RT-THREAD ALI-IOTKIT User Manual

RT-THREAD Document Center

WWW.RT-THREAD.ORG

Friday 28th September, 2018

Versions and Revisions

Date	Version	Author	Note
2018-07-21	v0.1	MurphyZhao	Initial version

Table of contents

Chapter 1

Software Package Introduction

ali-iotkit is a software package ported from RT-Thread for connecting to the Alibaba Cloud IoT platform. The basic SDK is the iotkit-embedded C-SDK provided by Alibaba .

The IoT Suite provides the following capabilities:

• Fast access to embedded devices (device-side SDK) •

Device

management • Device and data

information security • Bridging to other Alibaba Cloud products for device data storage/computing

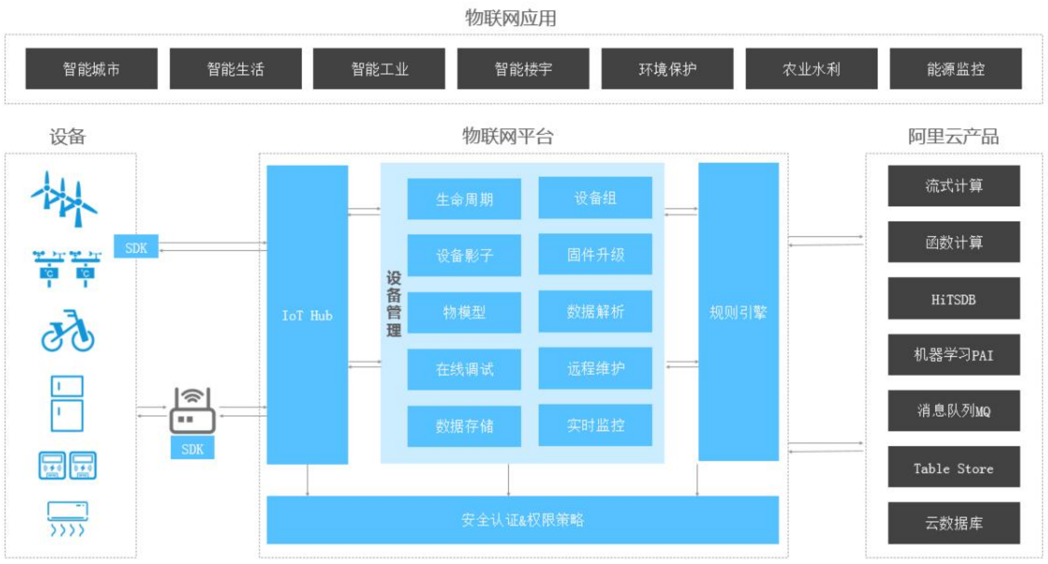


Figure **1.1**: Alibaba Cloud IoT Platform Scenario Architecture Diagram

In the IoT platform scenario architecture diagram, the IoT device SDK on the left is the part that connects the embedded device to Alibaba Cloud.

Machine Translated by Google

1.1 Software Framework Diagram

To facilitate device-to-cloud connectivity, the iotkit SDK encapsulates a rich set of connection protocols, such as MQTT, CoAP, HTTP, and TLS. It also abstracts the hardware platform, making it more flexible and independent of specific hardware platforms. In terms of code architecture, the iotkit SDK is divided into three layers, as shown in the following figure:

