TDMQ for MQTT

Data Integration

Product Documentation

Contents

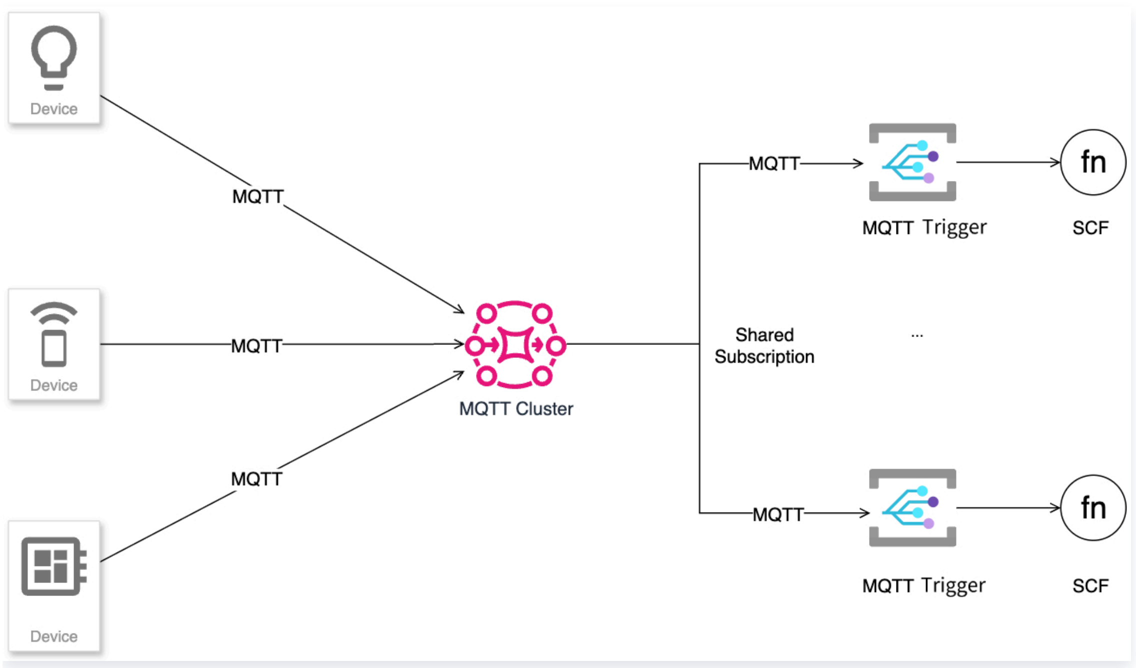
Data Integration

Integrating Data Into SCF

Last updated：2026-01-30 14:55:30

Implementation Principles

SCF supports MQTT Message Queue trigger. The trigger is based on the MQTT Shared Subscription mechanism, which can subscribe to and consume messages according to the Topic Filter defined in your business, thereby automatically triggering the corresponding function execution. Through this mechanism, SCF can establish efficient, asynchronous communication links with IoT devices. Messages are reliably routed and distributed via the MQTT cluster, and load balancing across multiple triggers is achieved through shared subscriptions, enabling high-concurrency IoT scenarios requiring real-time response, such as vehicle status monitoring and remote smart control.



Use Cases

New Energy Vehicle Battery SoC Management

The vehicle's TCU module triggers SCF via MQTT message trigger and calls different services subsequently based on the vehicle's status.

- Low power warning and response: When the vehicle TCU detects the battery level is below the threshold, it publishes a status message via MQTT topic, triggering SCF to perform operations such as notifying users and navigating to a charging station.
- Remote device control: SCF sends instructions to the vehicle via MQTT (such as limiting air conditioning power) to implement energy-saving control.

- Real-time status synchronization: Subscribe to MQTT topic via HTML/Mini Program Page to monitor battery SoC changes in real time.

Features and Advantages

- **Event-driven and real-time response**: SCF is automatically triggered by MQTT messages with no need for polling. Device status changes (e.g. low battery SoC) can be transmitted to the cloud as events immediately, triggering subsequent business chains.

  Achieves near real-time business processing, significantly reducing delay from event occurrence to system response, excellently meeting IoT application requirements for real-timeness.
- **Serverless architecture, ultimate elasticity and cost optimization**: SCF runs in serverless mode, automatically scaling based on MQTT message arrival frequency with no need to manage servers. Zero idle cost, auto-scaling copes easily with traffic spikes, simplifying operations.
- **Loose coupling integration method**: MQTT Topic serves as a message bus, completely decoupling device reporting from business processing. Different business features (such as notification, navigation, energy-saving control) are handled independently by different SCFs, offering high flexibility and maintainability.

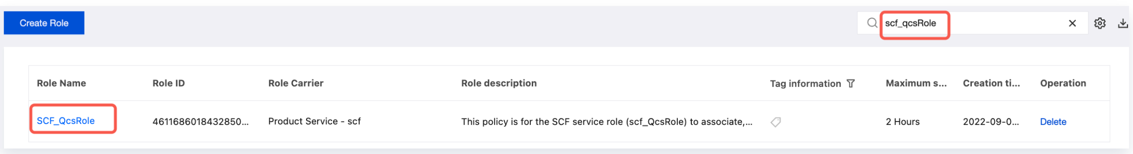
Operation Steps

Policy and Permissions

1. Check whether scf_QcsRloe has the following policies: QcloudMQTTReadOnlyAccess , QcloudAccessForSCFRoleInMQTT .

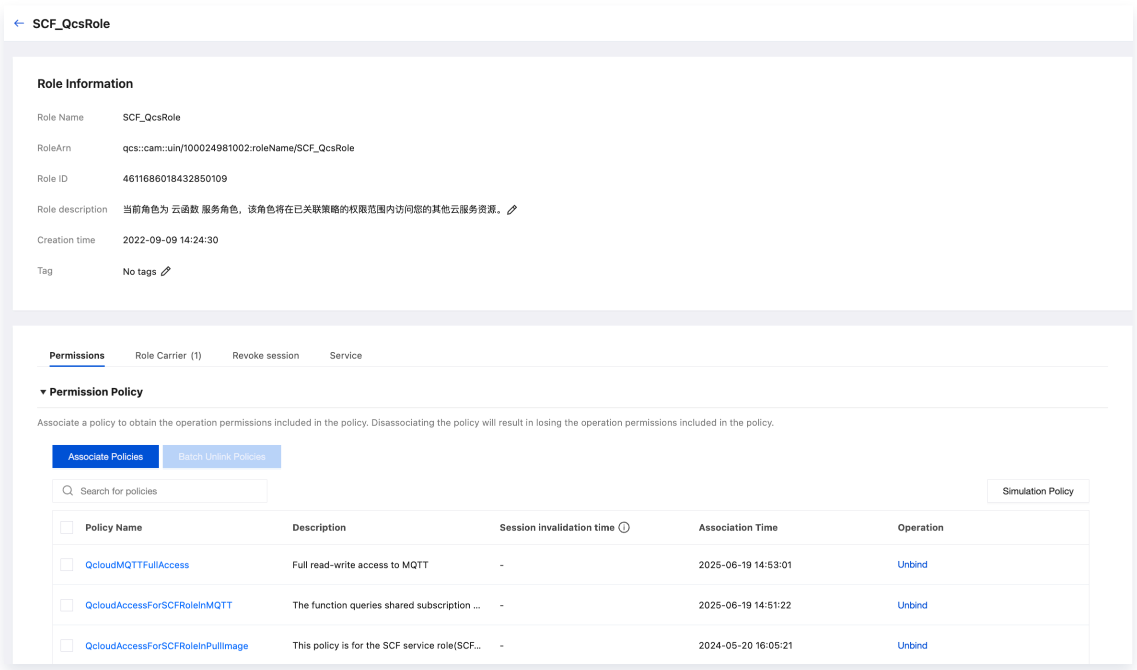
   1.1 Log in to the CAM console and navigate to the [Role] menu.

