ALIBABA CLOUD

Alibaba Cloud

物联网平台

Maintenance

Document Version: 20220622

Alibaba Cloud

Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.

2. No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company or individual in any form or by any means without the prior written consent of Alibaba Cloud.

3. The content of this document may be changed because of product version upgrade, adjustment, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and an updated version of this document will be released through Alibaba Cloud-authorized channels from time to time. You should pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.

4. This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides this document based on the "status quo", "being defective", and "existing functions" of its products and services. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity, applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not take legal responsibility for any errors or lost profits incurred by any organization, company, or individual arising from download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, take responsibility for any indirect, consequential, punitive, contingent, special, or punitive damages, including lost profits arising from the use or trust in this document (even if Alibaba Cloud has been notified of the possibility of such a loss).

5. By law, all the contents in Alibaba Cloud documents, including but not limited to pictures, architecture design, page layout, and text description, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of this document shall be used, modified, reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates. The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates.

6. Please directly contact Alibaba Cloud for any errors of this document.

Document conventions

Style	Description	Example
Danger	A danger notice indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.	**Danger:** Resetting will result in the loss of user configuration data.
Warning	A warning notice indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.	**Warning:** Restarting will cause business interruption. About 10 minutes are required to restart an instance.
Notice	A caution notice indicates warning information, supplementary instructions, and other content that the user must understand.	**Notice:** If the weight is set to 0, the server no longer receives new requests.
Note	A note indicates supplemental instructions, best practices, tips, and other content.	**Note:** You can use Ctrl + A to select all files.
>	Closing angle brackets are used to indicate a multi-level menu cascade.	Click **Settings**> **Network**> **Set network type**.
Bold	Bold formatting is used for buttons , menus, page names, and other UI elements.	Click **OK**.
`Courier font`	Courier font is used for commands	Run the `cd /d C:/window` command to enter the Windows system folder.
Italic	Italic formatting is used for parameters and variables.	`bae log list --instanceid` *Instance_ID*
[] or [a|b]	This format is used for an optional value, where only one item can be selected.	`ipconfig [-all|-t]`
{} or {a|b}	This format is used for a required value, where only one item can be selected.	`switch {active|stand}`

Table of Contents

1.Real-time monitoring ---- 05
1.1. What is real-time monitoring? ---- 05
1.2. View data metrics ---- 06
1.3. View the network status of a device ---- 07
1.4. Use CloudMonitor to monitor resource usage ---- 08
1.4.1. Configure alert rules ---- 09
1.4.2. Alert notifications ---- 12
2.Online debugging ---- 17
3.Device simulation ---- 20
4.Log Service ---- 25
4.1. IoT Platform logs ---- 25
4.2. Local device logs ---- 50
4.3. Dump IoT Platform logs ---- 52
4.4. Dump local device logs ---- 55
5.OTA update ---- 58
5.1. Overview ---- 58
5.2. Push an update package to devices ---- 61
5.2.1. Add an update package ---- 61
5.2.2. Verify an update package (Optional) ---- 65
5.2.3. Initiate a batch update ---- 69
5.2.4. View update status ---- 80
5.2.5. View statistics on update package versions and success.. ---- 84
5.3. Perform OTA updates ---- 86
5.4. Update sub-devices by using OTA ---- 98
6.Remote configuration ---- 100

1.Real-time monitoring

1.1. What is real-time monitoring?

IoT Platform provides real-time monitoring on metrics such as the number of online devices, number of upstream and downstream messages, number of messages forwarded by the rules engine, and device network status. In addition, you can monitor the IoT Platform data by setting alert rules in Cloud Monitor.

Use IoT Platform to monitor resource usage

IoT Platform monitors device data and network status under your Alibaba Cloud account in real time. The monitoring data is displayed on the **Real-time Monitoring** page.

- Device data metrics

  The following metrics are displayed in the console: Online Devices, Messages Sent to IoT Platform, Messages Sent from IoT Platform, and Messages Forwarded Through Rule Engine.

  For more information, see View data metrics.

- Device network status

  You can enable the devices whose network connection method is Wi-Fi to submit the network status data. After a device submits data, you can view the network status and network error messages of the device on the **Device Network status** tab of the **Real-time Monitoring** page. You can specify a device name and time range for a query.

  The displayed network status data of devices includes the signal collection time, received signal strength (RRS), signal-to-noise ratio (SNR), and packet loss rate. For more information, see View the network status of a device.

  For more information about topics that are used to submit network status data, format of network status data, and network errors, see Devices submit network status data.

  > **Note** Only devices whose network connection method is Wi-Fi can submit network status data.

Use Cloud Monitor to monitor resource usage

IoT Platform is integrated with Cloud Monitor. You can use the Event Alarm and Threshold Value Alarm features of Cloud Monitor to monitor the IoT Platform data. The metrics include OnlineDeviceCount, MessageCountSentFromIoT, MessageCountSentToIoT, MessageCountForwardedThroughRuleEngine, MessageCountPerMinute, DeviceEventReportError, DevicePropertyReportError, and DeviceServiceCallError.

On the **Real-time Monitoring** page, click **Alarm Settings** to go to the CloudMonitor console. Configure threshold-triggered and event-triggered alert rules as required.

For more information about alert rules and alert messages, see Configure alert rules and Alert notifications.

Authorize RAM users to use real-time monitoring

To authorize RAM users to use the real-time monitoring feature, you must log on to the RAM console and grant the `cms:QueryMetricList` permission to the RAM users. For more information about how to customize a RAM permission policy, see Custom permissions.

The following script shows the cms:QueryMetricList permission policy.

```
{
    "Statement": [
        {
            "Action": [
                "cms:QueryMetricList"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ],
    "Version": "1"
}
```

1.2. View data metrics

Go to the Real-time Monitoring page of the IoT Platform console. On the Overview tab, you can view the number of online devices, the number of upstream and downstream messages, and the number of messages that are forwarded by the rules engine.

View data metrics

1. Log on to the IoT Platform console.
2.
3. In the left-side navigation pane, choose **Maintenance > Real-time Monitoring**.
4. On the **Overview** tab, select a product and a time range that you want to query. You can specify 1 hour, 1 day, 1 week, or a custom time range within 7 days.

   > **Note** Delays exist during data collection.

Data metric description

The following table describes the data metrics that are displayed on the **Overview** tab.

Metric	Description
Online Devices	The number of devices that built persistent connections with IoT Platform. The data is displayed based on the types of protocols that are used to communicate with IoT Platform.

Metric	Description
Messages Sent to IoT Platform	The number of messages that devices send to IoT Platform. The data is displayed based on the types of protocols that are used to communicate with IoT Platform.
Messages Sent from IoT Platform	The number of messages that are sent from IoT Platform to devices and servers. The data is displayed based on the types of protocols that are used to connect to IoT Platform.
Messages Forwarded Through Rules Engine	The number of messages that are forwarded by the rules engine. The data is displayed based on the target cloud services to which messages are forwarded.

References

- CloudMonitor

  You can use CloudMonitor to monitor and configure alerts for the preceding data metrics.

  For more information, see the following topics:

  - Configure alert rules
  - Alert notifications

- Device network status

  You can enable the devices that use Wi-Fi to connect to networks to submit the network status data.

  For more information, see the following topics:

  - Devices submit network status data
  - View the network status of a device

1.3. View the network status of a device

IoT Platform allows you to check the network status of each device. A device that accesses a network by using Wi-Fi can send the network status data to IoT Platform by using a specified topic. This topic describes how to view the network status of a device in the IoT Platform console.

Context

For more information about device topics, data formats of Alink, and errors that are reported, see Device network status.

If your device uses AliOS Things V3.0 or later, the device automatically checks and sends network status data. For more information about network errors, see the err_stats table in Device network status.

Procedure

1. Log on to the IoT Platform console.
2.
3. In the left-side navigation pane, choose **Maintenance > Real-time Monitoring**.
4. On the **Real-time Monitoring** page, click **Device Network Status**.
5. Select a device and a time range.

   The network status data of the selected device and time range is displayed. Descriptions of the network status of the device

Field	Description
Reported At	The time when IoT Platform received the network status data.
Collected At	The time when the device collected the network status data. **Note** The device immediately sends the network status data to IoT Platform after a network error occurs or when the device collects data at a scheduled point in time. In other scenarios, the device may not immediately send the network status data after the device collects the data. If the data that the device sends to IoT Platform does not include a timestamp, no time is displayed.
RSSI (dBm)	The strength of the signal that is received.
SNR (dB)	The signal-to-noise ratio of the wireless signal.
Wireless Signal Packet Loss Ratio (‰)	The packet loss rate.
Network Connection Method	The method that is used to establish a network connection to the device. Only devices that can be connected by using Wi-Fi are supported.
Error Description	Click **View Details** to view the details of errors and the number of times that each error occurs. **Note** The **View Details** button appears only when network errors are reported by the device. For more information, see Device network status.

1.4. Use CloudMonitor to monitor resource usage

1.4.1. Configure alert rules

IoT Platform supports the CloudMonitor service. You can configure alert rules to monitor the resource usage of IoT Platform and receive alert notifications after the rules are triggered. This article describes how to configure an alert rule.

Features

Different alert rules support different metrics. You can configure an alert rule based on the following table.

Feature	Metrics	Description
Threshold-triggered alerts	If you set **Product** to **IoT Platform** when you create an alert rule, the following metrics are supported: • DeviceEventReportError, DevicePropertyReportError, DevicePropertySettingError, and DeviceServiceCallError • MessageCountForwardedThroughRuleEngine_REPUBLISH, MessageCountForwardedThroughRuleEngine_DATAHUB, MessageCountForwardedThroughRuleEngine_FC, MessageCountForwardedThroughRuleEngine_MNS, MessageCountForwardedThroughRuleEngine_MQ, MessageCountForwardedThroughRuleEngine_OTS, MessageCountForwardedThroughRuleEngine_RDS, and MessageCountForwardedThroughRuleEngine_TSDB • MessageCountSentFromIoT_MQTT and MessageCountSentFromIoT_LoRa • MessageCountSentToIoT_MQTT, MessageCountSentToIoT_CoAP, MessageCountSentToIoT_HTTP, and MessageCountSentToIoT_LoRa • OnlineDevicesCount_MQTT • MessageCountPerMinute • RuleEningeTransmitCountPerMinute • DeviceCount_Product	If the resource usage of IoT Platform or the number of failed operations exceeds a specified threshold within a specified time period, CloudMonitor sends an alert notification based on the alert rule. For more information, see Alert notifications.
	If you set **Product** to **IoT Platform-Instance** when you create an alert rule, the following metrics are supported: • DeviceNum_instance • MessageWatermarkTps_instance • RuleEngineWatermarkTps_instance	

Feature	Metrics	Description
	If you set **Product** to **IoT Platform-Server subscription** when you create an alert rule, the following metrics are supported: • AMQP_Msg_Accumulate • AMQP_Msg_Consume_rate	
Event-triggered alerts	• Device_Connect_QPM_Limit • Device_Uplink_QPS_Limit • Device_Downlink_QPS_Limit • Account_Connect_QPS_Limit • Account_Uplink_QPS_Limit • Account_Downlink_QPS_Limit • Account_RuleEngine_DataForward_QPS_Limit	

Create a threshold-triggered alert rule

1. Log on to the IoT Platform console.
2.
3. In the left-side navigation pane, choose **Maintenance > Real-time Monitoring**.
4. On the **Real-time Monitoring** page, click **Alert Settings**.
5. On the **Create Alert Rule** page, set the parameters and then click **Confirm**.

   The following table describes the parameters in the **Related Resource** section. You can set other parameters based on actual scenarios.

   For more information, see Create a threshold-triggered alert rule.

Parameter	Description
Product	Select **IoT Platform**, **IoT Platform-Instance**, or **IoT Platform-Server subscription**.
Resource Range	Valid values: ◦ All Resources ▪ If **Product** is set to **IoT Platform** or **IoT Platform-Instance**, all IoT Platform instances are included. ▪ If **Product** is set to **IoT Platform-Server subscription**, all consumer groups in IoT Platform instances are included. ◦ Instance: An alert notification is sent only if a specified product or consumer group in a specified instance matches the alert rule.
Region	This parameter is available if Resource Range is set to **Instance**. This parameter specifies the region where the IoT Platform instance resides.
Instance	Select one or more IoT Platform instances and products to be monitored.

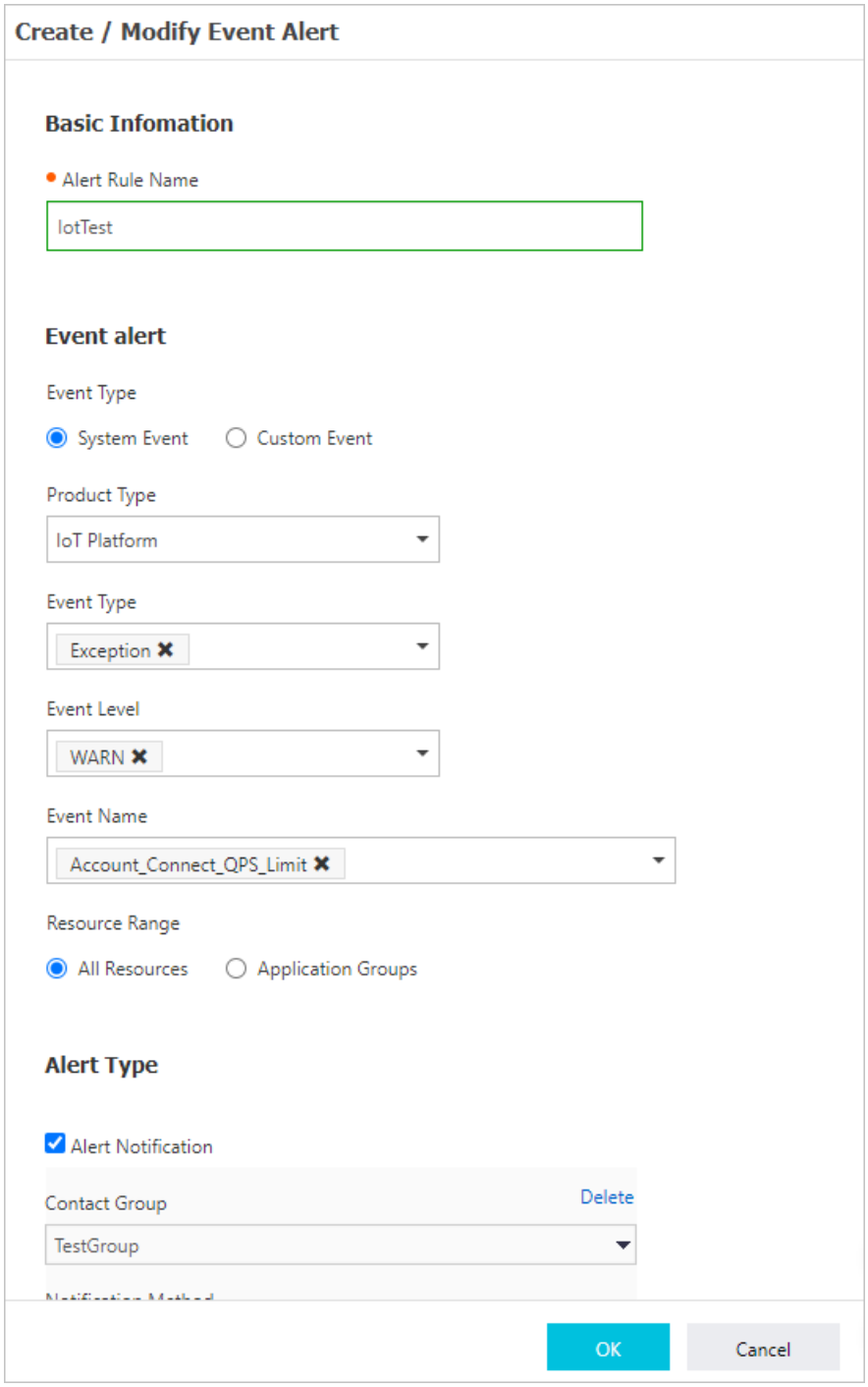
Parameter	Description
Consumer Group	The parameter is available if **Product** is set to **IoT Platform-Server subscription** and Resource Range is set to **Instance**. Select one or more consumer groups to be monitored. An alert notification is sent only if the number of accumulated messages or the consumption rate of a consumer group exceeds a specified threshold.

Create an event-triggered alert rule

1. Log on to the IoT Platform console.In the left-side navigation pane, choose **Maintenance > Real-time Monitoring**.
2. On the **Real-time Monitoring** page, click **Alert Settings**.
3. On the **Create Alert Rule** page, click **View the Detail** in the Set Alert Rules section.
4. On the **Event Monitoring** page, click **Create Event Alert**. In the Create / Modify Event Alert panel, configure the alert rule and click **OK**.

   Set **Product Type** to **IoT Platform** and set other parameters as needed.

   For more information, see Create an event-triggered alert rule.

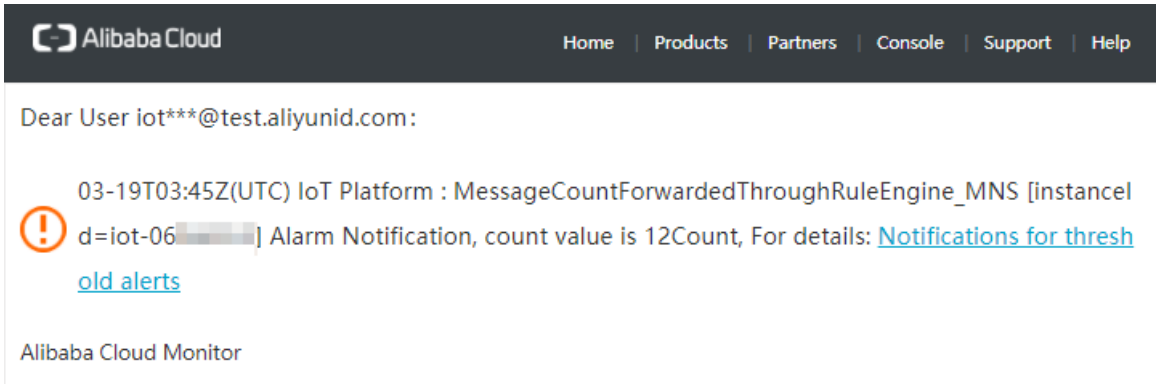


1.4.2. Alert notifications

If the resource usage of IoT Platform reaches the specified threshold in an alert rule, an alert is triggered. Then, Alibaba Cloud sends an alert notification to the specified alert group.

Notifications for threshold-triggered alerts

If an alert is triggered based on a threshold-triggered alert rule, alert contacts receive an alert notification email, as shown in the following figure.



Fields of an alert notification email

Field	Description
IoT Platform instance	The information about the instance that triggers the alert. The information contains the ProductKey (productKey), instance ID (instanceId), and region ID (regionId).
Metric	The code of the metric that you selected when you set the Rule Description parameter. In this example, the code MessageCountForwardedThroughRuleEngine_MNS indicates the number of messages that are forwarded by the rules engine. If the number of messages exceeds the specified threshold within a specified period of time, an alert is triggered. For more information about the codes of metrics, see Metric codes and descriptions.
Alert time	The time when the alert is triggered.
Count	The total number of messages, the number of forwarded messages, or the number of connected devices that are counted for the specified metric.
Duration	The period of time for which the alert exists.
Rule details	The details of the alert rule that you configured in the CloudMonitor console.

Metric codes and descriptions

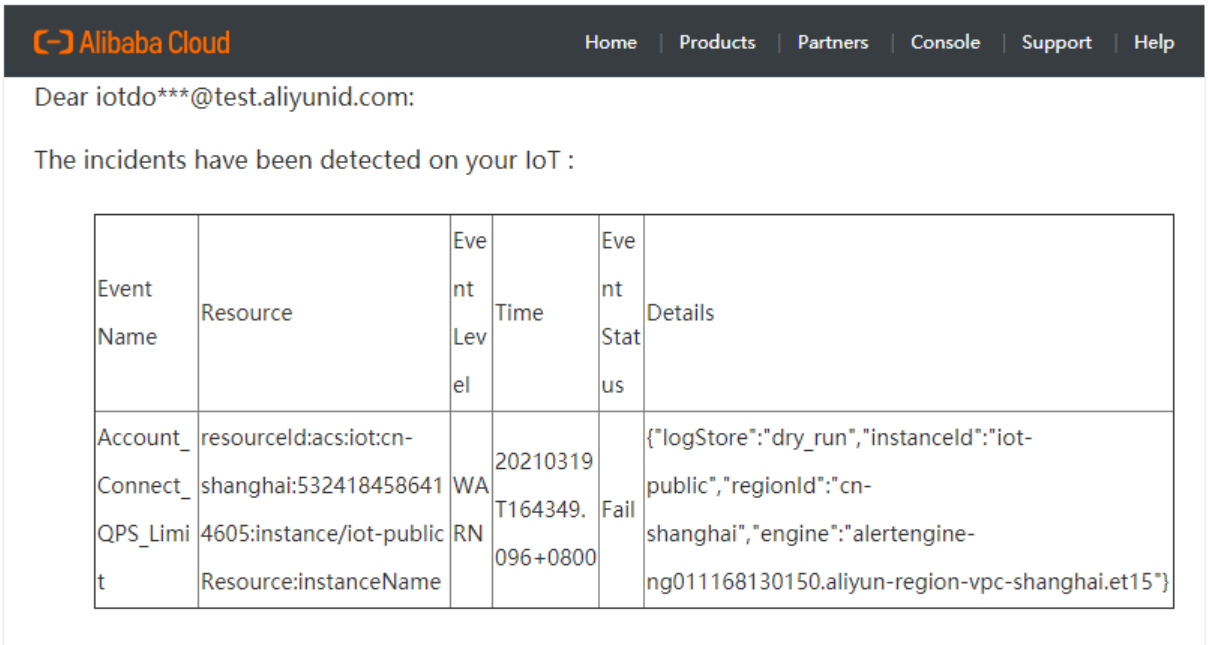
Code	Description
MessageCountForwardedThroughRuleEngine_FC	The number of messages that are forwarded by the rules engine. The number is the same as the number of times that the rules engine forwards data to Function Compute.
MessageCountForwardedThroughRuleEngine_MNS	The number of messages that are forwarded by the rules engine. The number is the same as the number of times that the rules engine forwards data to Message Service (MNS).

