ALIBABA CLOUD

Alibaba Cloud

物联网平台

Maintenance

Document Version: 20201030

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

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 device network status.

  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 Use CloudMonitor to monitor IoT resources and Alerts and 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, number of upstream and downstream messages, and number of messages that are forwarded by the rule engine.

View data metrics

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

   Real-time Monitoring

Data metric description

The following table lists 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. **Note** Delays exist during data collection.

Metric	Description
Messages Sent to IoT Platform	The number of messages that devices sent to IoT Platform. The data is displayed based on the types of protocols that are used to communicate with IoT Platform. Note Delays exist during data collection.
Messages Sent from IoT Platform	The number of messages sent from IoT Platform to devices and servers. The data is displayed based on the types of protocols that are used to communicate with IoT Platform. Note Delays exist during data collection.
Messages Forwarded Through Rule Engine	The number of messages forwarded by the rule engine. The data is displayed based on the target cloud services to which messages were forwarded. Note Delays exist during data collection.

References

- Use CloudMonitor to monitor resource usage

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

  For more information, see

  Use CloudMonitor to monitor IoT resources

  Alerts and notifications

- View device network status

  You can enable the devices whose network connection method is Wi-Fi to submit the network status data.

  For more information, see

  Devices submit network status data

  View device network status

1.3. View device network status

IoT Platform supports the monitoring of the network status. A device that accesses the network through Wi-Fi can report the network status to IoT Platform through a specified topic. On the **Real-time Monitoring** page of the console, you can select the **Device Network Status** tab to view the Wi-Fi signal quality of the device.

Internet of Things IoT Platform IoT Real-time monitoring Device network status

Context

For more information about the topic, data format, and errors of device network status messages, see Device network status.

If your devices use AliOS Things 3.0 or later, the system automatically monitors and reports 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. On the left-side navigation pane, choose **Maintenance > Real-time Monitoring**.
3. On the **Real-time Monitoring** page, select **Device Network Status**.
4. Select a device and a time range. The network status data of the selected device and time range is displayed.

Descriptions of the device network status

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 reports the network status data to IoT Platform after it encounters a network error or when it collects data at a scheduled time. For other scenarios, the device may not immediately report the network data. If the device does not report a timestamp, no time is displayed.
RSSI (dBm)	The received signal strength.
SNR (dB)	The signal-to-noise ratio of the wireless signal.
Wireless Signal Packet Loss Ratio (‰)	The data packet loss rate.
Network Connection Method	Currently, only Wi-Fi is supported.
Error Description	Click **View Details** to view errors and the number of such errors. **Note** The **View Details** button is only visible when the device reports network errors. For more information, see Device network status.

1.4. Use CloudMonitor to monitor resource usage

1.4.1. Use CloudMonitor to monitor IoT resources

You can set CloudMonitor to monitor the resource usage of your IoT Platform and receive alerts.

Create a threshold-triggered alarm rule

IoT Platform allows you to use CloudMonitor to monitor IoT Platform by multiple metrics. These metrics include the number of real-time online devices using a specific communication protocol, the number of messages sent from devices to IoT Platform, the number of messages sent from IoT Platform to devices, the number of messages forwarded by the rules engine to other Alibaba Cloud service, the number of property report failures, the number of event report failures, the number of service call failures, and the number of property setting failures.

To create a threshold-triggered alarm rule, follow these steps:

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

Alert rule parameters

Parameter	Description
Product	Select **IoT Platform**.
Resource Range	Includes the following value options: ○ All Resources: An alert is sent when any instance under your IoT Platform service meets the description of the alert rule. ○ Instance: An alert is sent only when the specified products meet the description of the alert rule.
Region	This parameter is available only when you set Resource Range to **Instance**. Select the region of the IoT Platform instance monitored by this alert rule.
Instance	Select an IoT Platform instance to be monitored and select one or more products.
Alarm Rule	Set the name of the alert rule.

Parameter	Description
Rule Description	Set the description of the alert rule. It defines the condition in which an alert is triggered. You must configure the following items: ○ Select a monitoring metric for the rule. ○ Select a scan period for the rule. For example, if the scan period is set to 60 minutes, scans are performed every 60 minutes. ○ Set the triggering condition. For example, an alert is triggered only if the number of devices exceeds 5,000 for three consecutive scan periods.
Mute for	Set the period of time before which the alert is sent again if the exception persists after the alert is triggered.
Effective Period	Set the time range when the alert rule is applied. CloudMonitor applies the alert rule to monitor the specified metric only during the specified effective period.
Notification Method	Set notification parameters, such as the notification contacts and notification methods.

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

Create an event-triggered alert rule

You can use an event-triggered alert rule to monitor the following IoT Platform events:

- The upstream QPS of any device reaches the upper limit.
- The downstream QPS of any device reaches the upper limit.
- The number of connection requests per second for the current account reaches the upper limit.
- The upstream QPS for the current account reaches the upper limit.
- The downstream QPS for the current account reaches the upper limit.
- The data forwarding QPS by the rules engine for the current account reaches the upper limit.

To create an event alert rule, follow these steps:

1. On the **Real-time Monitoring** page, click**Alert Settings**
2. On the **Create Alarm Rule** page, click View the Details in the event alarm description.
3. On the **Event Monitoring** page, click**Create Event Alert**. In the dialog box on the right side, configure the alarm rule, and then click**OK**.

   The following figure shows how to create an event-triggered alert rule.

Event-triggered alert rule parameters

Parameter	Description
Alarm Rule Name	Set the name of the alert rule.

Parameter	Description
Event Type	Select **System Event**.
Product Type	Select **IoT Platform**.
Event Type	Select **All types** or **Exception**.
Event Level	Select **All Levels** or select one or more specific event levels.
Event Name	Select one or more events to be monitored.
Resource Range	Select **All Resources**.
Alarm Type	Set the alert contacts and notification methods.

For more information about setting event-triggered alert rules, see Create a threshold-triggered alert rule.

Related documents

Alerts and notifications

1.4.2. Alerts and notifications

When the resource usage on IoT Platform reaches the value specified in an alert rule, the corresponding alert is triggered. Alibaba Cloud then sends a notification to the specified contact group. This topic describes the alerts and notifications of IoT Platform.

IoT Platform CloudMonitor alerts and notifications

Notifications for threshold alerts

When a threshold alert is triggered, the alert contact group receives a notification. The notification includes information shown in the following figure:

Notification content and descriptions

Field	Description
IoT Platform instance	The instance that triggers the alert. This field contains the product key (ProductKey), instance ID (instanceId), and region ID (regionId).
Metric	The metric is displayed as a code. It indicates the metric that you selected when you set the Rule Description parameter. In this example, the code "MessageCountForwardedThroughRuleEngine_MNS" represents the number of messages forwarded by the rules engine. If the number of messages exceeds the specified threshold during a time period, an alert is triggered. For more information about metrics and descriptions, see the following table: Metric codes and descriptions.
Alert time	The time when the alert is triggered.

Field	Description
Count	The total number of messages, the number of forwarded messages, or the number of connected devices counted for the specified metric.
Duration	The time period during which an alert is triggered upon a threshold violation.
Rule details	The alert rule details that you have set in the CloudMonitor console.

Metric codes and descriptions

Code	Description
MessageCountForwardedThroughRuleEngine_FC	The number of messages forwarded by the rules engine to Function Compute.
MessageCountForwardedThroughRuleEngine_MNS	The number of messages forwarded by the rules engine to Message Service.
MessageCountForwardedThroughRuleEngine_OTS	The number of messages forwarded by the rules engine. It equals the number of times that the rules engine forwards data to Table Store.
MessageCountForwardedThroughRuleEngine_RDS	The number of messages forwarded by the rules engine to ApsaraDB for RDS.
MessageCountForwardedThroughRuleEngine_REPUBLISH	The number of messages forwarded by the rules engine from the current topic to other topics.
MessageCountSentFromIoT_HTTP_2	The number of messages that are sent through IoT Platform over HTTP/2.
MessageCountSentFromIoT_MQTT	The number of messages that are sent through IoT Platform over MQTT.
MessageCountSentToIoT_CoAP	The number of messages that are sent through IoT Platform over CoAP.
MessageCountSentToIoT_HTTP	The number of messages that are sent to IoT Platform over HTTP.
MessageCountSentToIoT_HTTP/2	The number of messages that are sent to IoT Platform over HTTP/2.
MessageCountSentToIoT_MQTT	The number of messages that are sent to IoT Platform over MQTT.
OnlineDevicesCount_MQTT	The number of devices that are connected to IoT Platform over MQTT in real time.
DeviceEventReportError	The number of event reporting failures.
DevicePropertyReportError	The number of property reporting failures.