   1.2 Search scf_qcsRole



Role Name	Role ID	Role Carrier	Role description	Tag information	Maximum s...	Creation ti...	Operation
SCF_QcsRole	4611686018432850...	Product Service - scf	This policy is for the SCF service role (scf_QcsRole) to associate,...		2 Hours	2022-09-0...	Delete

   1.3 Click role details to view policy.


SCF_QcsRole

Role Information

Role Name	SCF_QcsRole
RoleArn	qcs::cam::uin/100024981002:roleName/SCF_QcsRole
Role ID	4611686018432850109
Role description	当前角色为 云函数 服务角色，该角色将在已关联策略的权限范围内访问您的其他云服务资源。
Creation time	2022-09-09 14:24:30
Tag	No tags

Permissions Role Carrier (1) Revoke session Service

Permission Policy

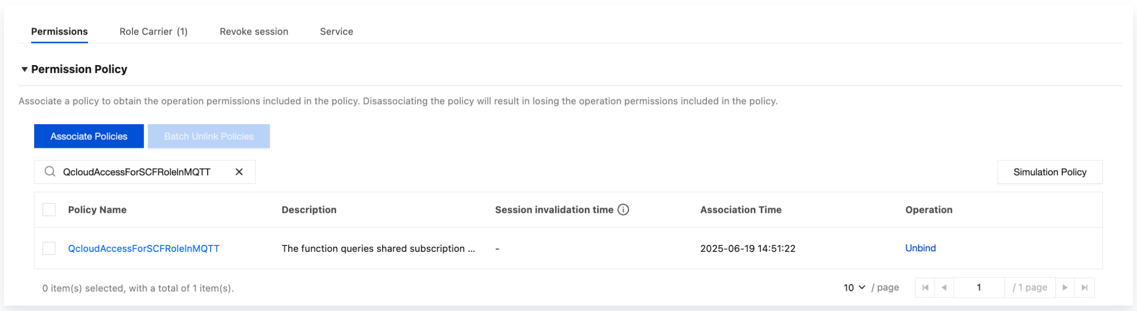
Associate a policy to obtain the operation permissions included in the policy. Disassociating the policy will result in losing the operation permissions included in the policy.

Associate Policies Batch Unlink Policies

Search for policies Simulation Policy

Policy Name	Description	Session invalidation time	Association Time	Operation
QcloudMQTTFullAccess	Full read-write access to MQTT	-	2025-06-19 14:53:01	Unbind
QcloudAccessForSCFRoleInMQTT	The function queries shared subscription ...	-	2025-06-19 14:51:22	Unbind
QcloudAccessForSCFRoleInPullImage	This policy is for the SCF service role(SCF...	-	2024-05-20 16:05:21	Unbind

1.4 Search QcloudMQTTReadOnlyAccess, QcloudAccessForSCFRoleInMQTT. If results are found, it signifies association succeeded.


Permissions Role Carrier (1) Revoke session Service

Permission Policy

Associate a policy to obtain the operation permissions included in the policy. Disassociating the policy will result in losing the operation permissions included in the policy.

Associate Policies Batch Unlink Policies

QcloudAccessForSCFRoleInMQTT Simulation Policy

Policy Name	Description	Session invalidation time	Association Time	Operation
QcloudAccessForSCFRoleInMQTT	The function queries shared subscription ...	-	2025-06-19 14:51:22	Unbind

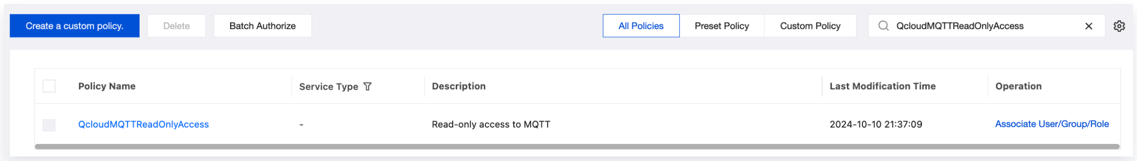
0 item(s) selected, with a total of 1 item(s). 10 / page 1 / 1 page

1.5 If the policy is missing, follow these steps to add a policy to the role:

1.5.1 Use the root account to log in to the Tencent Cloud console.

1.5.2 Click **Access Management** > **Policies**

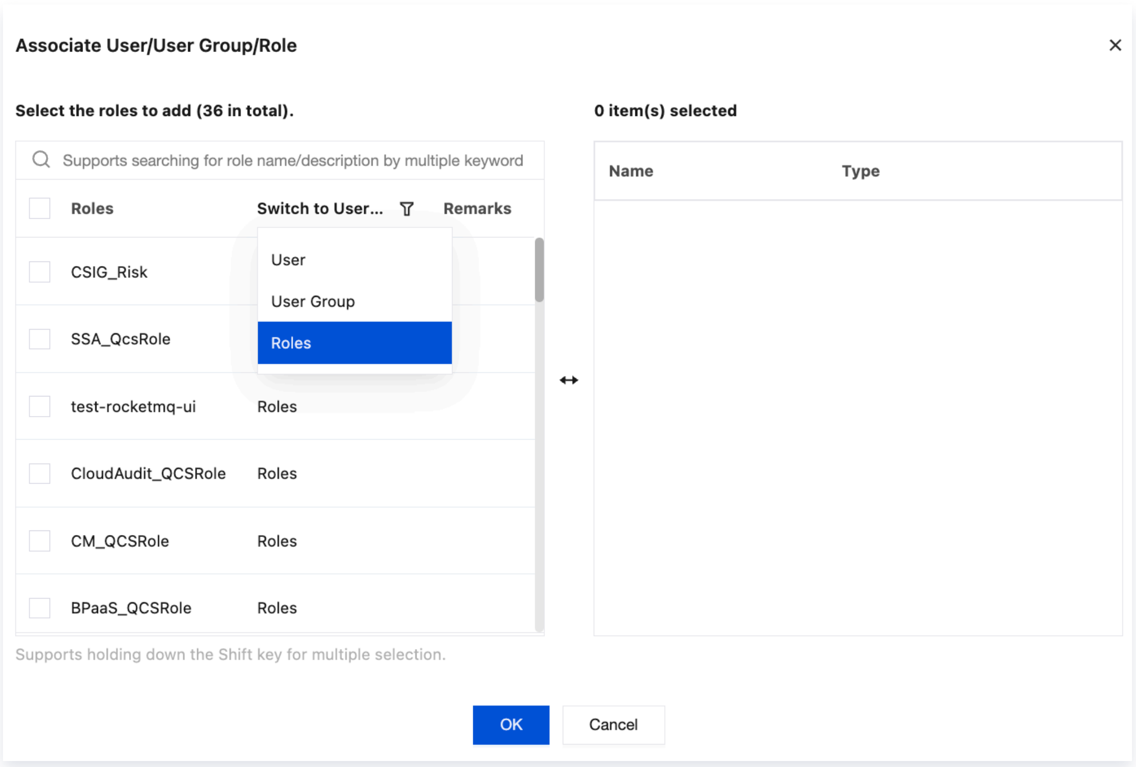
1.5.3 Search for missing policies, such as QcloudMQTTReadOnlyAccess


Create a custom policy. Delete Batch Authorize All Policies Preset Policy Custom Policy QcloudMQTTReadOnlyAccess

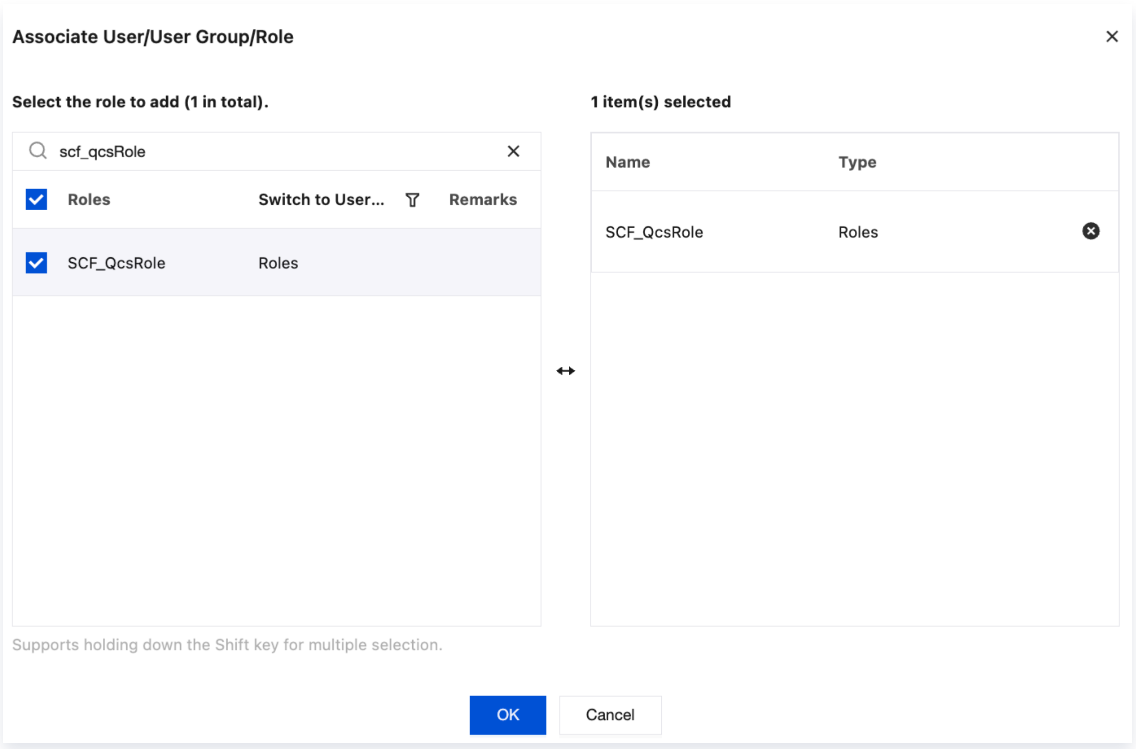
Policy Name	Service Type	Description	Last Modification Time	Operation
QcloudMQTTReadOnlyAccess	-	Read-only access to MQTT	2024-10-10 21:37:09	Associate User/Group/Role

