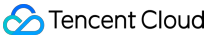
Internet of Things Hub Console Guide

Copyright Notice

Trademark Notice



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

Console Guide
Product Management
Device Connection Preparations

Last updated: 2025-03-19 14:46:15

Overview

Before connecting a device to IoT Hub, you need to create a virtual product and device in the IoT Hub console and match them with the real device. IoT Hub will assign each device a unique authentication identifier for connection. This document describes how to make preparations for connection to the platform.

Directions

Creating product

1. Log in to the Internet of Things Hub console and click **View details** on the upper right of the Overview module.
2. Click Product List on the left sidebar.
3. On the product list page, click **Create New Product** and fill in the following information as needed.
   - **Product Type**:
     - General Product: it communicates over 2G / 3G / 4G / Wi-Fi or wired connection. You can further develop its communication module for creation.
     - Gateway: it is used to proxy communication between subdevices and the backend.
   - **Product Name**: it can contain 1-40 letters, Chinese characters, digits, and underscores and can be modified.
   - **Authentication Method**:
     - Certificate: when you create a device, the platform will generate a certificate and a private key for authentication before the device can communicate with IoT Hub.
     - Key: when creating a device, you can use a custom PSK or the random PSK generated by the platform for the device.
   - **Data Format**:
     - JSON: you can match the rules and extract the content based on the data.
     - Custom: no data parsing is performed.
   - **Product Description**: it can contain up to 500 characters.

4. Click **OK** to generate the product (the product is a collection of a certain type of devices, and users manage all devices under the product).
5. After the product is created, you can enable the product's dynamic registration feature to generate a product key (ProductSecret).

Note:

> All devices under the same product can be programmed with unified product information during production, namely Product ID (ProductId) and Product Secret (ProductSecret). After leaving the factory, devices obtain and save their identity information through dynamic registration, and then use the obtained triplet or quadruplet information for device authentication. If dynamic registration is enabled and automatic device creation is selected, device names can be generated automatically, but it must be ensured that device names under the same Product ID (ProductId) are not duplicated, typically using IMEI or MAC addresses. For details on the dynamic registration feature, please refer to Device Access.

Deleting product

1. After completing the creation of a new product, you can view the product's basic information on the **Product Settings** page.
2. When you no longer need the product, you can delete it on the **Product List** page by clicking the **Delete** on the right side.

Create Device

After creating a product, you can add one device or batch add devices under it:

Single Addition

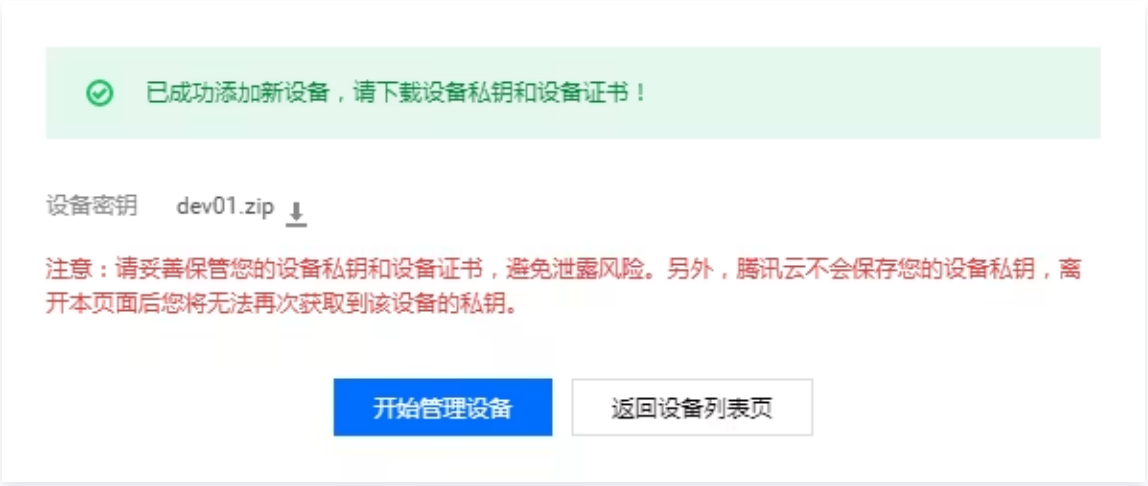
1. Click the **Device List** Tag of the corresponding product.
2. Click **Add New Device**, and fill in the relevant device information.

- Device Name: Supports a combination of letters, numbers, underscores, and "–", with a maximum of 48 characters.

- Device remarks: Supports a combination of Chinese, English, numbers, underscores, and "-", up to a maximum of 16 characters.

> **Note:**
> - The device name cannot be modified once confirmed.
> - The device name must be unique under the same product.
> - Key options will appear only when the product is authenticated by a key; custom keys must be Base64 encoded strings. You can input regular strings in the input box and click the convert to Base64 button to transform them into Base64 encoded strings.
> - Specific parameters that need to be entered during device creation vary by product type.

If the selected authentication method is certificate authentication, after the device is created, the device private key will be the unique identifier used by it to connect to the Internet of Things Hub backend, which does not store the device private key. Therefore, please keep it secure and confidential.



Batch adding

1. In the **Device List** tag, click **Batch Add**.
2. The batch addition method is divided into **Automatic Generation** and **Bulk Upload**.
   - Auto-generate: Set the desired number of devices, and the console will generate the same number of devices. In this case, a device name is a combination of 18 uppercase

and lowercase letters and digits.

- Upload: upload a CSV file of device names, and IoT Hub will create devices according to the device names in the CSV file.

> **Note:**
> The key option only appears when the product uses key authentication.

3. After successful addition, you can check the execution status on the **Add Results** page, download the device certificate/key file, or view and download the relevant information and files in **Batch Management**.

> **Note:**
> - If the selected product authentication method is key authentication, the files provided for download on the **Add Results** and **Batch Management** pages are CSV files, which include device name, device key, error code, and error information.

- If the selected product authentication method is certificate authentication, the files provided for download on the **Add Results** and **Batch Management** pages are ZIP archives, which include the same number of folders as there are devices and a CSV file. The folder names correspond to the device names, and the contents are certificate and private key files. The CSV file contains the device name, error code, error information, and the relative path to the device certificate and private key.

Device details

Certificate-authenticated device

Device details include all the content of the device: device name, device remarks, connection status, version information, device certificate (click to download), device private key (click to download), device log configuration, etc.

Key-authenticated device

Device details include all the content of the device: device name, device remarks, connection status, version information, device key (click to view), device log configuration, etc.

Gateway product connection

Last updated: 2025-03-19 14:46:30

This document describes how to manage subdevices.

Prerequisites

You have created Gateway Product and Equipment.

Operations

Binds subdevice

> **Note:**
> A subdevice is a device that can connect to IoT Hub only through a gateway device.

1. Log in to the Internet of Things Hub console and click **View details** on the upper right of the Overview module.
2. You will be taken to the **Product List** page by default.
3. Click the target **Product Name** to enter the product details page, then click **Device List** > **Sub-devices** to enter the sub-device page.

4. Click **Add Sub-device** > **Select Product** and check the sub-devices to be bound under the product, then click **Add** to complete the binding.

Unbinds subdevice

On the sub-device page, select the corresponding product and check the bound sub-devices of that product, then click **Bulk Unbinding** to unbind them.

Device Shadow

Last updated：2025-03-19 14:46:43

Initial State

After creating the product and device, the device is in an inactive state, and the device shadow is empty by default during initialization.

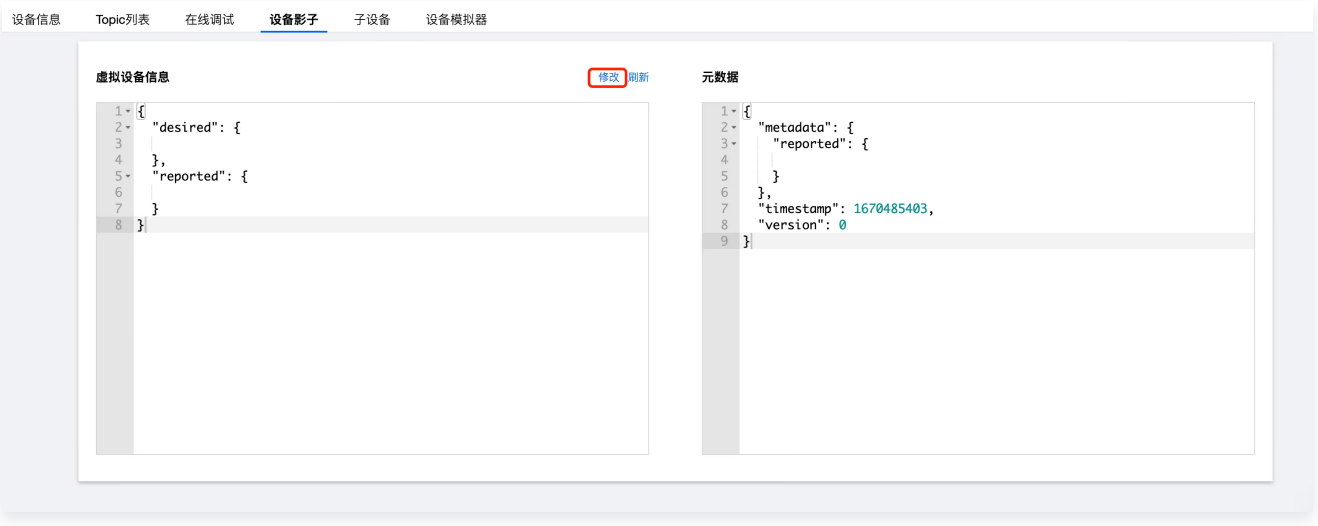
Device Status Reporting

After the device side reports the status to the IoT Cloud, the console will display the latest device shadow status.

Application updates device shadow status

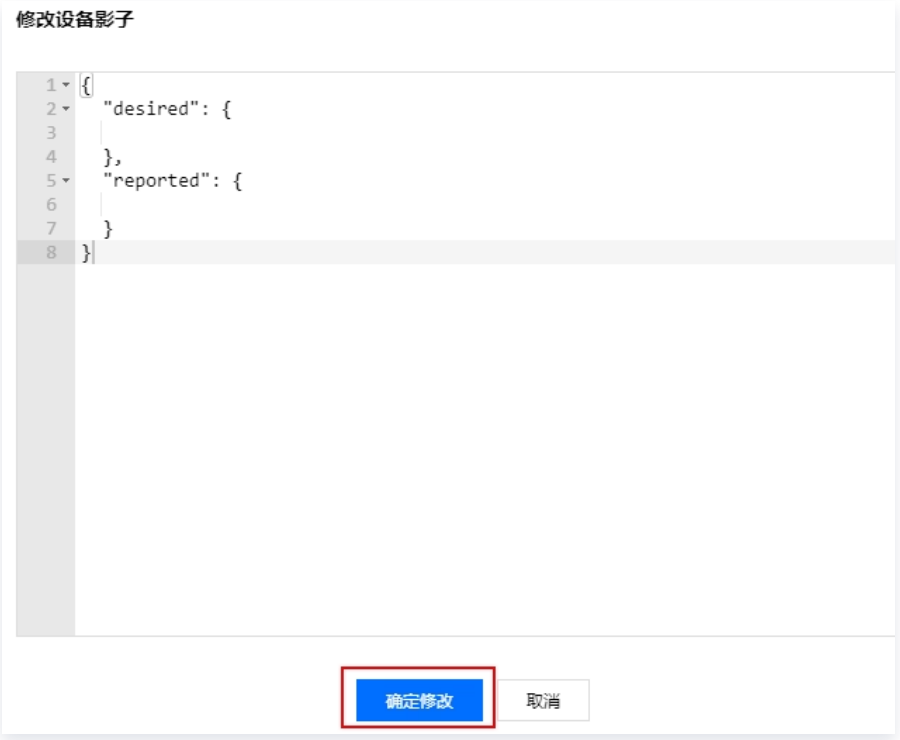
The console has the ability to modify the Virtual Device. Once the modification is saved, the device will receive the Virtual Device update.

1. Log in to the Internet of Things Hub console and click **View details** on the upper right of the Overview module.
2. Click on the left sidebar **Product List**.
3. Enter the product list page, click on the product name to go to the product details page.
4. Select **Device List**, click on a device name to enter the device details page, click on the **Device Shadow** tab, click on modify at the top right.



5. Follow the prompts on the right to add device shadow JSON data. The reported field can be empty, but the desired field cannot be. For detailed field explanations, refer to the document Introduction to Device Shadow.

6. After modification, click **Confirm Modification**.



Topic Management

Last updated：2025-03-19 14:46:56

Overview

After creating a product, you can configure the topics that its devices can publish or subscribe to. The device topic list is inherited from the product topic list. You can add, delete, or modify the items in the topic list only at the product level.

Operation Steps

Adding a custom topic

1. Log in to the Internet of Things Hub console and click **View details** on the upper right of the Overview module.
2. Click **Products** on the left sidebar.
3. Click a product name, and select **Topics**.
4. Click **Add Custom Topic** and set the topic name and grant device operation permissions.
   - Operation name: Naming supports letters, numbers, and underscore combinations; use / layer + to indicate the first level between different levels, use / + to name, cannot / + aaa /; length limit is 1 – 255 bits.
   - Operational permissions: You can select "subscription", "publish", or "subscription and publication", can be modified.

Editing a custom topic

By clicking **Edit** in the topic permission list, you can modify the name and permissions of the corresponding topic permission.

Deleting a topic permission

By clicking **Delete** in the topic permission list, you can delete the corresponding topic permission.

Cloud Logs

Last updated: 2025-03-19 14:47:10

The Internet of Things Hub Cloud Log Service module provides comprehensive, stable, and reliable log services, including multi-dimensional information such as device behavior, message content, and device anomalies. Users can search for critical device logs by time, log category, result, device name, RequestID, and keywords, helping you easily localize and resolve business issues.

Operations

1. Log in to the Internet of Things Hub console and click **View details** on the upper right of the Overview module.
2. Click Products on the left sidebar to enter the Products page.
3. Click the name of the target product to enter the product details page.
4. Click Cloud Logs to enter the log management page.

Run Logs

Running logs are divided into behavior logs, content logs, and device logs. You can search for logs by time and keyword.