Code	Description
DevicePropertySettingError	The number of property setting failures.
DeviceServiceCallError	The number of service calling failures.

Notifications for event alerts

When an event alert is triggered, Alibaba Cloud sends a notification to the specified contact group.

Notification content and descriptions

Field	Description
Event name	The event name is displayed as a code. In this example, the code "Device_Connect_QPM_Limit" represents the event of the maximum connection requests sent per minute by a device reaching the upper limit. For more information about event codes and descriptions, see the following table: Event codes and descriptions.
Object	The resource that triggers the alert. • resourceId: The resource ID. Format: `acs:iot:$regionid::instance/$instanceId/product/$productKey/device/$deviceName` • Resource name: The instance ID. iot-public indicates that this instance is a public instance. • Group ID: The ID of the group that the device belongs to. If the device does not belong to any group, the field displays an empty string.
Event level	Currently, all events are WARN events.
Event time	The time when the event occurs.
Event status	Currently, all events are set to the Fail status. This status indicates that the request failed because the number of connection requests sent per minute or messages sent per second has reached the upper limit.
Details	The information about the resource that triggers the alert. The information is in the JSON format. This field contains the region ID (regionId), instance ID (instanceId), product key (ProductKey), and the device name (DeviceName). The ProductKey and DeviceName parameters appear in the notification only when the number of connection requests sent per minute, messages sent per second, or messages received per second by a device reaches the upper limit.

Event codes and descriptions

Code	Description

Code	Description
Device_Connect_QPM_Limit	The number of connection requests sent per minute by a device has reached the upper limit.
Device_Uplink_QPS_Limit	The number of messages sent per second by a device has reached the upper limit.
Device_Downlink_QPS_Limit	The number of messages received per second by a device has reached the upper limit.
Account_Connect_QPS_Limit	The number of connection requests sent per second by the current account has reached the upper limit.
Account_Uplink_QPS_Limit	The number of messages sent per second by the current account has reached the upper limit.
Account_Downlink_QPS_Limit	The number of messages received per second by the current account has reached the upper limit.
Account_RuleEngine_DataForward_QPS_Limit	The number of messages forwarded per second by the rules engine for the current account has reached the upper limit.

2.Online debug

2.1. Debug physical devices

After you configure a physical device, you can use IoT Platform to debug the device online. You can directly dispatch commands onto the device from the IoT Platform console. Only physical devices that are connected by using the Message Queuing Telemetry Transport (MQTT) protocol are supported for online debugging.

Procedure

1. In the left-side navigation pane of the IoT Platform console, choose **Maintenance > Online Debug**.
2. On the **Online Debug** page, select the device to be debugged.

   After you select a device, the debugging page is displayed.
3. Click **Debug Physical Device**.
4. Dispatch the command.

   The following table lists the options for debugging a physical device.

Option	Procedure
Set properties	Set the properties of a physical device from the cloud. After the device receives the command, the device sets its properties and reports the property values to IoT Platform. i. Click **Property Debugging**. ii. From the **Debug Feature** list, select the property to debug and set the method to **Set**. The format for the property data is automatically displayed in the text box, for example, `{"Temperature":0}`. iii. Set a value for the property and click **Send Command**.
Get properties	Obtain the property values of the device. i. Click **Property Debugging**. ii. From the **Debug Feature** list, select the property to debug and set the method to **Get**. iii. Click **Send Command**. **Note** When you try to obtain the property of the device, you do not need to enter data in the text box. After you dispatch the command, the text box automatically shows the latest data of the property. If the device does not have a value for the property, the data is empty.

Option	Procedure
Call services	i. Click **Service Calls**. ii. From the **Debug Feature** list, select the service to debug. iii. In the text box, enter the input parameters for the service, and click **Send Command**. The input parameters must be in JSON format, for example, `{"Switch":0}`.

Result

After you dispatch the command, you can view the operation logs in **Real-time Logs** that is located on the right side of the page.

2.2. Debug virtual devices

IoT Platform provides virtual devices for application development and debugging. You can use virtual devices to debug property reporting, event reporting, property configurations, and service calls. Devices that use custom data formats can also be supported.

Context

You can implement an IoT Platform device by using the following process: After you develop a device, the device can connect with and report data to IoT Platform. Then, you can develop applications that are used on IoT Platform. However, this process is time-consuming.

To simplify the implementation process, Alibaba Cloud IoT Platform provides virtual devices. Virtual devices simulate the connections between physical devices and IoT Platform, report properties and events to IoT Platform, and receive commands from IoT Platform. You can use these commands to configure properties and call services. Then, you can develop and debug applications based on the virtual devices.

When you use virtual devices in the IoT Platform console, take note of the following limits:

- The minimum interval for pushing consecutive messages is 1 second.
- You can push a maximum of 1,000 consecutive messages.
- You can click **Push** 100 times per day to push debugging information.
- If a device uses custom data format, the Base64-encoded string of the binary data cannot exceed 4,096 characters.
- A virtual device cannot be started if the physical device is online or disabled. After a physical device goes online, the virtual device that corresponds automatically goes offline.

Procedure

1. Log on to the IoT Platform console.
2. In the left-side navigation pane, choose **Maintenance > Online Debug**.
3. On the **Online Debug** page, select the device that you want to debug.

   After you select the device, the debugging page is displayed.
4. Click the **Debug Virtual Device** tab, and then click **Start Virtual Device**.

5. Send a debug command.

Scenario	Procedure
Report properties	Use the virtual device to report property values to IoT Platform. i. Click **Properties**. ii. Enter a value in a property field. The value must match the data type and value range of the property. You can also use the *random()* function to generate a random value. iii. Push the data. You can select one of the following push methods: ■ **Push**: Immediately push the data. ■ **Push Policy**: Set a push policy. Valid values: ■ At Specific Time: Push data at a specified time. ■ At Specific Interval: Push data at a fixed interval during a specified period. The interval is measured in seconds.
Report events	Use the virtual device to report events to IoT Platform. i. Click **Events**. ii. Enter a value in the event field. The value must match the data type and value range of the event. You can also use the *random()* function to generate a random value. iii. Push the data. You can select one of the following push methods: ■ **Push**: Immediately push the data. ■ **Push Policy**: Set a push policy. Valid values: ■ At Specific Time: Push data at a specified time. ■ At Specific Interval: Push data at a fixed interval during a specified period. The interval is measured in seconds.
Set properties	Set the properties of a physical device from the cloud. After the device receives the command, the device sets its properties and reports the property values to IoT Platform. i. Click **Property Debugging**. ii. From the **Debug Feature** list, select the property to debug and set the method to **Set**. The format for the property data is automatically displayed in the text box, for example, `{"Temperature":0}`. iii. Set a value for the property and click **Send Command**.

Scenario	Procedure
Get properties	Obtain the property values of the device. i. Click **Property Debugging**. ii. From the **Debug Feature** list, select the property to debug and set the method to **Get**. iii. Click **Send Command**. **Note** When you try to obtain the property of the device, you do not need to enter data in the text box. After you dispatch the command, the text box automatically shows the latest data of the property. If the device does not have a value for the property, the data is empty.
Call service	i. Click **Service Calls**. ii. From the **Debug Feature** list, select the service to debug. iii. In the text box, enter the input parameters for the service, and click **Send Command**. The input parameters must be in JSON format, for example, `{"Switch":0}`.

Result

After data is pushed, you can view operations logs in the **Real-time Logs** section.