1.5.4 Click **Associate User/Groups/Role.**

1.5.5 In the pop-up window, select **Switch to Roles.**


Associate User/User Group/Role
Select the roles to add (36 in total).
0 item(s) selected
Supports searching for role name/description by multiple keyword
Roles
Switch to User...
Remarks
Name
Type
User
User Group
Roles
CSIG_Risk
SSA_QcsRole
test-rocketmq-ui
CloudAudit_QCSRole
CM_QCSRole
BPaaS_QCSRole
Supports holding down the Shift key for multiple selection.
OK
Cancel

1.5.6 Search scf_qcsRole, check and confirm.

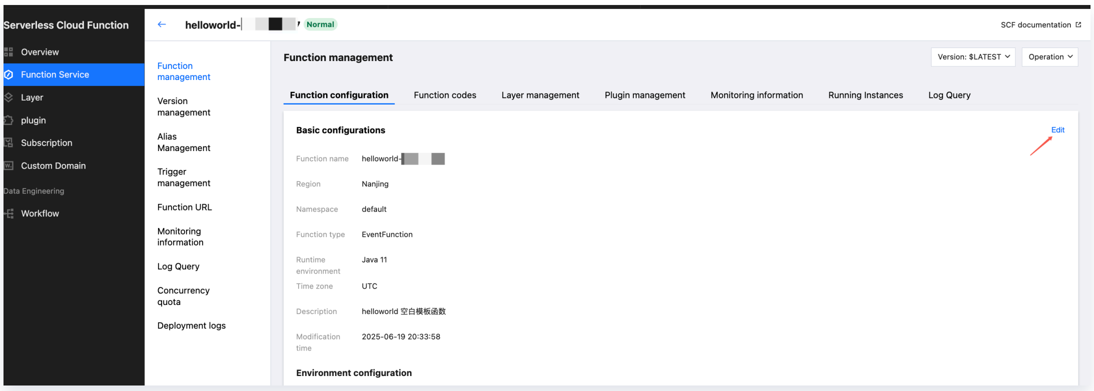

Associate User/User Group/Role
Select the role to add (1 in total).
1 item(s) selected
scf_qcsRole
Roles
Switch to User...
Remarks
Name
Type
SCF_QcsRole
Roles
SCF_QcsRole
Roles
Supports holding down the Shift key for multiple selection.
OK
Cancel

1.5.7 Page prompt: Associated successfully.

Configuring an MQTT Trigger

1. Log in to the Serverless console and navigate to the Function Service page.

2. Click the function name to enter the function management page.
3. Select the Trigger management TAB, click **Create trigger** in the upper right corner.

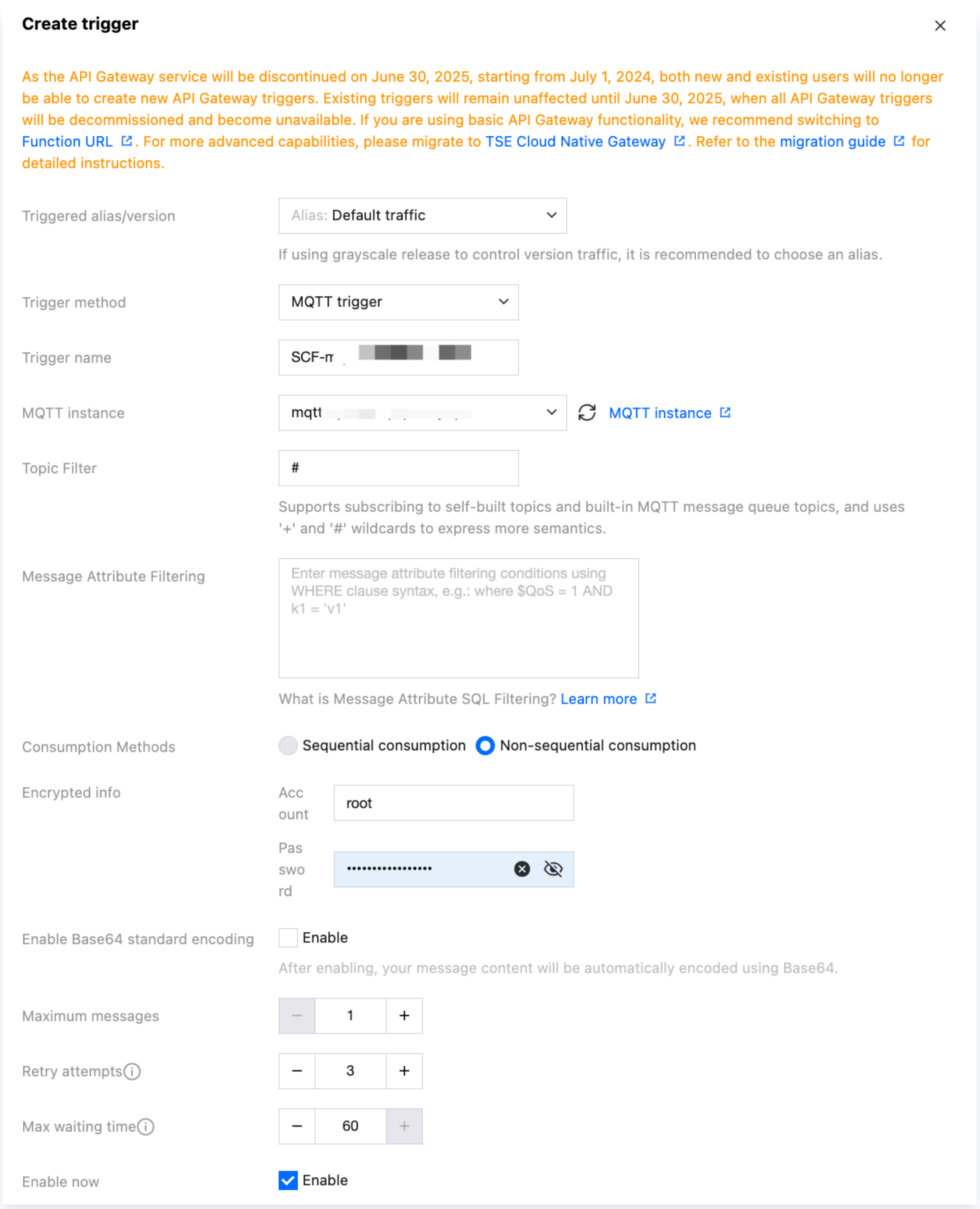


Basic configurations

Function name	helloworld-[redacted]
Region	Nanjing
Namespace	default
Function type	EventFunction
Runtime environment	Java 11
Time zone	UTC
Description	helloworld 空白模板函数
Modification time	2025-06-19 20:33:58

Environment configuration

4. According to business needs, fill in the required Topic Filter. Messages that meet the Topic Filter trigger execution of SCF.



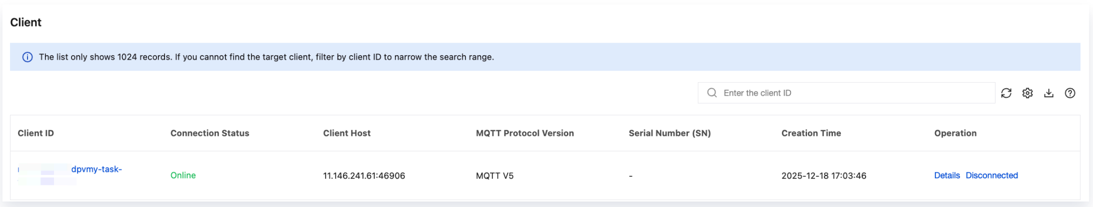
Create trigger

As the API Gateway service will be discontinued on June 30, 2025, starting from July 1, 2024, both new and existing users will no longer be able to create new API Gateway triggers. Existing triggers will remain unaffected until June 30, 2025, when all API Gateway triggers will be decommissioned and become unavailable. If you are using basic API Gateway functionality, we recommend switching to Function URL. For more advanced capabilities, please migrate to TSE Cloud Native Gateway. Refer to the migration guide for detailed instructions.