- Search by Time
  On the time panel, select the log time range you want to view, including the last 15 minutes, last 60 minutes, last 4 hours, last 24 hours, and a custom time range. Cloud Logs can be stored for up to 3 days.
- Search by Keyword
  In the log search box, you can filter the logs to be queried. When you need to filter target logs by multiple conditions, you can press Enter to separate different tags.

Behavior logs

You can search for logs of all communication behaviors between a device and the cloud, including device connection, disconnection, publishing, and subscribing.

Content logs

You can search for detailed logs of communication content between a device and the cloud.

Device logs

Device logs display device information at four levels: ERROR, WARN, INFO, and DEBUG. You can enable/configure device log collection on the device details page.

Loading more

Cloud Logs load 100 entries at a time. You can click Loading more at the bottom of the page to view more log information.

Status Monitoring

Last updated: 2025-03-19 14:47:25

IoT Hub provides monitoring capabilities for the device connection, message sending and receiving, device shadow, rule engine, and OTA features. You can view the monitoring data in the last 30 days on the Status Monitoring page in the IoT Hub console, which helps you identify and troubleshoot business issues.

Status Monitoring Types

Currently, IoT Hub provides the following five types of monitoring statistics.

Device connection statistics

Device connection statistics are used to monitor device connection.

Device message statistics

Device message statistics include the numbers of sent and received messages and failed messages.

- Upstream Messages: number of messages published by the device or by the application through TencentCloud API.
- Downstream Messages: number of messages transferred to the device by IoT Hub (messages subscribed to by the device).
- Failed Upstream Messages: number of messages published by the device or by the application through TencentCloud API which failed to be published due to invalid format (such as excessive length), frequency limit, traffic limit, or lack of permission.
- Failed Downstream Messages: number of messages transferred to the device by IoT Hub (messages subscribed to by the device) which failed to be received due to invalid format (such as excessive length), frequency limit, or traffic limit.

Device shadow update statistics

Device shadow update statistics include the numbers of active device shadow updates and failed updates.

Rule engine statistics

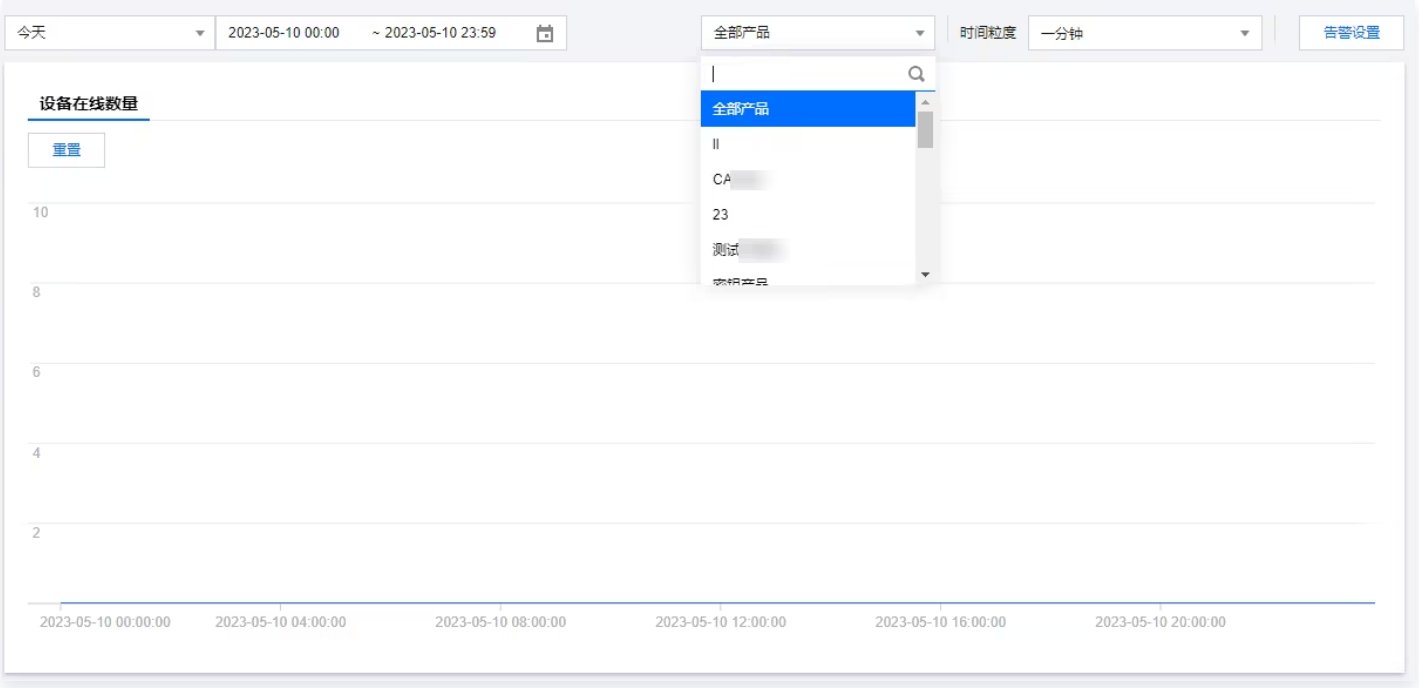
Rule engine statistics include the number of messages that hit the rule engine and the number forwarded to other services.

Firmware statistics

Firmware statistics include device version distribution and firmware version capacity statistics.

Product-Level Monitoring

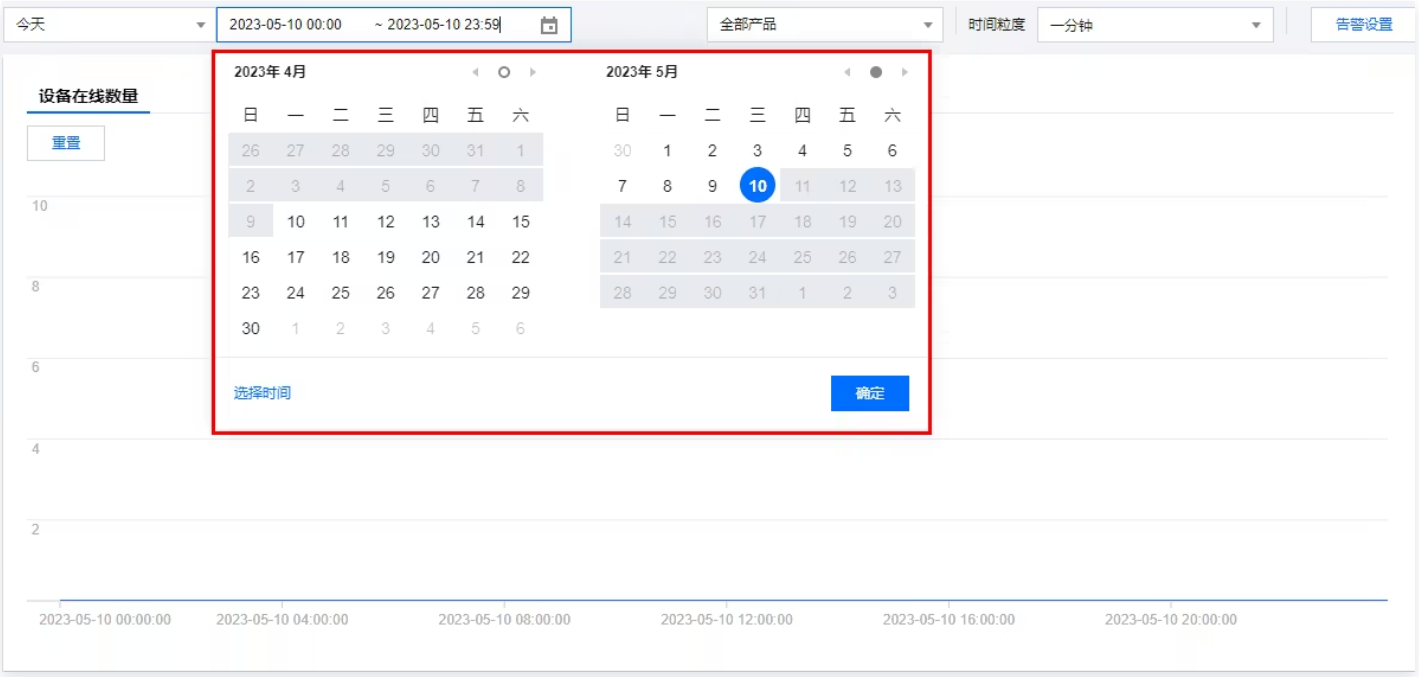
Status monitoring supports not only global monitoring but also product-level monitoring. You can search for desired product data by product name.



Note:
You can view the firmware statistics only after specifying a product.

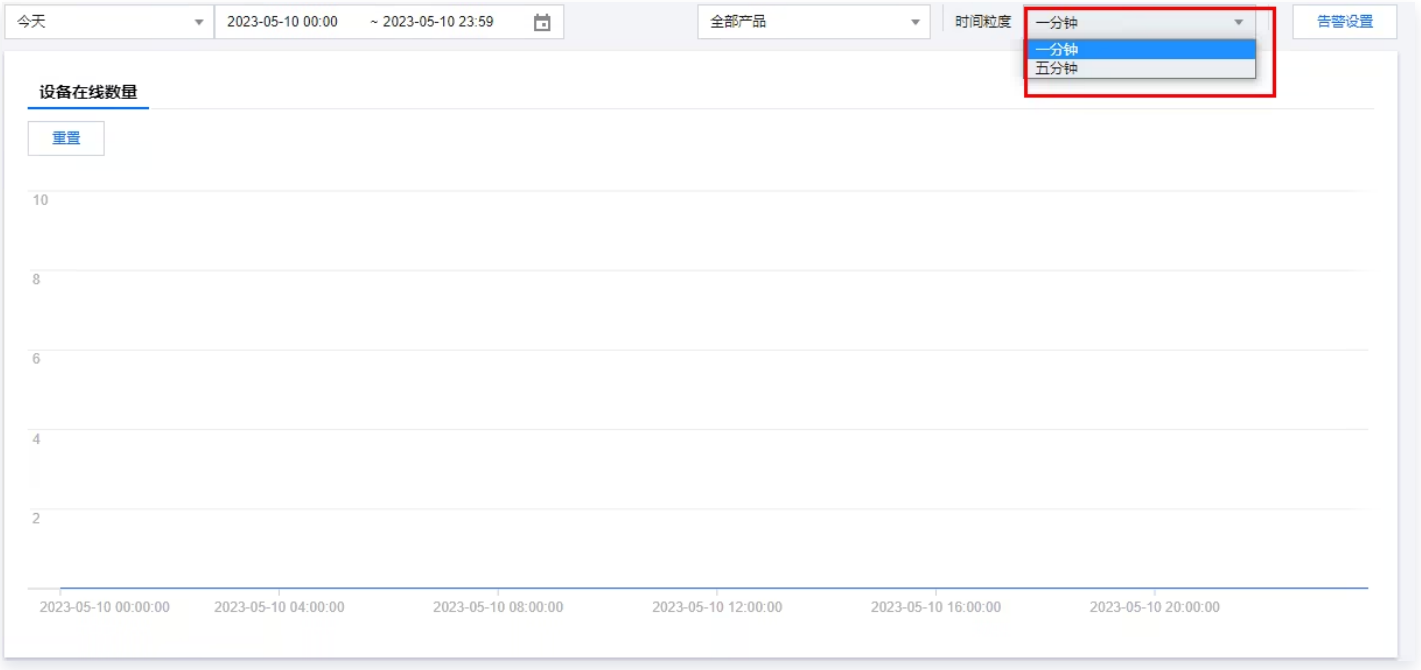
Search by Time

For status monitoring, you can select a time range to view data, including today, the last 7 days, the last 15 days, the last 30 days, and custom time range.



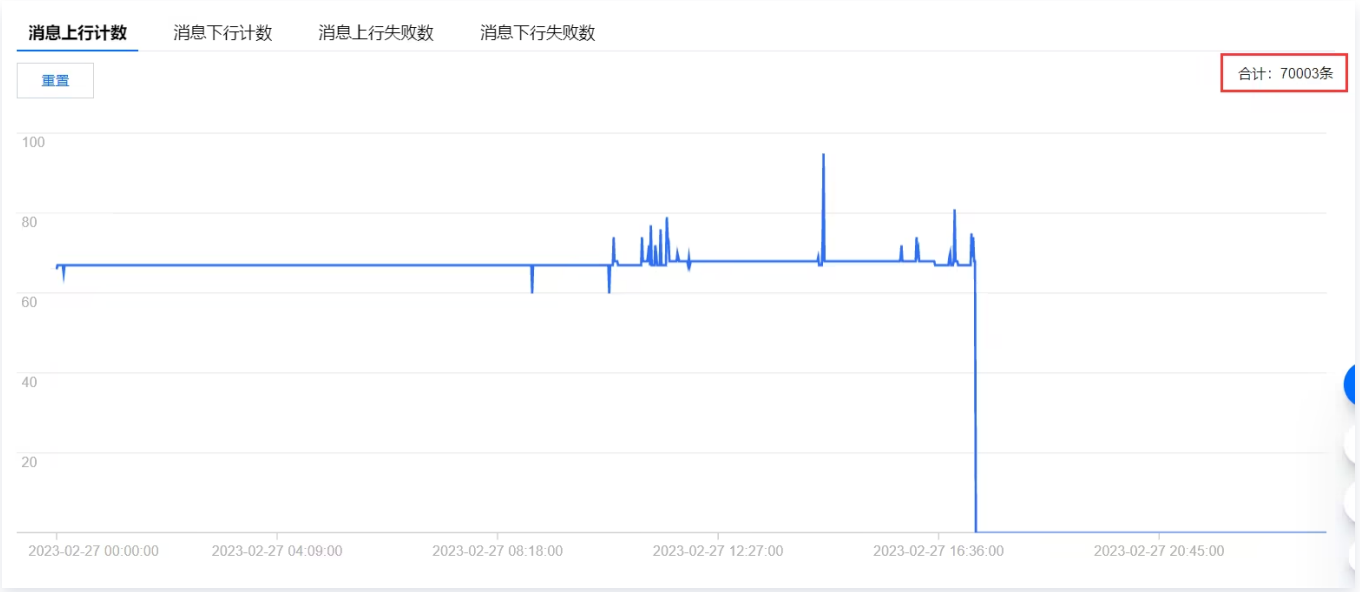
Data Point at Selected Time Granularity

In status monitoring, each displayed point is counted based on the selected time granularity. For the number of connected devices, the value displayed by a single point is the peak value over the selected time granularity. For other types, the value displayed by a single point is the sum calculated for the selected time granularity.