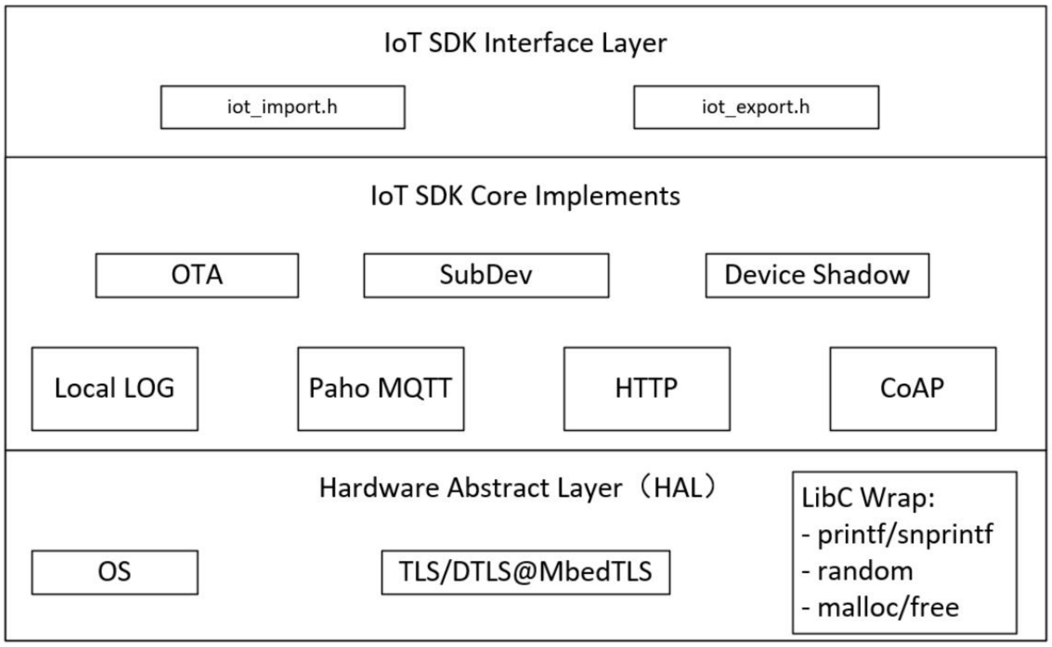


Figure **1.2:** *iotkit SDK* Software framework diagram

- The bottom layer is called the Hardware Abstraction Layer, also known as the HAL layer (Hardware Abstract Layer)

  Abstracts the operating system's support functions for the SDK on different embedded target boards, including network transmission and reception, TLS/DTLS channel establishment and reading and writing, memory allocation, and mutex locking and unlocking.

- The middle layer is called the SDK Core Implements layer (IoT SDK Core Implements)

  The core implementation part of the IoT platform C-SDK, which completes the functional encapsulation of MQTT/CoAP channels based on the HAL layer interface, including MQTT connection establishment, message sending and receiving, CoAP connection establishment, message sending and receiving, OTA firmware status query and OTA firmware download, etc.

  The encapsulation of the middle layer means that users do not need to worry about the internal implementation logic and can apply it without modification.

- The top layer is called the SDK interface declaration layer (IoT SDK Interface Layer)

  The top layer provides APIs for applications, and users use the APIs of this layer to complete specific business logic.

1.2 Software package directory structure

The ports directory contains the porting files involved in RT-Thread porting the iotkit-embedded software package and is rebuilt using scons.

The iotkit-embedded package is the source code of Alibaba IoT platform C-SDK, which contains the necessary

software package.

```
sub-iotkits
 | README.md |                                   // Software package instructions
 SConscript                                      //RT-Thread default build script
+---docs
 | +---figures | | api.md | |                    // Document using images
 introduction.md | | | |                         // API usage instructions
 principle.md | | README.md | |                  // Software package details
                   LICENSE                       // License file
 samples.md | | user-guide.md |                  // Implementation principle
 +---version.md                                  // Document structure description
                                                 // Package example
                                                 // Instructions
                                                 // Version description
+---ports | +---                                 // Migrate files
 rtthread | +---ssl +---                         // OS related porting files
 samples                                         // MbedTLS related porting files

 | +---mqtt | +---ota                            // Sample program for connecting the MQTT channel to Alibaba Cloud
                                                 // Alibaba Cloud OTA function demonstration program
+---iotkit-embedded                              // iotkit source code
```

1.2.1 iotkit-embedded software package directory structure

The iotkit-embedded package is the source code of Alibaba IoT Platform C-SDK, which is unmodified and includes the

The necessary software package for cloud IoT. After RT-Thread is transplanted, the default Makefile in iotkit-embedded is not used.

Build scripts, but rebuild them using scons.