Triggered alias/version	Alias: Default traffic If using grayscale release to control version traffic, it is recommended to choose an alias.
Trigger method	MQTT trigger
Trigger name	SCF-m [redacted]
MQTT instance	mqtt [redacted] — MQTT instance
Topic Filter	# Supports subscribing to self-built topics and built-in MQTT message queue topics, and uses '+' and '#' wildcards to express more semantics.
Message Attribute Filtering	Enter message attribute filtering conditions using WHERE clause syntax, e.g.: where $QoS = 1 AND k1 = 'v1' What is Message Attribute SQL Filtering? Learn more
Consumption Methods	Sequential consumption [✓ Non-sequential consumption]
Encrypted info	Account: root Password: •••••••••••••••••
Enable Base64 standard encoding	☐ Enable After enabling, your message content will be automatically encoded using Base64.
Maximum messages	1
Retry attempts	3
Max waiting time	60
Enable now	☑ Enable

5. After SCF is created, observe MQTT console > Client management.

   You can see the trigger subscriber list.



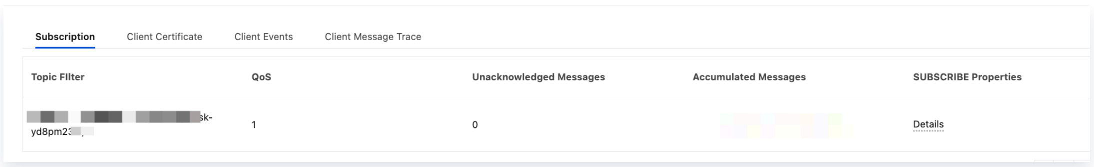
Client

The list only shows 1024 records. If you cannot find the target client, filter by client ID to narrow the search range.

Enter the client ID

Client ID	Connection Status	Client Host	MQTT Protocol Version	Serial Number (SN)	Creation Time	Operation
[redacted] dpvmy-task-	Online	11.146.241.61:46906	MQTT V5	-	2025-12-18 17:03:46	Details Disconnected

   View client details to see the subscription expression is configured the same.

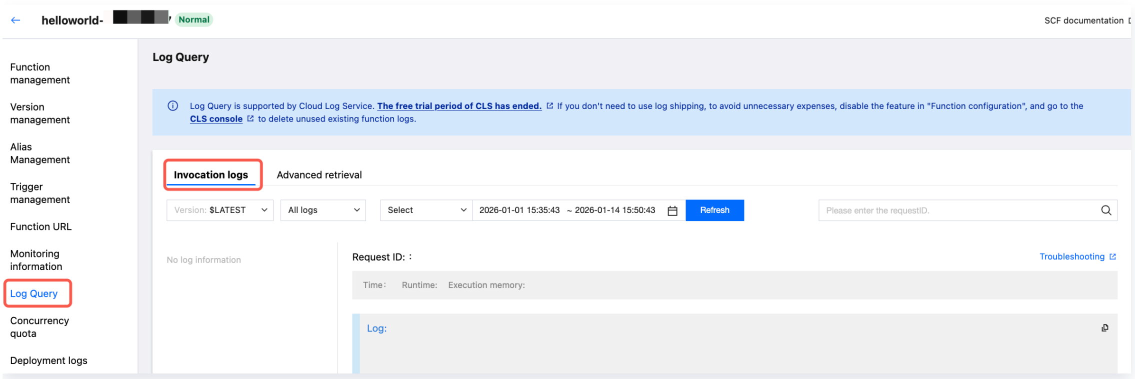


Subscription | Client Certificate | Client Events | Client Message Trace

Topic Filter	QoS	Unacknowledged Messages	Accumulated Messages	SUBSCRIBE Properties
[redacted]sk-yd8pm2[redacted]	1	0	[redacted]	Details

Verification

Send messages to the MQTT cluster and view the SCF function log.



helloworld-[redacted] Normal

SCF documentation

Log Query

Log Query is supported by Cloud Log Service. The free trial period of CLS has ended. If you don't need to use log shipping, to avoid unnecessary expenses, disable the feature in "Function configuration", and go to the CLS console to delete unused existing function logs.

Invocation logs | Advanced retrieval

Version: $LATEST | All logs | Select | 2026-01-01 15:35:43 ~ 2026-01-14 15:50:43 | Refresh | Please enter the requestID.

No log information

Request ID: :

Troubleshooting

Time: Runtime: Execution memory:

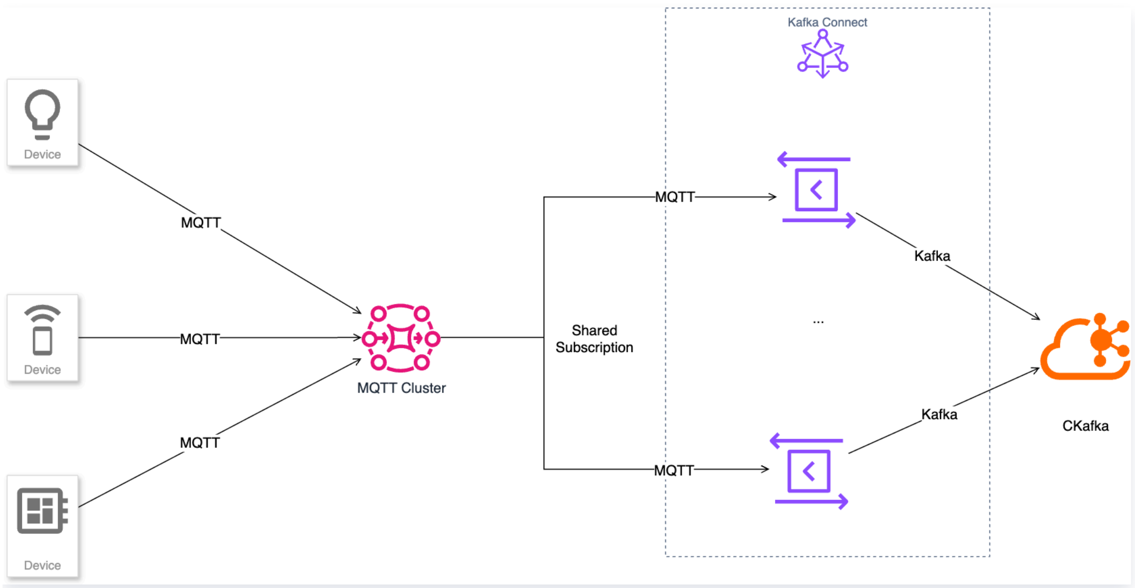
Log:

Integrating Data Into CKafka

Last updated：2026-01-30 14:55:30

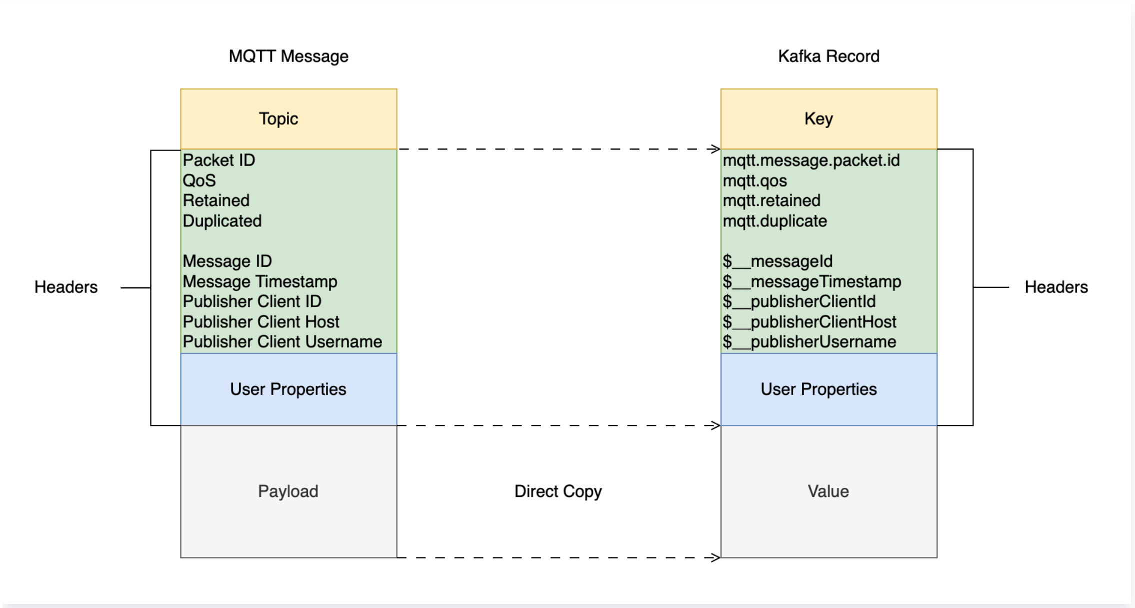
Implementation Principles

The CKafka connector has a built-in MQTT Source Plugin that leverages the MQTT shared subscription mechanism to ingest MQTT messages in real time and forward them to CKafka clusters. This shared subscription mode supports high-concurrency configuration, effectively ensuring data transmission throughput and fully meeting the requirements for high-traffic access and processing capabilities when Kafka is integrated with the big data ecosystem.