Code	Description
MessageCountForwardedThroughRuleEngine_OTS	The number of messages that are forwarded by the rules engine. The number is the same as the number of times that the rules engine forwards data to Tablestore.
MessageCountForwardedThroughRuleEngine_RDS	The number of messages that are forwarded by the rules engine. The number is the same as the number of times that the rules engine forwards data to ApsaraDB RDS.
MessageCountForwardedThroughRuleEngine_REPUBLISH	The number of messages that are forwarded by the rules engine. The number is equal to number of times that the rules engine forwards data from the current topic to other topics.
MessageCountSentFromIoT_HTTP_2	The number of messages that are sent from IoT Platform by using HTTP/2.
MessageCountSentFromIoT_MQTT	The number of messages that are sent from IoT Platform by using Message Queuing Telemetry Transport (MQTT).
MessageCountSentToIoT_CoAP	The number of messages that are sent to IoT Platform by using Constrained Application Protocol (CoAP).
MessageCountSentToIoT_HTTP	The number of messages that are sent to IoT Platform by using HTTP.
MessageCountSentToIoT_HTTP/2	The number of messages that are sent to IoT Platform by using HTTP/2.
MessageCountSentToIoT_MQTT	The number of messages that are sent to IoT Platform by using MQTT.
OnlineDevicesCount_MQTT	The number of devices that are connected to IoT Platform by using MQTT in real time.
DeviceEventReportError	The number of event reporting failures.
DevicePropertyReportError	The number of property reporting failures.
DevicePropertySettingError	The number of property setting failures.
DeviceServiceCallError	The number of service invocation failures.
DeviceNum_instance	The percentage of online devices that are connected to IoT Platform. The value is calculated by using the following formula: `Number of online devices that are connected to IoT Platform/Number of concurrently online devices supported by the current instance × 100%`.

Code	Description
MessageWatermarkTps_instance	The value of the metric is a percentage. The value is calculated by using the following formula: `Current TPS at which upstream and downstream messages are consumed/TPS at which upstream and downstream messages can be consumed under the current instance × 100%`.
RuleEngineWatermarkTps_instance	The value of the metric is a percentage. The value is calculated by using the following formula: `Current TPS at which the rules engine forwards messages/TPS at which the rules engine can forward messages under the current instance × 100%`.
AMQP_Msg_Accumulate	The number of accumulated messages.
AMQP_Msg_Consume_rate	The message consumption rate.

Notifications for event alerts

If an alert is triggered based on an event-triggered alert rule, alert contacts receive an alert notification email, as shown in the following figure.



Fields of an alert notification email

Field	Description
Event Name	The code of the event that triggers the alert. In this example, the code Device_Connect_QPM_Limit specifies the number of connection requests that are sent per minute by a device reaches the upper limit. For more information about the codes of events, see Event codes.

Field	Description
Resource	The resource that triggers the alert. • resourceId: the ID of the resource. Format: `acs:iot:$regionid::instance/$instanceId/product/$productKey/device/$deviceName` • Resource name: the ID of the instance. iot-public indicates a public instance. • Group ID: the ID of the group to which the device belongs. If the device does not belong to a group, the value of the field is an empty string.
Event Level	All events trigger WARN-level alerts.
Time	The time when the event occurs.
Event Status	All events are in the Fail state. This state indicates that the subsequent request failed because the number of connection requests that are sent per minute or the number of messages that are sent per second reaches the upper limit.
Details	The information about the resource that triggers the alert. The information is in the JSON format. The information contains the region ID (regionId), instance ID (instanceId), ProductKey (productKey), and DeviceName (deviceName). The productKey and deviceName parameters are included in the notification only when the number of connection requests that are sent per minute, the number of messages that are sent per second, or the number of messages that are received per second by a device reaches the upper limit.

Event codes

Code	Description
Device_Connect_QPM_Limit	The number of connection requests that are sent per minute by a device reaches the upper limit.
Device_Uplink_QPS_Limit	The number of messages that are sent per second by a device reaches the upper limit.
Device_Downlink_QPS_Limit	The number of messages that are received per second by a device reaches the upper limit.
Account_Connect_QPS_Limit	The number of connection requests that the current account sends to IoT Platform per second reaches the upper limit.
Account_Uplink_QPS_Limit	The number of messages that are sent per second by the current account reaches the upper limit.
Account_Downlink_QPS_Limit	The number of messages that are received per second by the current account reaches the upper limit.
Account_RuleEngine_DataForward_QPS_Limit	The number of messages that are forwarded per second by the rules engine for the current account reaches the upper limit.

2.Online debugging

After you configure a physical device in IoT Platform console, you can debug features of the device by pushing commands to the device from the IoT Platform console. This topic describes the online debugging procedure.

Prerequisites

The device is created in the IoT Platform console and connected to IoT Platform. For more information about how to configure and connect devices to IoT Platform, see What is Link SDK?.

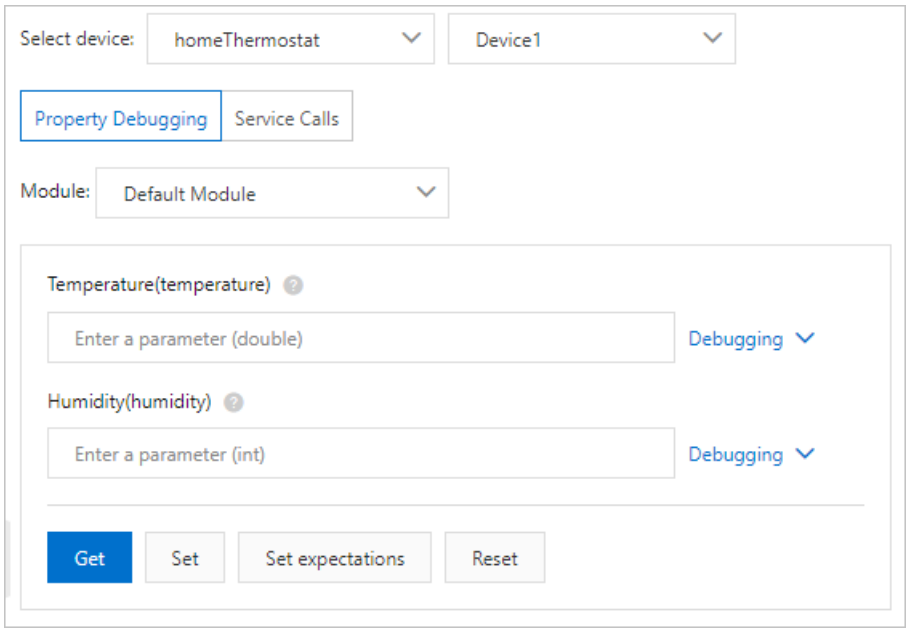
> **Note** You can also use the device simulator or MQTT.fx to simulate a device and connect the simulated device to IoT platform, and then debug the device online. For more information, see Device simulation or Connect a device to IoT Platform by using MQTT.fx.
>
> After the simulated device is connected, device topics are generated. For more information about how to subscribe to topics and use topics, see Generate topics.

Limits

You can debug only devices that are connected over MQTT.

Debug features

1. Log on to the IoT Platform console.
2.
3. In the left-side navigation pane, choose **Maintenance > Online Debug**.
4. On the **Online Debug** page, select the device that you want to debug.
5. Click the **Property Debugging** or **Service Calls** tab. On the tab, select a Thing Specification Language (TSL) model from the **Module** drop-down list.



Type	Procedure
Property Debugging	**Notice** You must have the read and write permissions on the property that you want to manage. The available operations include **Set** and **Set expectations**. ○ **Get**: obtains the latest value of a specified property from the device. If no value has been set for the property, no value is displayed in the field. On the right side of a property field, click **Debugging**. Then, click **Get**. ○ **Set**: sends a command from IoT Platform to the device to set a property value. After the device receives the command, the device sets the property value in the TSL model based on the command, and submits the latest property value to IoT Platform. Enter a value in a property field and click **Debugging**. Then, click **Set**. ○ **Set expectations**: sends a command from IoT Platform to the device to set a desired property value. Enter a value in the property field and click **Debugging**. Then, click **Set expectations**. When the command is sent: ■ If the device is online, the device receives the command, updates the property value, and then submits the new property value to IoT Platform. ■ If the device is offline, the device obtains the desired property value after it goes online, updates the property value, and then submits the new property value to IoT Platform. You can also click **Get**, **Set**, or **Set expectations** below all property fields to debug multiple properties.
Service Calls	i. Select the service that you want to debug from the **Debug Feature** drop-down list. ii. In the field, enter the input parameters of the service. Then, click **Send Command**. The input parameters must be in the JSON format. Example: `{"Switch":0}`.

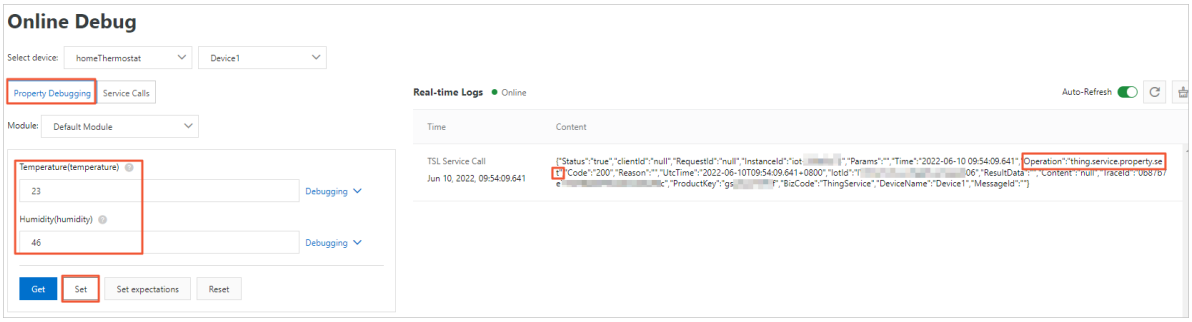
View debugging logs

After you push the command, you can view the operation logs in the **Real-time Logs** section on the right side of the page. You can also view the debugging results on the **TSL Data** tab of the Device Details page.

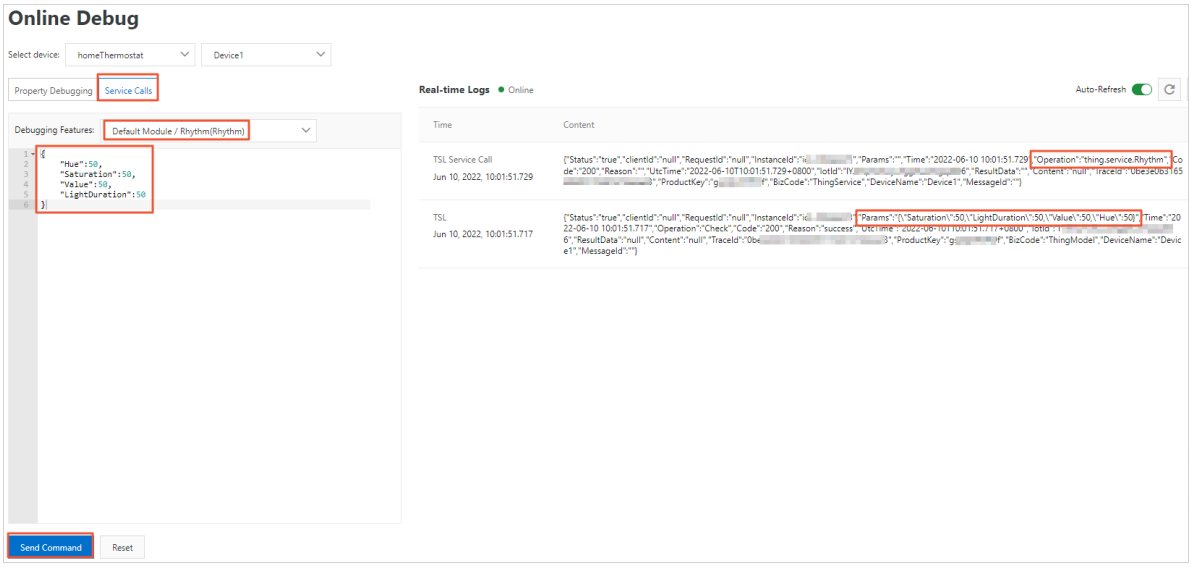
Notice During online debugging, IoT Platform uses the Revert-Remote Procedure Call (RRPC) synchronous communication mechanism to send commands to devices. This allows online simulated devices to receive debugging logs when the devices are not subscribed to topics. You can simulate devices by using tools such as MQTT.fx. You can choose **Maintenance > Device Log** to view detailed logs.

The following figures show how to view the debugging logs.

- Debug properties: Enter a property value, click **Debugging** on the right, and then click **Set**. The debugging logs are displayed in the Real-time Logs section on the right.



- Debug service calls: Select a service, enter the input parameters of the service, and then click **Send Command**. The debugging logs are displayed in the Real-time Logs section on the right.



3.Device simulation

IoT Platform provides the device simulator to simulate the connection between a physical device and IoT Platform. You can use simulated data to test the communication between the device and IoT Platform and identify problems.

Description

You can use the device simulator to debug the following features:

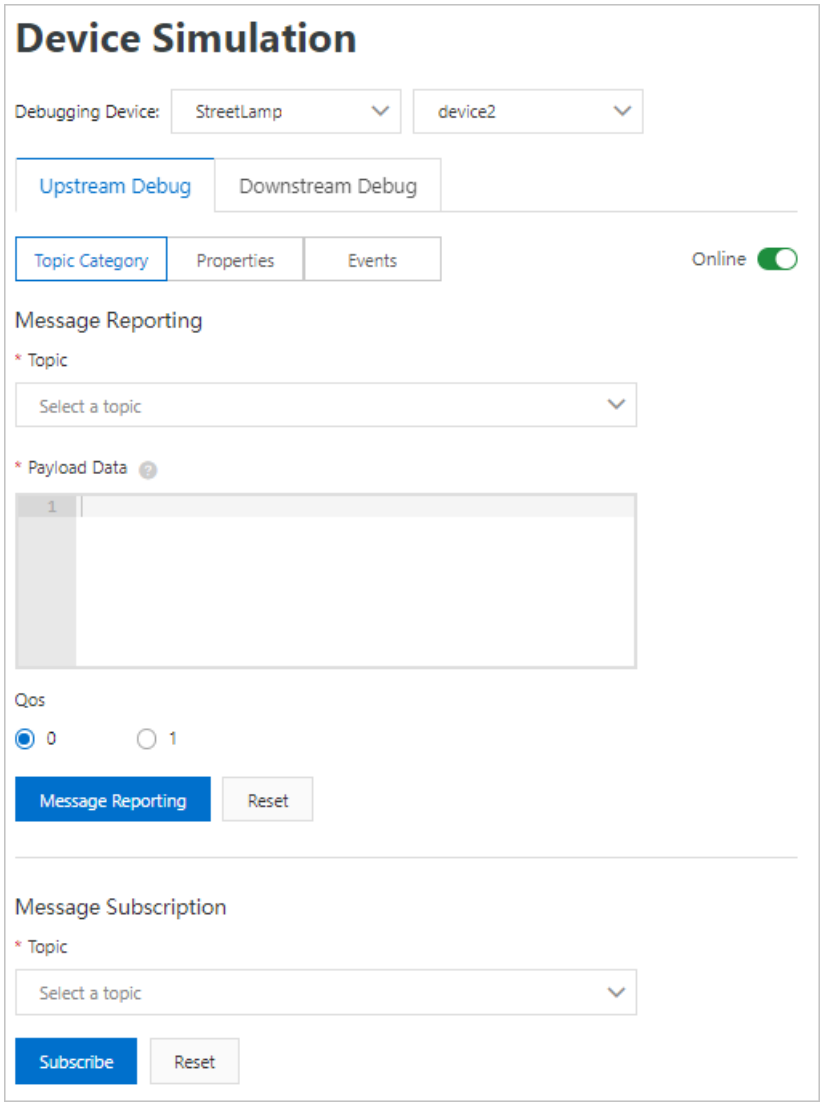
- Send upstream data
  - Custom topics that are used to send and receive data
  - Property submission
  - Event submission
- Send downstream commands
  - Custom topics
  - Property query and setting
  - Service calling

Limits

- You can set a push policy. If you set the push policy multiple times, only the last settings take effect.
- The minimum interval to push consecutive messages is 1 second.
- The maximum duration to push consecutive messages is 3 hours.
- The device simulation feature cannot be used to simulate devices that transmit data in custom formats.
- If a physical device is online or disabled, you cannot use device simulation. After you leave the Device Simulation page, the device simulator is disabled.

Procedure

1. Log on to the IoT Platform console.
2.
3. In the left-side navigation pane, choose **Maintenance > Device Simulation**.
4. Select a product and then a device that you want to simulate. On the Device Simulation page, click **Start Device Simulation**.
5. Select a feature and send data to debug the feature. The following table describes the features.



Feature type	Procedure
Custom topics that are used to send and receive data	Use the device simulator to send an upstream message to a custom topic. i. Click the **Upstream Debug** tab and then the **Topic Category** tab. ii. Select a custom topic that is used to send the message, enter payload data, and then set QoS to *0* or *1*. Click **Message Reporting**. iii. Select a topic that is used for message subscription. Click **Subscribe**.

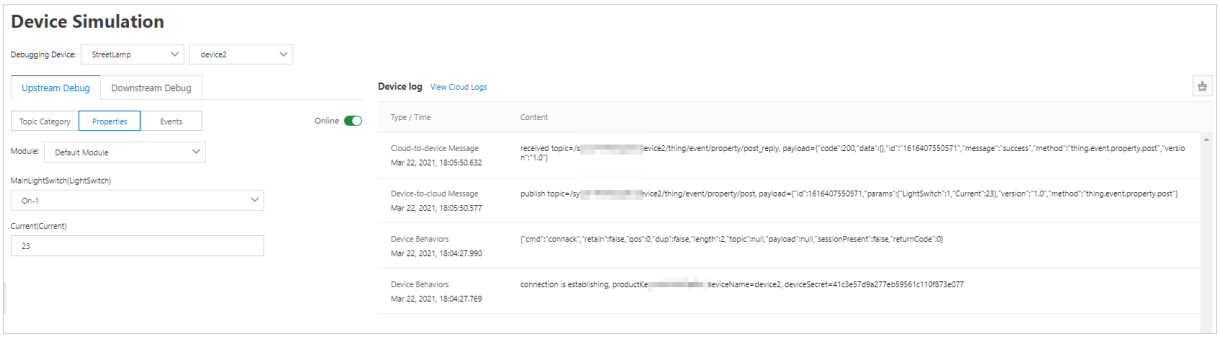
Feature type	Procedure
Property submission	Use the device simulator to submit a property to IoT Platform. i. Click the **Upstream Debug** tab and then the **Properties** tab. ii. Select a Thing Specification Language (TSL) module from the **Module** drop-down list. iii. Select or enter a value in a property field. The value must match the data type and value range of the property. iv. Push the data. Valid options: ■ **Send Command**: Immediately push the data. ■ **Push Policy**: Set a push policy. Valid values: ■ At Specific Time: The data is pushed only once at a specified time. ■ At Specific Interval: The data is pushed at a fixed interval during a specified period. The interval is measured in seconds.
Event submission	Use the device simulator to submit an event to IoT Platform. i. Click the **Upstream Debug** tab and then the **Events** tab. ii. Select a Thing Specification Language (TSL) module from the **Module** drop-down list. iii. Select an event and enter the event data in the JSON format, such as `{"Power": "on"}`. iv. Push the data. Valid options: ■ **Send Command**: Immediately push the data. ■ **Push Policy**: Set a push policy. Valid values: ■ At Specific Time: The data is pushed only once at a specified time. ■ At Specific Interval: The data is pushed at a fixed interval during a specified period. The interval is measured in seconds.
Custom topics to send downstream data	Enable IoT Platform to send a downstream message to a custom topic. i. Click the **Downstream Debug** tab and then the **Topic Category** tab. ii. Select a custom topic, enter payload data, and then set QoS to *0* or *1*. iii. Click **Send Command**.

Feature type	Procedure
Property query and setting	Enable IoT Platform to send a downstream command to set a property value and then retrieve the property value from the device simulator. i. Click the **Downstream Debug** tab and then the **Property debugging** tab. ii. Select a Thing Specification Language (TSL) module from the **Module** drop-down list. iii. Enter a value in a property field and click **Debugging**. Then, click **Set**. After the device simulator receives the command, it sets the property to the new value. iv. On the right side of a property field, click **Debugging**. Then, click **Get**. The latest property value is displayed in the field. If the property value does not exist on the device simulator, no data is retrieved.
Service calling	i. Click the **Downstream Debug** tab and then the **Invoke Service** tab. ii. Select a Thing Specification Language (TSL) module from the **Module** drop-down list. iii. In the field, enter the input parameters of the service. Then, click **Send Command**. The input parameters must be in the JSON format. Example: `{"Switch":0}`.

Result

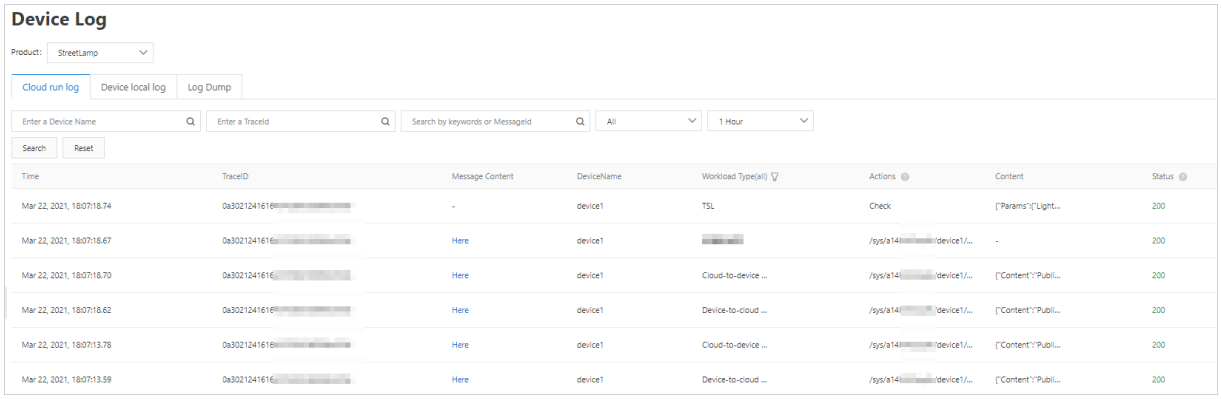
After data is pushed, you can view device logs in the **Device log** section on the right side of the page.

The following figure shows an example.



Click **View Cloud Logs**. On the **Cloud run log** tab, you can view the related IoT Platform logs.

For more information, see IoT Platform logs.