Zoom and Aggregation

The monitoring sequence diagram also supports scaling and total features. You can select a start time on the sequence diagram, hold and drag the mouse to the end time to view the monitoring data for that period.
The total feature automatically sums up the total number of data types displayed on the timeline and is shown in the top right corner of each statistical interface.



Click the **Reset** button to reset the scaling of the interface.

Monitoring and Alarms

Status monitoring supports the alert trigger feature of Tencent Cloud Observability Platform (TCOP) to monitor the number of devices online, the number of messages upstream and downstream, and the number of failed messages upstream and downstream. When an alert condition is met, alert messages will be sent to target users via SMS, email, or WeChat. Users can take appropriate actions based on the alert messages.

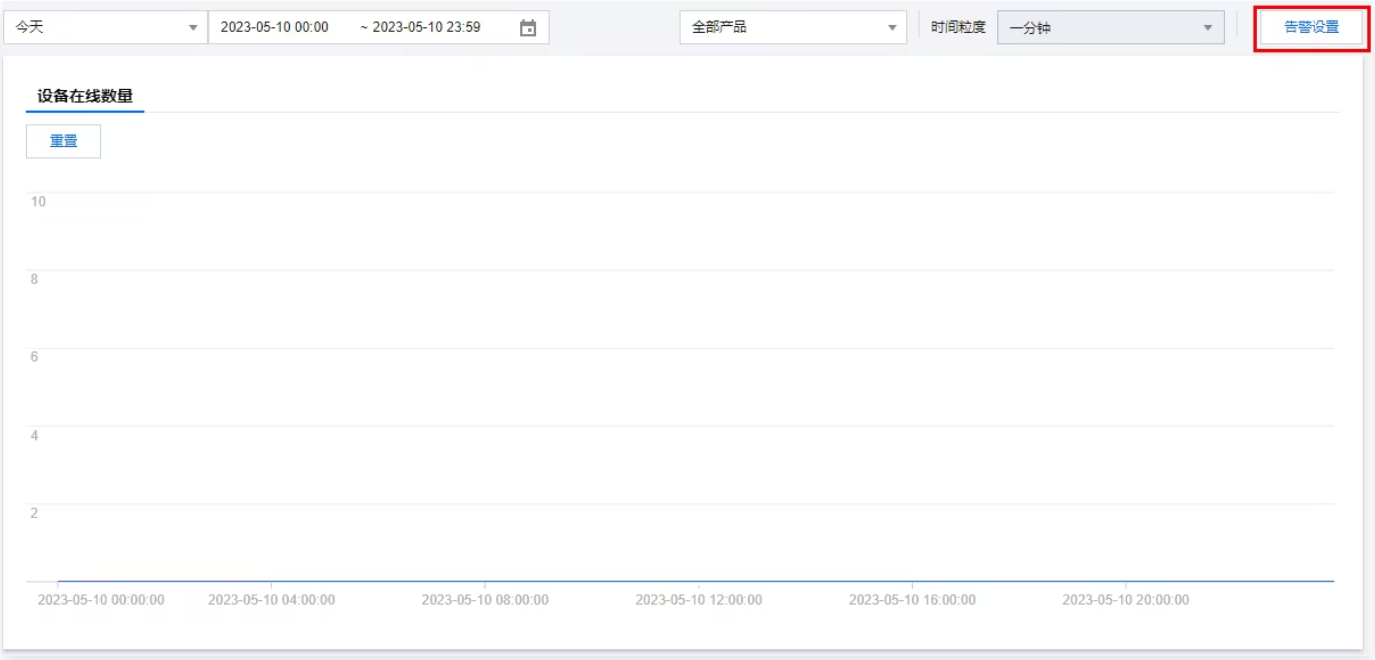
Creating Alarm Policy

IoT Hub supports setting alarms for the following metrics:

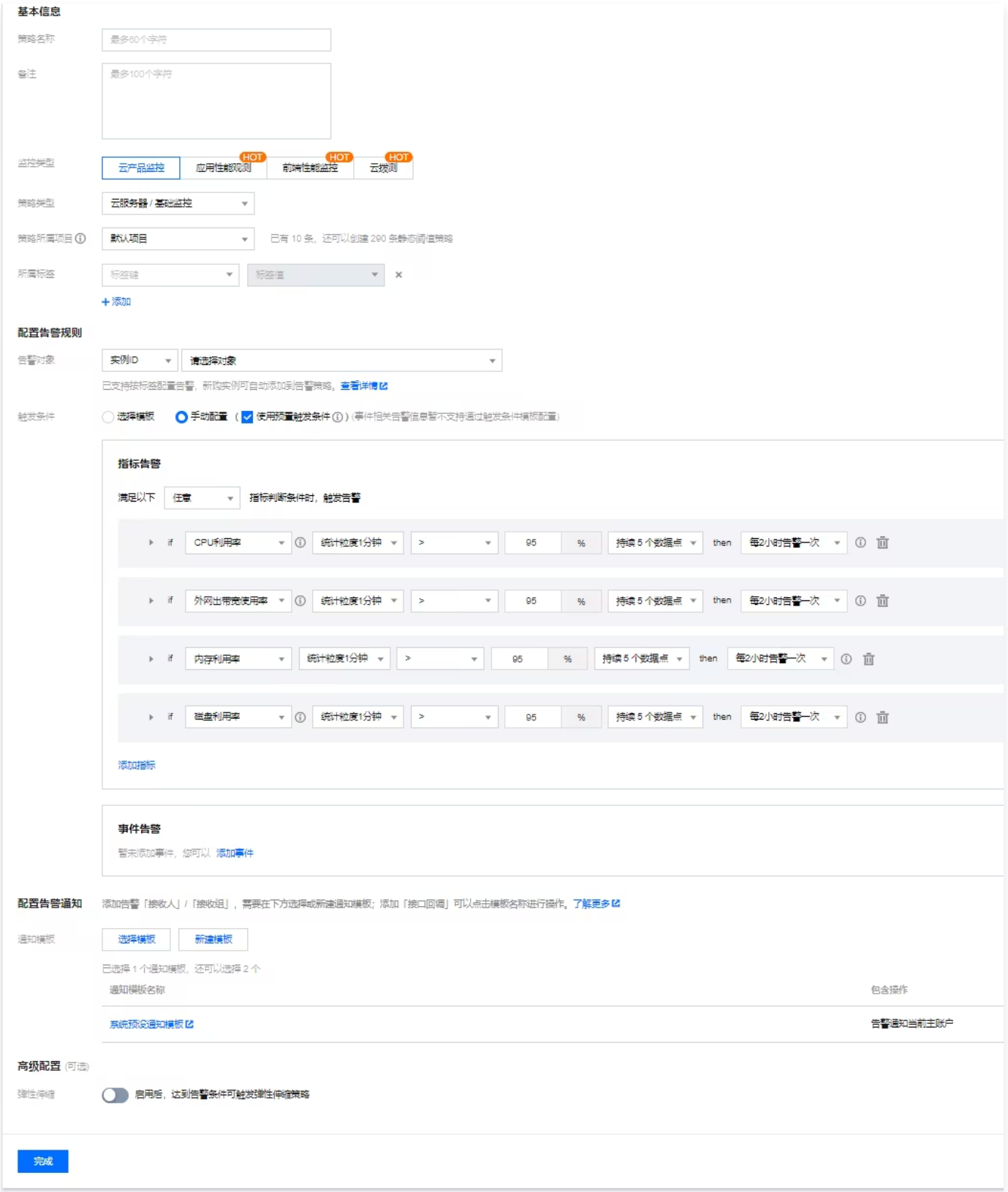
- Number of devices online.
- Message upstream count.
- Message downstream count.
- Message upstream failure count.
- Message downstream failure count.
- Number of device shadow updates.
- Number of device shadow failures.
- Number of rule hits.
- Number of rule forwards.

Operations

1. Log in to the IoT communication console, go to the Status Monitoring page, and click **Alert Settings** in the upper right corner to enter the alert settings page.



2. On the alert policy page, click **New Policy** to enter the new policy page and set the alert policy parameters.



- Policy Name: it can contain letters and underscores.
- Remarks: it describes the alarm policy and can contain letters and underscores.
- Policy type: The policy type corresponds to the alert policy type of the product. Here, select **IoT Hub** > **Status Monitoring**.

- Project: it is the project of the alarm policy. The default project is selected here.
- Alert object: Divided into all objects, partial objects, and instance groups.
  - All objects refer to all product objects in IoT Hub. When any product in IoT Hub meets the alert rules, an alarm notification will be sent.
  - Some objects are the selected product objects, and alarm notifications will only be sent when the selected product objects meet the alert rules.
  - The instance group contains product objects, and alarms can only be triggered when the objects in the instance group meet the alert conditions.
- Trigger conditions: You can set trigger condition templates and configure trigger conditions. The trigger conditions that can be set include:
  - You can set to trigger when any or all conditions are met.
  - You can set the monitoring category according to the status types supported by the IoT Hub platform.
  - You can set the statistical period and the number of continuous cycles for the policy.
  - You can set the comparative relationship for the alert policy.
  - You can set the alert threshold.
  - You can set the notification strategy. When an alarm is triggered, you can define the alarm to be repeatedly notified at a specific frequency. Options: do not repeat, once every 5 minutes, once every 10 minutes, and other exponentially increased frequencies. Exponential increase means that when an alarm is triggered for the first time, second time, fourth time, eighth time, ..., or 2 to the power of Nth time, an alarm notification will be sent to you. In other words, the alarm notification will be sent less and less frequently with longer time intervals in between, reducing the disturbance caused by repeated alarm notifications.

Note:
The default logic for repeated alarm notifications is as follows:
- The alarm notification will be sent to you at the configured frequency for 24 hours after an alarm is triggered.
- Following 24 hours after an alarm is triggered, the alarm notification will be sent once every day by default.
- The alarm notification will be sent for the last time 72 hours after the alarm is triggered and then will no longer be sent.

Trigger condition example: Set to trigger when all conditions are met; monitoring category is the number of devices online; statistical period is 1 minute, continuous cycle is 5 cycles;

comparative relationship is greater than; alert threshold is 100; notification strategy is to alert once a day. The number of devices online is counted every minute, and an alarm will be triggered when the number of devices online in the product objects exceeds 100 for 6 consecutive times.

- Alert channels: you can set the recipients, receiving time periods, receiving channels, and receiving languages.
- Advanced features: When enabled, the elastic scaling strategy can be triggered if the alert conditions are met.
- API Callback: alarm messages can be pushed to this address when the alarm conditions are met.

3. Click **save** to successfully create the alert policy.

Rule Engine Overview

Last updated: 2025-03-19 14:48:01

Purpose

When communication is performed based on a topic, you can use the rule engine to process the data in the topic and then forward it to other Tencent Cloud services or your business backend services, implementing services such as data acquisition, computing, and storage.

Step

1. Log in to the Internet of Things Hub console and click **View details** on the upper right of the Overview module.
2. Select the Rule Engine from the left menu.
3. On the rule engine page, click Create Rule, enter the rule name, and click Confirm.
   - Rule name: Supports combinations of letters, digits, and underscores, with a maximum of 32 characters. (The name cannot be modified after creation, please fill it in carefully.)
   - Rule Description: 0-256 characters. This can be modified.

4. After the rule is created successfully, you will be automatically redirected to the rule details page to edit data filtering and behavior operations.
   - Report filter: By selecting equipment, topic type, and setting the required extract content field and condition, generate the SQL statements for data extraction.

Field

JSON data reported by equipment supports extracting data content through fields, with multiple fields separated by commas. If all fields need to be forwarded, enter '*'. For non-json data reported by equipment, only '*' can be entered to forward all content. The field definitions are as follows:

- Only '*', ',', '.', spaces, letters, and numbers are supported in the field. It cannot be empty and must not exceed 300 characters.
- The field represents the key in JSON. If the data format is non-JSON, field filtering cannot be used. You can use '*' to forward all data.
- The reported JSON data format can be nested JSON. For example: {"device_status": {"switch":"on"}}, you can get the value of switch through device_status.switch.
- SQL and JSON subarrays are not supported.

Condition

Supported only when the device reports JSON data. Fill in the field (this field must be in the JSON data reported by the device) calculation expression in the condition. Data will be extracted and forwarded only when the message reported by the device meets the condition expression. Supported expressions are shown in the table below:

Operator	Description	Example
=	Equal to	color = 'red'
<>	Not equal to	color <> 'red'
AND	Logical AND	color = 'red' AND siren = 'on'
OR	Logical OR	color = 'red' OR siren = 'on'
( )	The content in the parentheses is a complete entity	color = 'red' AND (siren = 'on' OR siren ='isTest')
+	Arithmetic addition	age = 4 + 5
-	Arithmetic subtraction	age = 5 - 4
/	Remove	age = 20 / 4

*	Multiply	age = 5 * 4
%	Modulo	age = 0 % 6
<	Less than	5 < 6
<=	Less than or equal to	5 <= 6
>	Greater than	6 > 5
>=	Greater than or equal to	6 >= 5

- **Behavior operation**: Used to configure the target for receiving data, supporting the following behavior types:
  - Data Forwarding to Third-Party Service.
  - Data Forwarding to TDMQ.
  - Data Forwarding to CKafka.
  - Data Forwarding to TencentDB for MySQL.
  - Data Forwarding to TencentDB for MongoDB.
  - Data Forwarding to TDSQL for MySQL.
  - Data Forwarding to Tencent CloudBase.
- Action for forwarding error: By configuring a different behavior type from the behavior operation, if the behavior operation fails to forward after three retries, it will forward once to the error action to ensure no loss of device data.

Rule Function

Last updated: 2025-03-19 14:48:16

The rule engine provides a wide variety of functions, which you can use in the corresponding values of the rule engine fields, conditions, and database fields to process data in diverse ways.

Supported Functions

Function Name	Usage Description
productId()	Returns the ID of the product from which the message comes.
deviceName()	Returns the name of the device from which the message comes.
timestamp()	Returns the current Unix system timestamp in seconds.
topic()	Returns the original topic from which the message comes.
topic(n)	Return the n-th segment of the original Topic from the message source, divided by `/`.
payloadLen()	Returns the length of the payload in bytes.
bin_to_dec()	Convert the binary number 'data' to a decimal integer.
to_hex ()	Convert the entered original message to a hexadecimal string.
randint(min,max)	Return a random integer between min and max.
upper(string)	Return uppercase string (The input message format should be in JSON format, with the function object as the corresponding key value. For example, if the input message is `"tencent":"iot"`, then `upper(tencent)=IOT`).
lower(string)	Return lowercase string (The input message format should be in JSON format, with the function object as the corresponding key value).
crypto(field,String)	Encrypt the value of the field. The second parameter String is the algorithm string. Options: MD5, SHA1, SHA256, SHA384, SHA512. (The input message format must be in JSON format, and the function object is the corresponding key value).