Data Mapping

When converting MQTT messages to Kafka Records, the mapping relationship is as follows:



MQTT Message

An MQTT message consists of three parts: system fields, user attributes, and Payload. See MQTT Control Packet format.

System Fields

Field Name	Semantics
Packet ID	Control command ID, not unique, for quick reuse see Spec 2.2.1
Duplicated	Refer to Spec 3.3.1.1
QoS	Refer to Spec 3.3.1.2
Retained	Refer to Spec 3.3.1.3
Message ID	Extended fields Unique message number
Message Timestamp	Extended fields, server-side message storage time
Publisher Client ID	Extended fields, client identifier for publishing messages
Publisher Client Host	Extended fields, client IP for publishing messages
Publisher Username	Extended fields, client username for publishing messages

User Properties

The list of key-value pairs specified by the user. Refer to Spec 3.3.2.3.7.

Kafka Record

Field	Semantics
Key	Key-value of the record, optional
Headers	Key-value pairs associated with the record, often used to store metadata such as Content Type, event time, optional
Payload	Actual data of the record, message body

Headers Usage Scenarios

- Message route
- Metadata storage description
- Tracing and logs

- Customized service handling
- authentication
- Message priority
- Interoperability/compatibility command
- Stream processing

Use Cases

Smart City and Transportation Digital Twin

- The system collects multi-source traffic data (such as vehicle license plates, speeds, and travel trajectories) from cities in real time, reports them via MQTT topics, and integrates them into the big data ecosystem through Kafka connectors.
- It supports efficient search and analysis based on attributes such as license plate numbers and so on (such as vehicle trajectory restoration), providing data support for traffic monitoring, dispatching, and simulation.

Features and Advantages

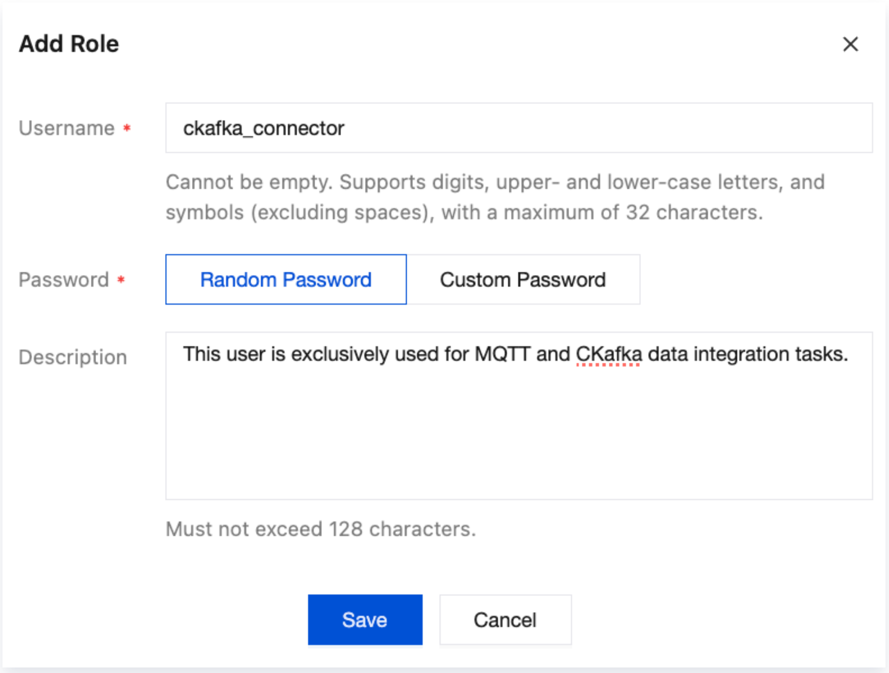
CKafka is a distributed, high-throughput, and highly scalable messaging system. However, it is not inherently designed for edge IoT communication scenarios, as its clients typically require stable network environments and substantial hardware resources. In contrast, the massive volumes of data generated by devices and applications in the IoT domain are often transmitted via the lightweight MQTT protocol. The CKafka MQTT Connector enables seamless integration between the MQTT protocol and the CKafka ecosystem, allowing MQTT messages published by devices to be streamed in real time into CKafka topics. This ensures data can be processed, stored, and analyzed immediately. The integration preserves MQTT's advantages in unstable network and low-resource environments while fully utilizing CKafka's capabilities in high throughput, reliability, and ecosystem compatibility. This achieves flexible, stable, and efficient integration between IoT data and big data systems.

Operation Steps

Policy and Permissions

1. Log in to the TDMQ for MQTT console, go to the **Cluster Details** page, and confirm whether the authorization policy management is enabled for the current MQTT cluster.
   1.1 If the permission policy is not enabled, the data plane resources have no permission management. You can use any "username + password" to connect, produce, and consume. For details, see Configure Data Plane Authorization. In this case, no additional configuration is required when data is integrated into CKafka. However, due to the lack of permission control, there are certain data security risks.
   1.2 If the permission policy is already enabled, follow the steps described below.

2. Go to Authentication > Username and Password, click **Create User** to create a dedicated account and password for the Data Integration task, with the username `ckafka_connector`. Specify in the description that this user is exclusively used for MQTT and CKafka data integration tasks, as shown in the figure.



Add Role

Username *: ckafka_connector

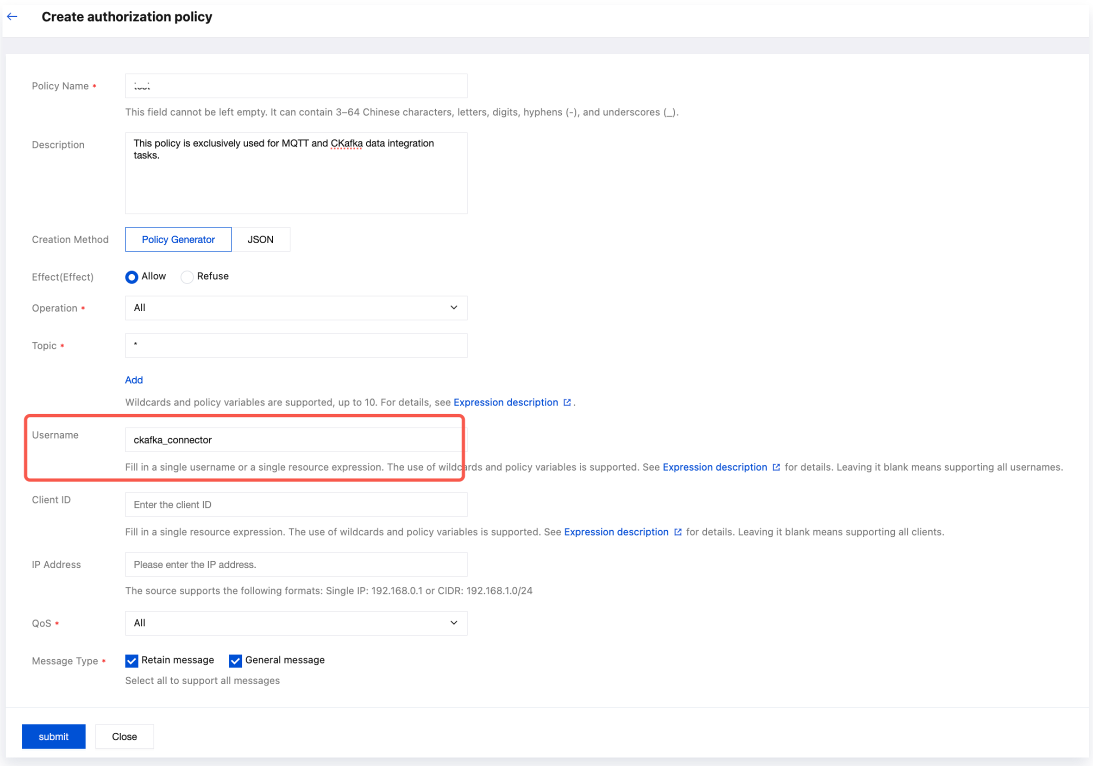
Cannot be empty. Supports digits, upper- and lower-case letters, and symbols (excluding spaces), with a maximum of 32 characters.

Password *: [✓ Random Password] Custom Password

Description: This user is exclusively used for MQTT and CKafka data integration tasks.

Must not exceed 128 characters.

3. Go to the **Authorization Policy** page, click **Create Policy**. It is strongly recommended to explicitly authorize the CKafka Data Integration account created in the previous step within this policy to achieve granular permission control. For specific configuration methods, refer to the figure below. Fill in other fields according to actual requirements. For details, refer to Configuring Data Plane Authorization.