After data is pushed, you can go to the **TSL Data** tab of the **Device Details** page. On the **Status** tab, you can view the reported properties. On the **Events** tab, you can view the reported events. On the **Invoke Service** tab, you can view the records of service calls.

Note If you select **Push Policy**, you can view the logs, properties, and events after the data is pushed.

3.Device simulation

IoT Platform allows you to simulate devices to facilitate IoT application development and testing.

Context

You can implement an IoT Platform device by using the following process: A device is developed, the device sends data to IoT Platform, IoT Platform receives data from the device, and then an application is developed in the IoT Platform console. This development process is time-consuming.

IoT Platform provides the device simulation feature that allows you to simulate a device and the connection between the device and IoT Platform. You can use this feature for debugging in the following scenarios:

- Debug the topics and procedures in the upstream:
  - Custom topics, including message reporting and message subscription
  - The procedure to report properties
  - The procedure to report events
- Debug the following topics and procedures in the downstream:
  - Custom topics
  - The procedures that are related to properties, including the procedures that are used to retrieve and set property values
  - The procedure to invoke services

Limits

- The device simulation feature cannot be used to simulate devices that adopt pass-through transmission or devices that transmit data in a custom format.
- Device simulation cannot be started if the selected physical device is online or disabled. After you leave the device simulation page, the simulated device becomes offline.

Procedure

1. Log on to the IoT Platform console.
2. In the left-side navigation pane, choose **Maintenance > Device Simulation**.
3. Select a device for simulation.
4. Click **Start Device Simulation**.
5. Push simulated data.

   Custom upstream topics

Scenario	Procedure
Debug a custom upstream topic	Enable a simulated device to use a custom topic to send IoT Platform a message. i. Choose **Upstream Debug > Custom Topics**. ii. Select a topic that is used to report messages, 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**.

Scenario	Procedure
Report property values	Enable a simulated device to report the property values to IoT Platform. i. Choose **Upstream Debug > Properties**. ii. Select or enter a value in a property field. The value must match the data type and value range of the property. iii. Click **Send command**.
Report events	Enable a simulated device to report an event to IoT Platform. i. Choose **Upstream Debug > Events**. ii. Select a feature and enter the event data in the JSON format, for example, `{"Power": "on"}`. iii. Click **Send Command**.
Debug a custom downstream topic	Enable IoT Platform to use a custom topic to send a simulated device a message. i. Choose **Downstream Debug > Custom Topics**. ii. Select a custom topic, enter payload data, and then set QoS to *0* or *1*. iii. Click **Send Command**.
Set and retrieve a property value	Enable IoT Platform to send a simulated device a command to set a property value and enable IoT Platform to retrieve a property value. i. Choose **Downstream Debug > Property debugging**. ii. Select a feature and select **Set** from the Method drop-down list. The syntax of the property data is displayed in the field, for example, `{"Temperature":0}`. Then, modify the value of the property. Click **Send Command**. After the simulated device receives the command, it sets the property to the new value. iii. Select a feature and select **Get** from the Method drop-down list. Click **Send Command**. After the command is sent and a response is received, the latest value of the property is displayed in the text box. If the data of the property is unavailable on the simulated device, a pair of braces {} is displayed.
Invoke a service	Enable IoT Platform to send a command to a simulated device to set a property to a specified value. The simulated device updates and reports the value to IoT Platform. If the simulated device is offline when IoT Platform sends the command, the simulated device will obtain, update, and report the specified property value to IoT Platform after going online. i. Choose **Downstream Debug > Invoke Service**. ii. Select a feature and enter the service input parameters in the JSON format, for example, `{"Switch":0}`. iii. Click **Send Command**.

Result

After data is pushed, you can view device logs in the **Device Logs** section on the right side of the page. Click **View Cloud Logs**. On the **Cloud run log** tab, you can view the related IoT Platform logs.

4.Log Service

4.1. IoT Platform logs

You can query IoT Platform logs on the Device Log page of the IoT Platform console.

Log types

The following figure shows the log types of upstream messages.

Upstream message logs

1. When devices submit messages to IoT Hub, the related message logs are printed. 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 printed.
3. If messages are sent to clients by using the rules engine or service subscription (Message Queue for AMQP and Message Service), logs for the rules engine and service subscription are printed.

The following figure shows the log types of downstream messages.

Downstream message logs

1. When users call API operations to publish messages, the logs of the API operations are printed. The API names are included in the logs.
2. When data is processed by different business modules in IoT Platform, the logs of the business modules are printed.
3. If messages are sent from IoT Hub to devices, logs for the messages are printed. The topics that are used to send messages are included in the logs.

Upstream messages for which the RequestId variable is specified

You can publish upstream messages for which the RequestId variable is specified to a device topic. This allows you to associate devices with IoT Platform logs. You can specify the RequestId parameter in upstream messages to associate devices and IoT Platform logs. This allows you to perform full-scale log analysis and identify issues at the earliest opportunity.

> **Note** The feature is available only for Link SDK 4.x for C. You also need to set the required parameters to implement the feature. For more information, see Logs.

When a device publishes a message to a topic, you can add `?_reqId=${RequestId}` to the end of the topic. After IoT Platform receives the message, IoT Platform parses the RequestId parameter and write the result to an IoT Platform log. You can search for the required message log by request ID.

For example, the RequestId parameter is required when a device publishes messages to a topic in the `/${productKey}/${deviceName}/user/update` format. Implement the device and set the RequestId parameter when the device reports a message to the topic, for example, `/${productKey}/${deviceName}/user/update?_reqId=1549265390d249fd`. After the device publishes the message, choose **Maintenance > Device Log > Cloud run log**. On the tab that appears, search for the required message log by request ID, as shown in the following figure.

Query IoT Platform logs

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

   The following table describes the supported search conditions.

Search condition	Description
DeviceName	Enter a device name. You can search for the logs of a device by device name.
TraceId	Enter a tracing ID to search for the logs of series modules.
Keyword	Enter a keyword to search for the logs that contain the keyword. Valid keywords include the values of the following parameters: ProductKey, DeviceName, Code, Operation, MessageId, RequestId, and TraceId.
Status	Search for logs by status. 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.

Field	Description	Remarks
Time	The time when a log entry is printed.	None
TraceId	The tracing ID. You can search for the logs of series modules by tracing ID.	None
The message contents.	You can click **View** in the Message Content column of a message to view the message contents. For logs whose workload type is *rules engine*, you can click **Filter** to obtain message logs that have the same message ID. These logs show the history that the rules engine forwards the message based on different data forwarding rules and retries after failed data forwarding.	None
DeviceName	The device name.	None

Field	Description	Remarks
Workload Type (all)	By default, logs of all business types are displayed. You can query logs of a specific business type. Valid values: • *Firmware Update* • *Data Parsing* • *TSL* • *Data Storage* • *Remote Configuration* • *Topological Relationship* • *TSL Service Call* • *Device Behavior* • *Device-to-cloud Messages* • *Cloud-to-device Messages* • *API Calls* • *Server-side Subscription* • *Device Shadow* • *Rules Engine* • *Other*	The first-level business identifier that identifies a business module.

Field	Description	Remarks

Actions	This field includes an operation that is performed, for example, API operation, service method, or message topic. The field value varies based on the following operation types:
	• Firmware update:
	  ○ *OTAFirmwarePush*: pushes notifications when a firmware update is initiated, confirmed, and completed.
	  ○ *OTAVersionReport*: submits the firmware 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*: checks the TSL model.
	  ○ For more information about the method parameter in the message body, see What is a TSL model?.
	• 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 tracing ID that can be used to search for the logs of series modules. • Message: the error message. Logs of failed operations include this field. • Params: the request parameters. Logs of some types include this field. • ResultData: the result. If operations generate results, printed logs include this field. Otherwise, printed logs do not include this field.	**Note** 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.

Field	Description	Remarks
Status	The HTTP status code in the response. The status code 200 indicates that a call was successful. Other status codes indicate that the call failed. For information about API operation-related error codes, see Error codes. For 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 and offline.