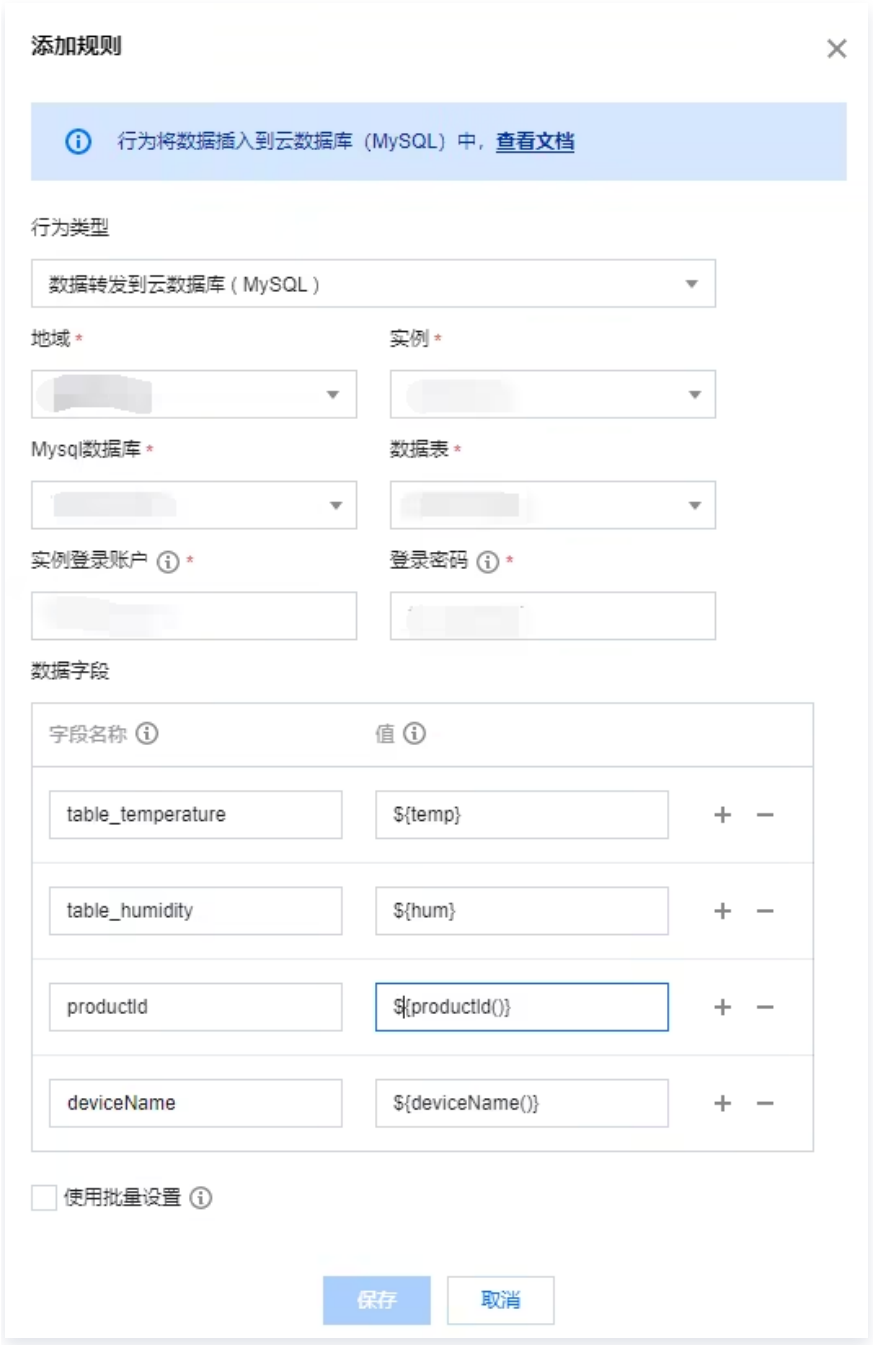
concat(string1, string2)	Concatenate strings, such as `concat(deviceid,'a')` or `concat(field1,field2)`.
requestId()	Returns the message ID generated by IoT Hub. The data is not unique, so do not use it as a database primary key.
newuuid()	Returns a random UUID string. The data is not unique, so do not use it as a database primary key.
replace(source, substring, replacement)	Replace the substring in the source using replacement.
substring(source, start, end)	String truncation, returns a substring from start (inclusive) to end (exclusive).

Use Case

The message content sent by a home temperature and humidity device dev00 to the Cloud is:

```
{"room1":{"temperature":31,"humidity":"63%"},
"room2":{"temperature":26,"humidity":"63%"}}
```

Under the temperature and humidity products, there are three devices: dev00, dev01, and dev02. They monitor the temperature and humidity of room1, room2, room3... room6 respectively. Data needs to be transferred to the MySQL database for processing only when the temperature of room1 exceeds 30 degrees Celsius. The rule engine is set as follows for this case:



Data Forwarding to Third-Party Service

Last updated: 2026-03-20 15:13:20

Overview

When forwarding message fields extracted by the rule to a third-party service, you can customize how to handle this data. This method provides the highest flexibility for users in message handling.

> **Note:**
> Third-party service must be provided via HTTP or HTTPS. To configure forwarding to a third-party service, a website URL and port that support HTTP or HTTPS must be provided. After the rule engine forwards successfully, the third-party service will receive data packets from `42.193.134.62,106.52.211.220`, `157.255.11.202`, `14.215.166.14`, `120.233.106.237`.

Fill in the server configuration

Follow the operation guide to create rules and filter data, then add action operations and select forwarding to a third-party service.

- Select action type as "Data Forwarding to Third-Party Service (Forward)".
- Enter your HTTP or HTTPS service address. The Internet of Things Hub platform will forward the data reported by the devices to the HTTP or HTTPS service address.
- To ensure your service request is legitimate and valid, check "Add Authentication Token" and enter the Token corresponding to your service; you can enter any Token to generate a signature (the Token will be compared with the Token included in the interface URL for security validation). When "Add Authentication Token" is checked, your service must implement the Message Verification logic.

Message Verification

> **Note:**
> To ensure stable use of your backend, please select Add Authentication Token.

Request Identification

If the user has selected "Add Authentication Token" for third-party service (Forward), i.e., HTTP forwarding, the Internet of Things Hub platform will add the following fields in the HTTP or HTTPS request header:

Parameters	Description
Signature	Signature combines the Token parameter entered in "Add Rule" and the Timestamp and Nonce parameters in the request.
Timestamp	Timestamp.
Nonce	A random number.

1. Sort the Token, Timestamp, and Nonce parameters in dictionary order.
2. Concatenate the three parameter strings into one string and perform SHA1 encryption.
3. Developers can compare the encrypted string with the Signature to identify that the request comes from the Internet of Things Hub platform.

The following is an example PHP code to verify the Signature:

```
private function checkSignature()
{
    $signature = $_GET["signature"];
```

```
    $timestamp = $_GET["timestamp"];
    $nonce = $_GET["nonce"];

    $token = TOKEN;
    $tmpArr = array($token, $timestamp, $nonce);
    sort($tmpArr, SORT_STRING);
    $tmpStr = implode( $tmpArr );
    $tmpStr = sha1( $tmpStr );

    if( $tmpStr == $signature ){
        return true;
    }else{
        return false;
    }
}
```

For example, in a request, the relevant parameters are as follows: the user sets the Token as aaa.

```
Nonce: IkOaKMDalrAzUTxC
Signature: c259ed29ec13ba7c649fe0893007401a36e70453
Timestamp: 1604458421
```

The sorted string is `1604458421IkOaKMDalrAzUTxCaaa`, and the final calculated sha1 result is `c259ed29ec13ba7c649fe0893007401a36e70453`.

Service Address Validation

1. When the rule engine is enabled, the Internet of Things Hub platform will send a GET Request to the specified server address URL, with the following fields added to the GET Request header:

Parameters	Description
Signature	Signature combines the Token parameter entered in "Add Rule" and the Timestamp and Nonce parameters in the request.
Timestamp	Timestamp.
Nonce	A random number.
Echostr	Random string.

Message example sent by the Internet of Things Hub platform to third-party service:

```
GET / HTTP/1.1
Host: **.**.**.**:4443
User-Agent: Go-http-client/1.1
Content-Type: application/json
Echostr: UPWIAFASvDUFcTEE
Nonce: testrance
Signature: abb6c316a8134596d825c5a1295bfa6f7657664d
Timestamp: 1623149590
Accept-Encoding: gzip
```

2. If the third-party service confirms that this GET request is from the Internet of Things Hub platform, please return the Echostr parameter content as is in the body.
Message example of the third-party service's reply to the Internet of Things Hub platform:

```
HTTP/1.1 200 OK
Date: Tue, 08 Jun 2021 10:53:10 GMT
Content-Length: 16
Content-Type: text/plain; charset=utf-8

UPWIAFASvDUFcTEE
```

3. The Internet of Things Hub platform verifies the returned Echostr parameter content to confirm whether the server address URL is valid.

Data Format

The message types include Topic messages reported by devices and notification messages of device status changes detected by the platform. After successful forwarding, the data format received by the third-party service for these two types of messages is different, as follows:

- Topic messages reported by the device:

  After successful forwarding, the data format received by the third-party service is as follows:

```
{
 "payload": {
```

```
        "device_inf": "car_device",
        "temperature": 26
    }
    "timemills": 1660210398035,
    "seq": 212934692,
    "timestamp": 1660210398,
    "topic": "RTOYL6STQ0/dev_01/event",
    "devicename": "dev_01",
    "productid": "RTOYL6STQ0"
}
```

The meanings of each field are as follows:

- payload: According to the data filtering rules, the data filtered from the device's reported topic message; if the device reports binary data, the payload will be the original data Base64-encoded.
- seq: Sequence number.
- topic: The topic of this message report.
- productid: Product ID.
- devicename: Device name.
- timestamp: the Unix timestamp when the forwarding action is triggered.
- timemills: The timestamp when the forwarding action is triggered, in milliseconds.

- Device Status Change Notification:

  When the platform detects a change in device status, it will trigger this message forwarding. After successful forwarding, the third-party service will receive the data in the following format:

```
{
    "MsgType": "Forward",
    "Event": "",
    "Topic": "$state/report/K72CRAIG98/pskDevice001",
    "Seq": 0,
    "PayloadLen": 178,
    "ProductId": "K72CRAIG98",
    "DeviceName": "pskDevice001",
    "Payload":
"eyJkZXZpY2VOYW1lIjoicHNrRGV2aWNlMDAxIiwiZXZlbnQiOiJFVl9PTkxJTkUiLCJwc
m9kdWN0SUQiOiJLNzJDUkFJRzk4IiwicmVhc29uIjoiUkVBU09OX0RFVklDRV9DT05ORUN
```

```
UIiwidGltZXN0YW1wIjoxNjc2OTY1MzUxLCJ0b3BpYyI6IiRzdGF0ZS9yZXBvcnQvSzcyQ
1JBSUc5OC9wc2tEZXZpY2UwMDEifQ==",
    "Time": "2023-02-21 15:42:31",
    "TimeMills": 1676965351605,
    "Reason": ""
}
```


After Base64-decoding the payload:


```
{
    "deviceName": "dev_01",
    "event": "EV_OFFLINE",
    "productID": "KXUCF9GJ9H",
    "reason": "REASON_DEVICE_DISCONNECT",
    "timestamp": 1677068839,
    "topic": "$state/report/KXUCF9GJ9H/dev_01"
}
```


The meanings of each field are as follows:

- event: "EV_ONLINE" for online, "EV_OFFLINE" for offline.
- reason: the reason for device status change:
  - "REASON_DEVICE_DISCONNECT": device disconnected.
  - "REASON_STATE_KICKED": server actively kicked offline.
  - "REASON_DEVICE_KICKED": device-side mutual kick offline.
  - "REASON_KEEPALIVE_TIMEOUT": device-side timeout disconnect.
- productID: Product ID.
- deviceName: Device name.
- timestamp: Timestamp.
- topic: Topic information.

Resending Mechanism

The resending mechanism is used to send the message again in case of a failure in the message forwarding process, which makes sure that the message is received. The details are as follows:

- If message forwarding fails, the system will retry forwarding at intervals of 1s, 3s, and 10s in sequence. If all three retries fail, the message will be discarded.

- If you have configured the "action for forwarding failure", then after three unsuccessful retries, the message will be forwarded again according to the configured action. If forwarding still fails, the message will be discarded.

Data Forwarding to CKafka

Last updated: 2025-03-19 14:48:42

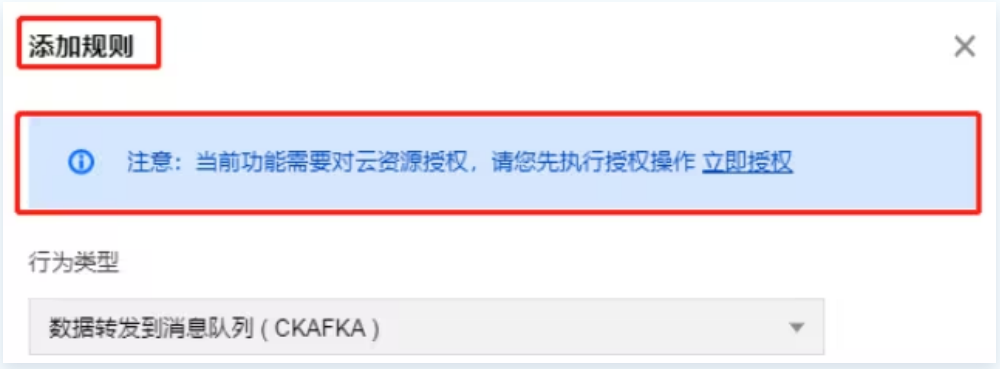
Overview

The rule engine allows you to configure rules to forward eligible data reported by devices to CKafka, and then your application server can read the data from CKafka for processing. This takes advantage of CKafka's high throughput to create a highly available message linkage.