Create authorization policy

Policy Name *: [illegible]

This field cannot be left empty. It can contain 3–64 Chinese characters, letters, digits, hyphens (-), and underscores (_).

Description: This policy is exclusively used for MQTT and CKafka data integration tasks.

Creation Method: [✓ Policy Generator] JSON

Effect(Effect): [✓ Allow] Refuse

Operation *: All

Topic *: *

Add

Wildcards and policy variables are supported, up to 10. For details, see Expression description.

Username: ckafka_connector

Fill in a single username or a single resource expression. The use of wildcards and policy variables is supported. See Expression description for details. Leaving it blank means supporting all usernames.

Client ID: Enter the client ID

Fill in a single resource expression. The use of wildcards and policy variables is supported. See Expression description for details. Leaving it blank means supporting all clients.

IP Address: Please enter the IP address.

The source supports the following formats: Single IP: 192.168.0.1 or CIDR: 192.168.1.0/24

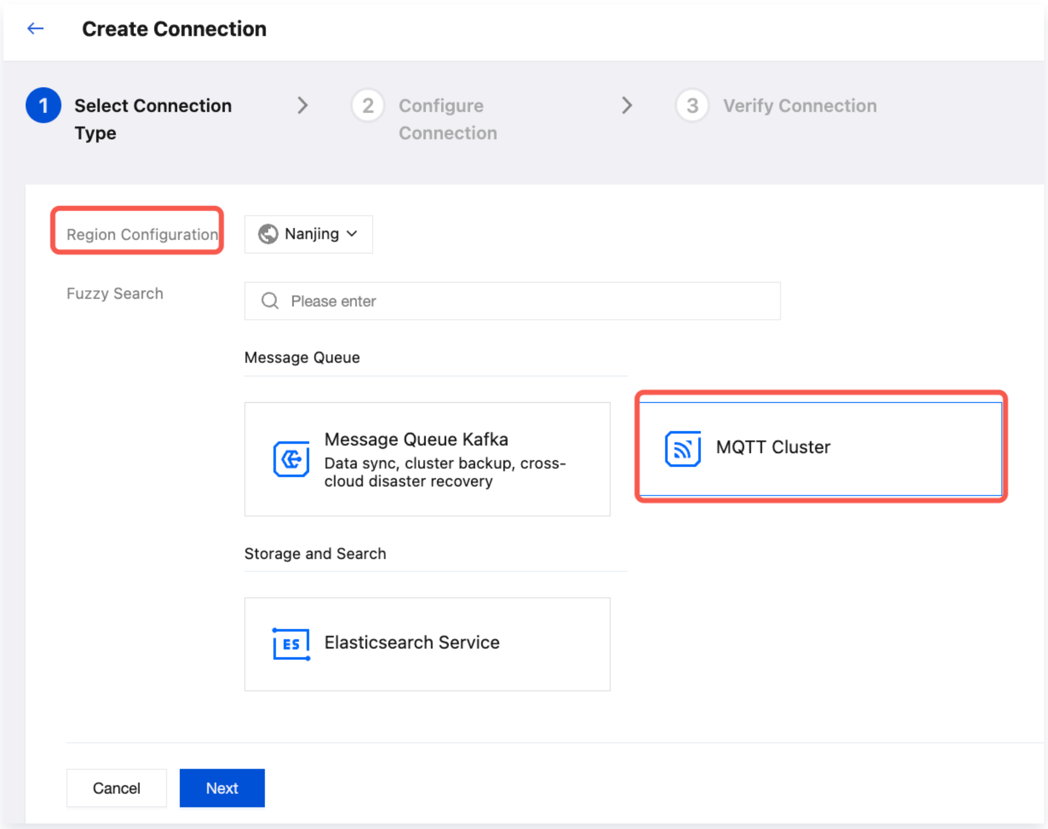
QoS *: All

Message Type *: [✓ Retain message] [✓ General message]

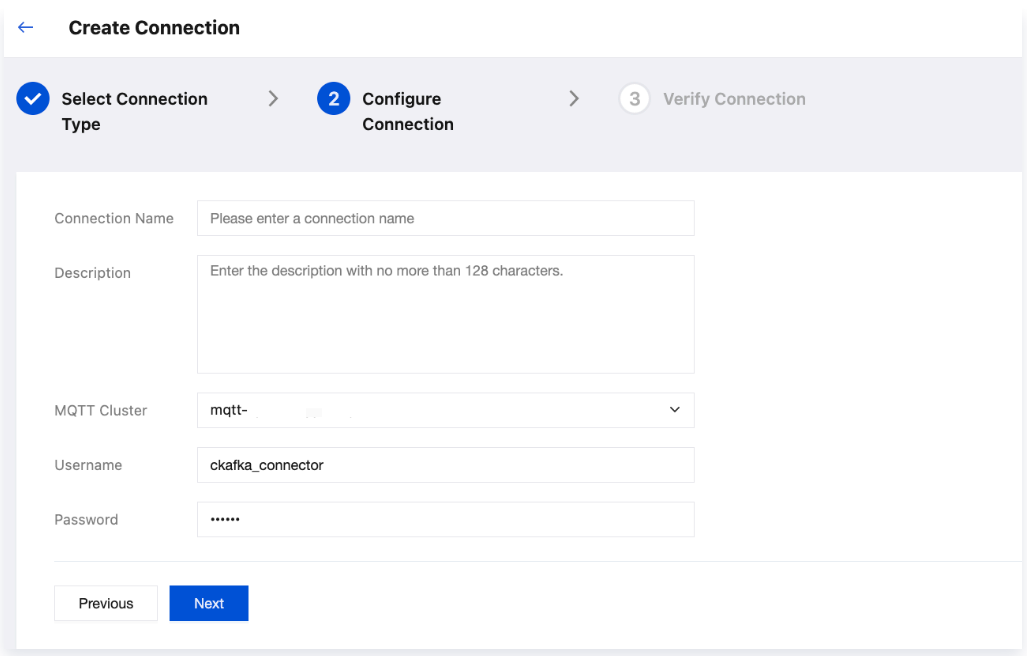
Select all to support all messages

Configure the CKafka Connector

1. Log in to the CKafka console, go to the **Connection List** page, and first confirm the **Region** to which the connection belongs at the top of the page.
2. Click **Create Connection** to create the connector.
3. Follow the steps below to select connection information. Select **MQTT Cluster** as the connection type, then click **Next** to go to the connection configuration page.



4. Enter the connection name, description, and other basic information, and select the target MQTT cluster from the dropdown. The username and password here are used for connection authentication, which are the dedicated Data Integration account credentials created in the MQTT cluster. For details, refer to the Policy and Permissions section. Click **Next** to enter the connection validation process.



Create Connection

Select Connection Type > 2 Configure Connection > 3 Verify Connection

Connection Name: Please enter a connection name

Description: Enter the description with no more than 128 characters.

MQTT Cluster: mqtt-

Username: ckafka_connector

Password: ••••••

5. After all validations pass, the connection is successfully created. You can view the newly added connection in **CKafka Console** > **Connector** > **Connection List**.
    - For created connections, the connections list displays their basic information, including ID, Name, Status, Connection Type, Bound Resources, Resource Region, Number of Associated Tasks, Creation Time, Description, and so on.
    - Click the **Edit** button in the Actions column to modify connection configurations. After the connection is updated, the system enables the "Update and restart all associated tasks" option by default. Exercise caution when performing this operation based on actual business requirements.
    - Click the **Delete** button in the Actions column to delete this connection.