Device Log
Product: StreetLamp
Cloud run log
Device local log
Log Dump
Enter a Device Name
Enter a TraceId
Search by keywords or MessageId
All
1 Hour
Search
Reset
Time
TraceID
Message Content
DeviceName
Workload Type(all)
Actions
Content
Status
Mar 22, 2021, 18:07:18.74
Mar 22, 2021, 18:07:18.67
Mar 22, 2021, 18:07:18.70
Mar 22, 2021, 18:07:18.62
Mar 22, 2021, 18:07:13.78
Mar 22, 2021, 18:07:13.59
Here
device1
TSL
Check
200

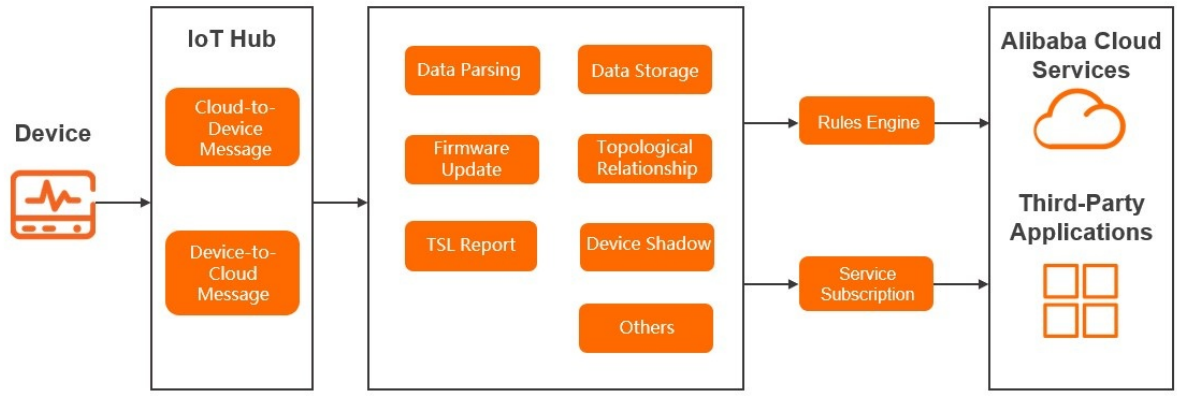
4.Log Service

4.1. IoT Platform logs

You can query IoT Platform logs in the IoT Platform console. IoT Platform logs contain the records of communication among IoT Platform, devices, and applications. This topic describes the error codes in IoT Platform logs and the troubleshooting methods.

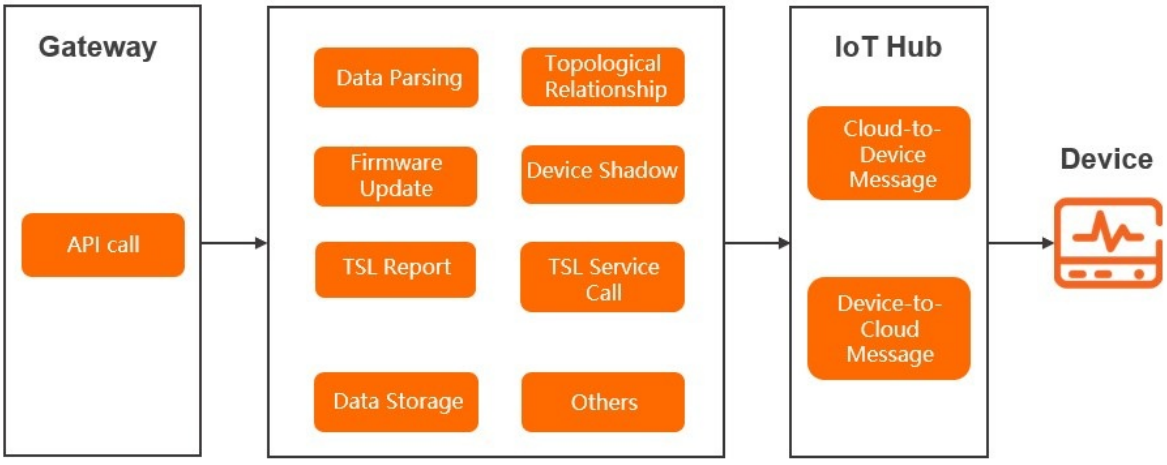
Log types

The following figure shows the log types of upstream messages.



1. If devices send messages to IoT Platform, the message logs are generated. The topics that are used to send messages are included in the logs.
2. When data is processed by different business modules in IoT Platform, the logs of the business modules are generated.
3. If messages are sent to consumers by using the data forwarding or server-side subscription feature, the logs of the feature are generated. If you use the server-side subscription feature, you can configure an Advanced Message Queuing Protocol (AMQP) or Message Service (MNS) client to receive messages.

The following figure shows the log types of downstream messages.



1. When users call API operations to publish messages, the logs of the API operations are generated.

The API operation names are included in the logs.

2. If data is processed by different business modules in IoT Platform, the logs of the business modules are generated.
3. If messages are sent from IoT Platform to devices, the corresponding message logs are generated. The topics that are used to send messages are included in the logs.

Query IoT Platform logs

1. Log on to the IoT Platform console.
2.
3. In the left-side navigation pane, choose **Maintenance > Device Log**. Then, click the **Cloud run log** tab.
4. Select a product, specify search conditions, and then click Search.

   The following table describes the supported search conditions.

   > **Notice** The search condition content is split into multiple words if the search condition, such as a device name or keyword, contains one of the following special characters: `, '";=()[]{}?@&<>/:\n\t\r`. In this case, the query fails. The following error message is returned: **The parameter is invalid when you query logs.**

Search condition	Description
DeviceName	Enter a DeviceName. You can search for the logs of a device by DeviceName.
TraceId	Enter a trace ID to search for the logs of series modules.
Keyword	Enter a keyword to search for the logs that contain the keyword.
MessageID	Enter a message ID. A message ID is a unique identifier that is generated by IoT Platform for a message. You can search for logs by message ID only when you analyze upstream and downstream messages.
State	Select a state to search for related logs. Valid values: ○ All ○ Successful: The HTTP status code is 200. ○ Failed: The HTTP status code is not 200.
Time Range	Select a time range.

Log fields

The following table describes the log fields.

Parameter	Description	Remarks
Time	The time when a log entry was generated.	None

Parameter	Description	Remarks
TraceId	The trace ID. You can use this ID to search for series modules.	None
MessageID	The ID of the message.	None
DeviceName	The DeviceName of the device.	None
Business type	By default, logs of all business types are displayed. You can query logs of a specific business type. The following log fields that correspond to business types are displayed when logs are stored in Log Service: • *OTA update*: *OTA* • *Data parsing*: *ScriptParsing* • *TSL data verification*: *ThingModel* • *Data storage*: *DataStorage* • *Remote configuration*: *RemoteConfig* • *Topological relationships*: *ThingTopo* • *TSL service calls*: *ThingService* • *Device behavior*: *device* • *Device-to-cloud messages*: *uplink* • *Cloud-to-device messages*: *downlink* • *API calls*: *ApiService* • *Server-side subscription*: *ServiceSubscribe* • *Device shadow*: *DeviceShadow* • *Data forwarding*: *RuleEngine* • *Subscription to topics*: *subscribe* • *Unsubscription from topics*: *unsubscribe* • *TSL messages*: *ThingModelMessage* • *Device file uploading*: *DeviceFileUpload* • *Others*: *Other*	The first-level business identifier that identifies a business module.

Parameter	Description	Remarks

Operation	This field specifies an operation that is performed, API operation, service method, or message topic. The field value varies based on the following operation types:
	• OTA update
	  ○ *OTAFirmwarePush*: pushes notifications when a firmware update is initiated, confirmed, and completed.
	  ○ *OTAFirmwareRequest*: requests the information about an OTA update package.
	  ○ *OTAVersionReport*: submits the OTA module version of a device.
	  ○ *OTAProgressReport*: submits the update progress of a device.
	• Data parsing
	  ○ *RawDataToProtocol*: converts raw data to Alink protocol data.
	  ○ *ProtocolToRawData*: converts Alink protocol data to raw data.
	• TSL data submission
	  ○ *check*: verifies the submitted TSL data based on the TSL definitions.
	  ○ For more information about the method parameter in the message body, see topics for TSL communications.
	• Device behavior management
	  ○ *online*: connects a device to IoT Platform.
	  ○ *offline*: disconnects a device from IoT Platform.
		The second-level business identifier.

Content	The log content may contain the following parameters: • TraceId: the trace ID that can be used to search for series modules. • Message: the error message. Logs of failed operations include this field. • Params: the request parameters. Logs of certain types include this field. • ResultData: the result. If operations generate results, printed logs include this field. Otherwise, printed logs do not include this field.	If the data format of a product is set to Custom, logs for TSL data parsing include the hexadecimal values of raw data that is submitted by devices.

Parameter	Description	Remarks
State	The HTTP status code in the response. A status code of 200 indicates that a call is successful. Other status codes indicate that the call failed. For more information about API operation-related error codes, see Error codes. For more information about other error codes, see the following section.	None

Device behavior-related error codes

Logs for device behaviors are generated when devices go online or offline.

Error code	Description	Cause	Troubleshooting
200	The device goes online or offline as expected.	• Offline: The device ends the connection with IoT Platform. • Online: The device establishes a connection with IoT Platform.	View the **Last Online** and **Current Status** parameters of the device on the **Device Details** page of the IoT Platform console.
1910	An error occurred due to an MQTT heartbeat timeout when you disconnected the device from IoT Platform.	If IoT Platform does not receive a message within the keep-alive interval, the device is disconnected from IoT Platform and must be reconnected to the server.	Check whether the MQTT heartbeat keep-alive interval exceeds the threshold.
1911	The device goes offline because the TCP connection between the device and IoT Platform is ended.	• If a firewall or network address translation (NAT) gateway on the device side detects an inactive TCP connection, the device ends the TCP connection. • The TCP connection is terminated due to the complex network environment of the Internet. If the device can reconnect to IoT Platform and the business is not affected, ignore this error code.	Change the network environment, or check the firewall and gateway settings to troubleshoot the issue. For example, you can disable the firewall. **Note** If a TCP connection error frequently occurs, you can use TCPDUMP to obtain packet capture files and then submit a ticket.
1913	The sub-device is disconnected.	The gateway is offline.	Check why the gateway goes offline by using Log Service.

Error code	Description	Cause	Troubleshooting
401	The device has no permissions.	When you add a topological relationship and authenticate the sub-device, the signature of the sub-device fails to be verified.	Verify the generated signature and submitted signature by using the signature method in the Alink protocol. For more information about the Alink protocol, see Register devices.
427	A disconnection error occurred.	The device is forced to go offline because the device certificate is used by another device. IoT Platform identifies a device only based on the device certificate (ProductKey, DeviceName, and DeviceSecret). • The same device certificate is burned on multiple devices. • The network or power supply of the device is unstable. The device immediately reconnected to IoT Platform after an instantaneous network outage or power failure. In this case, IoT Platform identifies the reconnected device as a new device. Even if the error message is returned, the device can work as expected.	Check whether the device certificate is reused on the **Device Details** page of the IoT Platform console.
520	An error occurred in the session between the sub-device and IoT Platform.	• The specified session does not exist because the sub-device is not connected to IoT Platform, or the sub-device is already disconnected from IoT Platform. • The session exists, but the session is not established by using the current gateway.	On the **Device List** tab of the Devices page, search for the required device and view the device status.
521	The device is deleted.	The device is deleted from IoT Platform.	On the **Device List** tab of the Devices page, search for the device to check whether the device is deleted.
522	The device is disabled.	The device is disabled in IoT Platform.	In the IoT Platform console, check whether the device is in the **Disabled** state.

Error code	Description	Cause	Troubleshooting
6100	The device does not exist.	The device is not created or is deleted.	On the **Device List** tab of the Devices page, search for the required device to check whether the device exists.
6204	The device is disabled.	If a device is disabled, you cannot manage the device. For example, you cannot add topological relationships, configure device properties, or call device services.	In the IoT Platform console, check whether the device is in the **Disabled** state.
6287	The signature is invalid.	The signature of a directly connected device or sub-device is invalid.	Verify the generated signature and submitted signature by using the signature method in the Alink protocol. For more information about the Alink protocol, see Register devices.
6288	The dynamic registration feature of the device is disabled.	The Dynamic Registration switch of the product to which the sub-device belongs is turned off.	On the **Product Details** page of the console, turn on **Dynamic Registration**.
6296	The Alibaba Cloud account information does not match the instance information.	The instance does not belong to the Alibaba Cloud account.	In the IoT Platform console, check whether the instance belongs to the Alibaba Cloud account.
6401	The topological relationship does not exist.	The topological relationship does not exist.	Log on to the IoT Platform console and choose **Devices > Devices**. On the page that appears, search for the device and view the device information.
6402	The gateway is the same as the sub-device when you add a topological relationship.	When you add a topological relationship, you cannot attach the gateway to itself as a sub-device.	Check whether the gateway is the same as the sub-device.
6619	The sub-device is attached to another gateway.	If a sub-device is already attached to another gateway, you cannot attach the sub-device to the current gateway.	On the **Device Information** tab of the console, check whether the sub-device is attached to a gateway.

Error code	Description	Cause	Troubleshooting
2043	The device authentication fails because the token is invalid.	The token failed to be verified during device authentication.	Check whether the token has expired or is invalid. You can refresh or recreate tokens.

Message-related error codes

Message-related logs are generated in the following business scenarios:

- Devices send messages to IoT Platform.
- IoT Platform sends messages to devices.
- The rules engine forwards messages by using the server-side subscription or data forwarding feature.

Error code	Description	Cause	Troubleshooting
1004	The format of the data that the device submits to IoT Platform is invalid.	The format of the upstream data is invalid in the following OTA update scenarios: • The device submits OTA module versions to IoT Platform. • The device requests the update package information from IoT Platform. • The device submits the update progress to IoT Platform.	Check the format of the upstream data. For more information about the data formats, see OTA updates.
1901	The message fails to be sent due to poor network conditions, such as the congestion of the TCP write buffer.	The data channel between the device and the server is blocked. A possible cause is that the network transmission speed is low, or the device cannot process a large number of messages	Check network conditions and the message consumption capabilities of the device.
1902	An exception occurred when the message was transmitted over the network.	The message transmission failed due to a network exception.	Check network conditions.
1903	The format of the topic is invalid.	The format of the message topic is invalid.	Check the topic format.

Error code	Description	Cause	Troubleshooting
1904	IoT Platform receives an invalid RRPC response.	The RRPC response received by IoT Platform does not have the corresponding RRPC request. This error may occur if the request times out.	Check whether the RRPC response from the device has timed out.
1905	IoT Platform does not receive an RRPC response within the timeout period.	After IoT Platform sent an RRPC request to the device, IoT Platform did not receive an RRPC response from the device within the timeout period.	Check whether the device responded to the RRPC request within the timeout period.
1941	The device fails to be authenticated.	The token failed to be verified.	Check and reobtain the token. Then, initiate the request again.
1942	The message communicatio n is throttled.	Excessive requests are sent to the topic.	Reduce the frequency at which messages are sent from a single device or contact technical support.
1950	A network connection exception occurred when the message was transmitted over the network.	The message failed to be transmitted due to a network exception.	Check network conditions.
1951	The response type is unknown.	The device sends a message of an unknown type to IoT Platform.	Check the type of the message that is sent by the device. If you are using the Alibaba Cloud device SDK, contact technical support or submit a ticket.
6733	The device cannot be located based on the network information	The device failed to be located based on the specified network information.	Modify the network information and relocate the device.
6736	The device cannot be located based on the IP address.	The device failed to be located based on the specified IP address.	Modify the IP address and relocate the device.

Error code	Description	Cause	Troubleshooting
6831	The specified topic or request method does not conform to the Alink protocol.	The topic to which the device submits data or the method parameter in the parsed data does not conform to the Alink protocol.	Check whether the topic to which the device submits data conforms to the Alink protocol. Check whether the method parameter in the parsed data conforms to the Alink protocol.
9200	The device is inactive.	The device is not activated in IoT Platform. After a new device is registered, the device is activated only after it is connected to IoT Platform and submits data to IoT Platform.	Check the status of the device in the IoT Platform console.
9201	The device is disconnected.	The device is offline.	Check the status of the device in the IoT Platform console.
9236	The topic fails to be authenticated.	The permission that is specified for the topic is invalid.	Go to the **Topic List** tab of the IoT Platform console. Check whether the permissions that are specified for topics are valid. The Publish permission must be specified for the topics that are used to publish messages. The Subscribe permission must be specified for the topics that are used to receive messages.
9307	The SQL statement fails to be parsed.	The syntax of the SQL statement or the parameter is invalid.	Check the syntax of the SQL statement and the parameters.
9324	A throttling error occurred.	The number of requests from the device or the tenant exceeded the limit.	Reduce the frequency at which messages are sent or contact technical support.
9325	The data fails to be forwarded to the cloud service.	The destination cloud service is unavailable.	Check whether the destination cloud service is available. A cloud service is unavailable if payments are overdue, instances are deleted, or the permissions that IoT Platform use to access the cloud service are deleted.
9321	The parameters are invalid.	The input parameters are invalid.	Check the parameters as prompted.
9320	The payload is invalid.	The format of the payload sent by the device is invalid.	Check whether the format of the payload is valid.

Error code	Description	Cause	Troubleshooting
9331	An internal error occurred on the destination cloud service.	An internal error occurred on the cloud service to which the message is sent.	Check the error code in the log entry and go to the official website of the cloud service to view the cause and solution. You can also contact technical support.
9332	The cloud service configuration is invalid.	The specified data forwarding destination is invalid. Therefore, an error occurred when IoT Platform connected to the destination cloud service.	View the data forwarding rule to check whether the configuration of the destination cloud service is valid and whether the resource exists. Check the error code in the log entry and go to the official website of the cloud service to view the cause and solution.
9362	An exception occurred when the script was being executed.	A parsing exception occurred when the rules engine executed a script. Possible causes: • An exception occurred when the parser script was running. • An exception occurred when functions in the script were called. You can determine the cause based on the log content.	Check the script syntaxes and function calls based on the log content. For more information about how to use the script and functions, see Script syntax and Functions.
9333	The permission to access the cloud service is invalid.	The permission that is granted to IoT Platform to access the destination cloud service is invalid.	Check your Alibaba Cloud RAM policy.
9389	The sub-device fails to send messages because the gateway is offline.	The gateway to which the sub-device is attached is offline.	On the **Device List** tab of the Devices page, search for the required gateway and view the gateway status. Make sure that the gateway is online or restart the gateway to connect the sub-device with IoT Platform.
9399	An unknown internal server error occurred.	An internal error occurred in IoT Platform.	Contact technical support or submit a ticket.
9600	The number of connections in a consumer group exceeds the limit.	The number of connections that IoT Platform can process exceeds the limit. For more information, see Limits of server-side subscriptions.	Clear unnecessary connections.

Error code	Description	Cause	Troubleshooting
9601	The heartbeat value is invalid.	The heartbeat value does not meet the requirement. For more information, see Limits of server-side subscriptions.	Specify a valid heartbeat value.
9602	IoT Platform ends the connection.	This error may occur in load balancing and IoT Platform iteration scenarios. The device must be reconnected to IoT Platform. This issue does not affect business continuity.	Submit a ticket.
9650	The ACK message times out.	No response is sent from the receiver within the timeout period. Therefore, the ACK message from the receiver timed out.	Check the message processing logic of the receiver.
9651	The receiver returns ACK released.	The receiver returns ACK released.	
9652	The receiver returns NACK.	The receiver returns NACK.	

TSL-related error codes

TSL-related logs are generated in the following business scenarios:

- Devices submit TSL data.
- IoT Platform calls TSL services.

If the data format of a product is set to Custom, TSL-related logs include the hexadecimal values of raw data that is submitted by devices.

The following table describes error codes that may be generated when IoT Platform calls services and configures properties.

When IoT Platform calls a service, IoT Platform checks whether the input parameters of the service follow the syntax defined in the TSL model.

Error code	Description	Cause	Troubleshooting
100000	The parameters are invalid.	The instance ID is not obtained when you query the configuration information about a product in a public or Enterprise Edition instance of IoT Platform.	Check whether you have accessed an IoT Platform instance. If you have accessed an IoT Platform instance, submit a ticket for troubleshooting.
9201	The device is disconnected.	The device is disconnected from IoT Platform.	Check the device status in the IoT Platform console.

Error code	Description	Cause	Troubleshooting
9200	The device is not activated.	The device is not activated in IoT Platform. A newly registered device is activated only after the device submits data to IoT Platform.	Check the device status in the IoT Platform console.
9237	Overdue payments exist for the IoT Platform service.	The Alibaba Cloud account has overdue payments.	In the console, click **Expenses** in the upper-right corner and view the balance in the user center. Make sure that the balance of your account is sufficient. Otherwise, IoT Platform devices are unavailable.
9389	The sub-device fails to send messages because the gateway is offline.	The gateway to which the sub-device is attached is offline.	On the **Device List** tab of the Devices page, search for the gateway and view the gateway status. Make sure that the gateway is online or restart the gateway to connect the sub-device with IoT Platform.
6208	The device is disabled.	After a device is disabled, IoT Platform cannot configure device properties or call device services.	Check the device status in the IoT Platform console. If the device is disabled, enable the device and then try again.
6300	The method parameter does not exist when IoT Platform verifies the input parameters based on the TSL model.	The method parameter that is required by the Alink protocol does not exist in the input parameters. The input parameters can be the device-submitted standard Alink data or the parsing result of custom data submitted by the device.	View the IoT Platform logs about device property reporting and check the data submitted by the device. You can also check the submitted data by viewing the on-premises device logs.
6206	An error occurred when you queried the service.	When IoT Platform calls a service, the information about the service is queried. This error occurs if the service does not exist.	Go to the Product Details page of the IoT Platform console. On the Define Feature tab, check whether the service is defined in the TSL model. If the service is defined, check whether the input parameters of the service contain invisible characters.

Error code	Description	Cause	Troubleshooting
6200	The script does not exist.	If the data format of a product is set to Custom, the script is used to parse data when IoT Platform calls the service of a device. This error occurs if you do not define a data parsing script.	Go to the Product Details page of the IoT Platform console. Check whether the data parsing script exists. If the data parsing script exists, resubmit the script and then try again.
6201	The parsing result is empty.	The data parsing script runs as expected, but returns an empty result. For example, the rawDataToProtocol() method returns null and the protocolToRawData() method returns null or an empty array.	Check the script to identify the cause.
6207	The data format is invalid.	This error may occur when IoT Platform synchronously calls a service or when the device submits data. When IoT Platform synchronously calls a service, this error may occur due to one of the following causes: • The format of the data returned by the device is invalid. • The format of parsed custom data is invalid. • The data format of the input parameters is invalid.	For more information about the valid data format required by the service, see the API documentation and TSL definitions. For information about the data format required by the Alink protocol, see Alink protocol.
6330	The data format does not conform to the defined format of the Long type.	The parameters or properties of the Long type are defined in the TSL model. The data format of the message does not conform to the defined format of the Long type.	• View the logs that are generated when the device submits properties. • View the TSL model in the console.
6335	The parameter in the message that the device submits as a response to the property configuration request is not empty.	When the device responds to the property configuration request sent from IoT Platform, the response data that is indicated by the data field must be empty.	• View the IoT Platform logs about device property reporting and check the data submitted by the device. • Check the submitted data by viewing the on-premises device logs.