Configuration

Create rules and filter data according to the operation guide, then add behavior operations and select forwarding to CKafka.

> **Notes:**
> The first time you use it, you will be prompted to authorize access to CKAFKA. You need to click **Authorize Access to CKAFKA** to continue creating.



1. In the pop-up "Add Rule" window, select the action "Data Forwarding to Message Queue (CKAFKA)"; sequentially select the CKAFKA instance and Topic, then click **Save** to

complete.

2. After the above configuration is completed, IoT Hub will forward eligible data reported by devices to the configured CKafka instance. You can refer to the creating instance and topic document to read and process the data on your own application server.

Data Format

Message types include topic messages reported by devices and notification messages of device status changes monitored by the platform. After successful forwarding, the data formats received by CKafka are different, as follows:

- Topic messages reported by the device:

  After successful forwarding, the data format received by CKafka is as follows:

```
{
 "MsgType": "Forward",
 "Event": "",
 "Topic": "7PQ0I75ZWY/dev_01/event",
 "Seq": 32569,
 "PayloadLen": 44,
 "ProductId": "7PQ0I75ZWY",
 "DeviceName": "dev_01",
```

```
 "Payload":
"eyJkZXZpY2VfaW5mIjoiY2FyX2RldmljZSIsInRlbXBlcmF0dXJlIjoxOX0=",
 "Time": "2022-08-11 19:17:24.943",
 "TimeMills": 1660216644943,
 "Reason": ""
}
```

The meanings of each field are as follows:

- MsgType: The value is "Forward".
- Topic: The topic when the device reports this message.
- Seq: serial number.
- PayloadLen: length of the message reported by the device.
- ProductId: Product ID.
- DeviceName: Device name.
- Payload: message content reported by the device after Base64 decoding.
- Time: time when the forwarding action is triggered, e.g., "2022-08-11 12:00:00".
- TimeMills: timestamp when the forwarding action is triggered, in milliseconds.

- Device Status Change Notification:

  When the platform detects a change in device status, it triggers this message forwarding. After successful forwarding, the received data format is as follows:

```
{
    "MsgType": "Forward",
    "Event": "",
    "Topic": "$state/report/K72CRAIG98/pskDevice001",
    "Seq": 0,
    "PayloadLen": 178,
    "ProductId": "K72CRAIG98",
    "DeviceName": "pskDevice001",
    "Payload":
"eyJkZXZpY2VOYW1lIjoicHNrRGV2aWNlMDAxIiwiZXZlbnQiOiJFVl9PTkxJTkUiLCJwc
m9kdWN0SUQiOiJLNzJDUkFJRzk4IiwicmVhc29uIjoiUkVBU09OX0RFVklDRV9DT05ORUN
UIiwidGltZXN0YW1wIjoxNjc2OTY1MzUxLCJ0b3BpYyI6IiRzdGF0ZS9yZXBvcnQvSzcyQ
1JBSUc5OC9wc2tEZXZpY2UwMDEifQ==",
    "Time": "2023-02-21 15:42:31",
    "TimeMills": 1676965351605,
    "Reason": ""
```

```
}
```

After Base64-decoding the payload:

```
{
    "deviceName": "dev_01",
    "event": "EV_OFFLINE",
    "productID": "KXUCF9GJ9H",
    "reason": "REASON_DEVICE_DISCONNECT",
    "timestamp": 1677068839,
    "topic": "$state/report/KXUCF9GJ9H/dev_01"
}
```

The meanings of each field are as follows:

- event: "EV_ONLINE" for online, "EV_OFFLINE" for offline.
- reason: the reason for device status change:
  - "REASON_DEVICE_DISCONNECT": device disconnected.
  - "REASON_STATE_KICKED": server actively kicked offline.
  - "REASON_DEVICE_KICKED": device-side mutual kick offline.
  - "REASON_KEEPALIVE_TIMEOUT": device-side timeout disconnect.
- productID: Product ID.
- deviceName: Device name.
- timestamp: Timestamp.
- Topic information.

Resending Mechanism

The resending mechanism is used to send the message again in case of a failure in the message forwarding process, which makes sure that the message is received. The details are as follows:

- If message forwarding fails, the system will retry forwarding at intervals of 1s, 3s, and 10s in sequence. If all three retries fail, the message will be discarded.
- If you have configured the "action for forwarding failure", then after three unsuccessful retries, the message will be forwarded again according to the configured action. If forwarding still fails, the message will be discarded.

Data Forwarding to TDMQ

Last updated：2025-03-19 14:48:56

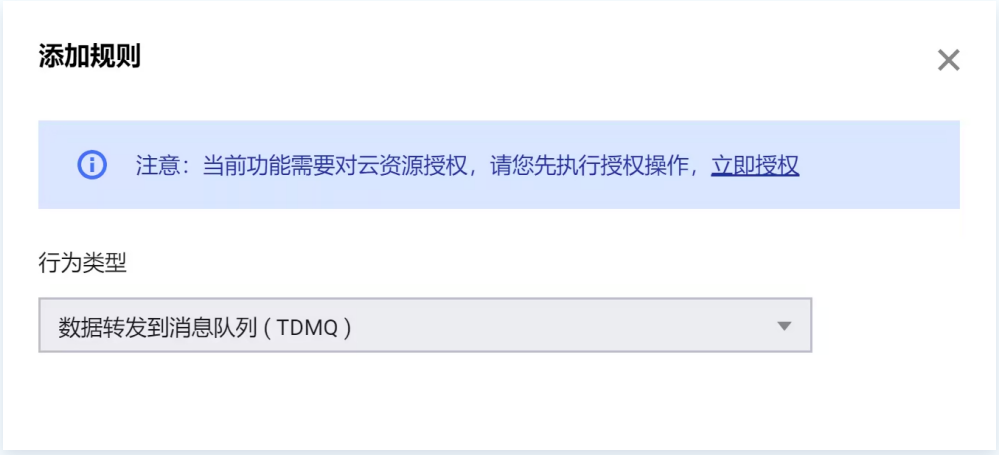
Overview

The rule engine allows you to configure rules to forward eligible data reported by devices to the message queue TDMQ topic. After subscribing to a TDMQ topic through TencentCloud API, you can receive messages pushed from the topic. The message push mechanism of the TDMQ topic provides the capability to receive messages asynchronously with high reliability.

Configuration

Create regulations and filter data according to the Operation Guide, then add behavior operations and select forwarding to TDMQ.

Note:
You will be prompted to authorize access to the TDMQ Topic if this is your first time using the rule engine. Click **Authorize Now** to continue creating.



In the "Add Rule" window that appears, select the "Data forwarding to message queue (TDMQ) option", region, and Topic, then click **Save** to complete.

Data Format

Message types include topic messages reported by devices and notification messages of device status changes detected by the platform. After successful forwarding of both types of messages, the data format received by TDMQ is the same as that forwarded to CKAFA. Please refer to the Data Format received by CKAFA.

Resending Mechanism

The resending mechanism is used to send the message again in case of a failure in the message forwarding process, which makes sure that the message is received. The details are as follows:

- If message forwarding fails, the system will retry forwarding at intervals of 1s, 3s, and 10s in sequence. If all three retries fail, the message will be discarded.
- If you have configured the "action for forwarding failure", then after three unsuccessful retries, the message will be forwarded again according to the configured action. If forwarding still fails, the message will be discarded.

Data Forwarding to CTSDB

Last updated: 2026-04-03 17:39:38

Overview

The rule engine allows you to configure rules to forward eligible data reported by devices to TencentDB for CTSDB (hereinafter referred to as CTSDB), and then your application server can read the data from CTSDB for processing. This takes advantage of CTSDB's high storage compression rate and aggregate display for massive amounts of data, effectively meeting the daily needs of devices for data storage, analysis, and visual display.

> **Note:**
> Only CTSDB 1.0 is supported.

Configuration Steps

Create rules and filter data according to the Operation Guide, then add actions and select forwarding to the time series database (CTSDB).

> **Note:**
> The first time you use it, you will be prompted to authorize access to CTSDB. You need to click **Authorize Access to CTSDB** to continue creating.
>
> 
>

In the pop-up "Add Rule" window, select the action "**Data Forwarding to Time Series Database (CTSDB)**", then select the CTSDB region and instance, fill in the basic information and the forwarding fields to be configured, and click **Save**.

After the above configuration is completed, IoT Hub will forward eligible data reported by devices to the configured CTSDB instance. You can refer to the CTSDB Development Guide to read the data on your own application server for processing or aggregate, search for, and query the data in the CTSDB Console.

Configuration Parameter Description

- **Instance Login Account**: this is the account name entered when you create the CTSDB instance before configuring the rule engine.
- **Login Password**: this is the account password entered when you create the CTSDB instance before configuring the rule engine.
- **Metric**: this configures to which CTSDB metric to forward the data. If the metric is not present when the rule engine is configured, IoT Hub will create it automatically.
- **Timestamp**: this is the timestamp when the data is written to CTSDB. Currently, 4 types of configurations are supported:
- Referencing the original message field value through `${}`.
- System function.
- timestamp(): the current system time of the message hitting the rule engine.
- Constant: it needs to be a Unix timestamp in seconds; if it is left empty, the current time of the message hitting the rule engine will be used by default.

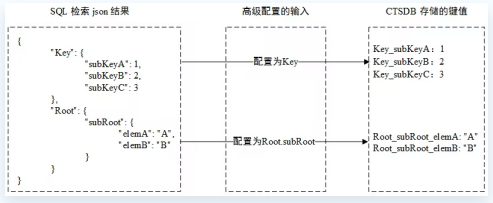
> **Note:**
> If you change the unit of the timestamp of the CTSDB metric to a unit other than second (such as millisecond) after the rule is created, subsequent data writes may fail.

- **Data field**: The type can be selected from tag type or field type in CTSDB. For field name input restrictions, please refer to CTSDB restrictions. There are three configuration methods for values: referencing the field value of the original message through ${}; constant; fixed value.

Advanced Configuration Description

The advanced configuration items are suitable for scenarios where the device-reported data fields are dynamically expanding and cannot be preconfigured. For example, if there are many sensors on the devices that need to transfer data, but different devices have different specifications and configurations, and the number of sensors is variable, then you can use the following advanced configuration to store the data from all device sensors to CTSDB through the rule engine:

> **Note:**
> - Default storage type: The storage type of fields with dynamic storage expansion in CTSDB is tag type by default.
> - key: The json key that needs to traverse the extended storage. The IoT Hub will traverse the json key-value nesting under this key, using '_' as the connector, and finally store it in the time series database. The json result obtained through the rule engine configured SQL SELECT retrieval and configuration (supports configuring sub-keys and multiple configurations) is stored in CTSDB as shown in the following example:



Resending Mechanism

The resending mechanism is used to send the message again in case of a failure in the message forwarding process, which makes sure that the message is received. The details are as follows:

- If message forwarding fails, the system will retry forwarding at intervals of 1s, 3s, and 10s in sequence. If all three retries fail, the message will be discarded.
- If you have configured the "action for forwarding failure", then after three unsuccessful retries, the message will be forwarded again according to the configured action. If forwarding still fails, the message will be discarded.

Forwarding Data to TencentDB for MySQL

Last updated: 2025-03-19 14:49:27

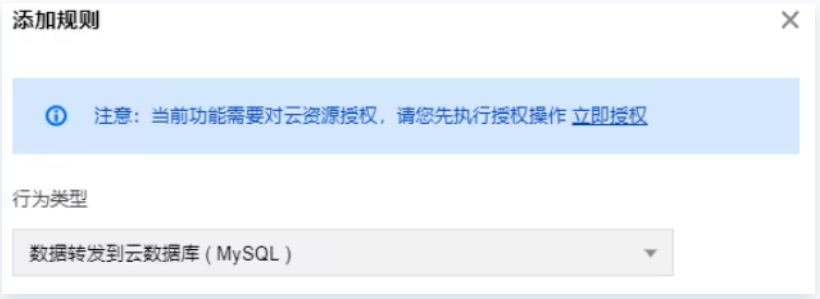
Overview

The rule engine supports user configuration of forwarding rules to forward eligible device-reported data to the cloud component MySQL. You can create a MySQL instance and tables in the MySQL console or use the TencentCloud API. Then, you can just write the specified fields in the device messages into the corresponding MySQL tables.

Configuration

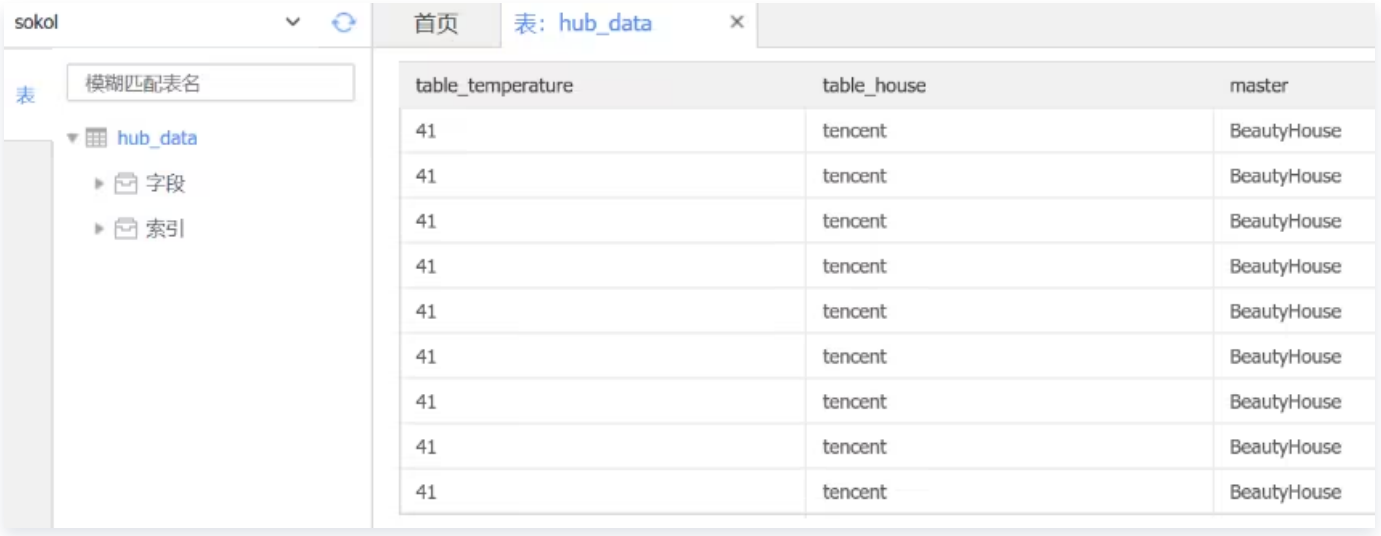
1. Create rules and filter data according to the operation guide, then add behavior operations and select Forward to CloudDB (MySql).