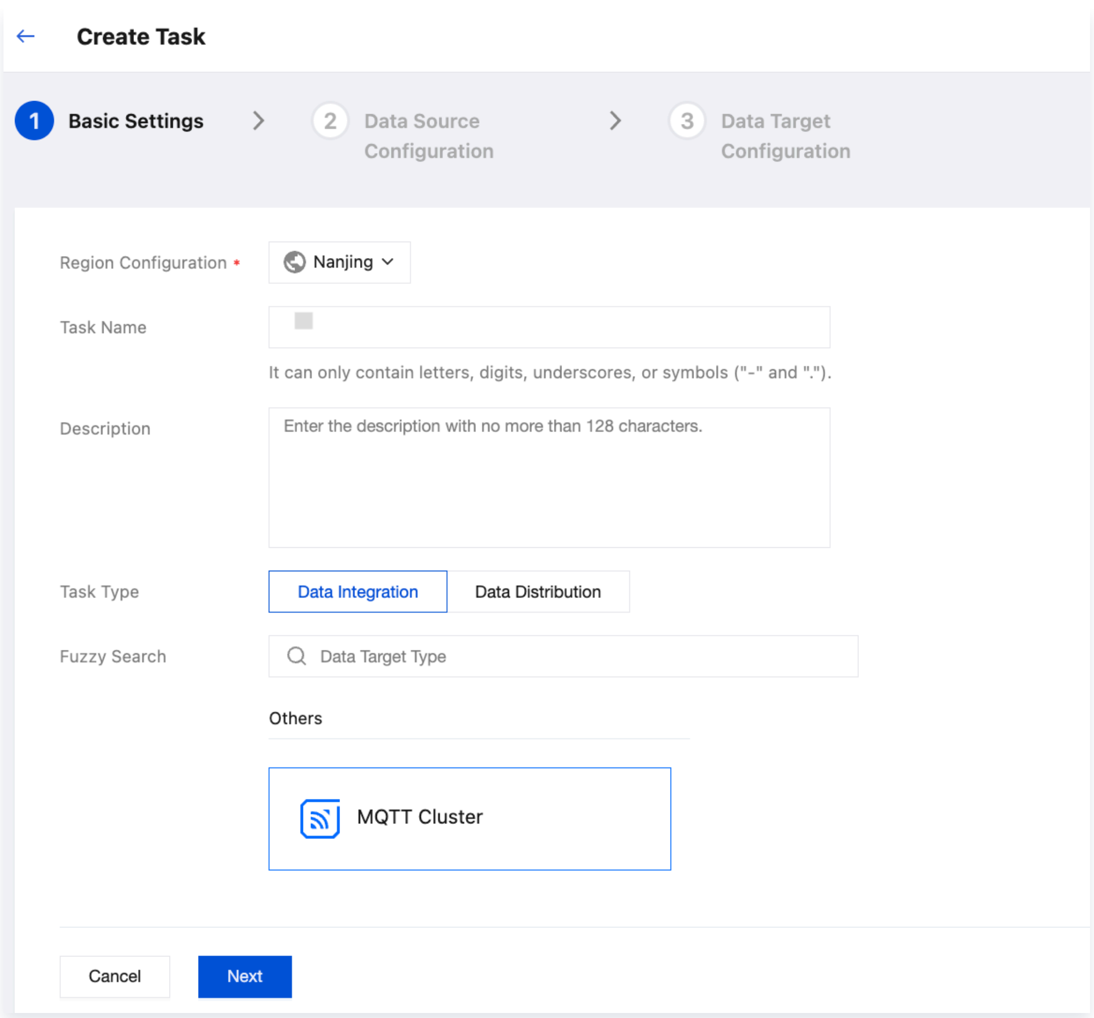
Create Data Integration Task

Prerequisites

In the same region as the MQTT cluster, a CKafka instance has been created. For details, refer to CKafka Quick Start.

Task Creation

1. Go to CKafka Console > **Connectors** > **Task List**, click Create Task in the upper-left corner to fill in the task-related information. Choose **Data Integration** > **MQTT Cluster** for the task type, then click **Next** to go to the data source configuration.



Create Task

1 Basic Settings > 2 Data Source Configuration > 3 Data Target Configuration

Region Configuration * Nanjing

Task Name [redacted]

It can only contain letters, digits, underscores, or symbols ("-" and ".").

Description: Enter the description with no more than 128 characters.

Task Type: [✓ Data Integration] Data Distribution

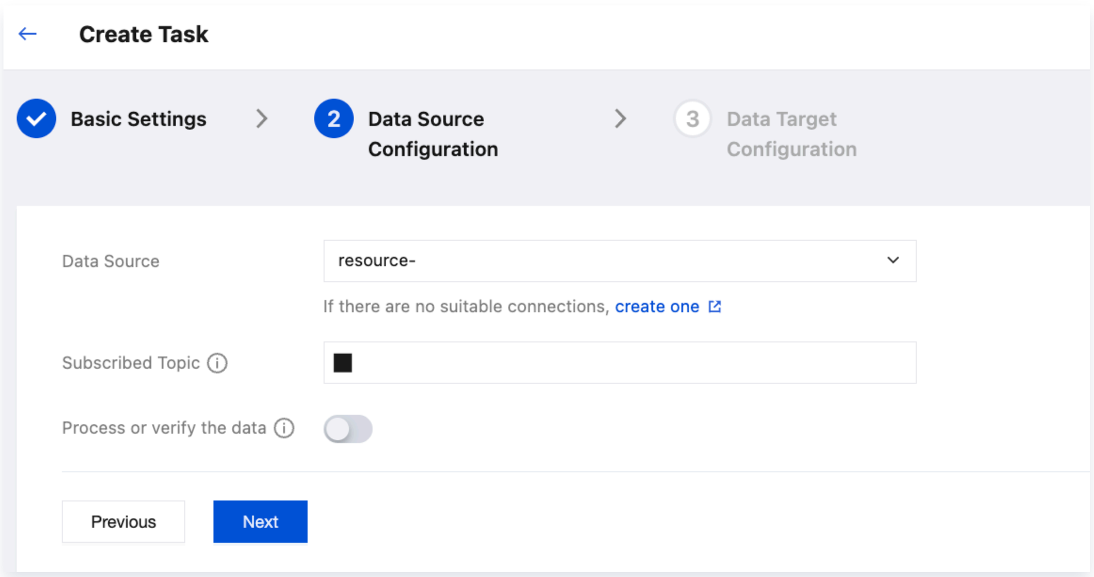
Fuzzy Search: Data Target Type

Others

MQTT Cluster

Cancel Next

2. Select an appropriate connection from the dropdown. If no suitable option is available, click the button below to go to the **New Connection** step. Enter the subscribed Topic. To subscribe to multiple topics, separate them with ",".



Create Task

✓ Basic Settings > 2 Data Source Configuration > 3 Data Target Configuration

Data Source: resource-

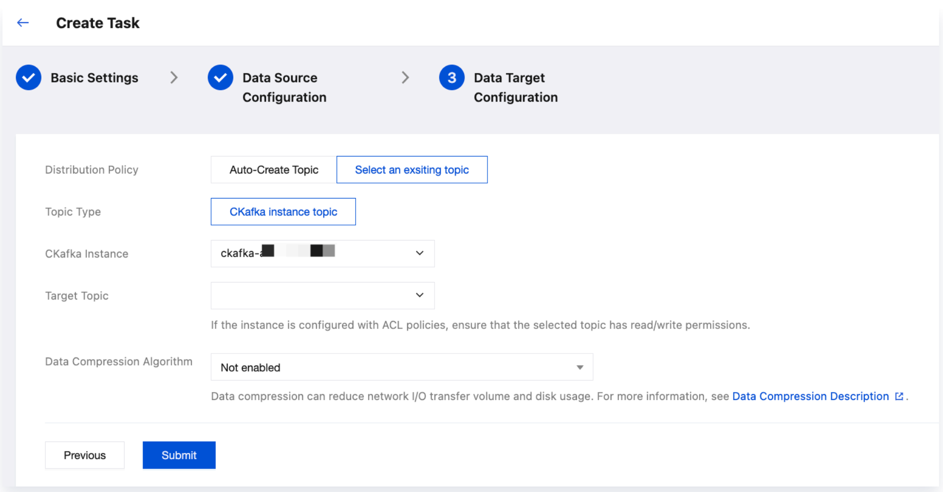
If there are no suitable connections, create one

Subscribed Topic ⓘ [redacted]

Process or verify the data ⓘ

Previous Next

3. Configure the data target, specify the **distribution policy** and **target CKafka instance**. Click **Submit** to complete task creation.



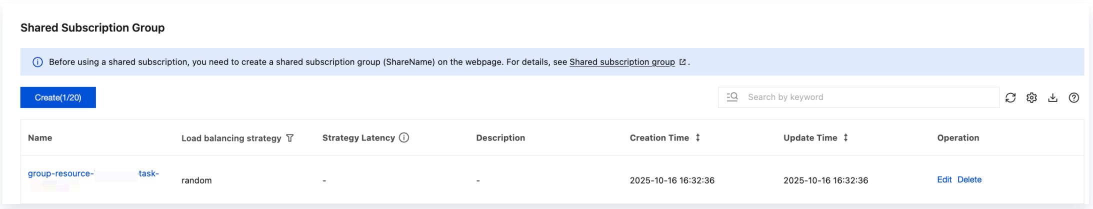
Create Task

Basic Settings > Data Source Configuration > 3 Data Target Configuration

Distribution Policy	Auto-Create Topic / Select an exsiting topic
Topic Type	CKafka instance topic
CKafka Instance	ckafka-[redacted]
Target Topic	
	If the instance is configured with ACL policies, ensure that the selected topic has read/write permissions.
Data Compression Algorithm	Not enabled
	Data compression can reduce network I/O transfer volume and disk usage. For more information, see Data Compression Description.

Previous Submit

4. When the task is successfully created, a shared subscription group will be automatically created under the MQTT cluster for performing Data Integration.