Error code	Description	Cause	Troubleshooting
5490	The specified TSL module does not exist.	The specified custom TSL module does not exist.	• In the IoT Platform console, view the identifier of the custom TSL module and check whether valid values are specified for the required parameters. • In the IoT Platform console, check whether the specified custom TSL module is deleted.
5092	The property does not exist in the TSL model.	The property in the upstream or downstream message is not defined in the TSL model. **Notice** For a property that is defined in a custom TSL module, you must combine the property with the module identifier in the format of `{tsl.functionBlockId}:{tsl.properties.identifier}`.	• View the logs that are generated when the device submits properties. • View the TSL model in the console.
5094	The service does not exist in the TSL model.	The service is not defined in the TSL model, or the service parameters are invalid. **Notice** For a service that is defined in a custom TSL module, you must combine the service with the module identifier in the format of `{tsl.functionBlockId}:{tsl.service.identifier}`.	• View the logs that are generated when the device submits properties. • View the TSL model in the console.

Error code	Description	Cause	Troubleshooting
5096	The event does not exist in the TSL model.	The event is not defined in the TSL model or the event parameters are invalid. **Notice** If an event is defined in a custom TSL module, you must combine the event with the module identifier in the format of `{tsl.functionBlockId}:{tsl.event.identifier}`.	• View the logs that are generated when the device submits properties. • View the TSL model in the console.
System error codes			
5159	An error occurred when IoT Platform retrieved the TSL property data.	A system exception occurred.	Submit a ticket.
5160	An error occurred when IoT Platform retrieved the TSL event data.		
5161	An error occurred when IoT Platform retrieved the TSL service data.		
6661	An error occurred when IoT Platform queried the tenant information.		
6205	An error occurred when IoT Platform called the service.		

Error code	Description	Cause	Troubleshooting
26015	An error occurred when IoT Platform ran the script to parse the data.		

The following table describes the error codes that are generated when devices fail to submit property and event data.

When a device submits property or event data, IoT Platform verifies the property data or the input parameters of the event based on the TSL model.

Error code	Description	Cause	Troubleshooting
6106	The number of properties submitted by the device exceeds the limit.	A device can submit a maximum of 200 properties at a time.	View the IoT Platform logs about device property reporting and check the number of properties that are submitted by the device. You can also check the submitted data by viewing the on-premises device logs.
6300	The method parameter does not exist when IoT Platform verifies the input parameters based on the TSL model.	The method parameter that is required by the Alink protocol does not exist in the input parameters. The input parameters can be the device-submitted standard Alink data or the parsing result of custom data submitted by the device.	View the IoT Platform logs about device property reporting and check the data submitted by the device. You can also view the on-premises device logs and check the submitted data.
6320	The property information does not exist when IoT Platform verifies the input parameters based on the TSL model.	The specified property failed to be found when the system queried the TSL data of the device.	Go to the Product Details page of the IoT Platform console. On the Define Feature tab, check whether the specified property is defined in the TSL model. If the property is not defined, define the property.
6450	The method parameter does not exist in the Alink JSON data.	The method parameter does not exist in the standard Alink data. The standard Alink data can be directly submitted by the device or converted from custom data submitted by the device.	View the IoT Platform logs about device property reporting, and check whether the submitted data includes the method parameter. You can also view the on-premises device logs.

Error code	Description	Cause	Troubleshooting
6207	The data format is invalid.	This error may occur when IoT Platform synchronously calls a service or when the device submits data. When the device submits data, this error may occur because the Alink data submitted by the device or the data parsed by using the script is not in the JSON format.	For more information about the data format required by the Alink protocol, see Alink protocol. You must use the required data format to submit data.
System error codes			
6452	A throttling error occurred.	Traffic throttling is triggered because excessive requests are submitted.	Submit a ticket.
6760	The storage quota of the tenant is exceeded.	A system exception occurred.	Submit a ticket.

The following table describes the error codes that are generated when devices fail to respond to the service calls and property configuration requests from IoT Platform.

Error code	Description	Cause	Troubleshooting
Common error codes			
460	One or more parameters are invalid.	The request parameters are invalid.	Submit a ticket.
500	An internal system error occurred.	An unknown error occurred in IoT Platform.	Submit a ticket.
400	A request error occurred.	An unknown error occurred when IoT Platform called the service.	Submit a ticket.
429	An excessive number of requests are submitted within a period of time.	Traffic throttling is triggered because an excessive number of requests are submitted within a period of time.	Submit a ticket.
System error codes			

Error code	Description	Cause	Troubleshooting
6452	A throttling error occurred.	Traffic throttling is triggered because an excessive number of requests are submitted.	Submit a ticket.

The following table describes the common TSL-related error codes.

IoT Platform verifies the input parameters of the service, the property data, and the input parameters of the event based on the TSL model.

Error code	Description	Cause	Troubleshooting
6321	The identifier of the property does not exist in the TSL model.	A system exception occurred.	Submit a ticket.
6317	The TSL model is invalid.	A system exception occurred.	Submit a ticket.
6332	The input parameter does not conform to the TSL definitions.	The input parameters must conform to the TSL definitions.	Go to the Product Details page of the IoT Platform console. View the TSL model on the Define Feature tab. Check the input parameters.
6302	The parameter does not exist.	IoT Platform failed to verify the input parameters of the service based on the TSL model because the parameters were not found in the request.	Go to the Product Details page of the IoT Platform console. View the TSL model on the Define Feature tab. Check the input parameters of the service in the TSL model and make sure that all required parameters are configured.
6306	The input parameter does not match the integer data type that is defined in the TSL model.	When IoT Platform verifies a parameter based on the TSL model, the following errors may occur: • The data type of the parameter is different from the data type that is defined in the TSL model. • The parameter value is not in the defined range of the TSL model.	Go to the Product Details page of the IoT Platform console. On the Define Feature tab, view the TSL model and make sure that the data type of the input parameter is the same as the data type defined in the TSL model.

Error code	Description	Cause	Troubleshooting
6307	The input parameter does not match the 32-bit float data type that is defined in the TSL model.	When IoT Platform verifies a parameter based on the TSL model, the following errors may occur: • The data type of the parameter is different from the data type that is defined in the TSL model. • The parameter value is not in the defined range of the TSL model.	Go to the Product Details page of the IoT Platform console. On the Define Feature tab, view the TSL model. Make sure that the data type of the input parameter is the same as the defined data type and the parameter value is in the defined value range.
6322	The input parameter does not match the 64-bit float data type that is defined in the TSL model.	When IoT Platform verifies a parameter based on the TSL model, the following errors may occur: • The data type of the parameter is different from the data type that is defined in the TSL model. • The parameter value is not in the defined range of the TSL model.	Go to the Product Details page of the IoT Platform console. On the Define Feature tab, view the TSL model. Make sure that the data type of the input parameter is the same as the defined data type and the parameter value is in the defined value range.
6308	The input parameter does not match the Boolean data type that is defined in the TSL model.	When IoT Platform verifies a parameter based on the TSL model, the following errors may occur: • The data type of the parameter is different from the data type that is defined in the TSL model. • The parameter value is not in the defined range of the TSL model.	Go to the Product Details page of the IoT Platform console. On the Define Feature tab, view the TSL model and make sure that the input parameter meets the requirements of the data type defined in the TSL model.
6309	The input parameter does not match the enum data type that is defined in the TSL model.	The data type of the parameter is different from the data type defined in the TSL model.	Go to the Product Details page of the IoT Platform console. On the Define Feature tab, view the TSL model. Make sure that the data type of the input parameter is the same as the defined data type.

Error code	Description	Cause	Troubleshooting
6310	The input parameter does not match the text data type that is defined in the TSL model.	When IoT Platform verifies a parameter based on the TSL model, the following errors may occur: • The data type of the parameter is different from the data type that is defined in the TSL model. • The length of the parameter exceeds the limit defined in the TSL model.	Go to the Product Details page of the IoT Platform console. On the Define Feature tab, view the TSL model. Make sure that the input parameter meets the requirements of the defined data type.
6311	The input parameter does not match the date data type that is defined in the TSL model.	When IoT Platform verifies a parameter based on the TSL model, the following errors may occur: • The data type of the parameter is different from the data type defined in the TSL model. • The input date parameter is not a UTC timestamp.	Go to the Product Details page of the IoT Platform console. On the Define Feature tab, view the TSL model. Make sure that the input parameter meets the requirements of the defined data type.
6312	The input parameter does not meet requirements of the structure data type defined in the TSL model.	When IoT Platform verifies a parameter based on the TSL model, the following errors may occur: • The data type of the parameter is different from the data type defined in the TSL model. • The number of the parameters in a structure is different from the number defined in the TSL model.	Go to the Product Details page of the IoT Platform console. On the Define Feature tab, view the TSL model and make sure that the data type of the input parameter is the same as the data type that is defined in the TSL model.
6304	The input parameter does not exist in the TSL structure.	IoT Platform failed to verify the input parameters based on the TSL model because the input parameters failed to be found in the structure.	Go to the Product Details page of the IoT Platform console. On the Define Feature tab, view the TSL model and make sure that the input parameter exists in the defined structure data type.

Error code	Description	Cause	Troubleshooting
6324	The input parameter does not match the array data type that is defined in the TSL model.	When IoT Platform verifies a parameter based on the TSL model, the following errors may occur: • The elements in the array do not conform to the array syntax defined in the TSL model. • The number of elements in the array exceeds the maximum number defined in the TSL model.	• Go to the Product Details page of the IoT Platform console. On the Define Feature tab, view the TSL model and check the array syntax that is defined in the TSL model. • View the upstream message logs to check the number of array elements in the data submitted by the device.
6328	The input parameter is not an array.	The input parameter is not an array when IoT Platform verifies the parameter based on the TSL model.	Go to the Product Details page of the IoT Platform console. On the Define Feature tab, view the TSL model and check the service parameter of the array data type. Make sure that the data type of the input parameter is array.
6325	The data type of elements in the array is not supported by IoT Platform.	The input parameter failed to be verified based on the TSL model because the data type of elements in the array is not supported. Only the following data types of elements are supported: int32, float, double, text, and struct.	Make sure that the data type of elements is supported by IoT Platform.
System error codes			
6318	A system exception occurred when IoT Platform parsed the TSL data.		
6329	An error occurred when IoT Platform parsed the array data in the TSL model.		

Error code	Description	Cause	Troubleshooting
6323	The data format of the parameter in the TSL model is invalid.	A system exception occurred.	Submit a ticket.
6316	An error occurred when IoT Platform parsed the parameter in the TSL model.		
6314	The data type of the parameter in the TSL model is not supported.		
6301	An error occurred when IoT Platform verified the data format of the input parameters based on the TSL model.		
Data parsing script-related error codes			
26010	Traffic throttling is triggered because an excessive number of requests are submitted.	An excessive number of requests were submitted within a period of time.	Submit a ticket.
26001	The content of the script is empty.	IoT Platform fails to obtain and run the script because the script content is empty.	Go to the Product Details page of the IoT Platform console, and check whether the data parsing script exists. If the script exists, check whether the script is saved. The script cannot be a draft.

Error code	Description	Cause	Troubleshooting
26002	An exception occurred when IoT Platform ran the script.	The script runs as expected but the script content is invalid. For example, the script contains syntax errors.	Log on to the IoT Platform console, use the same parameters to run the script for debugging, and then modify the script. **Notice** The console provides a basic script running environment but does not verify the script. We recommend that you check the script on premises before you save it.
26003	A timeout error occurred when you ran the script.	The logic in the script was too complex and the data failed to be parsed within the timeout period of 3 seconds.	View the script content in the console and check the logic. Make sure that the script does not contain infinite loops. We recommend that you check the script on premises before you save it.
26006	The required method does not exist in the script.	The script runs as expected but the script content is invalid. The script must contain the protocolToRawData() and rawDataToProtocol() methods. This error occurs if one of these two methods does not exist.	Go to the Product Details page of the IoT Platform console and check whether the protocolToRawData() and rawDataToProtocol() methods exist.
26007	The returned data format is invalid after data parsing.	The script runs as expected but the format of the returned data is invalid. The script must contain the protocolToRawData() and rawDataToProtocol() methods. The protocolToRawData() method returns a byte[] array. The rawDataToProtocol() method returns a JSON object. This error occurs if the returned data is not in the required format. For example, after a device submits data to IoT Platform, IoT Platform sends a response to the device. The returned data must also be parsed. Otherwise, the format of the returned data may be invalid.	Check the script in the IoT Platform console. Enter the input parameters, run the script on premises, and check whether the data format of the returned result is valid.

Subscription-related error codes

Error code	Description	Cause	Troubleshooting
9200	The device is inactive.	The device is not activated in the IoT Platform console. After a new device is registered, the device is activated only after it is connected to IoT Platform and submits data to IoT Platform.	Check the status of the device in the IoT Platform console.
500	An internal system error occurred.	An unknown error occurred in IoT Platform.	Submit a ticket.
403	The request is forbidden.	Your account has overdue payments or the topic fails to be authenticated.	Submit a ticket.

Topology-related error codes

Error code	Description	Cause	Troubleshooting
5005	An error occurred when you queried the product information.	The specified product does not exist.	Log on to the IoT Platform console and choose **Devices > Products**. On the page that appears, query the product information and check whether the **ProductKey** exists.

Remote configuration-related error codes

Error code	Description	Cause	Troubleshooting
6710	The remote configuration file has no content. You must edit and save the file in the console to obtain the file.	The remote configuration file is not saved in the console.	Log on to the IoT Platform console and choose **Maintenance > Remote Config**. Refresh the page that appears. Save the edited content under **Configure Template**.
6713	The remote configuration switch is turned off.	The remote configuration switch is turned off on the Remote Configuration page. This page appears when you choose **Maintenance > Remote Config** in the console.	In the IoT Platform console, choose **Maintenance > Remote Config** . On the page that appears, check whether the remote configuration feature is enabled.

File uploading-related error codes

Error code	Description	Cause	Troubleshooting

Error code	Description	Cause	Troubleshooting
78123	The file to be uploaded already exists in IoT Platform.	This error may occur due to one of the following causes: • The file to be uploaded has the same name as a file that has been uploaded to IoT Platform. • The specified processing policy for files with the same name does not allow an uploaded file to overwrite an existing file in IoT Platform. The two files have the same name.	Check the following items: • Check whether the file name is unique. • Check whether the processing policy for files with the same name is *overwrite*. For more information, see Sample code.
78129	The number of files that the device has uploaded to IoT Platform exceeds the limit. IoT Platform can store up to 1,000 files that are uploaded from each device.	The number of files that the device has uploaded to IoT Platform exceeds 1,000.	Check whether the number of files that the device has uploaded to IoT Platform exceeds the limit.

4.2. Local device logs

Devices, including gateway devices and sub-devices, can submit logs to IoT Platform. You can query local device logs and troubleshoot issues on the Device Log page of the IoT Platform console.

Prerequisites

- An SDK for Android or SDK for C is used to develop a device. Logs are submitted by the device. For more information about how to reset a password, see Device log reporting.
- Device local log reporting is turned on. To turn on the switch, you must log on to the IoT Platform and perform the following steps: 1. Choose **Devices > Devices**. 2. Find the required device in the device list and click **View**. 3. On the **Device Details** page, turn on **Device local log reporting**.

Query local device logs

1. Log on to the IoT Platform console.
2.
3. In the left-side navigation pane, choose **Maintenance > Device Log** .
4. Select a product and click the **Device local log** tab.

5. Specify the search conditions and click Search.

   The following table describes the supported search conditions.

Search condition	Description
DeviceName	Enter a DeviceName. You can search for the logs of a device by DeviceName.
TraceId	Enter a trace ID to search for the logs of series modules.
Module name	Enter a module name to search for the logs that are generated by the module.
Keyword	Enter a keyword to search for the logs that contain the keyword. Supported keywords: the name of a parameter in an API request, the failure cause, the ID of a message, the ID of a device, and the name of an operation.
Time range	Select a time range.

Log fields

The following table describes the log fields.

Parameters	Description
Reported At	The time when the device submitted the log.
Collected At	The time when the device collected the log.
TraceId	The trace ID. You can use this ID to search for series modules.
DeviceName	The DeviceName of the device.
The log level.	By default, logs of all levels are displayed. You can query only logs of a specific level. Log levels other than the OTHER level are in descending order: • FATAL • ERROR • WARN • INFO • DEBUG • OTHER: another log level
Module	The name of the module that generates the log. • If the device uses the SDK for Android, the module name is ALK-LK. • If the device uses the SDK for C, the module name is user-specified.
Description	The content of the log.

Analyze device logs

The log content includes the Code parameter. You can analyze device logs based on response codes.

- For more information about the status codes that may be returned if the device uses the SDK for C,

see Status codes for the SDK for C.

- If the device uses the SDK that is developed by you, customize the status code or leave it empty.

4.3. Dump IoT Platform logs

By default, the device log feature of IoT Platform allows you to retain the logs of the last seven days. This feature also allows you to export IoT Platform logs to a Logstore of Log Service for persistent storage. After you enable the IoT Platform log dump feature, you can query and analyze logs, view and subscribe to log reports, and configure alerts in the IoT Platform console.

Prerequisites

Log Service is activated. For more information, see Resource management overview.

Enable the IoT Platform log dump feature

> **Note** The log dump feature is available only for Alibaba Cloud accounts.

You must enable the log dump feature for each product.

1. Log on to the IoT Platform console.
2.
3. In the left-side navigation pane, choose **Maintenance > Device Log** .
4. Select a product and click the **IoT Platform Log Dump** tab.
5. Click **Enable**.
6. In the Log Configurations message, read the instructions and click **OK**.

   > **Note** If Log Service is not activated, you are navigated to the activation page of Log Service after you click **Enable**.

   After you enable the IoT Platform log dump feature for the product, IoT Platform automatically creates a log storage location and a service-linked role that is used to export logs.

   - Log storage location:
     - Project: `iot-log-${uid}-${regionId}`. Replace the `${uid}` variable with your Alibaba Cloud account ID and the `${regionId}` variable with the ID of the region in which IoT Platform is activated.
     - Logstore: **iot-logs**.

     All products share the log storage location. You can find the logs of a product based on the specified ProductKey. For more information, see IoT Platform logs.
   - Service-linked role: You can use this role to obtain permissions to export logs. For more information, see AliyunServiceRoleForIoTLogExport service-linked role.
7. Specify the retention period of logs.

   If the specified retention period of exported logs ends, the logs are deleted. By default, IoT Platform allows you to retain the logs of the last seven days. You can set the retention period to 1 to 3,000 days or permanent.

On the **IoT Platform Log Dump** tab, click **Set log save time**. In the message that appears, click OK. On the Logstore Attributes page, click **Modify**, set the **Data Retention Period** parameter, and then click **Save**.

Use the log search and analysis feature

After you enable the IoT Platform log dump feature, you can perform the following operations on the tabs of the **IoT Platform Log Dump** tab:

- **Raw Logs** tab

Operation	Description
Query and analyze logs	On the **Raw Logs** tab, you can use an SQL statement to query logs within a specified period of time and perform statistical analysis. For more information about SQL syntax, see Search syntax and Log analysis overview.
View charts	On the **Graph** tab, you can view the generated charts.
Aggregated logs	On the **LogReduce** tab, you can view aggregated logs with high similarity to get a full view of logs.
Perform quick analysis	After you search for logs, you can view the distribution of a field based on the search results.
Configure an alert rule	If an alert condition is met, IoT Platform sends an alert by using a notification method such as a text message, phone call, email, or DingTalk notification.

- **Log Report** tab

Operation	Description
View reports	You can view log reports in a specified time range. Log reports reflect the statuses and exceptions of devices. For more information about log reports, see the following *Log reports* table. By default, a log report shows the data of the last one hour (time frame). The time interval for a line chart is one minute. ◦ To specify a time range, you can click Time Range in the upper-right corner of the Log Report tab or choose > **Select Time Range** in the upper-right corner of a report chart. ◦ To set a time interval for a line chart, you can write an SQL statement on the **Raw Logs** tab. For more information about SQL syntax, see Date and time functions. For example, you can use the `bizCode:device | SELECT date_format(date_trunc('hour',__time__), '%m-%d %H:%i') AS Time, count(1) AS count , operation GROUP BY Time, operation ORDER BY Time limit 1440` statement to set the time interval to one hour.

Operation	Description
Subscribe to reports	IoT Platform converts reports into images on a regular basis and sends these images to the specified contacts by using emails or DingTalk chatbots.

Log reports

Report	Chart type	Description
Number of Times Device Online & Offline	Line chart	The line chart shows the number of times that devices are connected to IoT Platform and the number of times that devices are disconnected from IoT Platform in a specified period of time.
Device Uplink & Downlink Messages	Line chart	The line chart shows the number of upstream messages and the number of downstream messages in a specified period of time.
Top 20 Devices by Uplink&Downlink Message	List	The list shows the top 20 devices that send or receive the most upstream or downstream messages, and the number of messages of each device.
Error Distribution for Data Parsing Script	Pie chart	The pie chart shows the distribution of data parsing errors in a specified period of time. You can search for logs by error code and view the details of data parsing errors in logs. This helps efficiently improve a data parsing script.
Top 10 Devices by Data Parsing Script Error	List	The list shows the top 10 devices on which the most data parsing errors occur, and the number of errors that occur on each device in a specified period of time. You can search for logs by device name and view the details of data parsing errors in logs. This helps efficiently improve a data parsing script.
Error Distribution for TSL Validation	Pie chart	The pie chart shows the distribution of Thing Specification Language (TSL) validation errors in a specified period of time. You can search for logs by error code, view the details of TSL validation errors in logs, and troubleshoot errors.
Top 10 Devices by TSL Validation Error	List	The list shows the top 10 devices on which the most TSL validation errors occur, and the number of errors that occur on each device in a specified period of time. You can search for logs by device name, view the details of TSL validation errors in logs, and troubleshoot errors.