> **Note:**
> User authorization to access MySQL will be prompted for first-time use. You need to click **Authorize Now** to continue creating.



2. In the pop-up "Add Rule" window, select the "Forward Data to Cloud Database (MySQL)" option. After successful authorization, you need to configure the MySQL instance information and the field information to be written, as shown below. Once configured, click **Save**.

3. After successful forwarding, the information displayed in MySQL is as follows:



Configuration Instructions

Configuration is divided into the following steps:

1.1 Select a region and a MySQL instance.
1.2 Input the username of the MySQL instance just created.
1.3 Enter the instance login password.
1.4 Select the name of the database to write into. If no database has been created under the created mysql instance, please go to the MySQL console to create a new database. For specific operations, see Creating Databases and Tables.
1.5 Select the table to write into. If no table has been created under the database, go to the MySQL console to create a new table.
1.6 Configure the fields to be written. There are two columns here: "Field Name" and "Value".
- The "Field Name" corresponds to the field in the database table, indicating the field to be written.
- The "Value" indicates the value to be written to the corresponding field. The source of the value can be the message body (note that the message body must be in Json format for value extraction) or a constant filled in here.

> **Note:**
> - If the source is the message body, use `${}` to reference the fields in the message body. If you want to specify a constant, directly fill in the appropriate value, such as a digit like 5 or a string literal like hello.
> - You need to create the database, table, and field names in the Cloud Component MySQL before you can successfully write data to the database.

For more details, see Create database and table.

Resend Mechanism

The resend mechanism is used to resend again in case of failure during the message forwarding process, so as to achieve the purpose of receiving messages. The specific instructions are as follows:

- If the message forwarding fails, the system will perform a forwarding retry. The retries are performed sequentially at intervals of 1 s, 3 s, and 10 s. If all three retries fail, the message will be discarded.
- If the user has configured the "forwarding error behavior operation", after three retry failures, message forwarding will be proceeded with once according to the configuration of the "forwarding error behavior operation". If it still fails, the message will be discarded.

Data Forwarding to TencentDB for MongoDB

Last updated: 2025-03-19 14:49:47

Overview

The rule engine allows you to configure forwarding rules to forward eligible data reported by devices to TencentDB for MongoDB. After you create an instance in the TencentDB for MongoDB console or through TencentCloud API, device messages can be written to the corresponding TencentDB for MongoDB set.

> **Note:**
> The replica set instance version must not exceed v4.0, and the sharding instance version must be v4.0 or later.

Configuration

Follow the operation guide to create rules and filter data, then add behavior operations and select forwarding to TencentDB for MongoDB.

> **Note:**
> When using for the first time, users will be prompted to authorize access to MongoDB. You need to click **Authorize Now** to continue creating.

Enter the Add Behavior page and select the "Data Forwarding to Cloud Database (MongoDB) Option".

After successful authorization, you need to configure the TencentDB for MongoDB instance information. As shown below, the configuration is divided into the following steps:

1.1 Select the region and TencentDB for MongoDB instance. If there are no instances under the account, click **create instance** to jump to the TencentDB for MongoDB console to create one.

1.2 Enter the username of the TencentDB for MongoDB instance. The default username on the MongoDB official website is mongouser.

1.3 Enter the login password of the TencentDB for MongoDB instance.

1.4 Enter the name of the database to be written to.

1.5 Enter the name of the set to be written to.

Resending Mechanism

The resending mechanism is used to send the message again in case of a failure in the message forwarding process, which makes sure that the message is received. The details are as follows:

- If message forwarding fails, the system will retry forwarding at intervals of 1s, 3s, and 10s in sequence. If all three retries fail, the message will be discarded.
- If you have configured the "action for forwarding failure", then after three unsuccessful retries, the message will be forwarded again according to the configured action. If forwarding still fails, the message will be discarded.

Data Forwarding to Tencent CloudBase

Last updated：2025-03-19 14:49:59

Overview

The rule engine supports user-configurable forwarding rules to forward eligible device-reported data to Cloud Development Components. You can activate the development environment in the TCB Console. For specific steps, refer to Activating Environment.

Configuration

Follow the Operation Guide to create regulations and filter data, then add behavior operation, and choose to forward to CloudBase.

> **Note:**
> You will be prompted to authorize access to Cloud Development upon the first use. You need to click **Authorize Now** before you can proceed with creation.

In the pop-up "Add rule" window, select the "Data Forwarding to Cloud Development (CloudBase)" option, choose the region, existing environment, and function, and then click **Save**.

> **Note:**
> Currently, data can be forwarded only to functions in TCB. You need to create a development environment and function in TCB before you can select them here.

Resending Mechanism

The resending mechanism is used to send the message again in case of a failure in the message forwarding process, which makes sure that the message is received. The details are as follows:

- If message forwarding fails, the system will retry forwarding at intervals of 1s, 3s, and 10s in sequence. If all three retries fail, the message will be discarded.
- If you have configured the "action for forwarding failure", then after three unsuccessful retries, the message will be forwarded again according to the configured action. If forwarding still fails, the message will be discarded.

Data forwarding to TDSQL for MySQL

Last updated：2025-03-19 14:50:13

Overview

The rule engine allows you to configure forwarding rules to forward eligible data reported by devices to TDSQL for MySQL. After you create an instance and table in the tdsql console or through TencentCloud API, specified fields in device messages can be written to the corresponding TDSQL table.

Configuration

Create rules and filter data according to the operation guide, then add behavior operations and select "Forward to TDSQL for MySQL".

1. After successful authorization, in the pop-up "Add Rule" window, select "Data Forwarding to Cloud Database TDSQL-MySQL". You need to configure TDSQL-MySQL instance information and the field information to be written, as shown in the figure below:

2. Once the configuration is complete, click **Save**.

Configuration Instructions

The configuration is divided into the following steps:

1.1 Select a region and a TDSQL for MySQL instance.

1.2 Enter the username of the TDSQL for MySQL instance you just created.

1.3 Enter the login password of the instance.

1.4 Select the name of the database to be written to. If no database has been created under the created TDSQL for MySQL instance, go to the TDSQL for MySQL console to create one.

1.5 Select the table to be written to. If no table has been created under the created database, go to the TDSQL for MySQL console to create one.

1.6 Configure the fields to be written. There are two columns here: "field name" and "value".

- "Field name" corresponds to the field in the database table, indicating the field to be written.
- "Value" indicates the value of the field to be written. The value source can be a message body (which must be in JSON format to support value extraction) or a constant entered here.

> **Note:**
> - If the source is a message body, use `${}` to import the fields in the message body. If you want to specify a constant, just enter the corresponding value, such as 5 (number) or hello (string).
> - You must create the database, table, and fields in TDSQL for MySQL first before you can write data to the database.

Resending Mechanism

The resending mechanism is used to send the message again in case of a failure in the message forwarding process, which makes sure that the message is received. The details are as follows:

- If message forwarding fails, the system will retry forwarding at intervals of 1s, 3s, and 10s in sequence. If all three retries fail, the message will be discarded.
- If you have configured the "action for forwarding failure", then after three unsuccessful retries, the message will be forwarded again according to the configured action. If forwarding still fails, the message will be discarded.

Sub-account access to IoT
Creating Sub-account

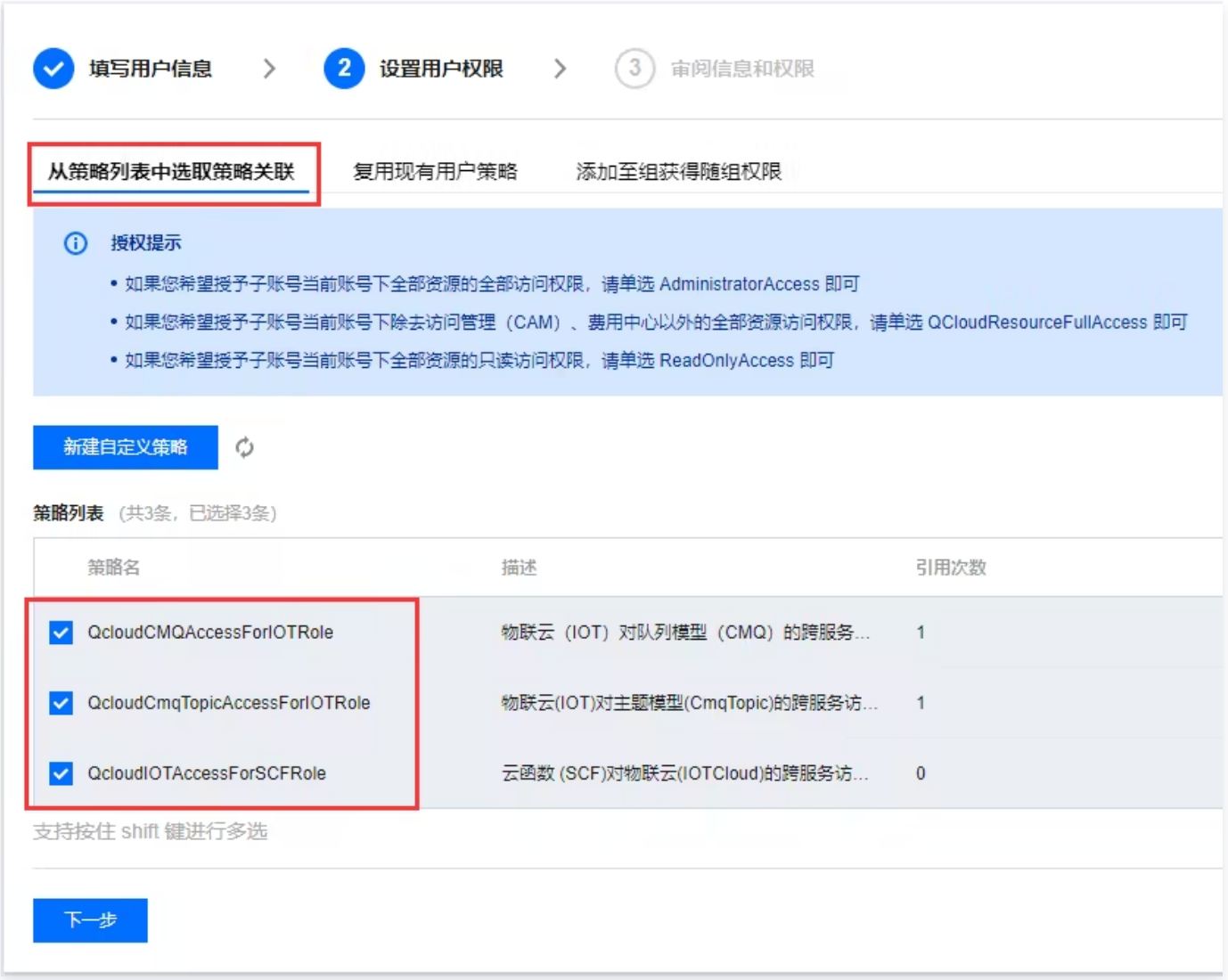
Last updated: 2025-03-19 14:50:36

Overview

This article will mainly introduce how to add a sub-account to the root account using "Collaborator" as an example.

Directions

1. Log in to the Tencent Cloud console with the **root account**. Select **Cloud Products** > **CAM** to enter the CAM console.
2. Select **Users** > **User List** from the left menu, then click **Create User** on the page.
3. A user type selection interface will pop up. Choose "Collaborator" or "Sub-user" to create.
4. Follow the instructions on the create Collaborator page, fill in the "username", "login account", "mobile phone", and "email information", then click **Next**.
5. Choose a user group for the newly created Collaborator. If there is no existing user group, click **Create User Group**.
6. Select "Associate policies from the policy list", search for IoT in the list, check the IoT-related policies shown in the figure, then click **Finish**.



After the collaborator is created, you can view the collaborator information in the user list.

Next steps

Click on the **Collaborator username** to enter the user information management interface, create an API key for the Collaborator, which is used for the Collaborator to access the root account's IoT resources through RestAPI.

Sub-account Permission Control

Last updated: 2025-03-19 14:50:52

Overview

This document mainly introduces how to grant product/device-level access control permissions to sub-accounts.

- Product-level access control permissions allow sub-accounts to manage access control for products they create or products created for them by the root account.
- Device-level access control permissions allow sub-accounts to manage access control only for devices created for them by the root account.