The directory structure of the iotkit-embedded software package is as follows:

```
+-- LICENSE                 : Software License, Apache-2.0 Software License
+-- make.settings           : Function tailoring configuration , such as MQTT|CoAP, or tailoring such as OTA|Shadow
+-- README.md               : Quick Start Guide
+-- sample | | | |          : Routine directory , demonstrating the use of communication modules and service modules
       +-- mqtt             : Demonstrates how to use the API of the communication module MQTT
       +-- coap +--         : Demonstrates how to use the API of the communication module CoAP
       device-shadow : Demonstrates how to use the API of the service module DeviceShadow
       +-- http             : Demonstrates how to use the communication module HTTP API
 |     +-- order            : Demonstrate how to use the service module OTA API
+-- src
```

Machine Translated by Google

```
+-- sdk-impl package        : SDK interface layer, providing overall header files and some API interfaces

+-- sdk-tests               : SDK unit tests
+-- mqtt                    : Communication module , realize access via MQTT protocol
+-- coap                    : Communication module , realize access via CoAP protocol
+-- http                    : Communication module , realize access via HTTP protocol
+-- order                   : Service module , implements firmware download based on MQTT|CoAP+HTTP+TLS
        aisle
+-- shadow                  : Service module , implement device shadow
+-- platform +--            : Hardware platform abstraction layer, needs to be ported and adapted
import library              : External input directory , storing header files/ binary files provided by chip /module manufacturers

+-- configs                 : Hardware platform compilation configuration , such as cross-compilation tool chain settings , functional modules
        tailoring, etc.
+-- scripts +--             : The compilation process will reference the script externally , and the user does not need to pay attention to it
packages +-- log            : External software modules referenced by SDK , users do not need to pay attention to
                            : Basic module , realizes running log
+-- system                  : Basic module , saves global information , such as TLS root certificate , device identification
        ID, etc.
+-- tls                     : Basic module , implementing TLS/DTLS, from tailored open source software
        mbedtls
+-- utils                   : Basic module , implements tool functions , such as SHA1 summary calculation, NTP pair
        Time
```

1.3 Features

• Different network access

Provide device access solutions for different networks, such as 2/3/4G, NB-IoT, LoRa, etc., to solve enterprise heterogeneous networks

The pain points of device access management.

• Access with different protocols

Provides device SDKs for multiple protocols, such as MQTT, CoAP, HTTP, etc., which can meet the needs of devices

Long connections ensure real-time requirements and can also meet the needs of devices that require short connections to reduce power consumption.

• Two-way communication

Providing uplink and downlink channels between devices and the cloud, it can stably and reliably support scenarios where devices report and send commands to devices.

• Device Shadow

Provides a device shadow cache mechanism to decouple devices from applications and solve communication problems when wireless networks are unstable.

Reliable pain points.

• Device authentication

Provides a one-machine-one-key device authentication mechanism to reduce the security risk of device compromise.

- Secure transmission

  Provides TLS standard data transmission channel to ensure data confidentiality and integrity.

Chapter 2

Sample Program

The **ali-iotkit** software package also supports Alibaba's existing **LinkDevelop** and **LinkPlatform** platform.

This article demonstrates sample programs for these two platforms respectively. Users can choose to use one of them according to their needs.

One.

2.1 LinkDevelop Platform

The LinkDevelop platform uses RGB_LED as an example to explain how to achieve two-way communication between the device and the cloud.

2.1.1 Preparation

• Register on the LinkDevelop platform

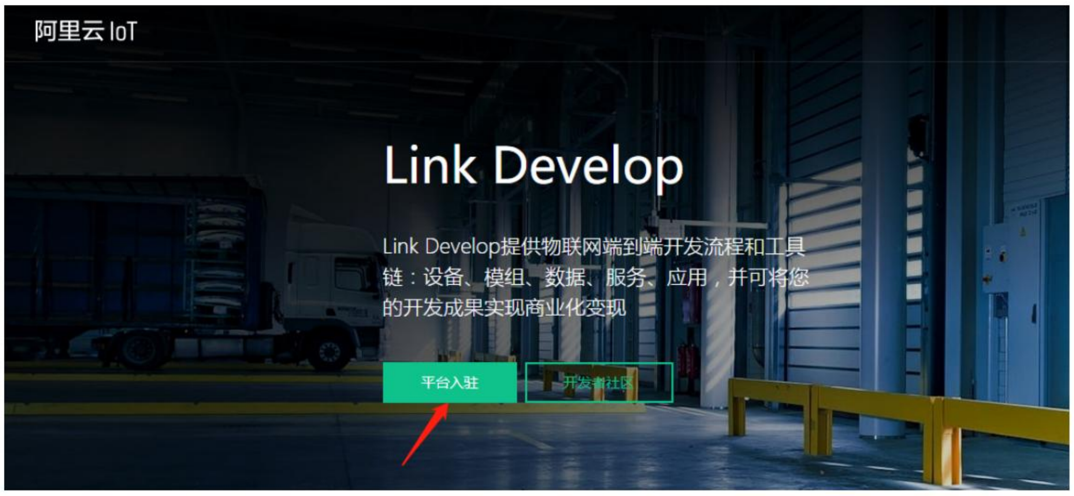

阿里云 IoT
Link Develop
Link Develop提供物联网端到端开发流程和工具链：设备、模组、数据、服务、应用，并可将您的开发成果实现商业化变现
平台入驻
开发者社区