Report	Chart type	Description
Server-side Subscription Messages Forwarded	Line chart	The line chart shows the total number of messages that are forwarded by an Advanced Message Queuing Protocol (AMQP) or Message Service (MNS) server-side subscription in a specified period of time.
Last 20 Device Anomaly Messages	Pie chart	The pie chart shows the last 20 device anomaly messages. You can search for logs by device name and error code, view the details of device anomaly messages in logs, and troubleshoot errors.
Cloud Service Messages Forwarded	Line chart	The line chart shows the total number of messages that are forwarded by the data forwarding feature in a specified period of time.
Last 20 Data Forwarding Errors	List	The list shows the last 20 data forwarding errors of the rules engine in a specified period of time. You can search for logs by device name and error code, view the details of data forwarding errors for the rules engine in logs, and troubleshoot errors.
Error Distribution for Cloud API Calls	Pie chart	The pie chart shows the distribution of errors in API calls in a specified period of time. You can search for logs by API operation name and troubleshoot errors based on log details and error codes.

Disable the IoT Platform log dump feature

You can disable the IoT Platform log dump feature for a specific product at any time as needed to save storage space.

1. Log on to the IoT Platform console . On the **Overview** page, find the instance that you want to manage and click the instance.
2. In the left-side navigation pane, choose **Maintenance > Device Log** .
3. Select a product and click the **IoT Platform Log Dump** tab.
4. Click **Stop Dump**. In the message that appears, click **OK**.
   After the IoT Platform log dump feature is disabled, the newly generated IoT Platform logs are no longer exported to your Logstore. Previous logs are not deleted until the specified retention period of these logs ends.

4.4. Dump local device logs

By default, the device log feature of IoT Platform allows you to retain the logs of the last seven days. This feature also allows you to export local device logs to a Logstore of Log Service for persistent storage. After you enable the local log dump feature, you can query and analyze logs and configure alerts in the IoT Platform console.

Prerequisites

- An SDK for Android or SDK for C is used to develop a device. Logs are submitted by the device. For more information about how to reset a password, see Device log reporting.
- Device local log reporting is turned on. To turn on the switch, you must log on to the IoT Platform and perform the following steps: 1. Choose **Devices > Devices**. 2. Find the required device in the device list and click **View**. 3. On the **Device Details** page, turn on **Device local log reporting**.
- Log Service is activated. For more information, see Resource management overview.

Enable the local log dump feature

> **Note** The log dump feature is available only for Alibaba Cloud accounts.

You must enable the log dump feature for each product.

1. Log on to the IoT Platform console.
2.
3. In the left-side navigation pane, choose **Maintenance > Device Log** .
4. Select a product and click the **Local Log Dump** tab.
5. Click **Enable**.
6. In the Log Configurations message, read the instructions and click **OK**.

   > **Note** If Log Service is not activated, you are navigated to the activation page of Log Service after you click **Enable**.

   After you enable the local log dump feature for the product, IoT Platform automatically creates a log storage location and a service-linked role that is used to export logs.

   - Log storage location:
     - Project: `iot-log-${uid}-${regionId}`. Replace the `${uid}` variable with your Alibaba Cloud account ID and the `${regionId}` variable with the ID of the region in which IoT Platform is activated.
     - Logstore: **iot-logs-device-local**.

     All devices of the product share the log storage location. You can find the logs of a specific product based on the ProductKey. For more information, see Local device logs.
   - Service-linked role: You can use this role to obtain permissions to export logs. For more information, see AliyunServiceRoleForIoTLogExport service-linked role.
7. Specify the retention period of logs.

   If the specified retention period of exported logs ends, the logs are deleted. By default, IoT Platform allows you to retain the logs of the last seven days. You can set the retention period to 1 to 3,000 days or permanent.

   On the **Local Log Dump** tab, click **Set log save time**. In the message that appears, click OK. On the Logstore Attributes page, click **Modify**, set the **Data Retention Period** parameter, and then click **Save**.

Use the log search and analysis feature

After you enable the local log dump feature, you can perform the following operations on the tabs of the **Local Log Dump** tab.

Operation	Description
Query and analyze logs	On the **Raw Logs** tab, you can use an SQL statement to query logs within a specified period of time and perform statistical analysis. For more information about SQL syntax, see Search syntax and Log analysis overview.
View charts	On the **Graph** tab, you can view the generated charts.
Aggregated logs	On the **LogReduce** tab, you can view aggregated logs with high similarity to get a full view of logs.
Perform quick analysis	After you search for logs, you can view the distribution of a field based on the search results.
Configure an alert rule	If an alert condition is met, IoT Platform sends an alert by using a notification method such as a text message, phone call, email, or DingTalk notification.

Disable the local log dump feature

You can disable the local log dump feature for a specific product at any time as needed to save storage space.

1. Log on to the IoT Platform console . On the **Overview** page, find the instance that you want to manage and click the instance.
2. In the left-side navigation pane, choose **Maintenance > Device Log** .
3. Select a product and click the **Local Log Dump** tab.
4. Click **Stop Dump**. In the message that appears, click **OK**.
   After the local log dump feature is disabled, the newly generated local device logs are no longer exported to your Logstore. Previous logs are not deleted until the specified retention period of these logs ends.

5.OTA update

5.1. Overview

IoT Platform provides the over-the-air (OTA) update and management feature. Before you update devices, make sure that your devices support the OTA service. If your devices support the OTA service, you can upload an update package on the OTA Update page in the IoT Platform console and specify information about the devices that you want to update. IoT Platform pushes OTA update notifications to the devices. The devices can download the OTA update package and perform OTA updates. This topic describes the limits and procedure for OTA updates.

Prerequisites

Before you use the OTA update feature, make sure that your devices support the OTA service.

- For more information about how to perform OTA updates by using device SDKs, see Perform OTA updates.
- For more information about how to perform OTA updates if AliOS Things chips are installed on your devices, see OTA tutorial for AliOS Things.

Usage notes and limits

Devices

Feature	Limits
Supported protocol	Only the devices that are connected to IoT Platform over Message Queuing Telemetry Transport (MQTT) support the OTA update feature.
OTA update of a distributed device	After a device is distributed to the destination instance, you can perform OTA updates on the device.

Update packages

Feature	Limits
Update packages	Each Alibaba Cloud account can have up to 500 update packages.
	The maximum size of an update package file is 2,000 MB. The file format can be `.bin`, `.dav`, `.tar`, `.gz`, `.zip`, `.gzip`, `.apk`, `.tar.gz`, `.tar.xz`, or `.pack`.

Update batches

Feature	Usage notes and limits

Feature	Usage notes and limits

Version-based updates	Limits:
	• Static update: You can use an update package to create one or more static update batches for different versions to be updated.
	• Dynamic update:
	  ◦ You can use an update package to create only one dynamic update batch. If you set the **Upgrade range** parameter to **All Devices**, you can specify multiple versions for the **Version number to be upgraded** parameter.
	  ◦ If you set the **Upgrade range** parameter to **All Devices** and specify the versions to be updated, up to 10 updates can be performed on a device in a dynamic update batch.
	  ◦ You can use multiple update packages to create multiple dynamic update batches for the same version. In this case, only the most recent dynamic update batch takes effect.

	Usage notes: • If you set the **The device supports simultaneous updates of multiple modules** parameter to **No** when you initiate an update batch for a device, the device can have only one ongoing update task for one module at the same time. Otherwise, an update conflict occurs. In this case, the update task can be in the To Be Pushed, Pushed, or In Upgrade state. • If you set the **The device supports simultaneous updates of multiple modules** parameter to **Yes** in an update batch for a device, the device can have multiple ongoing update tasks for different modules at the same time. In this case, the update tasks can be in the To Be Pushed, Pushed, or In Upgrade state. **Notice** In this case, the device must use Link SDK V4.x for C.
Group-based updates	Usage notes: If you delete a device group or remove a device from the group, the existing OTA task of the group is not affected. For more information about how to create and use a device group, see Device groups.

Concurrent updates on multiple modules

Feature	Description
Supported regions	Japan (Tokyo).
Supported instances	Enterprise Edition instances.
Supported Link SDKs	Link SDK V4.x for C.
Update policy configurations	You can set the **The device supports simultaneous updates of multiple modules** or **Override Previous Device Update Tasks** parameter to **Yes**.

Feature	Description
Limits	• You can run up to five update tasks on different modules on a device at the same time. • You can perform concurrent updates on multiple modules or overwrite previous updates in different update batches on the same device of a product. For example, you cannot use one or more update packages to set the **The device supports simultaneous updates of multiple modules** parameter to **Yes** and the **Override Previous Device Update Tasks** parameter to **Yes** for different batch updates to update the same product. If you want to update your devices, do not perform the preceding operations.

Procedure

1. Develop the OTA update feature: Configure the remote OTA update feature for a device.
2. Add an update package: Add an OTA module and an update package to a product.
3. Verify an update package (Optional): Verify the update package.

   > **Note**
   > - If you set the **Verify Update Package?** parameter to **No** when you add the update package, skip this step.
   > - After the tested devices are updated and the status of the update package changes to **Verified**, you can perform a batch update.

4. Initiate a batch update: IoT Platform sends OTA update information to the specified devices. The information includes the URL, version, and size of the update package.

   > **Note** When you initiate a batch update, you can set the **Whether IoT Platform Actively Pushes Update Task** parameter to one of the following values:
   > - **Yes**: IoT Platform pushes OTA update information to online devices.
   > - **No**: IoT Platform does not push OTA update information to devices. A device must send a request to IoT Platform to obtain the OTA update information.
   >
   > For more information, see Step 4 that is described in the Message formats section of the Perform OTA updates topic.

5. Perform OTA updates: Devices obtain the OTA update information, use the URL to download the update package, and then perform OTA updates.

   > **Note** The URL of an update package remains valid for 24 hours. A device can download the update package within the specified period. If the update package is not downloaded within 24 hours, the device can request the update information from IoT Platform and download the update package. For more information, see the following scenarios.

   Scenarios:

- A device immediately downloads the update package and performs an OTA update. For more information, see Step 5 that is described in the Message formats section of the "Perform OTA updates" topic.
- A device does not immediately download the update package. The device requests the update information from IoT Platform during off-peak hours and then performs an OTA update. For more information, see Step 4 that is described in the Message formats section of the "Perform OTA updates" topic.

6. View update status: View the update status of the devices and the information about the update package.
7. View statistics on update package versions and success rates: View the statistical information about the versions and the success rate of the update tasks after the update is completed. You can also identify the causes of update failures to improve the success rate.

References

- For more information about OTA update examples, see Configure OTA updates for devices.
- For more information about how to troubleshoot and fix OTA errors, see How do I troubleshoot OTA update errors.

5.2. Push an update package to devices

5.2.1. Add an update package

To perform over-the-air (OTA) updates on IoT devices, you must add an update package to a product in the IoT Platform console. This topic describes how to add an update package.

Prerequisites

The OTA update feature is enabled for your devices. For more information about how to configure Link SDK to perform an OTA update, see Overview.

If you add an update package that contains multiple files, the devices must support update tasks that are initiated by using multiple update package files. For more information, see Sample code.

> **Notice** Only Link SDK for C allows you to develop the OTA update feature by using an update package that contains multiple files.

Context

For more information about how to use the OTA update feature, see Overview. For more information about how to perform OTA updates, see Perform OTA updates.

Procedure

1. Log on to the IoT Platform console.
2.
3. In the left-side navigation pane, choose **Maintenance > OTA Update**.

> **Note** To provide better services, IoT Platform improves the OTA update feature and adds statistics on update package versions. When you use the new OTA update feature in the console for the first time, you must associate the uploaded update packages with products. You can associate an update package with only one product. For more information about how to associate update packages with products, see the instructions in the console.

4. Optional. If AliOS Things chips are installed on your devices, you can enable the secure update feature.

   We recommend that you enable this feature to ensure the integrity and security of update packages. If you use the secure update feature, verify the update package and update package signatures of devices. For more information, see OTA Tutorial for AliOS Things.

   i. On the **OTA Update** page, click **Secure Update**.

   ii. In the Secure Update panel, find the product to be updated and turn on the switch in the **Secure Update** column.

   When the secure update feature is in the **Activated** state, click **Copy** in the Public Key column to copy the public key. You can use the public key to verify the signature of a device.

5. Optional. Add a custom OTA module.

   An OTA module is the module of the devices to be updated in a product. OTA modules include the firmware, software, and driver. The default module is the firmware of a device. You can also customize an OTA module.

   On the **Modules** tab, click **Add Module**. In the Add Module dialog box, set the parameters and click **OK**. The following table describes the parameters.

Parameter	Description
Product	The product to which the module belongs.
Module Name	The name of the module. The module name must be unique in a product. The module name cannot be modified after the module is added. The module name must be 1 to 64 characters in length, and can contain letters, digits, periods (.), hyphens (-), and underscores (_).
Module Alias	The alias of the module. The module alias must be 4 to 64 characters in length, and can contain letters, digits, periods (.), hyphens (-), and underscores (_).
Module Description	The description of the module. The module description can be up to 100 characters in length.

6. On the **OTA Update** page, click the **Update Packages** tab and click **Add Update Package**.
7. In the Add Update Package dialog box, set the parameters as required, upload an update package file, and then click **OK**. The following table describes the parameters.

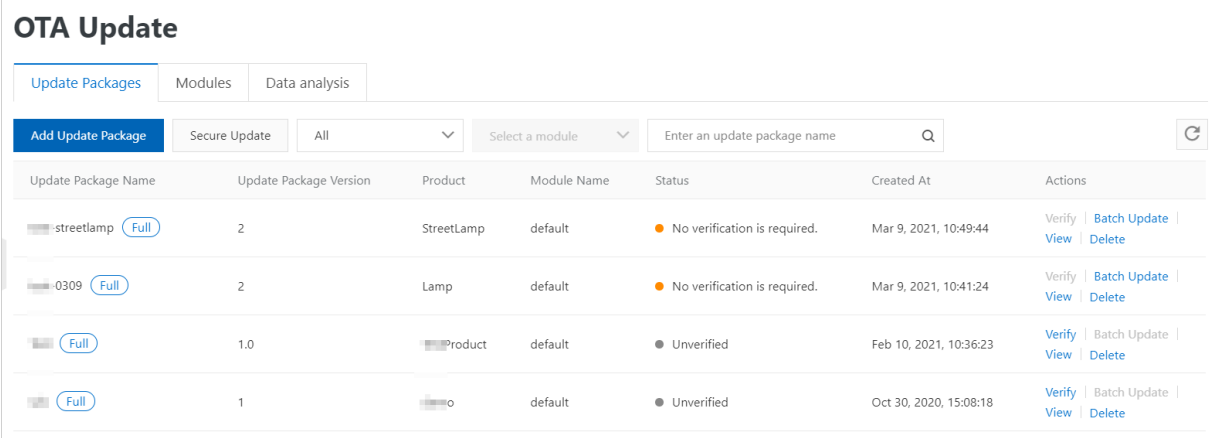
Parameter	Description

Parameter	Description
Types of Update Packages	◦ The type of the update package. Valid values: Full: If you select Full, you must upload a complete update package. IoT Platform pushes the complete update package to devices for update. ◦ Differential: If you select Differential, you must upload a file that contains only the differences between the previous update package version and the new update package version. IoT Platform pushes the differences to devices for update. Then, the differences are merged into the original update packages. Differential updates save device resources and reduce the traffic that is consumed when IoT Platform pushes update packages.
Update Package Name	The name of the update package. The update package name must be unique within an Alibaba Cloud account. The update package name cannot be modified after the update package is added. The name must be 1 to 40 characters in length, and can contain letters, digits, hyphens (-), underscores (_), and parentheses (). The name must start with a letter or digit. After you add the update package, you can click **Edit** on the **Update Package Information** tab of the **Update Package Details** page to modify the name of the update package.
Product	The product to which the update package belongs.
Update Package Module	The OTA module to which the update package applies. You can click **Add Module**. In the Add Module dialog box, set the parameters as required and click OK to add a module.
Update Package Version	The version number of the update package. The version number must be 1 to 64 characters in length, and can contain letters, digits, periods (.), hyphens (-), and underscores (_). This parameter is required if you set the Types of Update Packages parameter to **Full**.
Version number to be upgraded	The version number for the OTA module of the devices to be updated. The drop-down list displays the OTA module versions of all devices in the current product. You can enter a version number in the field, or select a version from the drop-down list. This parameter is required if you set the Types of Update Packages parameter to **Differential**.
Post-upgrade version number	The version number of the update package. This parameter is required if you set the Types of Update Packages parameter to **Differential**.
Signature Algorithm	The signature algorithm. Valid values: MD5 and SHA256. If you use Link SDK for Android and set the Types of Update Packages parameter to **Differential**, select the MD5 algorithm.

Parameter	Description
Select Update Package	The update package files to upload. You can upload a maximum of 20 files. The total file size cannot exceed 2,000 MB. The file format can be `.bin`, `.dav`, `.tar`, `.gz`, `.zip`, `.gzip`, `.apk`, `.tar.gz`, `.tar.xz`, or `.pack`. Take note of the following items when you upload multiple update package files: ◦ Each file name must be unique. Each file name cannot exceed 32 characters in length. ◦ IoT Platform checks whether the signatures of multiple update package files are the same. If the signatures are the same, IoT Platform determines that these files are duplicate.
Verify Update Package?	Specifies whether to verify the update package on several devices before you perform a batch update. Valid values: ◦ **Yes**: You can perform a batch update only after the update package is verified. ◦ **No**: You can perform a batch update without the need to verify the update package.
Update Package Description	The description of the update package. The description can be up to 1,024 characters in length. After you add the update package, you can click **Edit** on the **Update Package Information** tab of the **Update Package Details** page to modify the update package description.
Custom Information Pushed to Device	The custom information that you want to send to a device. After you add the update package and create an update task, IoT Platform sends the custom information to the specified device when IoT Platform pushes an update notification. The custom information can be up to 4,096 characters in length. No limits are applied to the content or format. After you add the update package, you can click **Edit** on the **Update Package Information** tab of the **Update Package Details** page to modify the custom information.

Results

After you add an update package, you can view the update package on the **Update Packages** tab, as shown in the following figure.



What to do next

If you set the **Verify Update Package?** parameter to **Yes** when you add an update package, you must verify the update package before you perform a batch update. For more information, see Verify an update package (Optional).

If you set the Verify Update Package? parameter to No when you add an update package, you can perform a batch update without the need to verify the update package. For more information, see Initiate a batch update.

Related API operations

Operation	Description
GenerateOTAUploadURL	Generates the URL and details of an update package to be uploaded to Object Storage Service (OSS).
CreateOTAFirmware	Adds an update package.
CreateOTAModule	Adds an OTA module for a product.
UpdateOTAModule	Modifies the alias and description of an OTA module.
DeleteOTAModule	Removes a custom OTA module.
ListOTAModuleByProduct	Queries the OTA modules of a product.
DeleteOTAFirmware	Removes an update package.
ListOTAFirmware	Queries update packages.

For more information about API operations related to the OTA update feature, see OTA updates.

5.2.2. Verify an update package (Optional)

If you set the **Verify Update Package?** parameter to **Yes** when you add an update package, you must verify the update package on tested devices before you perform a batch update. The update package can be pushed to devices for an over-the-air (OTA) update only after tested devices are updated. This topic describes how to verify an update package in the IoT Platform console.

Prerequisites

An update package is added. For more information, see Add an update package.

Procedure

1. Log on to the IoT Platform console.
2.
3. In the left-side navigation pane, choose **Maintenance > OTA Update**.

   > **Note** To provide better services, IoT Platform improves the OTA update feature and adds statistics on update package versions. When you use the new OTA update feature in the console for the first time, you must associate the uploaded update packages with products. You can associate an update package with only one product. For more information about how to associate update packages with products, see the instructions in the console.

4. On the Update Packages tab, find the update package that you want to verify and click **Verify** in the Actions column. In the Verify Update Package dialog box, set the parameters as required and click **OK** to verify the update package on one or more devices. The following table describes the parameters.

Parameter	Description
Version number to be upgraded	◦ The version number for the OTA module of the devices to be updated. If you perform a full update, this parameter is optional. The drop-down list displays the OTA module versions of all devices in the current product, except for the version to be updated to. You can select one or more versions. After you select the required versions, the related devices are added to the **Device to be verified** drop-down list as candidate devices. If you do not set this parameter, no limit is set on the OTA module versions of the devices to be verified. ◦ If you perform a differential update, the value of this parameter is the version number that you specify when you add the update package.
Device to be verified	The devices to be verified. You can select one or more devices.
Whether IoT Platform Actively Pushes Update Task	Specifies whether IoT Platform automatically pushes update tasks to devices. Valid values: ◦ **Yes**: After an update batch is created, IoT Platform automatically pushes update tasks to the specified online devices. This is the default value. In this case, a device can still initiate a request to obtain the information about the OTA update task from IoT Platform. ◦ **No**: A device must initiate a request to obtain the information about the OTA update task from IoT Platform.

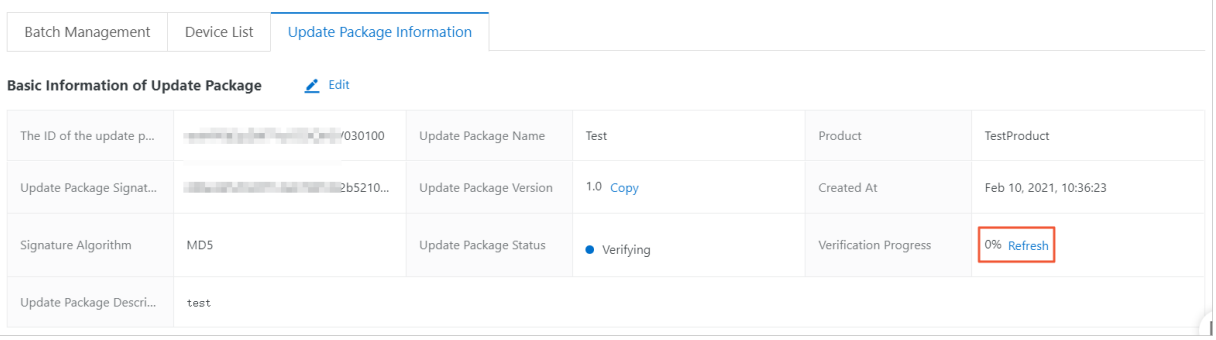
Parameter	Description
APP Confirm Upgrade	Specifies whether to control the update by using a mobile app. You must develop the mobile app as needed. Valid values: ◦ **Yes**: To perform an OTA update on a device, you must confirm the update by using your mobile app. You can call the ConfirmOTATask operation to confirm multiple update tasks that are pending for confirmation at a time. Then, the device can obtain the information about the OTA update task based on the setting of the **Whether IoT Platform Actively Pushes Update Task** parameter. ◦ **No**: A device obtains the information about the OTA update task based on the setting of the **Whether IoT Platform Actively Pushes Update Task** parameter. This is the default value.
Update Package Download Protocol	The protocol used to download the update package. Valid values: **HTTPS** and **MQTT**. After a device receives the update package URL pushed by IoT Platform, this protocol is used to download the update package. **Notice** If you need to download the update package by using the Message Queuing Telemetry Transport (MQTT) protocol, take note of the following items: ◦ Your service must be deployed in the China (Shanghai), China (Beijing), or China (Shenzhen) region. ◦ The OTA update package can contain only one file, and the size of the file cannot exceed 16 MB. ◦ You must use the latest version of Link SDK for C to develop the device features to perform OTA updates and download files over MQTT. For more information, see Sample code.