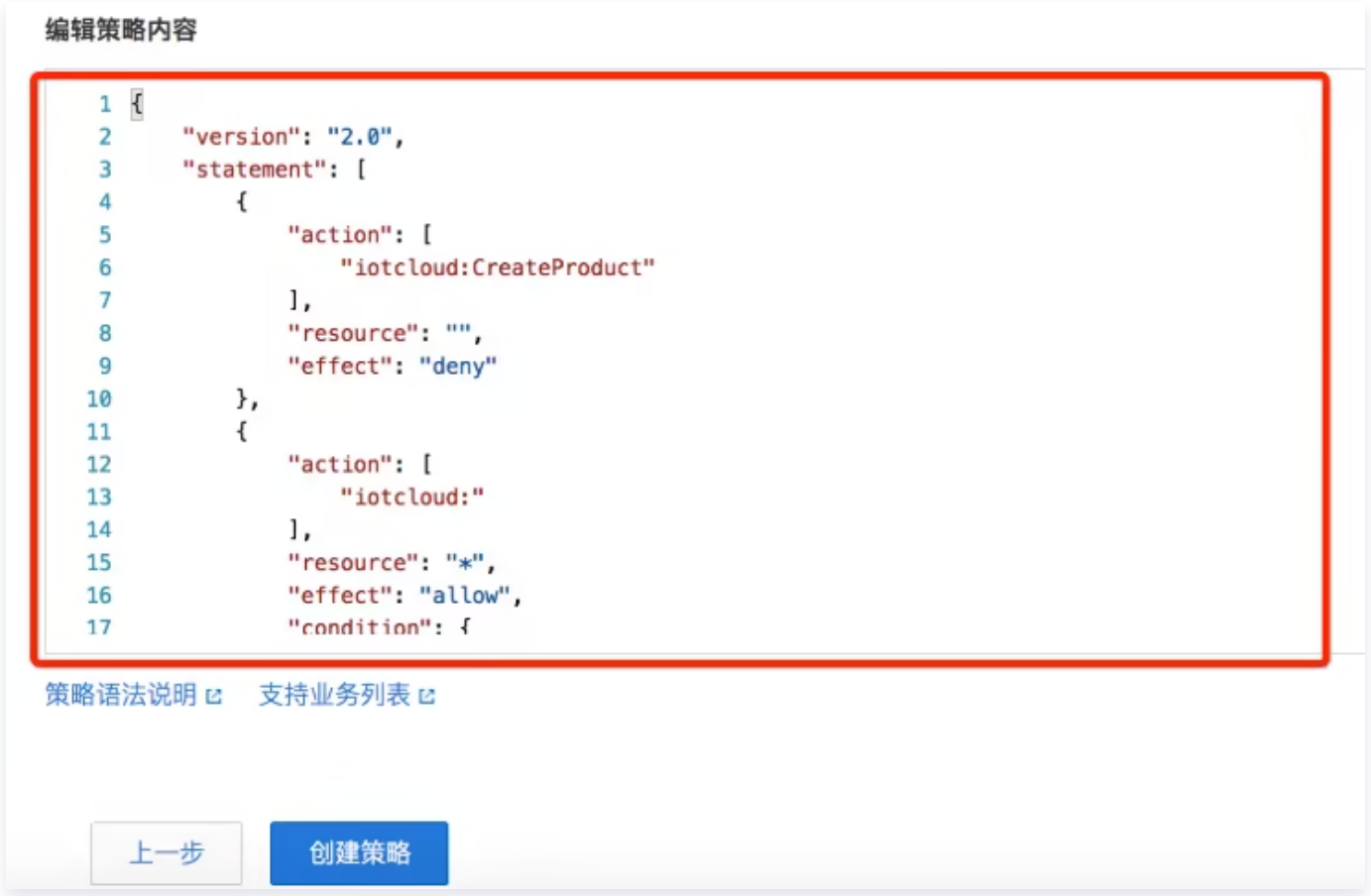
Granting permissions by policy syntax

CreatePolicy

1. Log in to the Tencent Cloud CAM console and click **Policies** in the left menu.
2. Enter the policy management page and click **Create a custom policy**.
3. On the page that pops up to select the policy creation method, choose **Creating by policy syntax**.

4. Select "Blank template" and click **Next**.

5. Fill in the custom policy name and edit the policy content according to the policy template. Example code is as follows:




```
{
    "version": "2.0",
    "statement": [
        {
            "action": [
                "iotcloud:CreateProduct"
            ]
            "resource": "*",
            "effect": "deny"
        }
        {
            "action": [
                "iotcloud:*"
            ]
            "resource": "*",
            "effect": "allow",
            "condition": {
```

```
                "string_equal_if_exist": {
                    "product": [
                        "${productID1}",
                        "${productID2}",
                        "${productID3}"

                    ]
                }
            }
        }
}
```

Association Strategy

1. After the custom policy is created, go to the list of user page.
2. Select the sub-account to which you want to grant permissions, and click **Associated Policy** in the "Permissions" column.
3. Search for the policy name you just created, select it, and click **OK** to complete granting the permissions defined in the policy.

Policy Description

- The following policy template prohibits the sub-account from creating product permissions. To prohibit other permissions for the sub-account, you can write the permission API name in the action, such as "iotcloud::DeleteDevice" to prohibit delete device permissions.

```
{
        "action": [
             "iotcloud:CreateProduct"
        ]
        "resource": "*",
        "effect": "deny"
 }
```

- The following policy template allows other permissions (create device, delete device, etc.). However, operations can only be performed under the specified product. Which product to open such permissions for depends on the PID filled in the product list. Users can replace ${productID*} with the productID of the internet of things product that needs authorization.

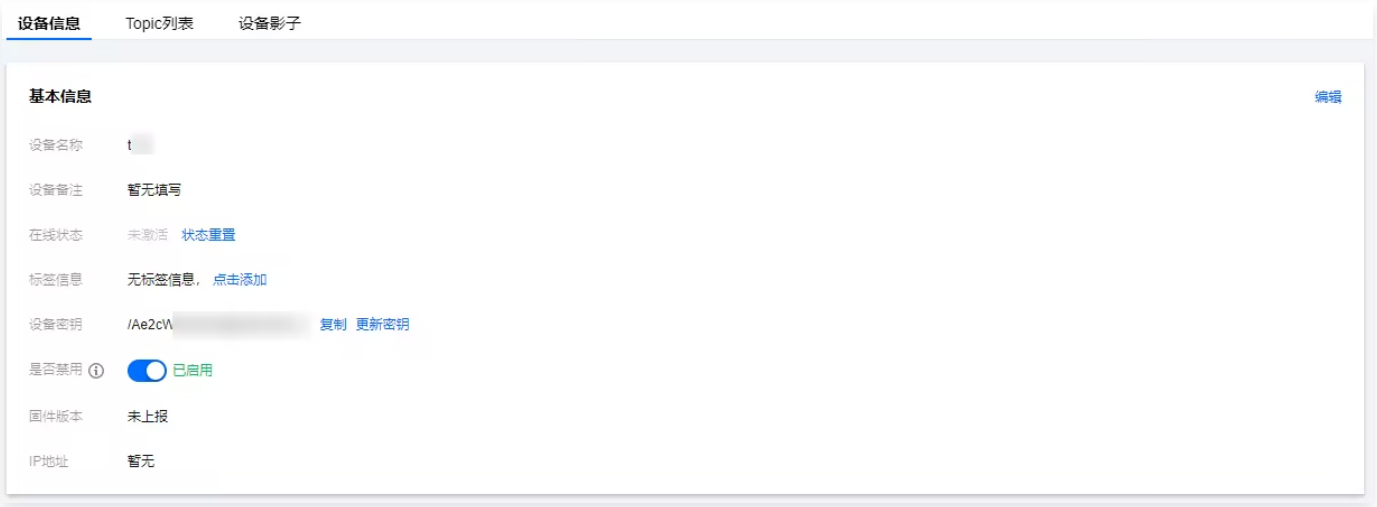
```
{
      "action": [
           "iotcloud:*"
       ]
       "resource": "*",
       "effect": "allow",
       "condition": {
            "string_equal_if_exist": {
                 "product": [
                      "${productID1}",
                      "${productID2}",
                      "${productID3}"

            }
      }
}
```

Now, you can obtain the basic information of the product through the IoT Hub console.

Tag-based authorization

Create device tags

1. Enter the IoT Hub console and click the target product name to enter the product information interface. If products and devices have not been added, you need to add them first. For detailed operation steps, please refer to Device Connection Preparations.



2. After clicking "Tag Information", click Add to fill in information such as key and value to add device tags.

- Tag key: Supports combinations of English letters, numbers, and underscores, up to 16 characters.
- Tag value: Supports combinations of English letters, numbers, and underscores, up to 16 characters.

3. After editing, click Confirm to complete the addition of "Tag Information". The corresponding tag content will be displayed in the device information.

Create policy and associated policy

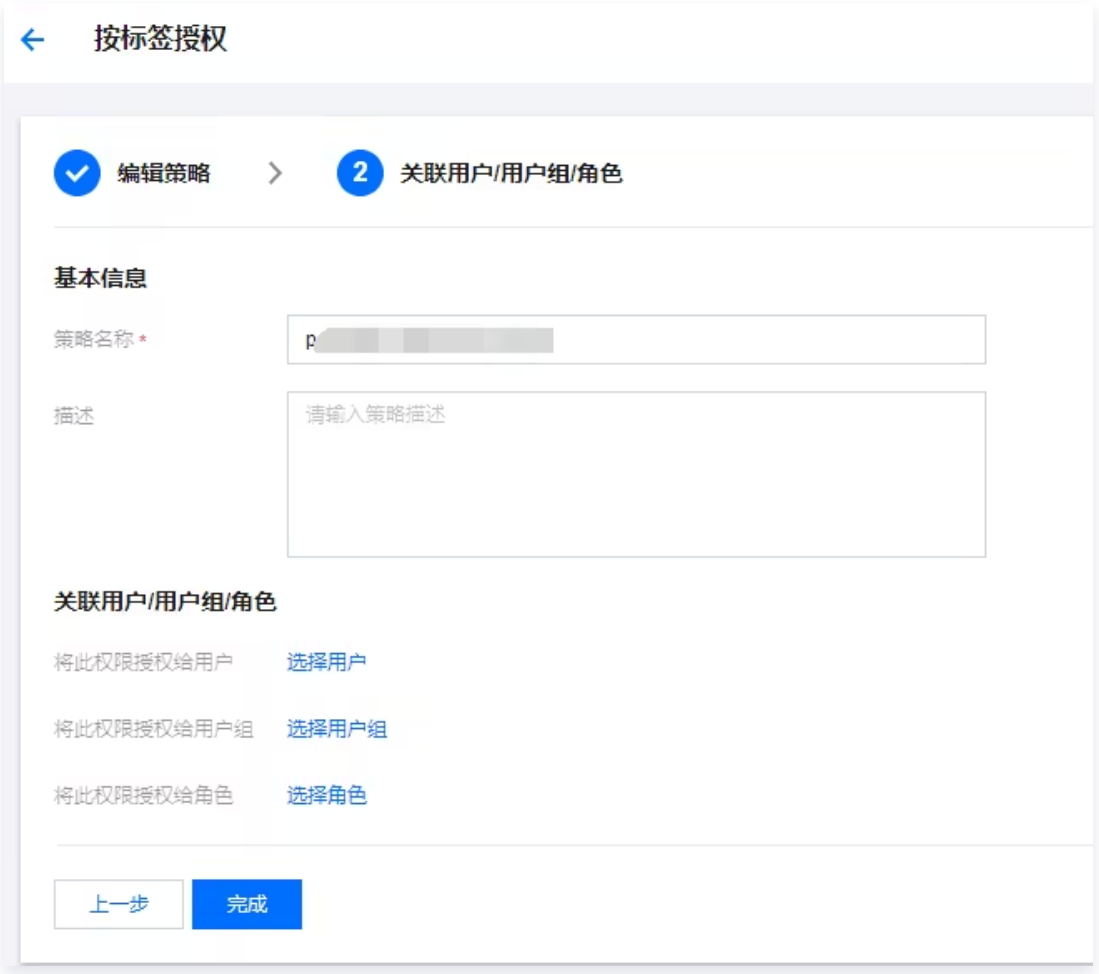
1. Log in to the Tencent Cloud CAM console and click **Policies** in the left menu.

2. Select Policies from the navigation bar and click Create a Custom Policy.
3. On the pop-up page for selecting the policy creation method, choose Tag-based Authorization.

4. Edit the policy, add services and operations on the visual strategy generator page, select tags, and click **JSON** to view the policy syntax content. After confirming, click **Next** to associate users/user groups/roles.

> **Note:**
> A single device can support multiple tags, and tag keys and tag values are not unique between devices. When selecting resources, you can select multiple tag keys and tag values, or choose a set of tag keys and tag values to allocate resources. A set of tag keys and tag values can allocate one or more device resources to a sub-user.

5. Select users or user groups or roles as needed, and click **Complete** to finish the tag-based authorization operation.



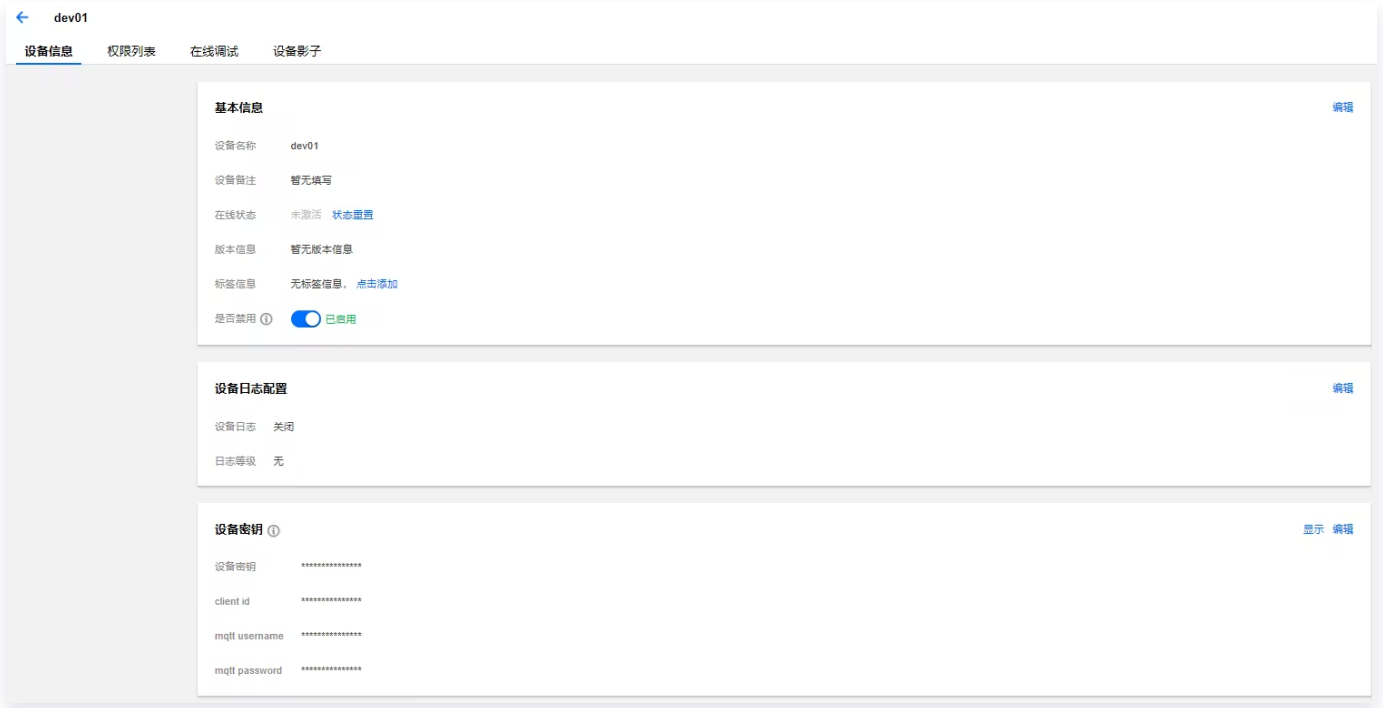
The policy name and policy information content can be changed. After confirming that everything is correct, click **Complete** to finish creating and associating the policy.