Figure **2.1:** register *LinkDevelop* platform

• New Project

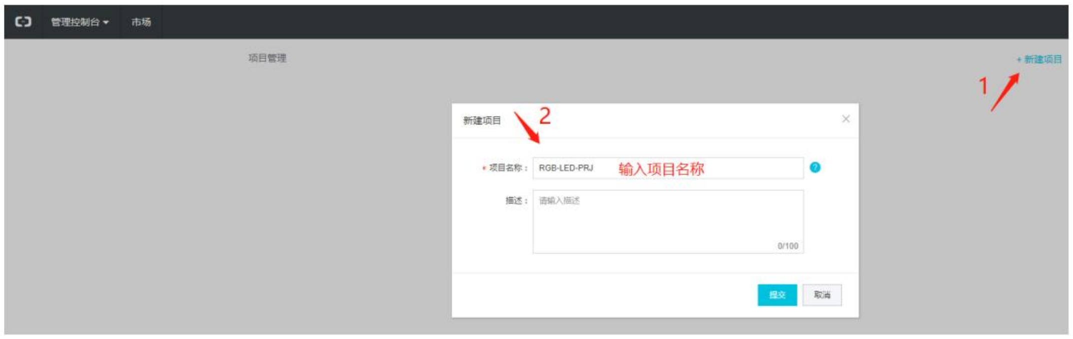
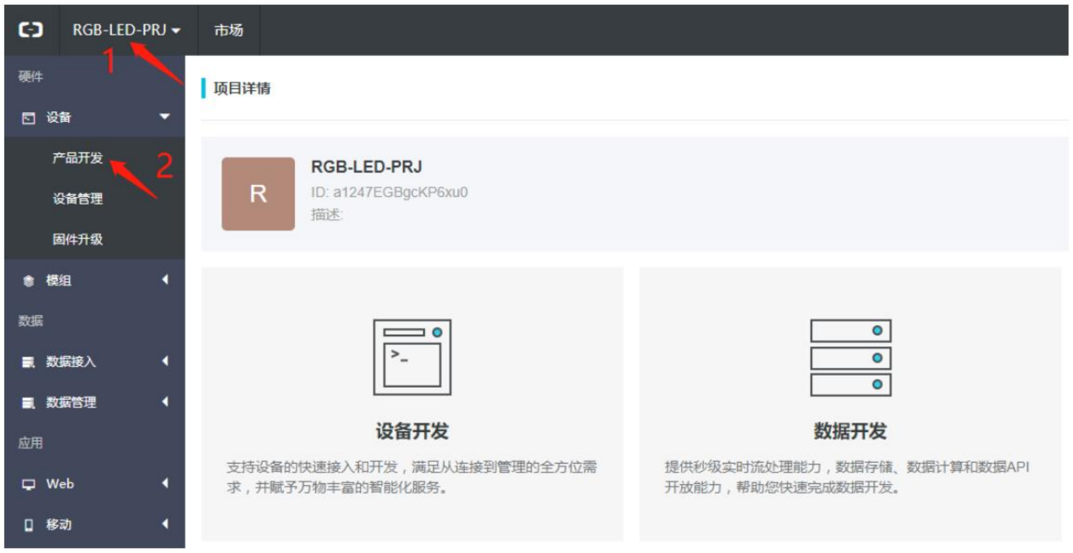


