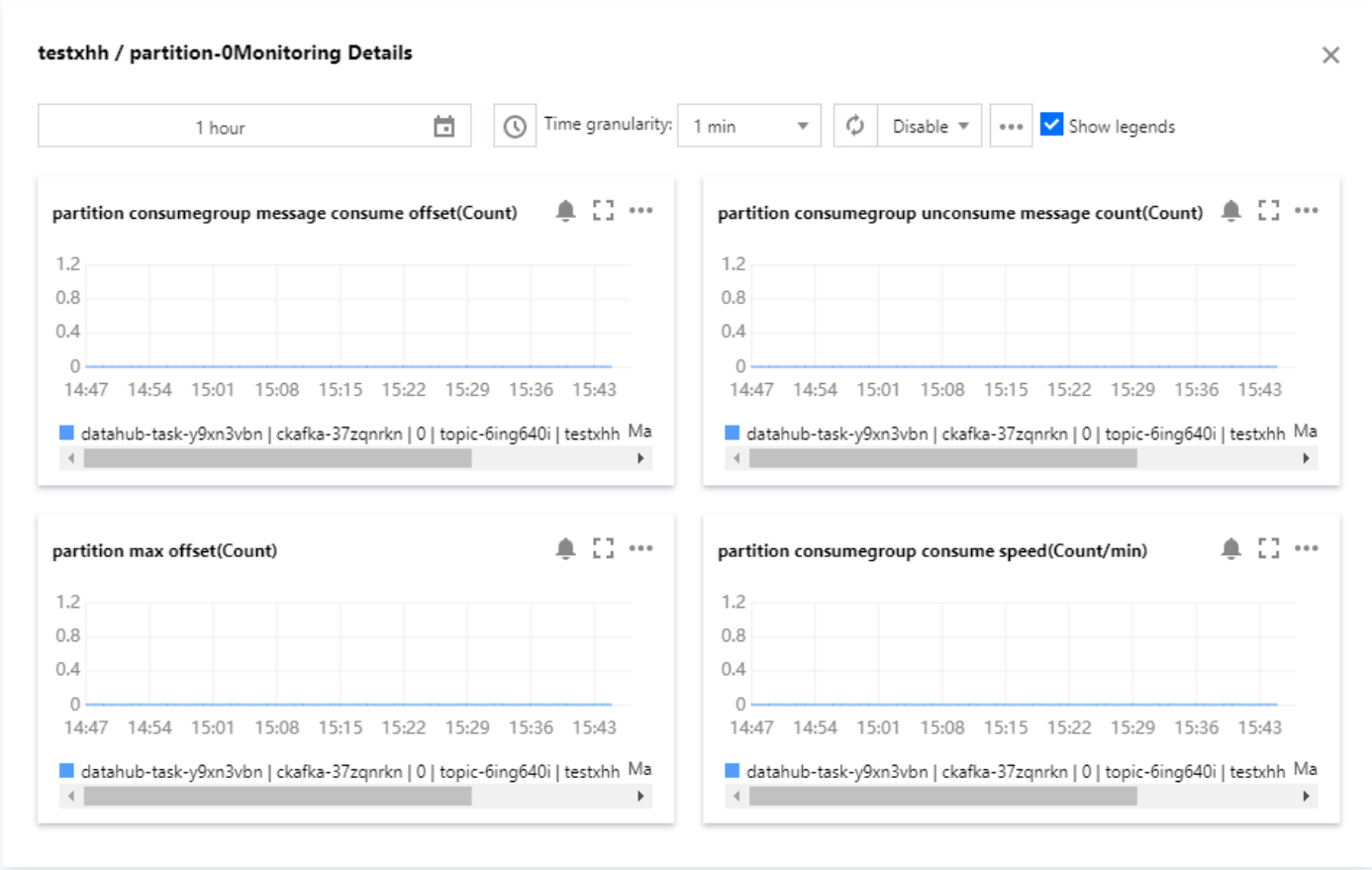
©2013–2023 Tencent Cloud. All rights reserved.

```
                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 (output).

```
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, and click **Query Details** to view the consumption details.



©2013–2023 Tencent Cloud. All rights reserved.

Running Kafka Client (Optional)

Last updated: 2023-09-07 16:26:15

Scenario

This guide demonstrates how to utilize Kafka APIs after purchasing the CKafka service. Download and extract the Kafka toolkit locally, and conduct basic testing on Kafka APIs.