Parameter	Description
Device upgrade time-out (minutes)	The timeout period of the update for a single device. If a specified device has not been updated within this period, the update times out. Valid values: 1 to 1440. Unit: minutes. If you perform an OTA update by using the update package for the first time, we recommend that you set this parameter to its maximum value. This increases the success rate of the update. The console shows a recommended value based on update records. **Note** ○ The update period starts from the first time the specified device reports the update progress. During the update, the update package may be repeatedly pushed to the specified device because the device goes online and offline multiple times. However, the start time of the update period does not change. ○ After the device is updated, the device must immediately report the updated version number. Otherwise, the update may fail due to a timeout error. For example, you set the timeout period to 60 minutes and the device first reports the update progress at 10:00. If the device does not report the updated version number before 11:00, the update fails.
Batch label	Click **Add Tag**. In the fields that appears, specify the tag key and tag value. If an update batch that you created is in the **updating** state, you can modify the tag or add more tags. For more information, see Manage update batches. The tags of an update batch are sent to devices when IoT Platform pushes update notifications to these devices. You can move the pointer over the icon to view the rules based on which you can configure tags.

Results

In the **Verify Update Package** dialog box, click **Close**.

On the **Update Packages** tab, find the update package that is in the **Verifying** state and click **View** in the Actions column. On the **Update Package Information** tab, you can view the verification progress.



What to do next

After tested devices are updated and the status of the update package is displayed as **Verified**, you can perform a batch update. For more information, see Initiate a batch update.

Related API operations

Operation	Description
CreateOTAVerifyJob	Creates a verification task for an update package.
QueryOTAFirmware	Queries the details of an update package.

For more information about API operations related to the OTA update feature, see OTA updates.

5.2.3. Initiate a batch update

This topic describes how to push an update package to multiple devices at a time in the IoT Platform console to perform an over-the-air (OTA) update.

Prerequisites

The following operations are performed:

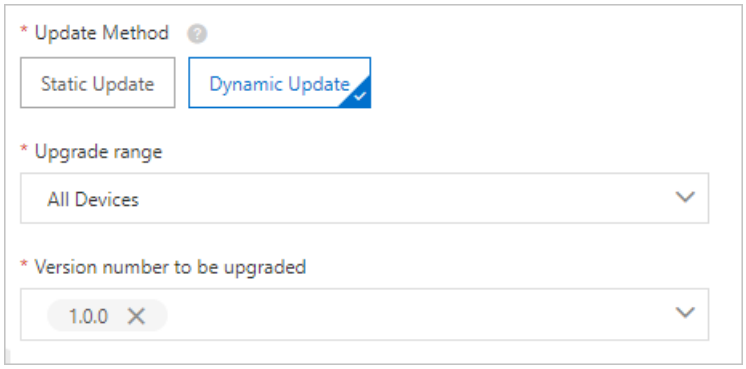
1. An update package is added.
2. Optional. The update package is verified.

Procedure

1. Log on to the IoT Platform console.
2.
3. In the left-side navigation pane, choose **Maintenance > OTA Update**.

   > **Note** To provide better services, IoT Platform improves the OTA update feature and adds statistics on update package versions. When you use the new OTA update feature in the console for the first time, you must associate the uploaded update packages with products. You can associate an update package with only one product. For more information about how to associate update packages with products, see the instructions in the console.

4. On the Update Packages tab, find the update package that you want to manage and click **Batch Update** in the Actions column. In the Update Scope Configuration step, configure the parameters and click **Next**. The following table describes the parameters.

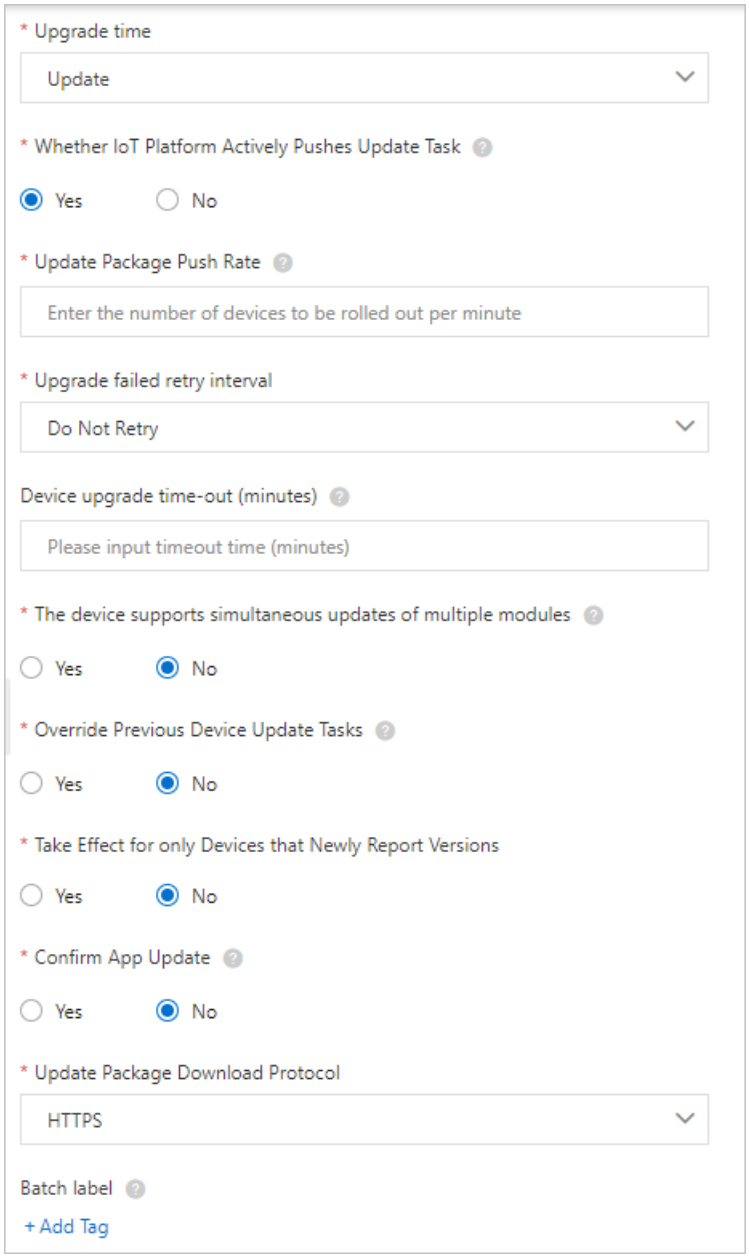


Parameter	Description
Update Method	The type of the update. Valid values: ○ Static Update: updates only the existing devices that meet the required conditions. ○ Dynamic Update: continuously updates the devices that meet the required conditions. Dynamic updates can be performed in the following scenarios: ■ The devices that are activated after a dynamic update is performed meet the required conditions. ■ The current OTA module versions that the devices report do not meet the required conditions. However, the devices continue to report the OTA module versions that meet the required conditions. **Note** ■ You can use an update package to create only one dynamic update batch. If you have created a dynamic update batch by using an update package, you must cancel this dynamic update batch before you create another one. ■ A device can be updated at most 10 times in a version-based dynamic update batch. If the device has been updated 10 times, no update can be initiated on the device even if the required conditions are subsequently met for a dynamic update.

Parameter	Description
Upgrade range	The scope of the update. Valid values: ○ All Devices: updates all devices that meet the required update conditions in the specified product. ○ Selected Devices: updates only the specified devices. If you select Selected Devices, you can use one of the following methods to select the required devices: ■ **Select**: Select the devices that you want to update from the Device Range drop-down list. If you use an Enterprise Edition instance in the Japan (Tokyo) region, you can use the Advanced search feature to search for devices. You can also download a CSV file that contains the names of the matched devices. ■ **Upload File**: Download a template file in the .csv format, enter the names of the required devices in the template file, and then upload the template file. Each template file can contain a maximum of 1,000,000 records. If a template file contains one or more invalid device names, an error occurs. Click **Download Invalid Device Name List** to download the file that contains invalid device names. Then, modify and re-upload the template file. ○ Phased Update: updates the specified devices. This option is displayed only if you set the Update Method parameter to Static Update. You must specify at least one device for a phased update. If you select Phased Update, the Update Adoption Rate (%) field appears. You must specify a percentage for the specified devices in the field. IoT Platform calculates the number of devices that can be updated based on the specified percentage. The calculation result is rounded down. ○ Group Update: updates the specified device group. This option is displayed only if you set the **Update Method** parameter to **Static Update**. The Group list drop-down list displays all parent groups in the current instance. For more information about how to create a device group, see Device groups.

Parameter	Description
Version number to be upgraded	Before you configure this parameter, take note of the following items: ◦ If you perform a full static update, this parameter is optional. If you perform a full dynamic update, this parameter is required. If you set the Upgrade range parameter to **Selected Devices**, this parameter is not displayed. The Version number to be upgraded drop-down list displays the OTA module versions of all devices in the current product, except for the version to be updated to. You can select one or more versions. If you do not configure this parameter, no limit is specified for the version number of the OTA module of the devices to be updated. ◦ If you perform a differential update, the value of this parameter is the version number that you specify when you add the update package. If you set the **Update Method** parameter to **Dynamic Update** when you create an update batch and the update batch is in the **updating** state, you can change the value of this parameter.

5. In the Update Policy Configuration step, configure the parameters and click **Complete**. Then, IoT Platform pushes update notifications to devices. The following table describes the parameters.



Parameter	Description
Upgrade time	The time when you want to perform the OTA update. Valid values: ○ Update: immediately performs the OTA update. ○ Scheduled Update: performs the OTA update within a specified time range. You can specify a start time and an end time for the OTA update. The start time must be 5 minutes to 7 days later than the current time. The end time must be 1 hour to 30 days later than the start time. The end time is optional. If you do not specify an end time, the update is not forcibly stopped. **Note** Scheduled updates are supported only if you set the Update Method parameter to **Static Update**.

Parameter	Description
Whether IoT Platform Actively Pushes Update Task	Specifies whether IoT Platform automatically pushes update tasks to devices. ○ **Yes**: After an update batch is created, IoT Platform automatically pushes update tasks to the specified online devices. This is the default value. In this case, a device can still initiate a request to obtain the information about the OTA update task from IoT Platform. ○ **No**: The device initiates a request to obtain the information about the OTA update task from IoT Platform.
	The number of devices to which you want to push the download URL of the update package per minute. Valid values: **Constant Rate** and **Variable Rate**. **Notice** ○ This parameter is not displayed if you set the **Whether IoT Platform Actively Pushes Update Task** parameter to **No**. ○ The **Variable Rate** option is available only for Enterprise Edition instances. The following list describes the parameters: ○ If you specify **Constant Rate**, you must configure the **Update at Constant Rate** parameter. Valid values: 10 to 10000. The value must be an integer. Unit: devices per minute. After you specify a value for the Update at Constant Rate parameter, the push rate remains unchanged. For example, if you want to fix a high-risk vulnerability, you must push an update package to all devices to perform an update. In this case, we recommend that you set the Update Package Push Rate parameter to Constant Rate. You can set the maximum constant push rate to 10,000 devices per minute. This way, you can push the update package to the devices that you want to update at the earliest opportunity. ○ **Variable Rate**: In some cases, you may need to push an update package at a low rate first (such as one device per minute), and then continuously increase the push rate based on specific conditions. You can set the Update Package Push Rate parameter to **Variable Rate**. For example, you want to push an update package to all devices at a low rate when a new feature is added. This way, you can perform an update first on a small portion of devices at the beginning of the update process. Then, you can check whether the updated devices run as expected and determine whether to increase the push rate based on the check result. This process is similar to the process of performing a phased update before a full update to ensure a successful update of all devices. If you specify Variable Rate, you must configure the following parameters:

Parameter	Description
Update Package Push Rate	■ **Basic Push Rate**: specifies the number of devices to which you want to push an update package per minute if the number of devices to which the update package is pushed or the number of updated devices does not reach the value of the Pushed Devices or Updated Devices parameter in the **Increase Push Rate** section. Valid values: 1 to 10000. The value must be an integer that is less than or equal to the value of the **Maximum Push Rate**parameter. ■ **Incremental Factor**: specifies an incremental factor based on which the system increases a push rate. This parameter applies if the number of devices to which the update package is pushed or the number of updated devices reaches the value of the Pushed Devices or Updated Devices parameter in the **Increase Push Rate** section. Valid values: 1.20 to 5.00. The value is rounded to two decimal places. ■ **Maximum Push Rate**: specifies the maximum number of devices to which you want to push the update package per minute. Valid values: 10 to 10000. The value must be an integer. If the rate at which the system pushes the update package reaches the maximum push rate, the system pushes the update package at the maximum push rate and no longer changes the push rate. ■ **Increase Push Rate**: specifies a threshold value for the **Pushed Devices** or **Updated Devices** parameter. Valid values: 1 to 100000. The value must be an integer. If the number of devices to which the update package is pushed or the number of updated devices reaches the threshold value, the system increases the push rate based on the value of the **Incremental Factor** parameter. Example: ■ You perform an OTA update by configuring the following settings: Set the Update Package Push Rage parameter to Variable Rate, set the **Basic Push Rate** parameter to 50, set the**Incremental Factor** parameter to 2, set the **Maximum Push Rate** to 10000, and then set the **Pushed Devices** parameter to 1000 in the **Increase Push Rate** section. ■ The following process describes how the OTA update is performed: An OTA update task is created to push an update package to 1,000 devices at a rate of 50 devices per minute at the beginning of the process. Then, the push rate increases based on the incremental factor.

Parameter	Description
	■ The following process describes how the system increases the push rate: a. The OTA update task pushes the update package to 1,000 devices at a rate of 50 devices per minute. Then, the system increases the push rate to 100 devices per minute based on the value of the **Incremental Factor** parameter. b. The OTA update task pushes the update package to another 1,000 devices at a rate of 100 devices per minute. A total of 2,000 devices have received the update package. Then, the system increases the push rate to 200 devices per minute. c. The OTA update task pushes the update package to another 1,000 devices at a rate of 200 devices per minute. A total of 3,000 devices have received the update package. Then, the system increases the push rate to 400 devices per minute. d. The OTA update task pushes the update package to another 1,000 devices at a rate of 400 devices per minute. A total of 4,000 devices have received the update package. Then, the system increases the push rate to 800 devices per minute. e. This way, the OTA update task pushes the update package at a rate of 800 devices per minute, 1,600 devices per minute, 3,200 devices per minute, and then 6,400 devices per minute until a total of 8,000 devices receive the update package and the push rate is increased to 12,800 devices per minute based on the value of the **Incremental Factor** parameter. The value 12800 exceeds the threshold value 10000 that you specified for the **Maximum Push Rate** parameter. Therefore, the OTA update task pushes the update package at the maximum push rate of 10,000 devices per minute and no longer changes the push rate.
Upgrade failed retry interval	The interval between an update failure and a retry after the failure. Valid values: ○ Do Not Retry ○ Retry Immediately ○ Retry in 10 Minutes ○ Retry in 30 Minutes ○ Retry in 1 Hour ○ Retry in 24 Hours **Notice** The value of the **Upgrade failed retry interval** parameter must be less than the value of the **Device upgrade time-out (minutes)** parameter. Examples: ○ If you set the timeout period to 60 minutes, the maximum retry interval that you can specify is 30 minutes. ○ If you set the timeout period to 1,440 minutes, the maximum retry interval that you can specify is 1 hour. If you want to set the **Upgrade failed retry interval** parameter to **Retry in 24 Hours**, we recommend that you do not configure the **Device upgrade time-out (minutes)** parameter. If an update times out, no retry is performed.
Max. Retry Times	The maximum number of retries that can be performed if an update fails. Valid values: ○ 1 ○ 2 ○ 5

If an update batch that you created is in the **updating** state, you can change the push rate that you specified to push an update package. However, you cannot change the push type from **Constant Rate** to **Variable Rate** or the other way around. For more information, see Manage update batches.

Parameter	Description
Device upgrade time-out (minutes)	The timeout period of the update for a single device. If a specified device has not been updated within this period, the update times out. Valid values: 1 to 1440. Unit: minutes. **Note** The update period starts from the first time the specified device reports the update progress. During the update, the update package may be repeatedly pushed to the specified device because the device goes online and offline multiple times. The start time of the update period remains unchanged. If you set the **Update Method** parameter to **Dynamic Update** when you create an update batch and the update batch is in the **updating** state, you can change the value of this parameter. For more information, see Manage update batches.
The device supports simultaneous updates of multiple modules	Specifies whether a device supports simultaneous updates on multiple modules. This parameter is displayed only for an Enterprise Edition instance or a public instance of the new version. Valid values: ○ **Yes**: A device supports simultaneous updates on multiple modules. In this case, IoT Platform uses the current update task to overwrite the previous update tasks for the same module and does not overwrite the update tasks that are in progress. **Notice** ■ Only a device that uses Link SDK V4.x for C supports simultaneous updates on multiple modules. For more information, see Overview. ■ You can set the **The device supports simultaneous updates of multiple modules** or **Override Previous Device Update Tasks** parameter to **Yes**. ■ The settings of the **The device supports simultaneous updates of multiple modules** and **Override Previous Device Update Tasks** parameters of a new dynamic update batch for a device group must be the same as those of the existing dynamic update batch for the device group. For more information, see the *Simultaneous updates of multiple modules* table in the Overview topic. ○ **No**: A device does not support simultaneous updates on multiple modules. Default value: No.

Parameter	Description
Override Previous Device Update Tasks	Specifies whether to overwrite the previous update task of a device. If a device has multiple update tasks, you must specify whether to use the current update task to overwrite the previous update tasks. An update task on a device can be in the Pending Confirmation, To Be Pushed, or Pushed state. Valid values: ○ **Yes**: Only the latest update task is performed. The previous update tasks are canceled. ○ **No**: Only the existing update task is performed. Default value: No. **Note** The update tasks that are in progress are not overwritten.
Take Effect for only Devices that Newly Report Versions	Specifies whether to update only the devices that subsequently report new OTA module versions. This parameter is displayed only if you set the Update Method parameter to **Dynamic Update**. Valid values: ○ **Yes**: updates only the devices that report new OTA module versions. ○ **No**: updates the existing devices that meet the update conditions and continuously checks whether the devices that report new OTA module versions meet the update conditions. Default value: No.
APP Confirm Upgrade	Specifies whether you can use a mobile app to perform the update. If you select Yes, you must develop the mobile app. Valid values: ○ **Yes**: To perform an OTA update on a device, you must confirm the update by using your mobile app. You can call the ConfirmOTATask operation to confirm multiple update tasks that are in the pending confirmation state at a time. Then, the device can obtain the information about the OTA update task based on the value of the **Whether IoT Platform Actively Pushes Update Task** parameter. ○ **No**: A device obtains the information about the OTA update task based on the setting of the **Whether IoT Platform Actively Pushes Update Task** parameter. Default value: No.

Parameter	Description
Update Package Download Protocol	The protocol that you want to use to download the update package. Valid values: **HTTPS** and **MQTT**. After a device receives the update package URL from IoT Platform, you can use this protocol to download the update package. **Notice** If you want to download the update package by using the Message Queuing Telemetry Transport (MQTT) protocol, take note of the following items: ○ Your service must be deployed in the China (Shanghai), China (Beijing), or China (Shenzhen) region. ○ The OTA update package can contain only one file, and the size of the file cannot exceed 16 MB. ○ You must use the latest version of Link SDK for C to develop the device features to perform OTA updates and download files over MQTT. For more information, see Sample code.
Batch label	Click **Add Tag**. In the fields that appears, specify the tag key and tag value. If an update batch that you created is in the **updating** state, you can modify the tag or add more tags. For more information, see Manage update batches. The tags of an update batch are sent to devices when IoT Platform pushes update notifications to these devices. You can move the pointer over the icon to view the rules based on which you can configure tags.

6. (Optional)On the Batch Management tab of the **Update Package Details** page, find the dynamic update batch and click **Edit** in the Actions column. In the **Update Scope Configuration** and **Update Policy Configuration** steps, you can change the values of the **Version number to be upgraded** and **Device upgrade time-out** parameters. You can cancel the timeout settings.

   Notice
   - Before you modify dynamic update settings, take note of the following items:
     - **Version number to be upgraded**: If you add a version number, the existing devices that use the version and new devices that match the dynamic policy are updated. If you remove a version number, the existing devices are not affected.
     - **Device upgrade time-out**: This parameter takes effect only for dynamic OTA updates on new devices. The existing devices are not affected.
   - You cannot change the value of the **Version number to be upgraded** parameter for a group-based dynamic update batch.

Result

After you initiate a batch update, IoT Platform pushes an update notification to the specified devices based on your settings. You can view the update status of each device and the update package information in the IoT Platform console. For more information, see View update status.

Related API operations

API	Description
CreateOTAStaticUpgradeJob	Creates a static update batch.
CreateOTADynamicUpgradeJob	Creates a dynamic update batch.
CancelOTAStrategyByJob	Cancels an update policy that is associated with a dynamic update batch.
CancelOTATaskByDevice	Cancels the pending device update tasks of an update package.
CancelOTATaskByJob	Cancels the device update tasks of an update batch.

For more information about API operations related to the OTA update feature, see OTA updates.

5.2.4. View update status

After you submit a batch update request, IoT Platform pushes an update notification to your devices based on your settings. You can view the update status of a device and the update package information in the IoT Platform console.

Prerequisites

A batch update request is submitted. For information about how to detach a tag policy from an object, see Initiate a batch update.

Procedure

1. Log on to the IoT Platform console.
2.
3. In the left-side navigation pane, choose **Maintenance > OTA Update**.

   > **Note** To provide better services, IoT Platform improves the OTA update feature and adds statistics on update package versions. When you use the new OTA update feature in the console for the first time, you must associate the uploaded update packages with products. You can associate an update package with only one product. For more information about how to associate update packages with products, see the instructions in the console.