Figure 2.2: New Project

• New Products

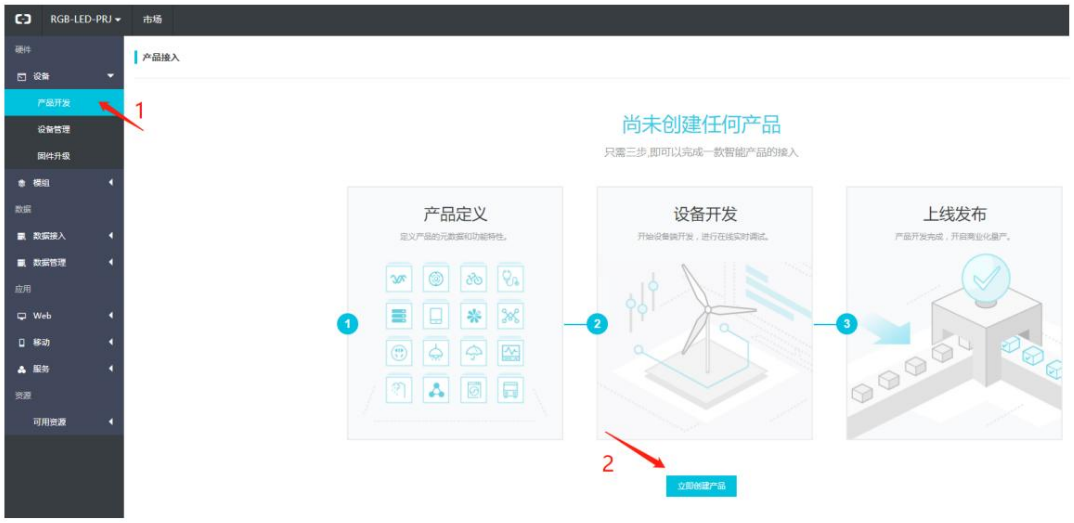
When adding a new product, select the data format as needed. Here we use the **Alink** data format demonstration and select WiFi communication method.