Shared Subscription Group

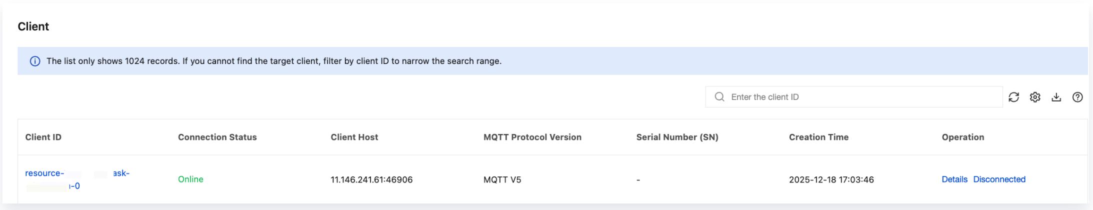
Before using a shared subscription, you need to create a shared subscription group (ShareName) on the webpage. For details, see Shared subscription group.

Create(1/20)

Search by keyword

Name	Load balancing strategy	Strategy Latency	Description	Creation Time	Update Time	Operation
group-resource-[redacted]task-[redacted]	random	-	-	2025-10-16 16:32:36	2025-10-16 16:32:36	Edit Delete

You can also go to the **Client** page to view details of the connector client for this task.



Client

The list only shows 1024 records. If you cannot find the target client, filter by client ID to narrow the search range.

Enter the client ID

Client ID	Connection Status	Client Host	MQTT Protocol Version	Serial Number (SN)	Creation Time	Operation
resource-[redacted]ask-[redacted]-0	Online	11.146.241.61:46906	MQTT V5	-	2025-12-18 17:03:46	Details Disconnected

Integrating Data into RocketMQ

Last updated：2026-01-30 14:55:30

Note: The feature is currently in gradual rollout. If you need to use it, please contact us by submitting a ticket.

Implementation Principles

TDMQ for MQTT supports message routing and forwarding between Message Queue MQTT and TDMQ for RocketMQ: MQTT messages are forwarded to the corresponding topics in RocketMQ via the cross-cluster replication capability provided by RocketMQ, according to pre-configured routing rules; similarly, messages from RocketMQ can also be delivered to MQTT and then further delivered to device endpoints.
RocketMQ, as a distributed message queue, is responsible for persisting and distributing messages for consumption and processing by multiple downstream business services, thus forming a closed-loop mechanism from IoT terminal state perception to cloud business response. The entire process achieves transparent conversion between the MQTT protocol and the RocketMQ protocol, as well as reliable message synchronization, ensuring that device status changes can reach the backend business system in real time and accurately.

Use Cases

Intelligent Ops and Status Management for EV Charging Stations

- Device Connection and Will Message Configuration: When an EV charging station device connects to the cloud server via the MQTT protocol, it pre-configures a will message in its connection request and designates the MQTT topic to which the message will be published.
- Abnormal Offline and Will Triggering: When an EV charging station abnormally disconnects due to sudden power outages, network interruptions, or device failures, the MQTT server will immediately publish the device's preset will message to the designated topic, indicating to the cloud that the device has gone offline abnormally.
- Cross-protocol Message Routing: Through the message routing rules pre-configured in the TDMQ console, the will messages on the specified topics of the MQTT server will be automatically and reliably forwarded to a specific topic in TDMQ RocketMQ.
- Ops Service Consumption and Status Awareness: The backend cloud maintenance service continuously subscribes to the RocketMQ topic. Once a new will message (that is, failure notification) arrives, the service immediately consumes and parses the message to accurately obtain the faulty device's identity ID, location information, and potential error codes. Upon detecting the device's abnormal status, it automatically triggers subsequent business processes.

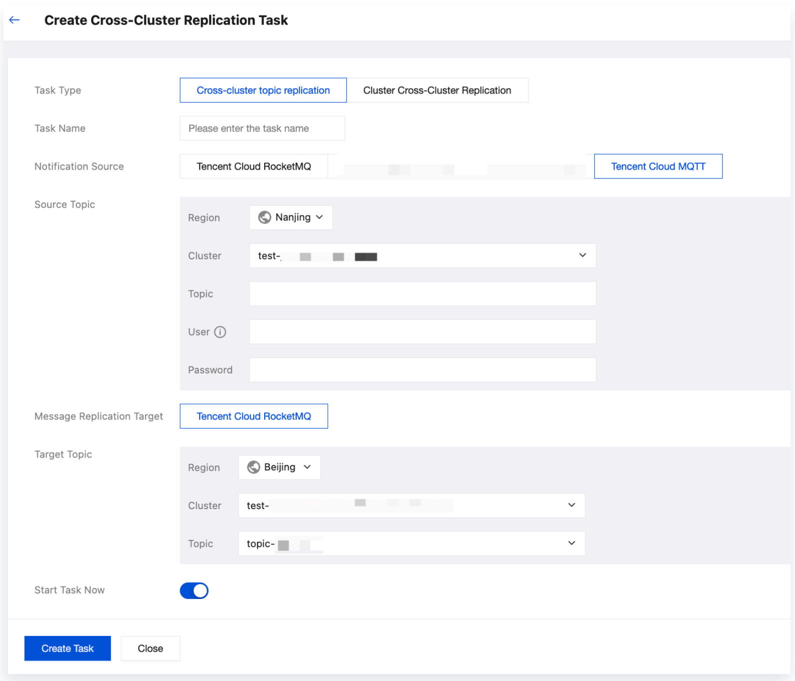
Features and Advantages

- High Reliability and Automation: Reliably batch and send MQTT messages to RocketMQ to achieve integration between IoT devices and RocketMQ and application systems.
- Flexible Topic Mapping: Data Integration supports flexibly mapping MQTT topics to RocketMQ topics.
- Business Scalability: As a message middleware, RocketMQ allows multiple business services to consume device status messages simultaneously, supporting smooth system scaling and flexible business iteration.

Operation Steps

Create Cross-Cluster Replication Task

1. Log in to the TDMQ for RocketMQ console and go to the **Cross-Cluster Replication** page.
2. Click **Create Task** to create a task.
3. Select the **Notification Source** and Message Replication Target, then fill in the fields as required:


Create Cross-Cluster Replication Task

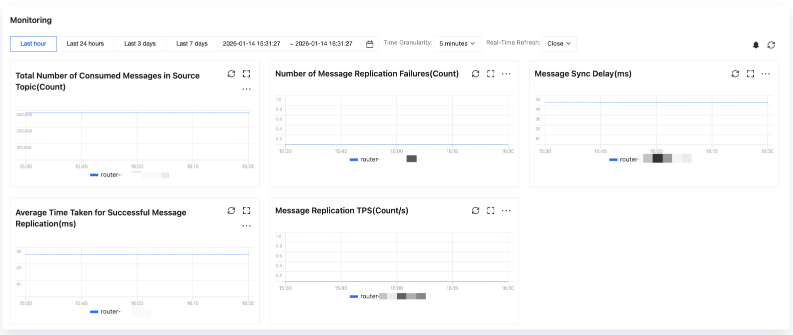
Task Type	Cross-cluster topic replication / Cluster Cross-Cluster Replication
Task Name	Please enter the task name
Notification Source	Tencent Cloud RocketMQ / Tencent Cloud MQTT
Source Topic	Region: Nanjing; Cluster: test-[redacted]; Topic; User; Password
Message Replication Target	Tencent Cloud RocketMQ
Target Topic	Region: Beijing; Cluster: test-; Topic: topic-
Start Task Now	

   - Task Name: contains no more than 200 characters. It can only contain English words, digits, letters, hyphens (-), and underscores (_).
   - Source Topic: Select the region, cluster, and Topic sequentially from the dropdown. If no suitable cluster or Topic is available, a new one can be created on the cluster list page.
   - Target Topic: Select the region, cluster, and Topic sequentially from the dropdown. If no suitable cluster or Topic is available, a new one can be created on the cluster list page.
   - Start Task Now: If the switch is enabled, the task will be replicated according to the configuration of the current task after it is created.

3.1 After you click **Create Task**, the Task List page will be displayed. After the task is initialized, it indicates that it has been created.

Viewing Task Details

1. After the task is created, you can view the newly added replication task on the task list page and quickly check its status. Click **Start/Pause** in the Operation column to start or pause the task with one click.
2. You can click the task name to go to the task details page and view the detailed configuration of the task. Running tasks cannot be modified. To adjust the replication task configuration, pause the task first, then click **Edit** in the Operation column. Alternatively, go to the task details page and click Edit in the upper-right corner of "Basic Information" to modify the task details.
3. In the monitoring section, you can view real-time metrics of the current message replication task, such as the total number of source messages consumed, the number of failed message replications, and message synchronization latency.



Monitoring

2026-01-14 15:31:27 ~ 2026-01-14 16:31:27

Time Granularity: 5 minutes

Real-Time Refresh: Close