Error code	Description	Cause	Troubleshooting

Error code	Description	Cause	Troubleshooting
400	A request error occurs.	This error may occur due to one of the following causes: • The device is disconnected because another device uses the same device certificate to connect to IoT Platform. IoT Platform identifies a device only based on the device certificate information (productKey, deviceName, and deviceSecret). The same device certificate may be used by different devices in the following scenarios: ◦ The same device certificate is burned on multiple devices. ◦ The network or power supply of the device is unstable. The device is reconnected to IoT Platform immediately after an abrupt 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. • The device is removed from IoT Platform. • The device is disabled in IoT Platform.	• Log on to the IoT Platform console. On the **Device Details** page, view the time when the device is activated in the **Activated At**. Check whether different devices use the same device certificate to connect to IoT Platform. • On the **Device List** tab of the Devices page, search for the required device to check whether the device is removed. • In the IoT Platform console, check whether the device is in the **Disabled** state.

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.

Error code	Description	Cause	Troubleshooting
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. The potential cause is that the network transmission speed is slow, or the device cannot process this number of messages.	Check network conditions and device message consumption capabilities.
1902	When the message is transmitted over the network, an exception occurs.	The sending failure is caused by a network exception.	Check network conditions.
1903	The format of the topic is invalid.	The format of the topic is invalid.	Check the topic format.
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 fails due to the timeout issue.	Check the RRPC response that is received from the device to determine whether the timeout issue occurs.
1905	IoT Platform does not receive an RRPC response before the timeout period is exceeded.	After IoT Platform sends an RRPC request to the device, IoT Platform does not receive an RRPC response from the device before the timeout period is exceeded.	Check whether the device responds to the RRPC request in time.
1950	When the message is transmitted over the network, a network connection error occurs.	The sending failure is caused by a network exception.	Check the network status.
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 an Alibaba Cloud device SDK, contact technical support or submit a ticket.

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 reports data or the request method in the data parsing result does not conform to the Alink protocol.	Check whether the topic to which the device reports data conforms to the Alink protocol. Check whether the request method that is specified in upstream data conforms to the Alink protocol.
9200	The device is not activated.	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 disconnected from IoT Platform.	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.
9324	A throttling error occurs.	The requests from the device or the tenant exceed the limit.	Reduce the frequency of message sending, or contact technical support.
9321	The parameters are invalid.	The input request 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.
9331	An internal error occurs on the destination cloud service.	An internal error occurs 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 corresponding cloud service to troubleshoot the error. You can also contact technical support.

Error code	Description	Cause	Troubleshooting
9332	The cloud service configuration is invalid.	The specified data forwarding destination is invalid. Therefore, an error occurs when IoT Platform connects 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 corresponding cloud service to troubleshoot the error.
9333	The permission to access the cloud service is invalid.	The permission that is granted on IoT Platform to access the destination cloud service is invalid.	Check your Alibaba Cloud RAM policy.
9399	An unknown internal server error occurs.	IoT Platform has an internal error.	Contact technical support or submit a ticket.

TSL-related error codes

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

- Submit TSL data.
- Call TSL services.

If the data format of a product is set to Custom, TSL-related logs include the hexadecimal values of raw data 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
9201	The device is disconnected.	The device is disconnected from IoT Platform.	Check the device status in the IoT Platform console.
9200	The device is inactive.	The device is not activated in IoT Platform. A newly registered device must submit data to IoT Platform before the device can be activated.	Check the device status in the IoT Platform console.
6208	The device is disabled.	After a device is disabled, you cannot configure properties or call services of the device.	Check the device status in the IoT Platform console. If the device is disabled, enable the device and then try again.

Error code	Description	Cause	Troubleshooting
6300	The method parameter is not found 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 standard Alink data or the parsed custom data that is submitted by the device.	Check the submitted data by viewing the IoT Platform logs that are generated when property data is submitted. You can also check the submitted data by viewing the on-premises device logs.
6206	An error occurs when you query the service.	When IoT Platform calls a service, the information of the service is queried. This error occurs if the service is not found.	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.
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. However, the script 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.

Error code	Description	Cause	Troubleshooting
6207	The data format is invalid.	This error may occur when IoT Platform synchronously calls services or when the device submits data. When IoT Platform synchronously calls services, this error may occur due to one of the following causes: • The format of the data that is 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 required data formats when you call services, see API reference and TSL-related topics. For information about the data format that is required by the Alink protocol, see Alink protocol.
System error codes			
5159	An error occurs when IoT Platform retrieves the TSL property data.	A system exception occurs.	Submit a ticket.
5160	An error occurs when IoT Platform retrieves the TSL event data.		
5161	An error occurs when IoT Platform retrieves the TSL service data.		
6661	An error occurs when IoT Platform queries the tenant information.		
6205	An error occurs when IoT Platform calls the service.		

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 and the input parameters of the event based on the related definitions in 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.	Check the number of properties by viewing the IoT Platform logs that are generated when property data is submitted. You can also check the submitted data by viewing the on-premises device logs.
6300	The method parameter is not found 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 standard Alink data or the parsed custom data submitted by the device.	Check the submitted data by viewing the IoT Platform logs that are generated when property data is submitted. You can also check the submitted data by viewing the on-premises device logs.
6320	The property information is not found when the system verifies the input parameters based on the TSL model.	The specified property is not found when IoT Platform queries 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-formatted data.	The method parameter does not exist in the standard Alink data or the parsed custom data submitted by the device.	View the IoT Platform logs that are generated when property data is submitted, and check whether the 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 the service or when the device submits data. When the device submits data, this error may occur due to the following cause: the Alink data submitted by the device or the data parsed by using the script is not in the JSON format.	For 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 code			
6452	A throttling error occurs.	Traffic throttling is triggered because excessive requests are submitted.	Submit a ticket.
6760	The storage quota of the tenant is exceeded.	A system exception occurs.	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	The parameters are invalid.	The request parameters are invalid.	Submit a ticket.
500	An internal system error occurs.	An unknown error occurs in the system.	Submit a ticket.
400	A service request error occurs.	An unknown error occurs when IoT Platform calls the service.	Submit a ticket.
429	Excessive requests are submitted in a short period of time.	Traffic throttling is triggered because excessive requests are submitted in a short period of time.	Submit a ticket.
System error code			

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

The following table describes the common error codes of TSL models.

IoT Platform verifies the input parameters of a service call, property data, and input parameters of an event based on the related definitions in the TSL model. This applies to service invocation, property data submission, and event data submission.

Error code	Description	Cause	Troubleshooting
6321	The identifier of the property does not exist in the TSL model.	A system exception occurs.	Submit a ticket.
6317	The TSL model is invalid.	A system exception occurs.	Submit a ticket.
6302	The parameters do not exist.	When the system verifies the input parameters of the service, the required parameters are 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 you have specified all required parameters.
6306	The input parameter does not meet the integer data type that is defined in the TSL model.	When the system 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 range that is 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 defined in the TSL model.

Error code	Description	Cause	Troubleshooting
6307	The input parameter does not meet the 32-bit float data type that is defined in the TSL model.	When the system 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 range that is 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 data type defined in the TSL model. Make sure that the parameter value is in the value range that is defined in the TSL model.
6322	The input parameter does not meet the 64-bit float data type that is defined in the TSL model.	When the system 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 range that is 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 data type defined in the TSL model. Make sure that the parameter value is in the value range that is defined in the TSL model.
6308	The input parameter does not meet the Boolean data type that is defined in the TSL model.	When the system 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 range that is 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.
6309	The input parameter does not meet the ENUM data type that is defined in the TSL model.	The data type of the parameter is different from the data type that is 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.

Error code	Description	Cause	Troubleshooting
6310	The input parameter does not meet the text data type that is defined in the TSL model.	When the system 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 that is defined in the TSL model.	Go to the Product Details page of the IoT Platform console, and view the TSL model. Make sure that the data type of the input parameter is the same as the data type that is defined in the TSL model and that the parameter length does not exceed the upper limit.
6311	The input parameter does not meet the date data type that is defined in the TSL model.	When the system verifies a parameter based on the TSL, 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 input data is not a UTC timestamp.	Go to the Product Details page of the IoT Platform console, and view the TSL model. Make sure that the data type of the input parameter is the same as the data type that is defined in the TSL model and that the input data is a UTC timestamp.
6312	The input parameter does not meet the struct data type that is defined in the TSL model.	When the system 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 number of the parameters that are included in a structure is different from the number that is 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 structure of the TSL model.	The input parameter is not found in the structure when the system verifies the parameter based on the TSL model.	Go to the corresponding Product Details page of the IoT Platform console, and view the TSL model. Check the input parameters for inconsistencies based on the TSL model.