4. On the Update Packages tab, find the update package that you want to use for a batch update and click **View** in the Actions column. Then, you can view the information and perform operations on the following tabs:
   - **Batch Management**
     - View the type of an update batch, such as Verify Update Package or Batch Update.
     - Find the update batch and click **View** in the Actions column. On the **Batch Details** page, you

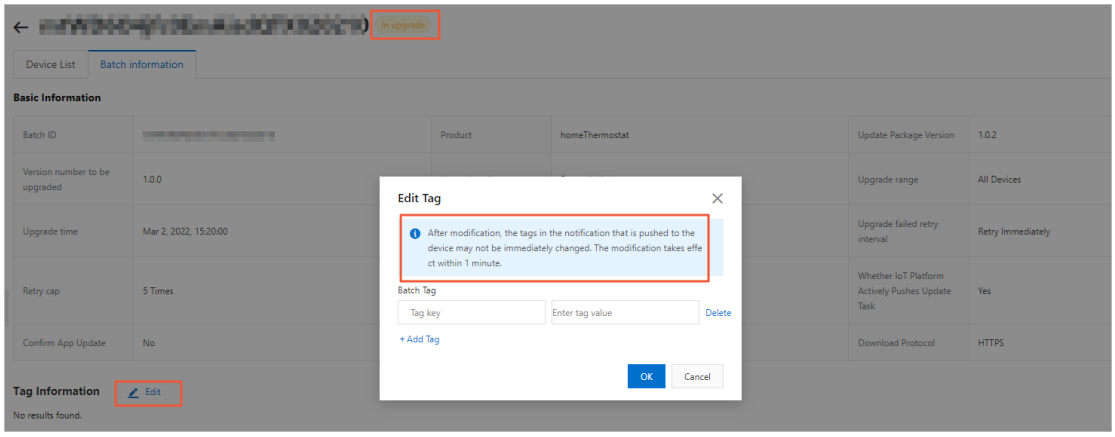
can view devices in different states on the **Device List** tab.

Status	Description
Pending Confirmation	The OTA update task has not been confirmed by using the mobile app. If you set the **APP Confirm Upgrade** parameter to **Yes** when you create an update batch, the update tasks in the batch are in this state.
To Be Pushed	The OTA update notification has not been pushed to the device. The update task may be in this state due to the following reasons: 1. The device is offline. 2. The notification is scheduled to be pushed. 3. The pushing rate exceeds the specified threshold value. The following states that correspond to these reasons are displayed: ■ **To Be Pushed (Device Offline)** ■ **To Be Pushed (Scheduled:XX XX, XXXX, XX:XX:XX)** ■ **To Be Pushed**
Pushed	The device has received the OTA update notification but has not submitted the update progress.
In upgrade	The device has received the OTA update notification and submitted the update progress.
Updated	The device has submitted a valid version number after the update. **Note** After a device is updated, make sure that the device submits a valid version number at the earliest opportunity. Otherwise, the update may fail due to a timeout issue.
	OTA updates may fail due to the following reasons: ■ For example, you initiate a new batch update for a device and you do not overwrite the previous update task that is not completed on the device. In this case, you can perform the following operations: ■ After the previous update task is completed, initiate a new update. ■ Before you initiate a new update for a device, overwrite the previous task that is running on the device. **Note** If the device is in the **In upgrade** state, an update task that is running on the device cannot be overwritten. ■ The device submits the value -1, the value -2, the value -3, or the value -4 that indicates that a failure occurs when the device uses a specific topic to submit the update progress to IoT Platform. Description of the values: ■ -1: The update failed. ■ -2: The download failed. ■ -3: The verification failed. ■ -4: The flashing failed.

Status	Description
Update Failed	■ During a device update, the first time that the device submits the update progress, the system starts calculating the update duration. If the device fails to submit the updated version number in the specified timeout period, the update fails. ■ The device that is in the **In upgrade** state submitted a version number that is not the source version or destination version. If you specify a version number to be updated and specify a retry interval when you initiate a batch update, retries are performed after the update fails. An update may fail due to one of the following reasons: ■ The device that is in the **In upgrade** state submitted a version number that is not the source version or destination version. ■ The device submits one of the following values when the device uses a specific topic to submit the update progress to IoT Platform: *-1*, *-2*, *-3*, and *-4*. During automatic retries, the update status of a device in IoT Platform remains unchanged. For example, if a retry is performed on a device that is in the **Pushed** state, the device status is displayed as **Pushed** in the IoT Platform console. If a retry is performed on a device that is in the **In upgrade** state, the device status is still displayed as **In upgrade** in the IoT Platform console. **Note** If an update fails due to one of the following reasons, automatic retries are not performed by IoT Platform: ■ The update timed out. ■ The update is canceled.
Canceled	The update for the device is canceled.

If an update batch is in the **In upgrade** state, the **Edit** button is displayed in the **Actions** column of the update batch. You can click **Edit**. In the Update Policy Configuration step of the page that appears, you can change the value of the **Update at Constant Rate** parameter if the **Update Package Push Rate** parameter is set to **Constant Rate** or change the values of the required parameters if the **Update Package Push Rate** parameter is set to Variable Rate. You can also modify existing tags or add new tags. For more information, see Initiate a batch update.

You can also click the **Batch information** tab to view the basic information about the batch. If the update batch is in the **In upgrade** state, the **Edit** button is displayed next to **Tag Information**. You can click **Edit** to modify or add tags.



- Find the update batch and click **Cancel** in the Actions column.
  - If you cancel a static update batch, all update tasks of the batch are automatically canceled, including the tasks that are in the Pending Confirmed, To Be Pushed, Pushed, and Updated states.
  - If you cancel a dynamic update batch, only the dynamic update policy is canceled. You can cancel all ongoing update tasks based on your business requirements, including the tasks that are in the To Be Pushed, Pushed, and In upgrade states.

  You can cancel all device update tasks that are in **Pending Confirmation**, **To Be Pushed**, or **Pushed** state.

- **Device List**

  View the information about the devices to which the update package is pushed, including the update status of devices, and the statistical information about successful updates, failed updates, and canceled updates.

  Find the device and click **View** in the Actions column. On the **Batch Details** page, you can cancel the update task of the device.

- **Update Package Information**
  - View the basic information about the update package, such as the ID, name, signature, signature algorithm, version number, status, and module of the update package.
  - To obtain the update package, click **Download** next to **Update Package Signature**.

What to do next

View statistics on update package versions and success rates: After the OTA update task is completed, you can go to the Data analysis tab to view the distribution of update package versions and the statistical information about the success rate of updates.

Related API operations

API	Description
ListOTAJobByFirmware	Queries the update batches of an update package.
ListOTAJobByDevice	Queries the update batches of an update package by device.

API	Description
ListOTATaskByJob	Queries the update tasks of a device by update batch.
QueryOTAJob	Queries the details of an update task.

For more information about API operations related to the OTA update feature, see OTA updates.

5.2.5. View statistics on update package versions and success rates

The over-the-air (OTA) update feature of IoT Platform provides statistics. On the Data analysis tab, you can view statistics on the update package versions of a product and the success rate of each update batch. This helps you troubleshoot OTA update failures and improve the success rate of device updates.

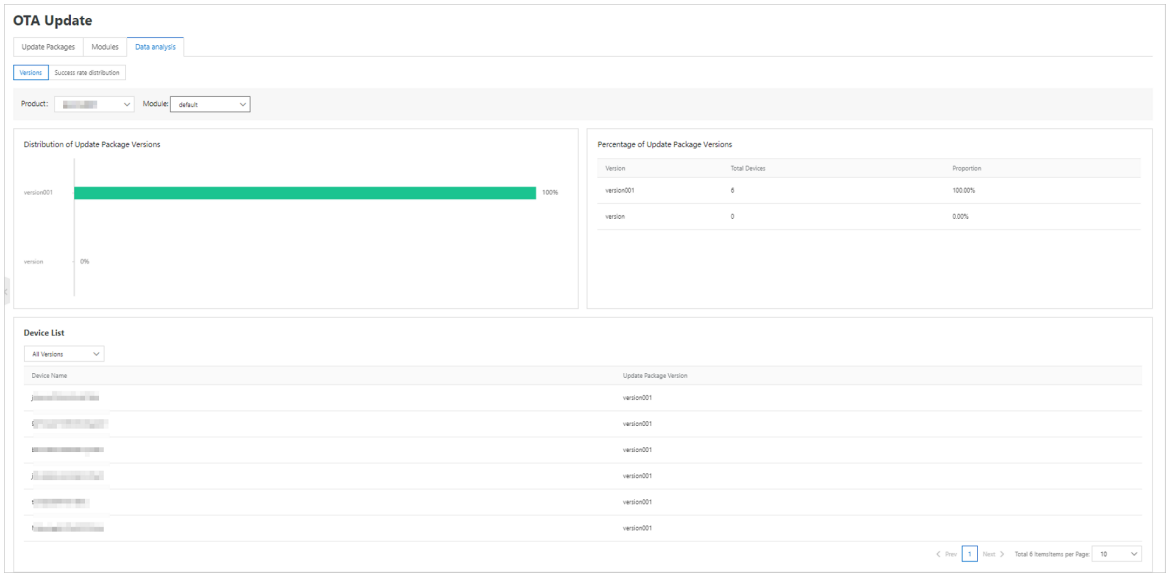
Overview

Metric	Chart type	Descripition
Versions		
Distribution of Update Package Versions	Bar chart	Displays update package versions in descending order based on quantities. The top five versions are displayed and the remaining versions are merged as Others. You can move the pointer over a bar to view the update package version.
Percentage of Update Package Versions	Table	Displays the proportions of update package versions in descending order. The columns of the table include Version, Total Devices, and Proportion.
Device List	Table	Displays update package versions and devices that correspond to each update package version. The columns of the table include Device Name and Update Package Version.
Success rate distribution		
Upgrade success rate	Donut chart	Displays the proportions of successful and failed updates of source versions in all batches or a single batch.

Metric	Chart type	Descripition
Total distribution of source versions	Bar chart or table	Displays the information about update packages in descending order based on the quantities of source versions. The detailed information includes Source version number, Number of upgrades, and Distribution proportion. • If the quantity of source versions is at most 10, only the top five versions are displayed in a bar chart. The remaining versions are merged as Others. You can move the pointer over the bar of a source version to view the information about the update package. • If the quantity of source versions is greater than 10, the information about update packages is displayed in a table.
Source version upgrade success rate ranking	Bar chart or table	Displays the information about update packages in descending order by success or failure rate based on the Total distribution of source versions metric. The detailed information includes Source version number, Success, Failure, and Success rate or Failure rate. Only top five source versions are displayed in a bar chart. An average value is calculated from the remaining source versions and displayed on the Others bar.
Source version upgrade failure rate ranking		
Failure Cause Distribution	Bar chart	Displays update failure causes in descending order based on percentages. The failure causes include: **Upgrade timeout**, **Version Error**, **Progress Error**, and **Upgrade Conflict**. For more information, see How do I troubleshoot OTA update errors?.

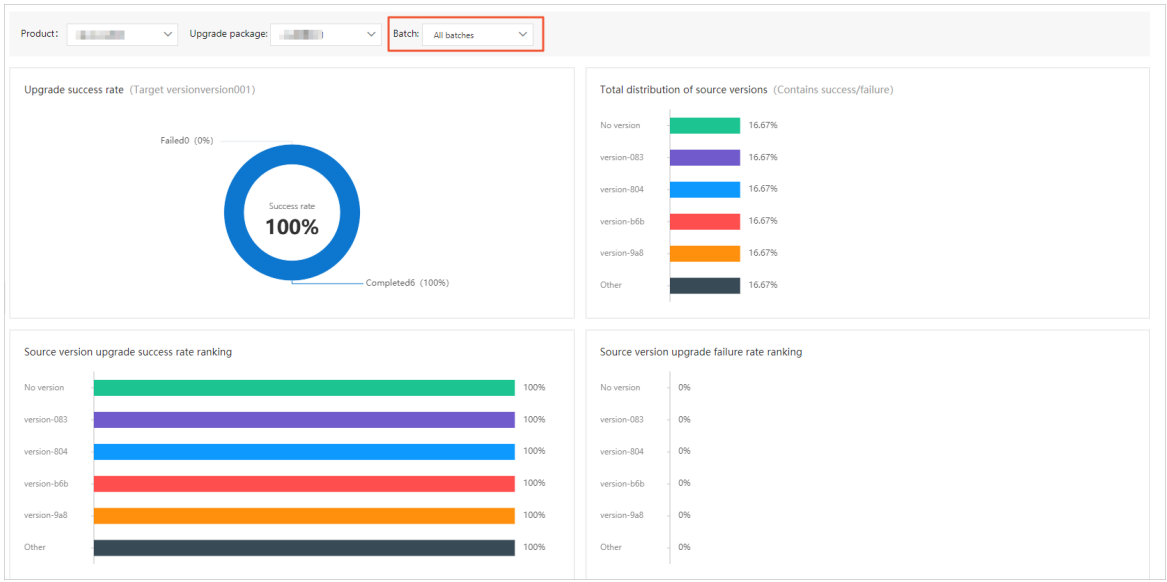
Procedure

1. Log on to the IoT Platform console.
2.
3. In the left-side navigation pane, choose **Maintenance > OTA Update**. On the OTA Update page, click the **Data analysis** tab.
4. On the **Versions** tab, select a product and a module.
   This tab displays the distribution information about module versions in the product. In the **Device List** section, you can select a version number and view the device information.



5. Click the **Success rate distribution** tab. Select a product and an update package.
   This tab displays statistics on the success rate of each update batch. You can select an update batch to view the statistics on the success rate of the update batch.

> **Note** Statistics are generated after OTA update tasks succeed or fail.

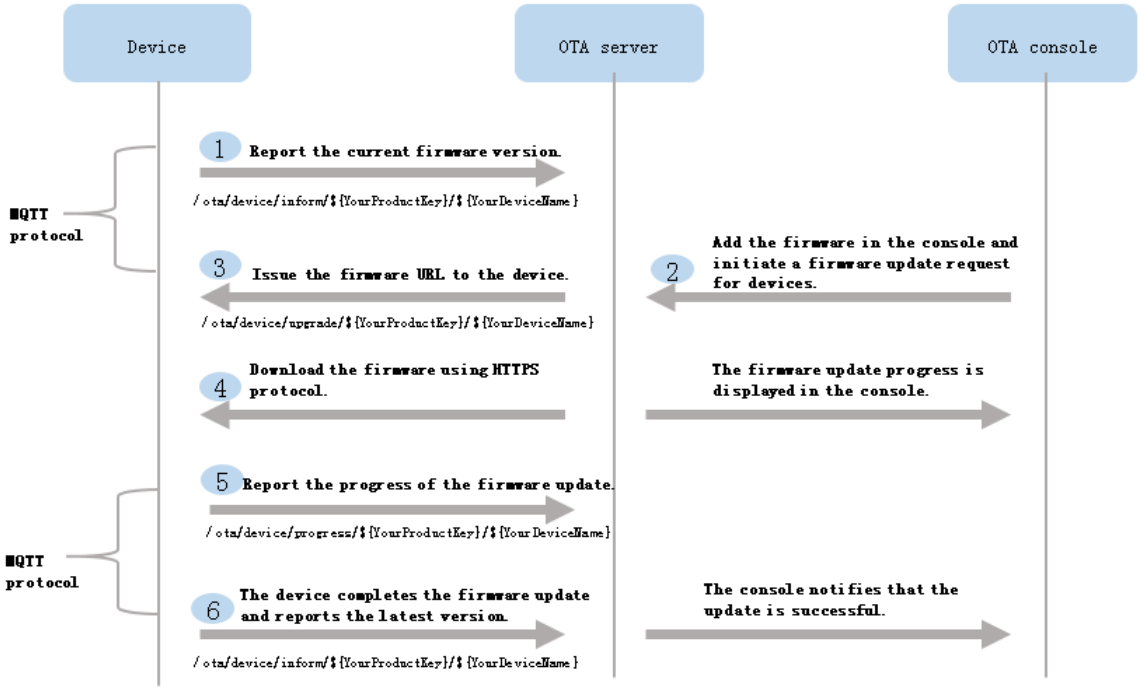


5.3. Perform OTA updates

IoT Platform supports the over-the-air (OTA) feature. You can use the OTA feature to update devices. This article describes how to use the OTA feature to update devices when Message Queuing Telemetry Transport (MQTT) is used to connect the devices to IoT Platform. It also describes topics and data formats that are used during data forwarding.

Procedure

The following figure shows the procedure of an OTA update over MQTT.



Notes

- To perform a differential update, the device must submit the version number of an OTA module. To perform a full update, the device does not need to submit the version number of an OTA module. If the version number is not submitted, you cannot specify a source version when you configure a batch update. For more information, see Initiate a batch update.

  A device needs to submit the version number only once during the startup before the first update. After the update succeeds, the device must immediately submit the current version number.

- In the IoT Platform console, after you start updating multiple devices, each device is in the Pending Update state.

  When IoT Platform receives the update progress that is submitted by a device, the status of the device changes to Updating.

  > **Note** After a device receives an update notification from IoT Platform, the device can download the update package and performs an update immediately or during off-peak business hours.

- IoT Platform checks whether an OTA update succeeds based on the version number that is submitted by the device.
- An offline device cannot receive an update notification from IoT Platform.

  After the status of the device changes to online, IoT Platform identifies whether the device needs to be updated. If an update is required, IoT Platform sends an update notification to the device. Otherwise, no notification is sent.

Message formats

For more information about how to use language-specific SDKs to implement OTA updates on devices, see the Link SDK documentation.

The following steps describe the procedure of an OTA update.

1. Optional. Connect a device to the OTA service of IoT Platform and submit the version number.

   The version number is pushed to the following topic over MQTT: `/ota/device/inform/${YourProductKey}/${YourDeviceName}`. Sample message:

   ```
   {
       "id": "123",
       "params": {
           "version": "1.0.1",
           "module": "MCU"
       }
   }
   ```

   Parameters

Parameter	Type	Description
id	String	The ID of the message. Valid values: 0 to 4294967295. Each message ID must be unique for the device.
version	String	The version of the OTA module.
module	String	The name of the OTA module. **Note** ◦ If the device submits the version of the default module, the module parameter is optional. ◦ The version of the default module indicates the version of the device firmware.

2. In the IoT Platform console, add an update package, check the update package, and then start a batch update.

   For more information, see Overview.

3. After you start an update in the console, the device receives the URL of the update package from a topic. The message that includes the URL is sent from the OTA service of IoT Platform to the topic.

   The format of the topic is `/ota/device/upgrade/${YourProductKey}/${YourDeviceName}`. After IoT Platform sends an OTA update request to the device, the device receives the URL of the update package from this topic.

   Sample message:

   ◦ Sample information about an OTA update package that contains a single file:

- Download the OTA update package over HTTPS:

```
{
    "id": "123",
    "code": 200,
    "data": {
        "size": 93796291,
        "sign": "f8d85b250d4d787a9f483d89a974***",
        "version": "10.0.1.9.20171112.1432",
        "isDiff": 1,
        "url": "https://the_firmware_url",
        "signMethod": "MD5",
        "md5": "f8d85b250d4d787a9f48***",
        "module": "MCU",
        "extData":{
            "key1":"value1",
            "key2":"value2",
            "_package_udi":"{"ota_notice":" Update the camera driver to prevent blu
rry videos."}"
        }
    }
}
```

- Download the OTA update package over MQTT:

```
{
    "id": "123",
    "code": 200,
    "data":{
        "size":432945,
        "digestsign":"A4WOP***SYHJ6DDDJD9***"
        "version":"2.0.0",
        "isDiff":1,
        "signMethod":"MD5",
        "dProtocol":"mqtt",
        "streamId":1397345,
        "streamFileId":1,
        "md5":"93230c3bde***",
        "sign":"93230c3bde42***",
        "module":"MCU",
        "extData":{
            "key1":"value1",
            "key2":"value2"
        }
    }
}
```

- You can download an OTA update package that contains multiple files only over HTTP. Sample information:

```
{
    "id": "123",
    "code": 200,
    "data": {
        "version": "2.0.0",
        "isDiff": 1,
        "signMethod": "MD5",
        "files":[
            {
                "fileSize":432944,
                "fileName":"file1-name",
                "fileUrl":"https://iotx***.aliyuncs.com/nop***.tar.gz?Expires=1502955
804&OSSAccessKeyId=***&Signature=XfgJu7***U%3D&security-token=CAISu***",
                "fileMd5":"93230c3bde425a9d7984a594ac55ea1e",
                "fileSign":"93230c3bde425a9d7984a594ac55****"
            },
            {
                "fileSize":432945,
                "fileName":"file2-name",
                "fileUrl":"https://iotx-***.aliyuncs.com/no***.tar.gz?Expires=1502955
804&OSSAccessKeyId=***&Signature=XfgJu7P***KU%3D&security-token=CAISuQJ***",
                "fileMd5":"93230c3bde425a9d7984a594ac56ea1f",
                "fileSign":"93230c3bde425a9d7984a594ac56****"
           }
        ],
        "module": "MCU",
        "extData":{
            "key1":"value1",
            "key2":"value2",
            "_package_udi":"{"ota_notice":" Update the camera driver to prevent blurr
y videos."}"
        }
    }
}
```

Parameters

Parameter	Type	Description
id	Long	The ID of the message. Each message ID is unique for the device.
message	String	The response message.
code	String	The HTTP status code.
version	String	The version of the OTA update package.
size	Long	The size of the update package. Unit: bytes. This parameter is available if the OTA update package contains a single file.

Parameter	Type	Description
url	String	The Object Storage Service (OSS) URL of the OTA update package. This parameter is available if the OTA update package contains a single file and the download protocol is HTTPS.
dProtocol	String	The protocol that is used to download the OTA update package. This parameter is available if the download protocol is MQTT.
streamId	Long	The unique ID generated when you download the OTA update package over MQTT. This parameter is available if the download protocol is MQTT.
streamFileId	Integer	The unique ID of the OTA update package that contains a single file. This parameter is available if the download protocol is MQTT.
isDiff	Long	This parameter is available if an update package is a delta update package. Set the value to 1. This value specifies that the update package contains only the differences between the new version and the previous version. In this case, a delta update is performed.
digestsign	String	The signature of the OTA update package after a secure update is performed. This parameter is available if the secure update feature is enabled for an OTA update package. For more information about how to enable the secure update feature, see Add an update package.
sign	String	The signature of the OTA update package. This parameter is available if the OTA update package contains a single file.
signMethod	String	The signature algorithm. Valid values: ◦ SHA256 ◦ MD5 Delta update packages for Android support only the MD5 algorithm.