Instructions

Step 1. Install a JDK.

1. Check Java installation.

Open a terminal window and run this command:

```
java -version
```

If the output displays a Java version number, it indicates that Java is successfully installed. If Java is not installed, please download and install a Java Development Kit (JDK).

2. Set up the Java environment.

Set the `JAVA_HOME` environment variable and point it to the Java installation directory on your machine.
For example, if you use Java JDK 1.8.0_20, the outputs on different operating systems are as follows:

Operating System	Output
Windows	Set the environment variable JAVA_HOME to C:\Program Files\Java\jdkjdk1.8.0_20
Linux	export JAVA_HOME=/usr/local/java-current
Mac OSX	export JAVA_HOME=/Library/Java/Home

Add the Java compiler path to the system path:

Operating System	Output

©2013-2023 Tencent Cloud. All rights reserved.

Windows	Add `;C:\Program Files\Java\jdk1.8.0_20\bin` to the end of the system variable `Path`
Linux	export PATH=$PATH:$JAVA_HOME/bin/
Mac OSX	not required

Use the `java -version` command to check your Java installation.

Step 2. Download the Kafka installation file.

Download and extract the Kafka installation package from the official website.

Step 3. Test Kafka APIs.

1. Configure an ACL policy locally.
   1.1 In the `./config` directory of the toolkit, append the following content to the end of both the producer.properties and consumer.properties files:

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

   1.2 Create a file named `ckafka_client_jaas.conf`, and add the following content.

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

> **Note**
> The username is the combination of the `Instance ID`, `#`, and the `recently configured username`, while the password is the recently configured user password.

   1.3 In the `./bin` directory of the toolkit, add a declaration for the JAAS file path (which must be a complete path) at the beginning of the kafka-console-producer.sh and kafka-console-consumer.sh files:

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

©2013-2023 Tencent Cloud. All rights reserved.

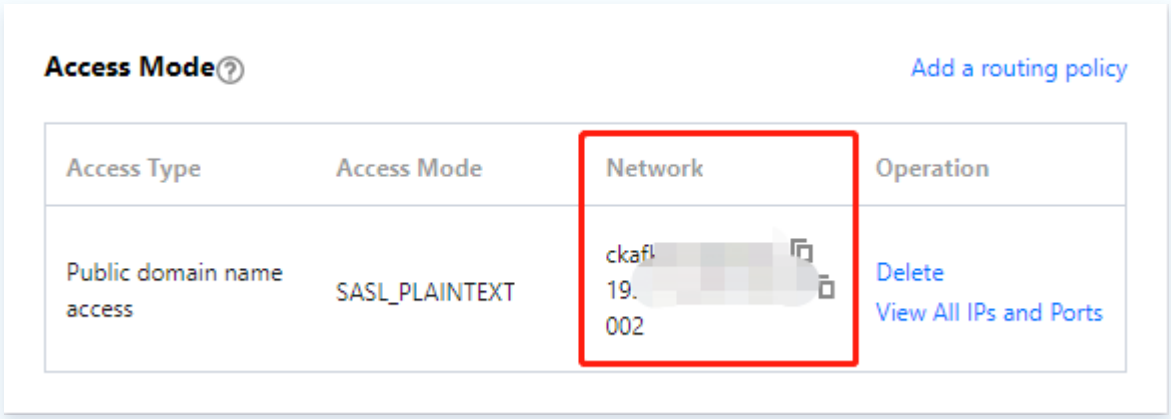
2. Go to the `./bin` directory, and produce and consume a message via CLI commands.

2.1 Open a terminal window 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 access, which can be obtained from the **Access Method** module on the console instance details page.



- Topic: Replace XXXX with the topic name, which can be obtained from the 'Topic Management' page on the console.