Error code	Description	Cause	Troubleshooting
6324	The input parameter does not meet the array data type that is defined in the TSL model.	When the system verifies a parameter based on the TSL model, the following errors may occur: • The elements in the array do not follow the array syntax that is defined in the TSL model. • The number of elements in the array exceeds the upper limit that is 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 that is submitted by the device.
6328	The input parameter is not an array.	The input parameter is not an array when the system 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 data type of the element in the array is not supported when the system verifies the parameter based on the TSL model. Only the following data types of elements are supported: int32, float, double, text, and struct.	Make sure that the data type of the element is supported by IoT Platform.
System error code			
6318	A system exception occurs when IoT Platform parses the TSL model.		
6329	An error occurs when IoT Platform parses the array data in the TSL.		
6323	The data format of the parameter in the TSL model is invalid.		

Error code	Description	Cause	Troubleshooting
6316	An error occurs when IoT Platform parses the parameter in the TSL model.	A system exception occurs.	Submit a ticket.
6314	The data type of the parameter in the TSL model is not supported.		
6301	An error occurs when IoT Platform verifies the data format of the input parameters based on the TSL model.		
Data parsing script-related error codes			
26010	Traffic throttling is triggered because excessive requests are submitted.	The message returned because the maximum number of requests has been reached.	Submit a ticket.
26001	The content of the script is empty.	The script content is empty when IoT Platform runs the script.	Go to the Product Details page of the IoT Platform console, and check whether your data parsing script exists. If the script exists, check whether the script is saved. The script cannot be a draft.
26002	An exception occurs when IoT Platform runs the script.	The script runs as expected. However, 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. The console provides a basic script running environment, but does not verify the script. We recommend that you check the script before it is submitted.

Error code	Description	Cause	Troubleshooting
26006	The required method does not exist in the script.	The script runs as expected. However, the script content is invalid. The script must contain the protocolToRawData() and rawDataToProtocol() methods. This error occurs if these two methods do 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. However, the data format of the returned result is invalid. The script must contain the protocolToRawData() and rawDataToProtocol() methods. The protocolToRawData() method returns a byte[] array, and the rawDataToProtocol() method returns a JSON object. This error occurs if the returned result is not in the required data format. For example, after a device submits data, a result is returned to the device. The returned result must also be parsed. Otherwise, the data format of the returned result may be invalid.	Check the script in the IoT Platform console. Enter the input parameters, run the script on premises, and then check whether the data format of the returned result is valid.

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 problems on the Device Log page of the IoT Platform console.

Prerequisites

- The C SDK is used to develop a device so that the device can submit logs. For more information, see Submit device logs.
- The Device local log reporting switch is turned on. To turn on the switch, you must log on to the IoT Platform. Choose **Devices > Devices**, find the required device in the device list, and click **View**. On the **Device Details** page, turn on the **Device local log reporting** switch.

Query local device logs

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

   The following table lists the supported search conditions.

Search condition	Description
Device name	Enter a device name. You can search for the logs of a device by specifying a device name.
Tracing ID	Enter a tracing 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.
Content keyword	Enter a keyword of the log content to search for the logs that contain the keyword. Valid keywords include the values of the following parameters: ProductKey, DeviceName, Code, LogLevel, Module, IotId, LogContent, TraceContext, and TraceId.
Time range	Select a time range.

Log fields

The following table lists the log fields.

Parameter	Description
Reported At	The time when the device submits the log.
Collected At	The time when the device collects the log.
TraceId	The tracing ID. You can use this ID to search for series modules.
DeviceName	The name of the device.
Log Level	By default, logs of all levels are displayed. You can only query logs of a certain level. In addition to the OTHER level, log levels are in descending order: • FATAL • ERROR • WARN • INFO • DEBUG • OTHER: another log level
Module Name	The name of the module that generates the log, the module name is user-specified.
Content	The content of the log.

Analyze device logs

The log content includes the Code parameter. You can analyze device logs based on response codes, see Error codes for device SDKs.

4.3. Dump logs

The device log feature of IoT Platform allows you to retain logs of the last seven days by default. This feature also allows you to export IoT Platform operational logs to the Logstores of Log Service for persistent storage. After you enable the log dump feature, you can search and analyze logs in the IoT Platform console. This feature provides multiple functions, such as log reports, dashboard subscription, and alerts.

Prerequisites

Log Service is activated. For more information, see Activate Log Service.

Enable the log dump feature

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

You must enable the log dump feature for products in sequence.

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

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

   After you enable the log dump feature for a product, IoT Platform creates a log storage location and 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. Replace the ${regionId} variable with the ID of your region where IoT Platform resides.
     - Logstore: iot-logs.

     All products share the log storage location. You can find the logs of a product based on the specified product key. For more information, see IoT Platform logs.
   - Service linked role: to export logs. For more information, see Roles.
6. Set the retention period of logs. If exported logs reach the end of the specified retention period, these logs are deleted. The default retention period is seven days. The retention period can be 1 to 3,000 days or permanent.After you click **Set Log Save Time**, you are redirected to the Logstore Attributes page. You can click **Modify**, set the required parameters, and then click **Save**.

Use the log search and analysis feature

After you enable the log dump feature, you can log on to the IoT Platfrom console, choose **Maintenance > Device Log > Log Dump**, and then perform the following operations:

- Search and analyze logs: On the **Raw Logs** tab, you can use SQL statements to search logs of a specified period of time. Then, you can create statistical charts based on analytical results. For more information about the SQL syntax, see Search syntax and Analysis syntax. For more information about statistical charts, see Analytical charts.

- Quick analysis: After you search logs on the **Raw Logs** tab, you can view the distribution of a field in a specified period of time based on the search results. For more information, see Quick analysis.
- View reports: On the **Log Report** tab, you can view log reports in a specified period of time. Log reports reflect the statuses and exceptions of devices. The following table describes the reports.

Report	Description
Number of Times Device Online & Offline	The line chart shows the number of times that devices connect to IoT Platform, and number of times that devices disconnect from IoT Platform in a specified period of time.
Device Uplink & Downlink Messages	The line chart shows the number of upstream messages and number of downstream messages in a specified period of time.
Top 20 Devices by Uplink&Downlink Message	The list shows the top 20 devices that report or receive the most upstream or downstream messages, and the number of messages of each device.
Error Distribution for TSL Validation	The pie chart shows the distribution of data parsing errors in a specified period of time. You can search logs by error code and view the details of data parsing errors in logs. This allows you to improve a data parsing script in an efficient manner.
Top 10 Devices by Data Parsing Script Error	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. You can search logs by device name and view the details of data parsing errors in logs. This allows you to improve a data parsing script in an efficient manner.
Error Distribution for TSL Validation	The pie chart shows the distribution of Thing Specification Language (TSL) validation errors in a specified period of time. You can search logs by error code, view the details of TSL validation errors in logs, and troubleshoot issues.
Top 10 Devices by TSL Validation Error	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 logs by device name, view the details of TSL validation errors in logs, and troubleshoot issues.
Server-side Subscription Messages Forwarded	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	The pie chart shows the last 20 device anomaly messages. You can search logs by device name and error code, view the details of device anomaly messages in logs, and troubleshoot issues.

Report	Description
Cloud Service Messages Forwarded	The line chart shows the total number of messages that are forward by the data forwarding feature in a specified period of time.
Last 20 Data Forwarding Errors	The line chart shows the last 20 data forwarding errors of the rules engine in a specified period of time. You can search logs by device name and error code, view the details of data forwarding errors for the rules engine in logs, and troubleshoot issues.
Error Distribution for Cloud API Calls	The pie chart shows the distribution of API invocation errors in a specified period of time. You can search logs by API operation name and troubleshoot issues by using log details and error codes.