Parameter	Type	Description
md5	String	If the signature algorithm is MD5, IoT Platform specifies values for the sign and md5 parameters. This parameter is available if the OTA update package contains a single file.
module	String	The name of the module to which the OTA update package is applied. **Note** If the OTA update package is applied to the default module, IoT Platform does not send the module parameter.
extData	Object	The tags of the update batch and the custom information that you want IoT Platform to push to the device. _package_udi specifies the custom information. Format of each tag: `"key":"value"`.
files	Array	The information about files in an update package. This parameter is available if the OTA update package contains multiple files. Information about a single file: ○ fileSize: the size of the file. ○ fileName: the name of the file. ○ fileUrl, fileMd5, and fileSign: These parameters correspond to the url, md5, and sign parameters in this table.

4. Optional. Download the update package within 24 hours after the device SDK receives the URL of the update package. Otherwise, the URL expires.

   To request an update task from IoT Platform, the device can send a message to the following topic: `/sys/{productKey}/{deviceName}/thing/ota/firmware/get`. Sample message:

```
{
    "id": "123",
    "version": "1.0",
    "params": {
        "module": "MCU"
    },
    "method": "thing.ota.firmware.get"
}
```

   Parameters

Parameter	Type	Description

Parameter	Type	Description
id	String	The ID of the message. Valid values: 0 to 4294967295. Each message ID must be unique for the device.
version	String	The version of the protocol. Set the value to 1.0.
params	Object	The request parameters.
module	String	The name of the module to which the OTA update package is applied. **Note** If you do not configure this parameter, the update package information of the default module is requested.
method	String	The request method. Set the value to *thing.ota.firmware.get*.

After IoT Platform receives the request, IoT Platform sends the update package information to the following topic: `/sys/{productKey}/{deviceName}/thing/ota/firmware/get_reply`.

- IoT Platform sends the information about the latest update package to the device. Sample responses:

- Information about an OTA update package that contains a single file:
  - Download the update package over HTTPS:

```
{
    "id": "123",
    "code": 200,
    "data": {
        "size": 93796291,
        "sign": "f8d85b250d4d787a9f483d89a974***",
        "version": "1.0.1.9.20171112.1432",
        "isDiff": 1,
        "url": "https://the_firmware_url",
        "signMethod": "MD5",
        "md5": "f8d85b250d4d787a9f48***",
        "module": "MCU",
        "extData":{
            "key1":"value1",
            "key2":"value2",
            "_package_udi":"{"ota_notice":" Update the underlying camera driver t
o prevent blurry videos."}"
        }
    }
}
```

  - Download the update package over MQTT:

```
{
    "id": "123",
    "code": 200,
    "data":{
        "size":432945,
        "version":"2.0.0",
        "isDiff":1,
        "signMethod":"MD5",
        "dProtocol":"mqtt",
        "streamId":1397345,
        "streamFileId":1,
        "md5":"93230c3bde***",
        "sign":"93230c3bde42***",
        "module":"MCU",
        "extData":{
            "key1":"value1",
            "key2":"value2"
        }
    }
}
```

- You can download an OTA update package that contains multiple files only over HTTP. Sample information:

```
{
    "id": "123",
    "code": 200,
    "data": {
        "version": "2.0.0",
        "isDiff": 1,
        "signMethod": "MD5",
        "files":[
            {
                "fileSize":432944,
                "fileName":"file1-name",
                "fileUrl":"https://iotx***.aliyuncs.com/nop***.tar.gz?Expires=15029
55804&OSSAccessKeyId=***&Signature=XfgJu7***U%3D&security-token=CAISu***",
                "fileMd5":"93230c3bde425a9d7984a594ac55ea1e",
                "fileSign":"93230c3bde425a9d7984a594ac55****"
            },
            {
                "fileSize":432945,
                "fileName":"file2-name",
                "fileUrl":"https://iotx-***.aliyuncs.com/no***.tar.gz?Expires=15029
55804&OSSAccessKeyId=***&Signature=XfgJu7P***KU%3D&security-token=CAISuQJ***",
                "fileMd5":"93230c3bde425a9d7984a594ac56ea1f",
                "fileSign":"93230c3bde425a9d7984a594ac56****"
           }
        ],
        "module": "MCU",
        "extData":{
            "key1":"value1",
            "key2":"value2",
            "_package_udi":"{"ota_notice":" Update the underlying camera driver to
prevent blurry videos."}"
        }
    }
}
```

Parameters

Parameter	Type	Description
id	String	The ID of the message. Valid values: 0 to 4294967295. Each message ID is unique for the device. The message ID in the response is the same as that in the request. You can view the id parameter in the data that is submitted to the `/sys/{YourProductKey}/{YourDeviceName}/thing/ota/firmware/get` topic.
code	Integer	The response code. The value 200 indicates a successful request.

Parameter	Type	Description
data	String	The information about the update package. For more information, see Push OTA update package information to a device.

- IoT Platform sends a response if no update package information exists. Sample response:

```
{
    "id": "123",
    "code": 200,
    "data": {
    }
}
```

5. Call the operation to download the update package over HTTPS after the device SDK receives the URL of the update package. For more information, see OTA.

> **Note** The device cannot automatically download the update package. You must call the required operation in the SDK to download the update package. If the device fails to download the update package from the URL within 24 hours, the URL expires.

6. During an update, the device must send the update progress to the following topic: `/ota/device/progress/${YourProductKey}/${YourDeviceName}`.

   Sample message:

```
{
    "id": "123",
    "params": {
        "step": "-1",
        "desc": "OTA update failed because no update package information is found.",
        "module": "MCU"
    }
}
```

   Parameters

Parameter	Type	Description
id	String	The ID of the message. Valid values: 0 to 4294967295. Each message ID must be unique for the device.

Parameter	Type	Description
step	String	The progress of the OTA update. Valid values: ○ An integer from 1 to 100: indicates a percentage that represents the update progress. ○ -1: indicates that the update failed. ○ -2: indicates that the download failed. ○ -3: indicates that the verification failed. ○ -4: indicates that the firmware flashing failed.
desc	String	The description of the current step. The description cannot exceed 128 characters in length. If an exception occurs, this parameter contains the error message.
module	String	The name of the module to which the OTA update package is applied. For more information, see Add an update package. Note If the device submits the update progress of the default module, the module parameter is optional.

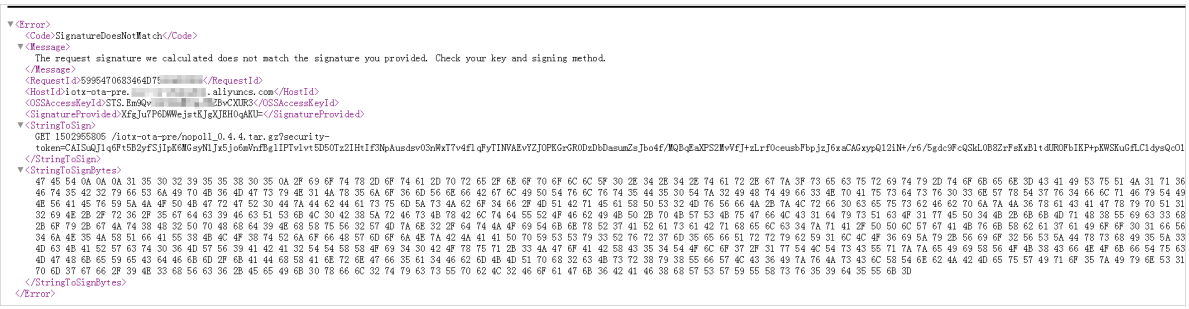
7. After the OTA update of a device is completed, the device must send the current firmware version to a topic in the following format: `/ota/device/inform/${YourProductKey}/${YourDeviceName}`. If the submitted version is the same as the version that the OTA service sends to the topic, the OTA update is successful. Otherwise, the OTA update fails.

   Note A successful OTA update is indicated by the submitted firmware version. For example, a device submits an update progress of 100%. However, if the current firmware version is not submitted within a specified timeout period, IoT Platform considers that the update fails.

   We recommend that you restart the device immediately after the OTA update is completed. After the device goes online, submit the current version number. The interval between a request to connect the device with IoT Platform and a request to submit the current version number cannot exceed 2 seconds.

Common errors

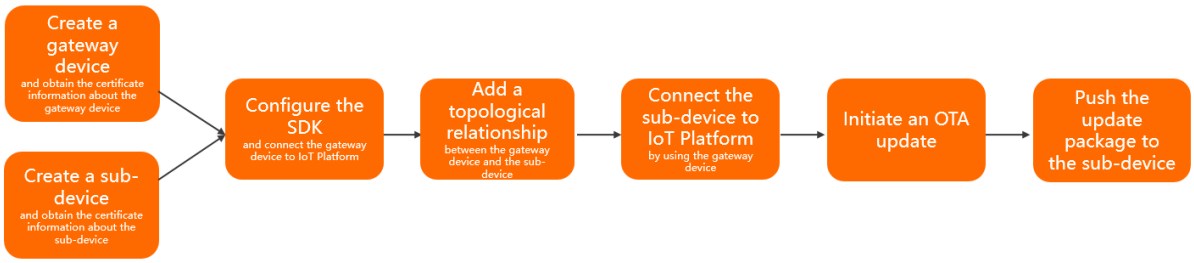
- Sign failed. If the URL of an update package is invalid or changed, this error occurs, as shown in the following figure.



- The access is denied. This error occurs due to the expiration of a URL. Each URL is valid for 24 hours.

5.4. Update sub-devices by using OTA

Sub-devices do not directly connect to IoT Platform. Instead, they connect to IoT Platform by connecting to gateways and by using the communication channel between the gateways and IoT Platform. After a sub-device connects to a gateway, you can implement an Over-the-Air (OTA) update on a sub-device by using the communication channel between the gateway and IoT Platform. To update a sub-device, you must use the topics of the sub-device and specify the ProductKey and DeviceName of the sub-device in the topics. This article describes how to implement an OTA update on a sub-device by using a gateway device.



1. Create a gateway device and a sub-device and obtain the ProductKeys, DeviceNames, and DeviceSecrets.
   - When you create a product, set the **Node Type** parameter to **Gateway device**. Then, create a device in the product.
   - When you create a product, set the **Node Type** parameter to **Gateway sub-device** and select a gateway protocol. Then, create a sub-device in the product.

   For more information, see Create a product and Create a device.
2. Specify the certificate information of the gateway device and connect the gateway device to IoT Platform. The certificate information includes the ProductKey, DeviceName, and DeviceSecret.

   For more information, see Link SDK documentation.
3. Add a topological relationship between the gateway device and the sub-device.

For more information, see Manage sub-devices.

4. Connect the sub-device to IoT Platform by using the gateway device.

   You can connect a sub-device to IoT Platform by using one of the following methods:

5. Configure the gateway device to perform the following operations for the sub-device: initiate a request for an OTA update, report the version of the OTA module to be updated, monitor messages that are pushed from IoT Platform, report the update progress, and pull the information about the update packages.

   > Note
   >
   > - If you want to implement an OTA update on a sub-device by using a gateway device, you must use the topics of the sub-device.
   >
   >   For more information about the topics that are used to transmit OTA update messages and the Alink JSON format, see OTA update.
   > - When you add an OTA update package, you must specify the product of the sub-device as the product to which the package belongs.

   For more information, see Overview.

6.Remote configuration

The remote configuration feature of IoT Platform allows you to update the configurations of remote devices without the need to restart or stop the devices. The configurations include the system parameters and network parameters.

Prerequisites

The remote configuration feature is supported in the device SDK. You must specify `FEATURE_SERVICE_OTA_ENABLED = y` in the device SDK. You can call the linkkit_cota_init operation that is provided by the SDK to initialize remote configurations. For more information, see Remote configuration.

Context

You can push an update package to a device to update device configurations such as the system parameters, network parameters, and security policies. If you use this method, the workloads of firmware version maintenance increase, and the device must stop running to implement an update.

To resolve these issues, IoT Platform provides the remote configuration feature. This feature allows you to update configurations without the need to restart or stop the device.

> **Note** The remote configuration feature is applied on a product basis. The configuration file that is uploaded to IoT Platform takes effect on all devices of a product. You cannot specify a single device for remote configuration.

Overview

You can perform the following operations that are related to the remote configuration feature:

- Enable or disable the remote configuration feature.
- Edit configuration files and manage the file versions in the IoT Platform console.
- Push configuration files from IoT Platform to multiple devices at a time to update the configurations.
- Enable devices to request configuration files.

Scenarios

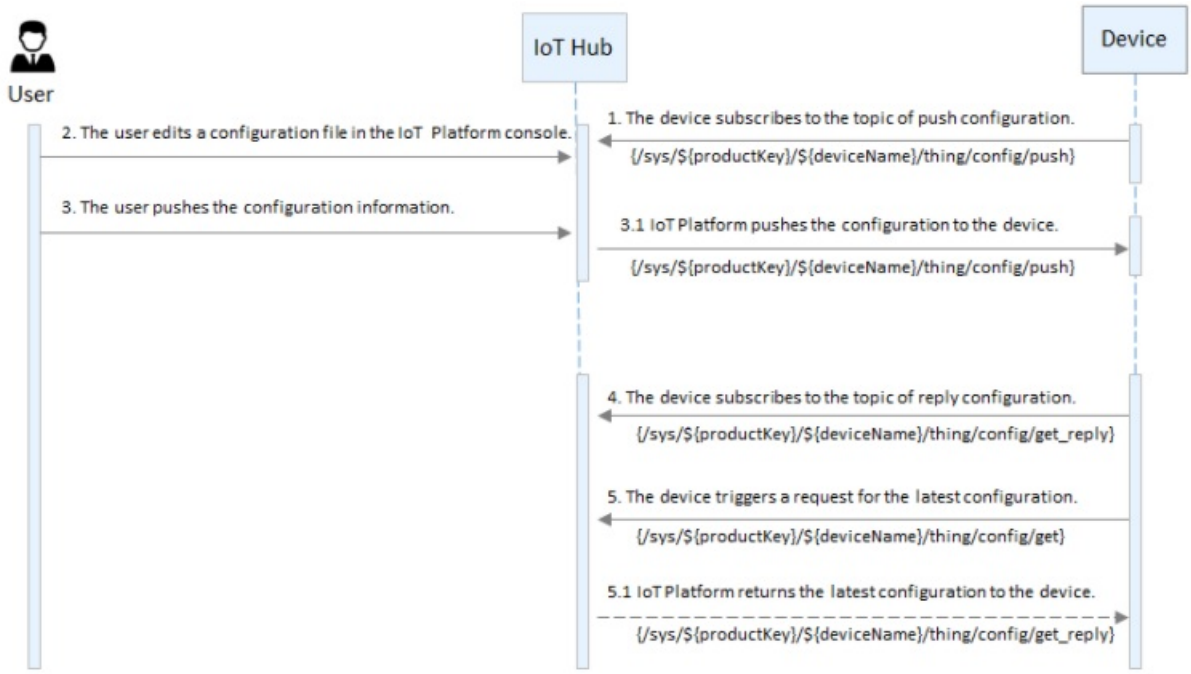
The remote configuration feature is suitable for the following scenarios:

- IoT Platform pushes configuration information to multiple devices at a time. Each device receives the information and modifies the local configuration file based on the information.
- A device requests a configuration file from IoT Platform and performs an update.

The following sections describe the procedures in the preceding scenarios.

Scenario 1: IoT Platform pushes configuration information to devices

In the IoT Platform console, you can push a configuration file to all devices of a product at a time.

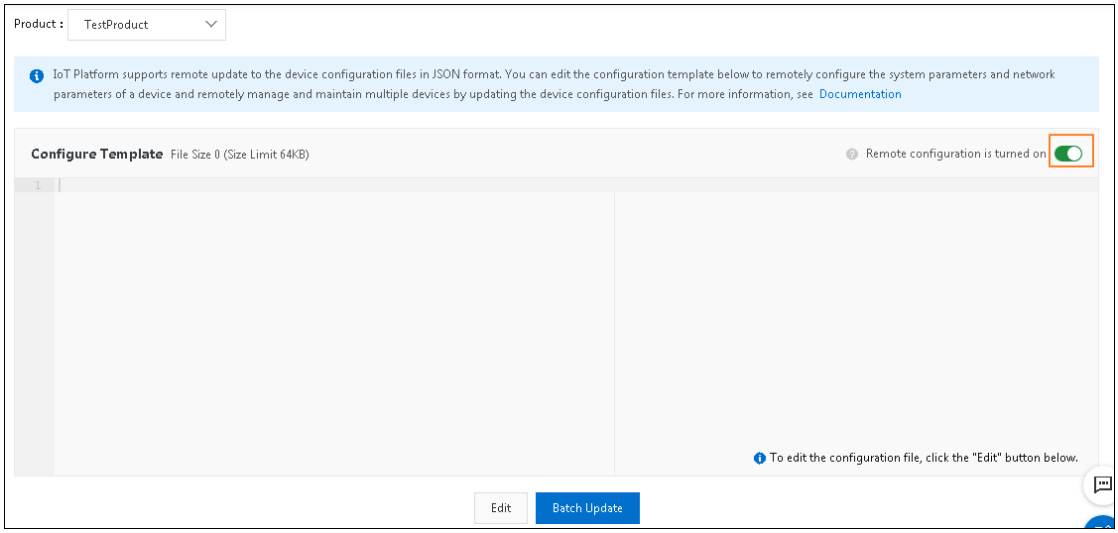


1. Connect devices to IoT Platform.

   For more information about how to use the sample code to develop devices, see the Link SDK documentation.

   > **Notice** When you develop the devices, subscribe to the following topic that is used to push configuration information: `/sys/${productKey}/${deviceName}/thing/config/push`.

2. In the IoT Platform console, edit a configuration file.
   i. Log on to the IoT Platform console.
   ii.
   iii. In the left-side navigation pane, choose **Maintenance > Remote Config**.
   iv. Select a product, turn on Remote Configuration, and then click **Edit**.



v. In the code editor of the **Configure Template** section, write or paste the configuration information in the JSON format.

> **Note** The configuration template is applied to all devices of a product. You cannot push a configuration file to a single device in the IoT Platform console.

vi. Click **Save**, and then click **OK**.

You can push a configuration file to all devices of a product at a time. You can also enable a device to request a configuration file.

3. Click **Batch Update**, and then click **Confirm Update**.

Then, IoT Platform pushes the configuration file to all devices of the product.

If IoT Platform determines that the operation is not performed in a trusted environment after you click **Batch Update**, IoT Platform sends you an SMS message for verification. IoT Platform sends the configuration file to the devices only after the verification is complete.

> **Notice**
> - After you push a configuration file to a product, you cannot push another configuration file to the product within 1 hour.
> - You can push the configuration file to a product only once in the IoT Platform console. The next time you repeat the preceding steps to push the same configuration file to the product, the push operation fails.
> - If you want to stop batch updates, you can turn off Remote Configuration for the product. Then, IoT Platform stops pushing configuration files and denies update requests from devices.

4. Devices automatically update the configurations after the devices receive the download URL of the configuration file from IoT Platform.

> **Note** The download URL is valid for 30 minutes. Devices must download the configuration file within the validity period of the download URL.

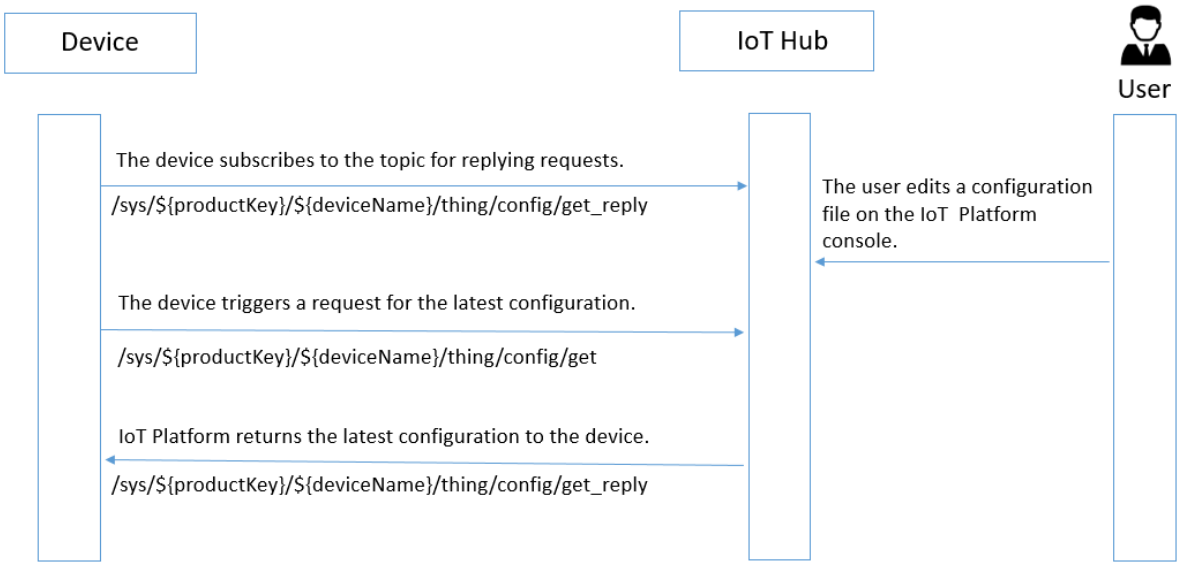
5. Optional. View and manage the versions of configuration files.

By default, the latest five configuration files are saved in the IoT Platform console. After you edit and save a new version of the configuration file, the previous version is automatically displayed in the Configuration Version Record section.

You can view the update time and content of a configuration file. Find the configuration file whose update time and content that you want to view and click **View** in the Actions column. In the dialog box that appears, you can view the content. You can also click **Restore to This Version** to restore the content of the selected version to the code editor. Then, you can modify the content and update the configurations of multiple devices at a time.

Scenario 2: Enable a device to request a configuration file

The following figure shows the process.



1. Connect devices to IoT Platform.

   For more information about how to use the sample code to develop devices, see the Link SDK documentation.

   > **Note** When you develop the device, subscribe to the following topic that is used to send responses to update requests: `/sys/${productKey}/${deviceName}/thing/config/get_reply`.

2. In the IoT Platform console, enable remote configuration and edit a configuration file. For more information, see Step 2 in Scenario 1.
3. Call the linkkit_invoke_cota_get_config operation to generate a request for remote configuration.
4. Send the request to the following topic to query the latest configuration information: `/sys/${productKey}/${deviceName}/thing/config/get`.
5. IoT Platform receives the request, and then returns the information to the following topic: `/sys/${productKey}/${deviceName}/thing/config/get_reply`.
6. Then, the device uses the URL that is sent from IoT Platform to download the configuration file.

   For example, you can invoke the cota_callback callback function that is provided by the device SDK for C 3.x to process the download URL of the configuration file and perform an update.

   > **Note** The download URL is valid for 30 minutes. Devices must download the configuration file within the validity period of the download URL.