6. Due to the limitations of the IoT Hub console, after device resources are allocated to a sub-user, the sub-user needs to obtain product list and device list information to enter the device information interface and view the authorized device resources. Therefore, authorization for the product list and device list is required. You can complete the authorization of product list and device list information by creating policy syntax. The authorization code is as follows:

```
{
    "version": "2.0",
    "statement": [
        {
            "action": [
                "iotcloud:DescribeProducts",
                "iotcloud:DescribeDevices"
            ]
            "resource": "qcs::iotcloud:::ProductId/*",
            "effect": "allow"
```

```
            }


}
```

7. After the operation is completed, the authorized sub-user can perform management operations on the corresponding device resources in the console.



Unauthorized device resources will not be viewable.

Updating firmware

Last updated：2025-03-19 14:51:27

Overview

This document describes how to quickly use the firmware update service in IoT Hub.

Operation Steps

Adding firmware

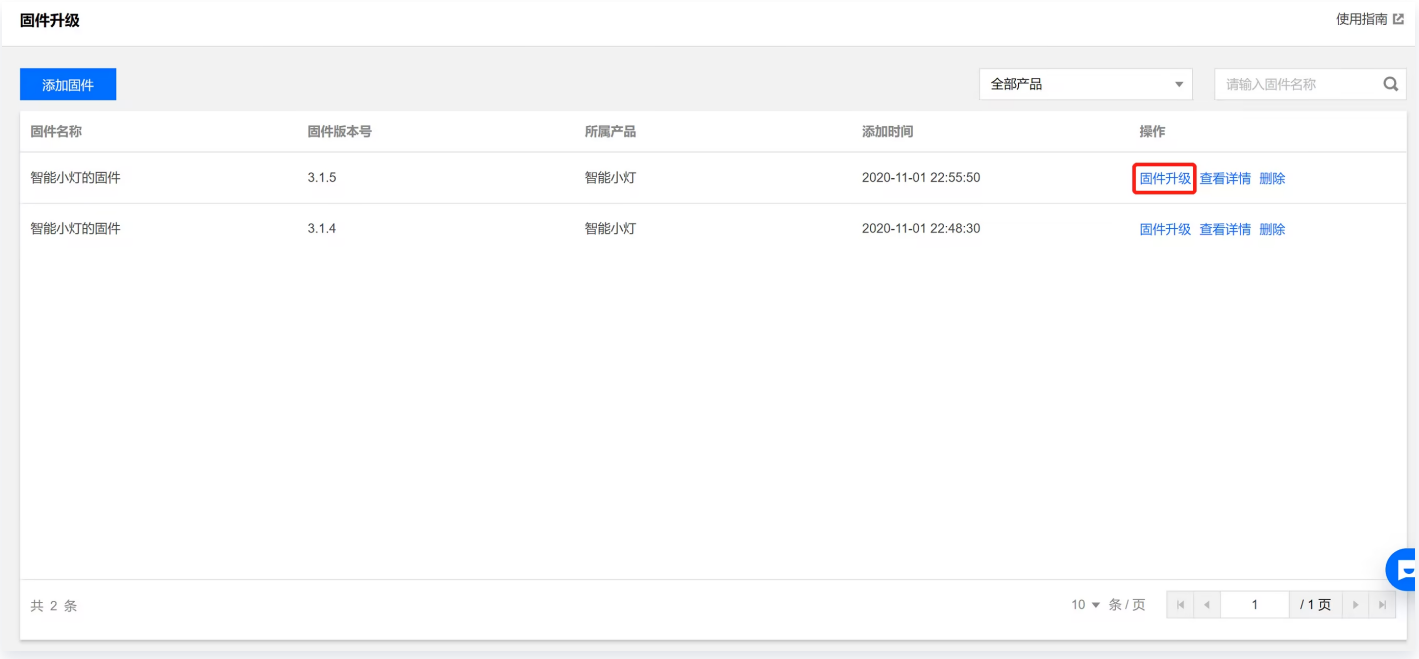
1. Log in to the Internet of Things Hub console and click **View details** on the upper right of the Overview module.
2. Click Left navigation**Firmware Upgrade** to enter the Firmware List Page, where all firmware in the current project can be viewed.
3. Click **Add Firmware** to add new firmware.

- Firmware Name: it can contain up to 32 letters, digits, underscores, minus signs, and parentheses and must begin with a letter or digit.
- Product: select the product of the firmware to be uploaded.
- Firmware Version Number: it can contain 1-32 letters, digits, dots, hyphens, and underscores.
- Firmware File: the firmware file to be uploaded must be a .bin file or a .tar, .gz, or .zip package and cannot exceed 1,024 MB in size.
- Firmware Description: it is the description and record of the firmware to be uploaded and can contain 0-100 characters.
- Up to 100 firmware files can be uploaded under each account. If you want to upload more, you need to delete some old firmware versions first.

4. After successful upload, the firmware will be displayed in the list. You can perform operations such as updating, adding, deleting, querying, and modifying the firmware file and viewing its details.

Updating firmware

Select the target firmware version you want to upgrade to, click the **Firmware Upgrade** on the right side of the firmware list to initiate the upgrade task. The firmware upgrade method supports two batch upgrades: by firmware version number and by device name.



Updating by firmware version number

1. Go to the firmware update page, which displays the information of the target firmware, such as firmware name, product, and version number.
2. Select **Batch Upgrade Method** as "Update by firmware version".

- Source Versions: select one or multiple firmware version numbers in the drop-down list for update.
- Upgrade Scope: Two upgrade scopes are supported. You can target either all devices under the selected firmware version number or specifically designated devices for the firmware upgrade. Designated equipment upgrade feature is commonly used for gray scale validation of firmware content. When selecting the upgrade scope as designated equipment, click the right side of the drop-down list **Select Device**, and you can batch

select target upgrade devices from all devices under the product.

3. After clicking **Save**, the system will execute the upgrade task and deliver the selected target version firmware to the target devices within the upgrade scope.

> **Note:**
> Update by firmware version requires the target devices to report their currently running firmware versions. If the versions are not reported, you can select updating by device name as described below.

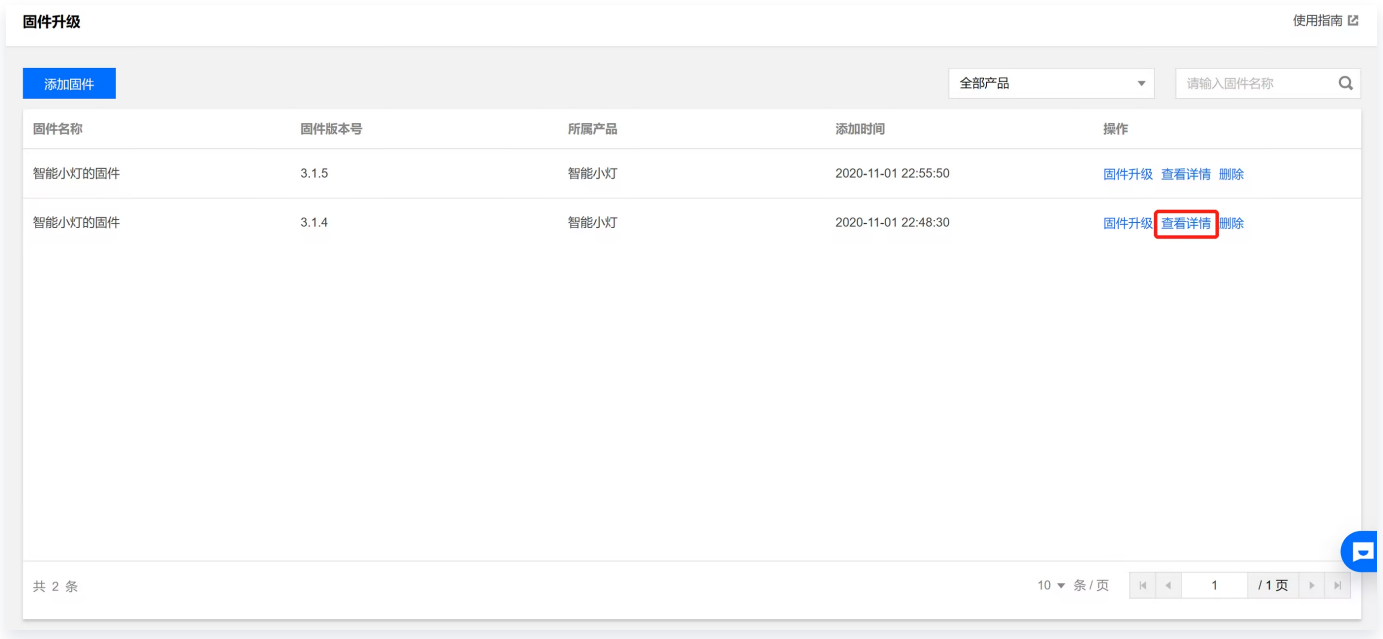
Updating by device name

1. Go to the firmware update page, which displays the information of the target firmware, such as firmware name, product, and version number.
2. Select **bulk upgrade method** as "Upgrade by device name".

3. Upload the specified device list that needs firmware updates. Click **Download Template** to get the template file, enter the exact DeviceName in the template file, and then click **Upload File** to upload. A maximum of 10,000 devices can be updated at a time. The file only supports csv format.
4. Click **Save**, and the system will execute the upgrade task, delivering the firmware to the target devices.

Viewing firmware details

1. In the firmware list, click **View Details** on the right side of the firmware list to view the firmware details.

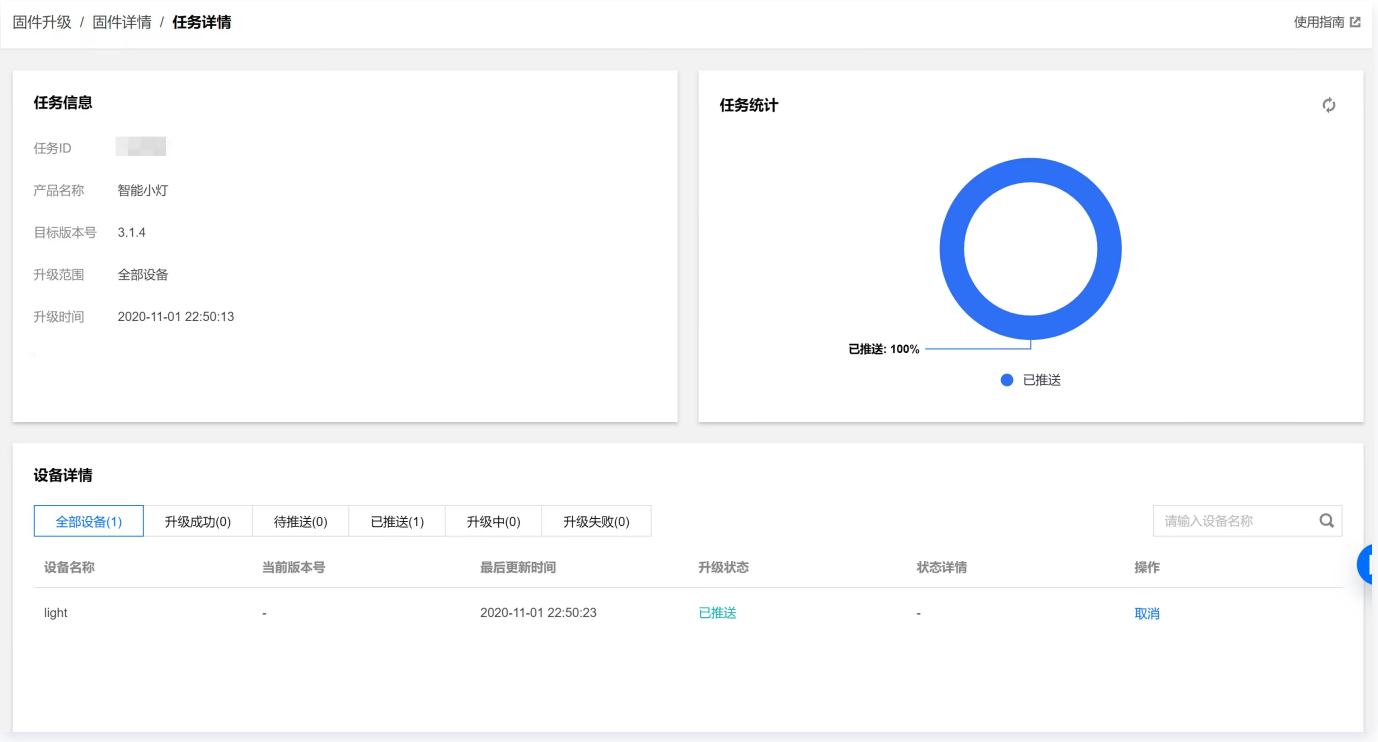


2. On the firmware details page, you can view the firmware details, statistics of devices updated to the firmware version, and update task management list.

- Firmware Information: It includes the firmware name, belongs to product, firmware version number, firmware signature, signature algorithm, add time, and firmware description. Click the edit button in the top right corner to modify the firmware name and description.
- Firmware Update Statistics: it includes the total number of devices in all batch update tasks of the firmware and the numbers of corresponding devices in firmware update tasks in different update status.
- Task management list:
  - Click **Task details** to view all upgrade tasks of the firmware. The task status of the upgrade task includes 4 types: not started, creating, created, and creation failed.

- Click **Device Details** to view the record details of device upgrades in all upgrade tasks associated with the firmware. The device upgrade status includes 5 types: pending push, pushed, upgrading, upgrade successful, and upgrade failure.

3. In the **Task details** or **Device Details** of task management, click the **View Details** on the right side of a task to enter the task detail page, where you can view the list of devices upgraded in this task, the upgrade status, and the statistics of device numbers in different upgrade status.



In the device details list, you can view the current upgrade status and status details of all devices in the batch upgrade of the task:

- When the upgrade status is "pending push" and "pushed", the status details are not displayed.
- When the upgrade status is "upgrading", the status details include: downloading, burning, and the progress percentage is displayed.

- When the upgrade status is "upgrade failure", the status details will provide error information.

Additionally, on the right side of the device details list, you can cancel or retry device upgrades based on the upgrade progress. The device upgrade status of the canceled upgrade will be marked as upgrade failure; you can click **Retry** to re-upgrade the device that failed to upgrade.