Note:

- Reports show data of the last one hour (time frame). The time interval for line charts is one minute.
- To set a time range, you can select the required value from the Time drop-down list in the upper-right corner of the Log Dump tab or choose >**Select Time Range** in the upper-right corner of a report card.
- To set a time interval for a line chart, you must write an SQL statement on the **Raw Logs** tab. For more information about the SQL syntax, see 日期和时间函数.

  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` query statement to set the time interval to one hour.

- Subscribe to a dashboard: IoT Platform converts dashboards into images on a regular basis and sends these images to the required customers by using emails or DingTalk chatbots. For more information about how to subscribe to a dashboard, see Subscribe to a dashboard.
- Set alerts: If one of the alert conditions is met, IoT Platform sends alerts by using multiple notification methods. These methods include the text message, phone call, email, and DingTalk chatbot. For more information about how to set alerts, see Create an alert rule.

Disable the log dump feature

You can disable the log dump feature for a product at any time. This allows you to save storage space. Procedure:

1. Log on to the IoT Platform console.
2. In the left-side navigation pane, choose **Maintenance > Device Log**.
3. Select a product and click the **Log Dump** tab.
4. Select the destination product, click **Stop Dump**, and click **OK**.
   New logs are no longer exported to the Logstore. Previous logs will not be deleted until these logs reach the end of the retention period.

5.OAT update

5.1. Push firmware files to devices

IoT Platform provides the firmware update and management feature. To update firmware, you must ensure that your devices support over-the-air (OTA) updates. Then, you can upload firmware files to the IoT Platform console and push update notifications to the devices. This article describes how to use the IoT Platform console to add, verify, and push firmware files to devices.

Prerequisites

Before you use the firmware update feature, ensure that your devices support OTA updates.

- For information about how to configure OTA updates by using device SDKs, see Update devices by using OTA.
- If you use the Link SDK for Android on devices and need to perform differential updates, you must integrate the algorithms to authenticate the devices and generate full update packages. For more information, see Implement a differential firmware update on an Android device.
- For information about how to configure OTA updates if your devices are installed with AliOS Things chips, see OTA tutorial for AliOS Things.

Context

Note the following limits of firmware updates:

- Each Alibaba Cloud account can have a maximum of 500 firmware files.
- The maximum size of a firmware file is 1,000 MB. The file format can only be .bin, .tar, .gz, .tar.gz, .zip, or .gzip.
- Note the following limits of update batches:

  Update batches: IoT Platform displays created update tasks as different update batches. You can go to the **Firmware Details** page and view update batches of the firmware on the **Batch Management** tab.

  - You can use a firmware file to create different update batches for multiple firmware versions to be updated.
  - You can use a firmware file to create only one dynamic update batch for a firmware version to be updated.
  - You can use different firmware files to create multiple dynamic update batches for a firmware version to be updated.
  - Each device can have only one ongoing update batch. A device that has an ongoing update batch is in the To Be Pushed, Pushed, or In Upgrade state.

- Only devices that are connected to IoT Platform by using the MQTT protocol support the firmware update feature.
- If devices are online, they can immediately receive update notifications. If devices are offline, IoT Platform pushes update notifications when devices go online.

Procedure

1. Log on to the IoT Platform console.

2. In the left-side navigation pane, choose **Maintenance > Firmware Update**.

   > **Note** To provide better services, IoT Platform improves the firmware update feature and allows you to manage firmware versions based on products. When you use the new firmware update feature in the console for the first time, you must associate the uploaded firmware files with products. You can associate a firmware file with only one product. For more information about how to associate firmware files with products, see the instructions in the console.

3. Optional. If your devices are installed with AliOS Things chips, you can enable the Secure Update feature.

   We recommend that you enable this feature to ensure integrity and confidentiality of firmware. If you use the Secure Update feature, you must validate the firmware and firmware signatures of devices. For more information, see OTA Tutorial for AliOS Things.

   i. Go to the **Firmware Update** page, and click **Secure Update**.

   ii. In the dialog box that appears, turn on the **Secure Update** switch for the product that is pending for update.When Secure Update is in the **Activated** state, click **Copy** in the Public Key column to copy the public key. You can use the public key to validate the signature of the device.

4. On the **Firmware Update** page, click **Add Firmware**.

5. In the **Add Firmware** dialog box, specify the parameters, upload a firmware file, and then click **OK**.

Parameter	Description
Type	◦ Full: If you select Full, you must upload a complete firmware file. The system pushes the complete firmware file to devices for update. ◦ Differential: If you select Differential, you must upload a file that contains only the differences between the destination firmware version and the original firmware version. IoT Platform pushes the differences to devices for update. Then, the differences are merged into the original firmware packages. Differential updates optimize the usage of device resources and minimize the traffic that is consumed during firmware pushing.
Firmware Name	The name of the firmware. The name can contain Chinese characters, English letters, digits, hyphens (-), underscores (_), and parentheses (). The name must be 1 to 40 characters in length.
Firmware Version	The version number of the firmware. The version number can contain letters, digits, periods (.), hyphens (-), and underscores (_). It must be 1 to 64 characters in length. You must specify this parameter if you set the Type parameter to **Full**.
Version number to be upgraded	The firmware versions to be updated. The drop-down list shows the firmware versions of all devices under the current product. You can select one or more firmware versions from the drop-down list. You must specify this parameter if you set the Type parameter to **Differential**.

Parameter	Description
Post-upgrade version number	The destination firmware version. You must specify this parameter if you set the Type parameter to **Differential**.
Affiliated Products	The product to which the specified firmware belongs.
Firmware Module	The module of the firmware. Firmware updates are based on different modules of devices under a product. ○ Select: Select one or more existing firmware modules. ○ Add: Customize a new module. The module name can contain letters, digits, periods (.), hyphens (-), and underscores (_). The module name must be 1 to 64 characters in length. Default value: default.
Signature Algorithm	The signature algorithm. Only MD5 and SHA256 are supported. If you use the Link SDK for Android and set the Type parameter to **Differential**, select the MD5 algorithm.
Select Firmware	Uploads a firmware file. The maximum size of a firmware file is 1,000 MB. The file format can only be .bin, .tar, .gz, .tar.gz, .zip, or .gzip.
Description	The description of the firmware features. The description can be up to 100 characters in length. Each Chinese symbol occupies two characters.

6. Validate the firmware on one or more devices. In the firmware list, find the required firmware and click **Validate Firmware** in the Actions column. In the dialog box that appears, specify the parameters, and click **OK**.

> **Note** After you upload a firmware file to IoT Platform, you must validate the firmware on devices. Then, you can use the firmware file to create batch update tasks.

Parameter	Description
Version number to be upgraded	The drop-down list shows the firmware versions of all devices under the specified product. You can select one or more firmware versions from the drop-down list. After you specify firmware versions, the devices that use these firmware versions are displayed in the **Device to be verified** field.
Device to be verified	Select one or more devices to be validated.
Device upgrade time-out (minutes)	The timeout period of the firmware update. If the specified device has not been updated within this period, the update times out. The period starts from the first time the specified device submits the update progress. Valid values: 1 to 1440. Unit: minutes.

7. Push update notifications to the devices. After the firmware is validated, click **Batch Update**. In the dialog box that appears, specify the parameters, and click **OK**.

Parameter	Description
Version number to be upgraded	The drop-down list shows the firmware versions of all devices under the specified product. You can select one or more firmware versions to be updated from the drop-down list. You must specify this parameter if you set the Type parameter to Full.
Update Policy	○ Static Update: updates only the current devices that meet the conditions. ○ Dynamic Update: constantly updates the devices that meet the conditions. Dynamic updates can be applied to the following scenarios: ■ The devices that are subsequently activated meet the conditions. ■ The current firmware versions that devices submit do not meet the conditions. However, the devices subsequently submit the firmware versions that meet the conditions. Note You can use a firmware file to create only one dynamic update batch. If you have created a dynamic update batch by using a firmware file, you must cancel this dynamic update batch before creating another one.
Apply Update to	○ All Devices: updates all devices under the specified product. ○ Selected Devices: If you select Selected Devices, you must specify a range for the devices to be updated. In the dialog box that appears, select the devices to be updated. Only the selected devices are updated. Note You can select multiple firmware versions if you select the Selected Devices option. By default, the firmware versions that are specified by the Version number to be upgraded parameter are selected. ○ Phased Update: updates the specified devices. This option appears if you select Static Update from the Update Policy drop-down list. After you select Phased Update, specify a range for devices in the Range (%) field that appears. IoT Platform calculates the number of devices to be updated based on the specified range. The calculation result is rounded down. You must specify at least one device for a phased update.