支持设备的快速接入和开发，满足从连接到管理的全方位需求，并赋予万物丰富的智能化服务。

提供秒级实时流处理能力，数据存储、数据计算和数据API开放能力，帮助您快速完成数据开发。

Figure 2.3: Open the product development page



尚未创建任何产品

只需三步,即可以完成一款智能产品的接入

产品定义

定义产品的元数据和功能特性。

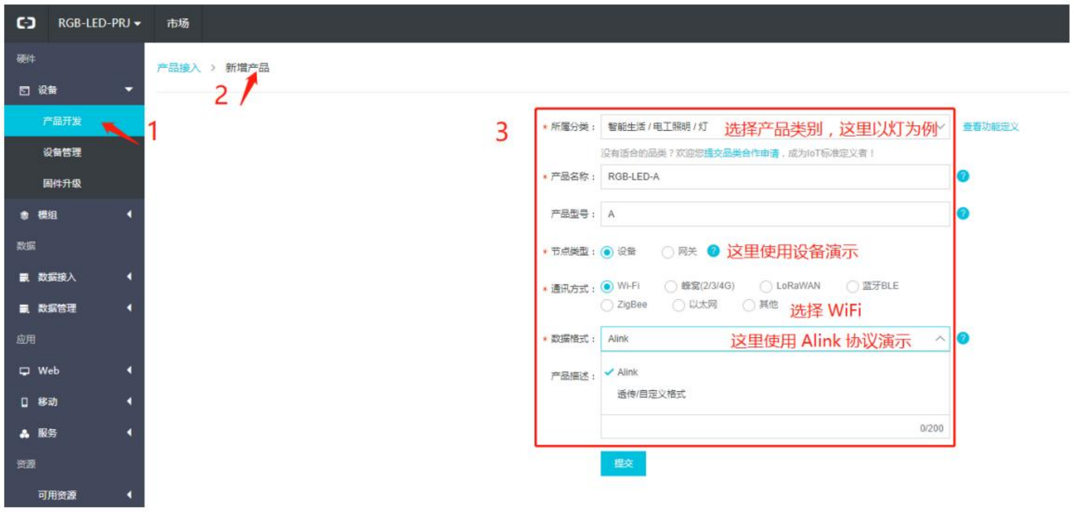
设备开发

开始设备端开发，进行在线实时调试。

上线发布

产品开发完成，开启商业化量产。

Figure 2.4: Create a product



* 所属分类： 智能生活 / 电工照明 / 灯 选择产品类别，这里以灯为例

没有适合的品类？欢迎您提交品类合作申请，成为IoT标准定义者！

* 产品名称： RGB-LED-A

产品型号： A

* 节点类型： [✓ 设备] 网关 这里使用设备演示

* 通讯方式： [✓ Wi-Fi] 蜂窝(2/3/4G) LoRaWAN 蓝牙BLE ZigBee 以太网 其他 选择 WiFi

* 数据格式： Alink 这里使用 Alink 协议演示

✓ Alink

透传/自定义格式

产品描述： 0/200

Figure 2.5: Write product information

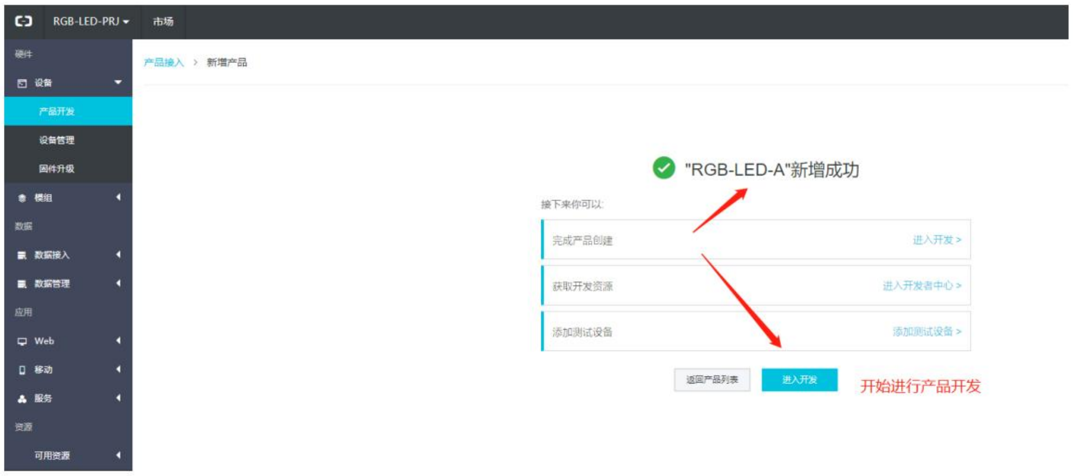


Figure 2.6: Enter product development

• Add functionality

Add RGB color adjustment function to the RGB LED demonstration product, as shown below:

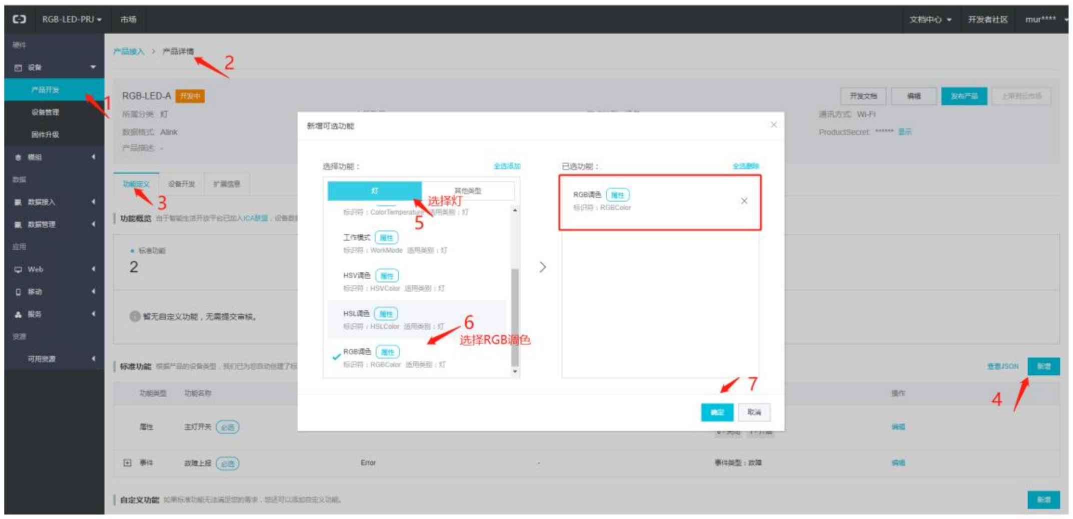
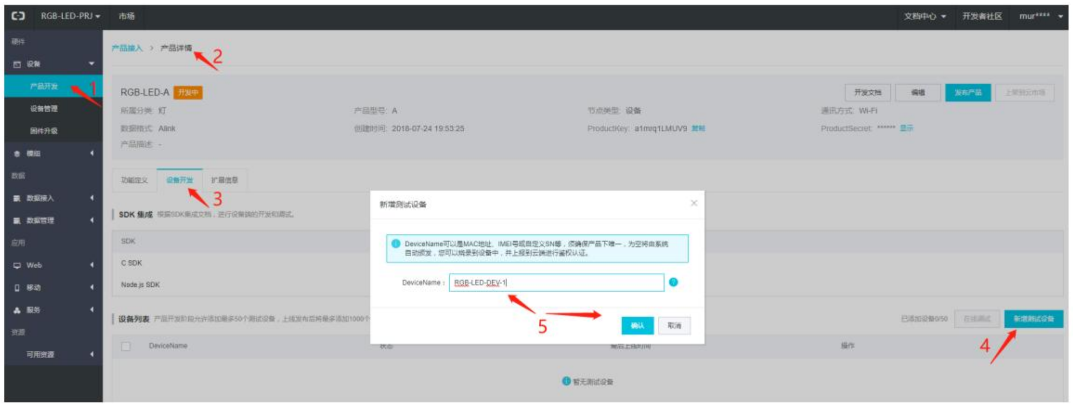