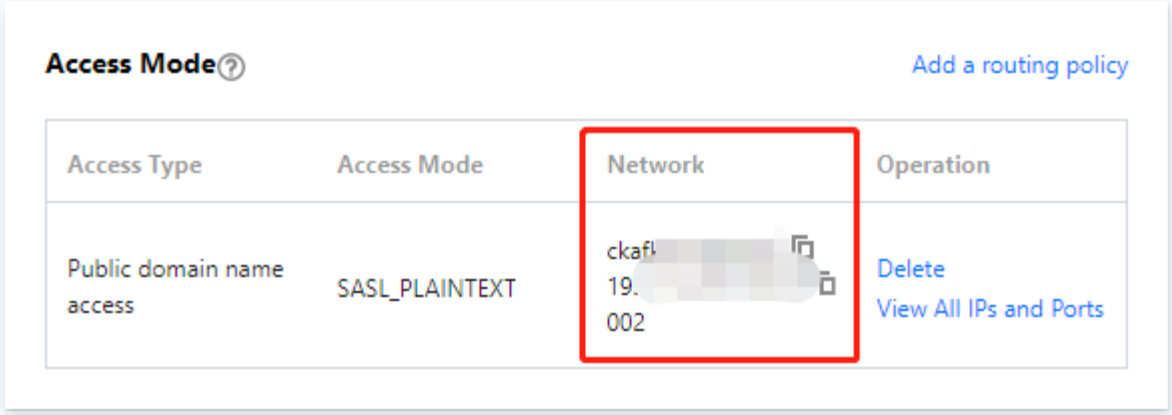
2.2 Open another terminal window to start the producer.

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

Note

- broker-list: Replace XXXX:port with the domain name and port for public access, which can be obtained from the **Access Method** module on the console instance details page.

©2013-2023 Tencent Cloud. All rights reserved.



- Topic: Replace XXXX with the topic name, which can be obtained from the 'Topic Management' page on the console.

Enter the content of the message, press Enter, and you can see that the consumer received the message almost at the same time.