Parameter	Description
Update Time	The time when the device firmware is updated. ○ Update: immediately updates the firmware. ○ Scheduled Update: updates the firmware at a specified time range. You can specify a start time and end time. 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 available if you select Static Update from the Update Policy drop-down list.
Recipients per Minute	The number of devices to which you want to push the download URL of the firmware file per minute. Valid values: 10 to 1000.
Retry Interval	The interval after which a retry is performed if the update fails. Valid values: ○ Do Not Retry ○ Retry Immediately ○ Retry in 10 Minutes ○ Retry in 30 Minutes ○ Retry in 1 Hour ○ Retry in 24 Hours
Maximum Retries	The maximum number of retries after the update fails. Valid values: ○ 1 ○ 2 ○ 5
Timeout (Minutes)	The timeout period of the firmware update. If the specified device has not been updated within this period, the update times out. The period starts from the first time the specified device submits the update progress. Valid values: 1 to 1440. Unit: minutes.
Override Previous Device Update Tasks	Specifies whether to overwrite the previous update task. Each device can be in only one ongoing update task at a time. The To Be Pushed, Pushed, or In Upgrade state is displayed if a device is in an ongoing update task. ○ If you select Yes, only the new update task is performed. ○ If you select No and the device already has an update task, the previous task is implemented. The update task that is in progress is not overwritten.

Parameter	Description
Take Effect for only Devices that Newly Report Versions	This parameter is available if you set the Update Policy parameter to **Dynamic Update**. This parameter specifies whether to update only the devices that submit firmware versions later. Ensure that the submit firmware versions are used as the firmware versions to be updated. ○ If you select Yes, only the devices that subsequently submit firmware versions are updated. ○ If you select No, the current devices that meet the conditions are updated. In addition, IoT Platform constantly checks the devices. If the devices that subsequently submit firmware versions meet the conditions, these devices are updated.

Result

After you submit a batch update request, IoT Platform pushes an update notification to your devices based on your settings.

Find the required firmware and click **View** in the Actions column. On the **Batch Management** tab, you can perform the following operations:

- View the status of an update batch.
- Cancel all update tasks of an update batch.

View the status of an update batch

On the **Batch Management** tab, find the required update batch and click **View** in the Actions column. On the **Batch Details** page, you can view devices in different update statuses on the **Device List** tab.

State	Description
To Be Pushed	The firmware update notification has not been pushed to the device. This state may appear due to the following three causes: 1. The device is offline. 2. The notification is scheduled to be pushed. 3. The pushing rate exceeds the limit. The following states that correspond to these three causes are displayed: • **To Be Pushed (Device Offline)** • **To Be Pushed (Timing: 2020/XX/XX XX:XX:XX)** • **To Be Pushed**
Pushed	The device has received the firmware update notification but has not submitted the progress.
In Upgrade	The device has received the firmware update notification and submitted the update progress.
Updated	The device has submitted the update progress 100% and valid version number after the update.

State	Description
Updated Failed	Firmware updates may fail due to the following causes: • If a device receives a new update notification before the previous update is complete, the new update fails. In this case, you can update the device after the previous update is complete. • During the updating process, a firmware file may fail to be downloaded, extracted, or validated. If a failure occurs, you can initiate a new update. If you configure a batch update and specify automatic update retries, the retries are performed after the update fails. An update may fail due to one of the following causes: • The device in the **To Be Pushed** or **Pushed** state submits a version that is not the current version or the destination version. • The device in the **In Upgrade** state submits a version that is not the destination version. • The device submits one of the following values when using a specified 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 occurs on a device that is in the **Pushed** state, the device status is still displayed as **Pushed** in IoT Platform. If a retry occurs on a device that is in the **In Upgrade** state, the device status is still displayed as **In Upgrade** in IoT Platform. **Note** IoT Platform does not perform update retries if an update fails due to one of the following causes: • The update times out . • The update is canceled.
Canceled	The update for the device is canceled.

Cancel all update tasks of an update batch

On the **Batch Management** tab, find the required update batch and click **Cancel** in the Actions column.

- For a static update batch, only scheduled update tasks are canceled by default. You can cancel all ongoing update tasks, including those in the To Be Pushed, Pushed, and In Upgrade states.
- For a dynamic update batch, the dynamic update policy is canceled by default. You can cancel all ongoing update tasks, including those in the To Be Pushed, Pushed, and In Upgrade states.

5.2. Update devices by using OTA

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

OTA update process of device firmware

The following figure shows the process of a firmware update.

The following topics are used during the firmware update:

- Devices use the following topic to report firmware versions to IoT Platform.

  ```
  /ota/device/inform/${YourProductKey}/${YourDeviceName}
  ```

- Devices receive firmware update notifications by subscribing to the following topic.

  ```
  /ota/device/upgrade/${YourProductKey}/${YourDeviceName}
  ```

- Devices use the following topic to report the progress of the firmware update.

  ```
  /ota/device/progress/${YourProductKey}/${YourDeviceName}
  ```

> **Note**
>
> - A device needs to report the firmware version only once during the startup.
> - In the IoT Platform console, after you start updating multiple devices, each device is in the Pending Update state.
>
>   When the OTA service of IoT Platform receives the update progress that is reported by a device, the status of the device changes to Updating.
> - You can check if an OTA update is successful based on the device version.
> - An offline device cannot receive an update notification from IoT Platform.
>
>   After the status of the device changes to online, IoT Platform identifies the change and the OTA system verifies whether the device requires an update. If an update is required, the OTA system sends an update notification to the device. Otherwise, no notification is sent.

Message format

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

1. To connect to the OTA system, a device must report the firmware version to a topic.

   The format of the topic is `/ota/device/inform/${YourProductKey}/${YourDeviceName}`. Sample message:

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

Parameters

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

2. In the IoT Platform console, you can add firmware, verify firmware, and update firmware.

   For more information, see Push firmware files to devices.

3. After you start a firmware update in the console, the device receives the URL of the firmware update 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}` .

   Sample message:

```
{
 "code": "1000",
 "data": {
  "size": 432945,
  "version": "2.0.0",
  "isDiff": 1,
  "url": "https://iotx-ota-pre.oss-cn-shanghai.aliyuncs.com/nopoll_0.4.4.tar.gz?Expires=1502955804&OSSAccessKeyId=XXXXXXXXXXXXXXXXXXXX&Signature=XfgJu7P6DWWejstKJgXJEH0qAKU%3D&security-token=CAISuQJ1q6Ft5B2yfSjIpK6MGsyN1Jx5jo6mVnfBglIPTvlvt5D50Tz2IHtIf3NpAusdsv03nWxT7v4flqFyTINVAEvYZJOPKGrGR0DzDbDasumZsJbo4f%2FMQBqEaXPS2MvVfJ%2BzLrf0ceusbFbpjzJ6xaCAGxypQ12iN%2B%2Fr6%2F5gdc9FcQSkL0B8ZrFsKxBltdUROFbIKP%2BpKWSKuGfLC1dysQcO1wEP4K%2BkkMqH8Uic3h%2Boy%2BgJt8H2PpHhd9NhXuV2WMzn2%2FdtJOiTknxR7ARasaBqhelc4zqA%2FPPlWgAKvkXba7aIoo01fV4jN5JXQfAU8KLO8tRjofHWmojNzBJAAPpYSSy3Rvr7m5efQrrybY1lLO6iZy%2BVio2VSZDxshI5Z3McKARWct06MWV9ABA2TTXXOi40BOxuq%2B3JGoABXC54TOlo7%2F1wTLTsCUqzzeIiXVOK8CfNOkfTucMGHkeYeCdFkm%2FkADhXAnrnGf5a4FbmKMQph2cKsr8y8UfWLC6IzvJsClXTnbJBMeuWIqo5zIynS1pm7gf%2F9N3hVc6%2BEeIk0xfl2tycsUpbL2FoaGk6BAF8hWSWYUXsv59d5Uk%3D",
  "md5": "93230c3bde425a9d7984a594ac55ea1e",
  "sign": "93230c3bde425a9d7984a594ac55****",
  "signMethod": "Md5",
  "module": "MCU"
 },
 "id": "1507707025",
 "message": "success"
}
```


Parameters

Parameter	Value	Description
id	String	The message ID. Valid values: 0 to 4294967295. Each message ID must be unique for the device.
message	String	The result.
code	String	The HTTP status code.
version	String	The version of the firmware.
size	Long	The size of the firmware. Unit: bytes.
url	String	The endpoint of the Object Storage Service (OSS) bucket that stores the firmware.

Parameter	Value	Description
isDiff	Long	If the firmware type is differential, the message includes the isDiff parameter that is set to 1. The value 1 indicates that the firmware file is a differential package. The differential package includes only the differences between the original firmware and the new firmware. The device must integrate the differential package with the original firmware to form the new firmware. If the firmware type is full, this parameter is not included.
sign	String	The firmware signature.
signMethod	String	The signature algorithm. Valid values: ○ SHA256 ○ Md5 The MD5 algorithm is available only for Android differential packages.
md5	String	If the signature algorithm is MD5, IoT Platform assigns values to the sign and md5 parameters.
module	String	The name of the module to which the firmware belongs. **Note** If the name of the default module is specified, IoT Platform does not send the module parameter.

4. After the device receives the URL, the device connects to the URL by using HTTPS and download the firmware update from the URL.

   Note If the device fails to download the firmware update from the URL within 24 hours, the URL becomes invalid.

   During a firmware update, a device must send the update progress to a topic in the following format: `/ota/device/progress/${YourProductKey}/${YourDeviceName}`. Sample message:

```
{
 "id": "123",
 "params": {
  "step": "-1",
  "desc": "Firmware update has failed. No firmware information is available.",
  "module": "MCU"
 }
}
```

Parameters

Parameter	Value	Description
id	String	The message ID. Valid values: 0 to 4294967295. Each message ID must be unique for the device.
step	String	The progress information of the firmware update. Valid values: ○ 1 to 100: indicates a percentage of 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 writing failed.
desc	String	The description of the step parameter. If an exception occurs, this parameter contains the error message.
module	String	The name of the module to which the firmware belongs. Note If the device reports the firmware version of the default module, the module parameter is optional.

5. After the firmware 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 reported version is the same as the version that the OTA service sends to the topic, the firmware update is successful. Otherwise, the firmware update fails.

   Note A successful firmware update is indicated based only on the reported version. For example, a device reports an update progress of 100%. However, no firmware version is reported. This case is also considered as a failed update.

Common errors during firmware downloads

- Sign failed. If the URL of a firmware update is invalid or changed, this issue occurs, as shown in the following figure.

  

- Access denied. This error occurs due to an expired URL. The URL is valid for 24 hours.

  

References

For more information, see Implement OTA updates for devices.

6.Remote configuration

IoT Platform provides the remote configuration function, which allows device configurations to update online when the device is in service.

Prerequisites

- You have activated the remote configuration function in the IoT Platform console. If you have not activated this function, log on to the IoT Platform console and then, in the left-side navigation pane, click **Maintenance > Remote Config.**. Then, turn on remote configuration.
- You have configured your device SDK to support the remote configuration function. Define `FEATURE_SERVICE_OTA_ENABLED=y` in the device SDK. The SDK provides the linkkit_cota_init operation to initialize remote configurations such as Config Over The Air (COTA).

Introduction to the remote configuration function

Developers often need to update device configurations, such as the system parameters, network parameters, and security policies of devices. Generally, device configurations are updated using the firmware update function. However, firmware update requires more time for firmware version maintenance, and devices must stop their services in order to install the update. To streamline the device configuration update process, IoT Platform provides the remote configuration function. This function enables you to complete configuration updates without service interruption.

With the remote configuration function, you can perform the following operations:

- Enable or disable remote configuration.
- Edit configuration files and perform version management in the IoT Platform console.
- Update the configuration information for all devices of a product at one time.
- Enable devices to send requests for configuration update from IoT Platform.

Remote configuration flow chart:

The processes involved in remote configuration include the ability to:

- Edit and save configuration files in the IoT Platform console.
- Push configuration updates to all devices of a product in the IoT Platform console. Then, when the devices receive the update requests, they immediately update their configurations.
- Devices can also send requests for configuration updates from IoT Platform, and then perform update when configuration information is received.

Use the remote configuration function

The remote configuration function is mainly designed for two scenarios, namely, you want to push configuration updates to devices from IoT Platform, or you want to allow devices to send requests for configuration updates. The process of using the remote configuration function varies based on different scenarios.

Scenario 1: Push configuration information to devices from IoT Platform.

In the IoT Platform console, you can push device configuration updates to all devices of a product.

1. Connect the devices to IoT Platform and configure the devices to subscribe to the topic `/sys/${productKey}/${deviceName}/thing/config/push`.

2. In the IoT Platform console, edit a configuration file.
   i. In the left-side navigation pane, click **Maintenance > Remote Config.**.
   ii. Select the product for which you want to use the remote configuration function, and enable the function.

   > **Note**
   > - Only if you enable the remote configuration function for the selected product can you edit a configuration template file for it.
   > - If the remote configuration function is not enabled, devices of the product cannot be updated in this way.
   > - A configuration template file that you edit here is used by all the devices of the product. Currently, you cannot push a configuration file to a specified device.

   iii. Click **Edit**, and then edit a configuration template in the area of **Configuration Template**.

   > **Note**
   > - Remote configuration files are JSON files. IoT Platform does not have special requirements for the configuration content. The system only checks the format of the data when you submit the configuration file. This is to prevent errors that are caused by format errors.
   > - The configuration file can be up to 64 KB. The file size is dynamically displayed in the upper-right corner of the editing area. Configuration files larger than 64 KB cannot be submitted.

   iv. After you have completed editing the configuration information, click **Save** to generate the configuration file. The system then allows devices to send requests for the configuration file.
3. Push the configuration file to devices. Click **Batch Update** and then IoT Platform sends the configuration file to all the devices of the product.

   After you click **Batch Update**, the system may initiate SMS authentication to verify your account. If authentication is required, you need to first complete account verification, and then the system sends the configuration file to the devices.

   > **Note**
   > - Operation frequency limit: You can only perform a batch update once per hour.
   > - If you want to stop pushing configuration updates, disable the remote configuration function for the product. The system then stops pushing the update file and will deny update requests from devices.

4. Devices automatically update the configuration after receiving the configuration file from IoT Platform.

Configuration file management:

The latest five configuration files are saved in the console by default. After you edit and save a new version of configuration file, the previous version is automatically displayed in the configuration version record list. You can view the update time and content of the displayed five versions.

Click **View** to view the configuration content of the version. Click **Recover to This Version**, and the configuration content of this version will be displayed in the editing box. You can edit the content and then save it as a new version.

Scenario two: Devices send requests for configuration information

If devices are configured to send requests for configuration information, you need to enable the remote configuration function. To do so, follow these steps:

1. Configure the devices to subscribe to the topic `/sys/${productKey}/${deviceName}/thing/config/get_reply`.
2. In the IoT Platform console, enable the remote configuration function and edit a configuration file. For detailed steps, see the related procedures in Scenario 1.
3. Configure the devices to call the linkkit_invoke_cota_get_config operation to trigger requests for remote configuration.
4. Configure the devices to send requests for the latest configuration updates through the topic `/sys/${productKey}/${deviceName}/thing/config/get`.
5. IoT Platform returns the latest configuration information to the devices after receiving the requests.
6. The devices use the cota_callback function to process the configuration file that is sent through the remote configuration function.