- **Producing a 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
>
```

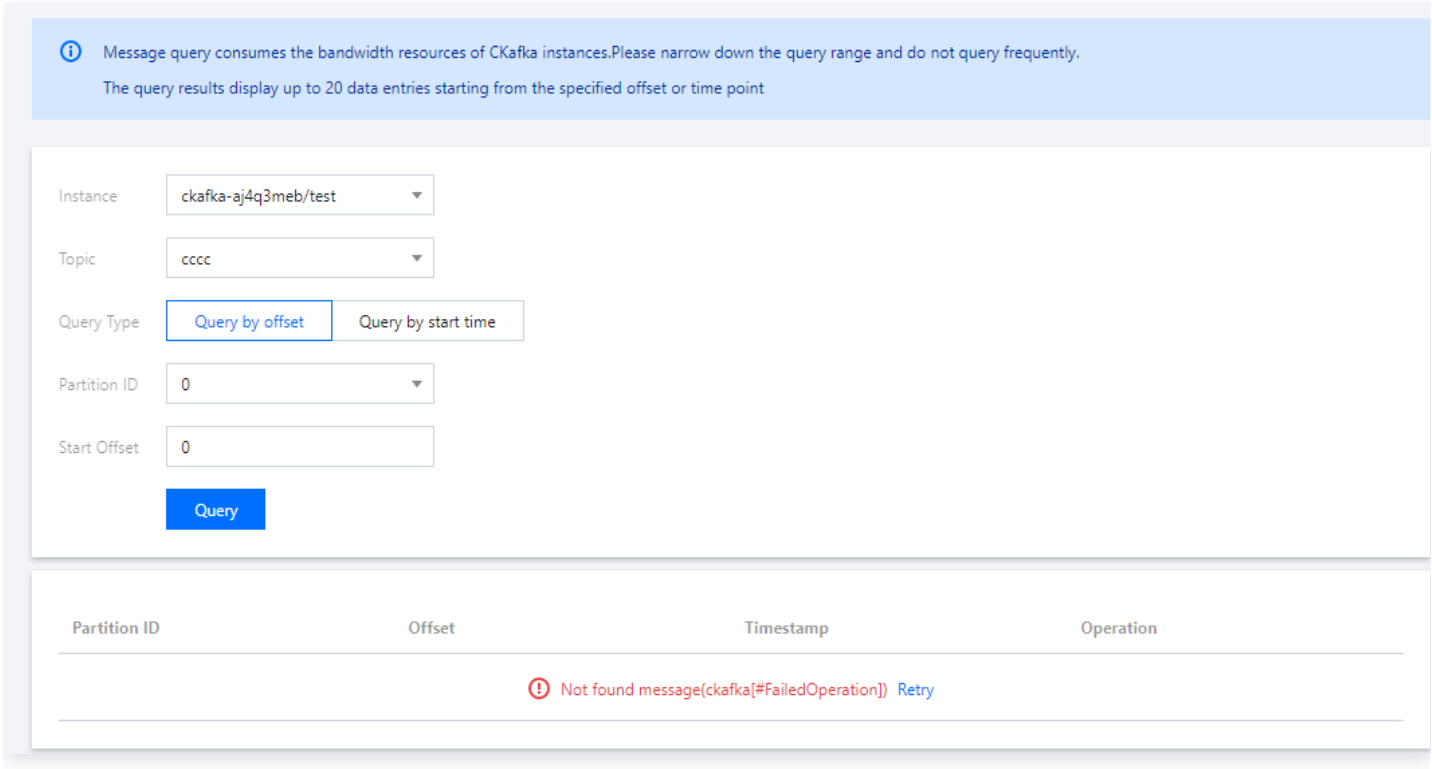
- **Consuming a message:**

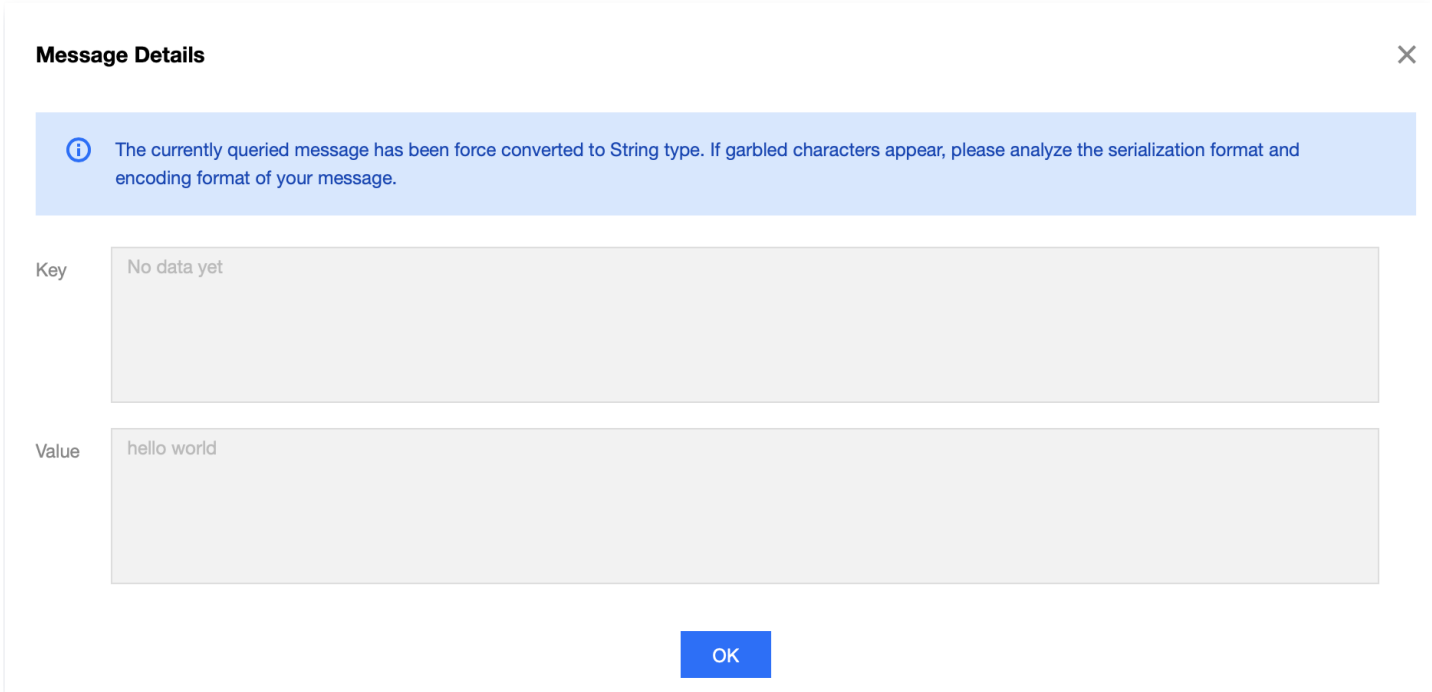
```
[ bin % bash kafka-console-consumer.sh --bootstrap-server c
kafka- .ap-guangzhou.ckafka.tencentcloudmq.com:6014 --topic test --consum
er.config ../config/consumer.properties
hello world
this is a message
this is another message
```

3. In the message querying page of the CKafka console, query the message sent.

The details of the message are as follows:

©2013–2023 Tencent Cloud. All rights reserved.





©2013–2023 Tencent Cloud. All rights reserved.