Figure 2.7: Add product features

• Add a device

After creating a product, click View to enter the product details page, click Device Development, and add a new debugging device.



新增测试设备

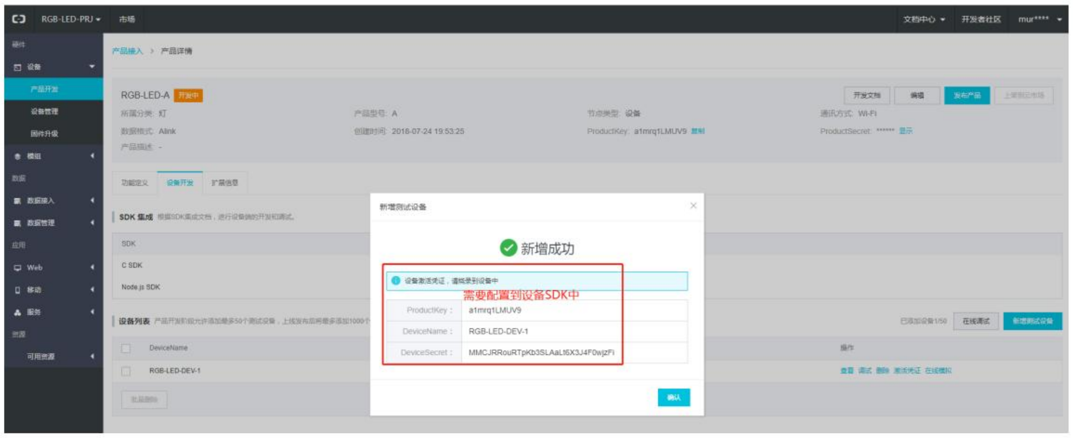
DeviceName可以是MAC地址、IMEI号或自定义SN等，须确保产品下唯一，为空将由系统自动颁发，您可以烧录到设备中，并上报到云端进行鉴权认证。

DeviceName：RGB-LED-DEV-1

Figure 2.8: Add a device

After successfully creating the device, you can obtain the triplet required for device activation (ProductKey, DeviceName,

DeviceSecret), which needs to be configured in the device SDK using **menuconfig** later.



新增测试设备

新增成功

设备激活凭证，请烧录到设备中

需要配置到设备SDK中

ProductKey：	a1mrq1LMUV9
DeviceName：	RGB-LED-DEV-1
DeviceSecret：	MMCJRRouRTpKb3SLAaLt6X3J4F0wjzFi

Figure 2.9: Obtaining device activation credentials

- Get the software package

Open the ENV tool provided by RT-Thread and use **menuconfig** to configure the software package.

–Configure iotkit software package

Configure and enable the iotkit software package and fill in the device activation credentials.

In menuconfig , select **LinkDevelop** for Alibaba Cloud Platform and MQTT for OTA channel **(with**

Taking MQTT as an example), the detailed configuration is as follows:

```
RT-Thread online packages --->
    IoT - internet of things --->
        IoT Cloud --->
          [*] Ali-iotkit: Ali Cloud SDK for IoT platform --->
                Select Aliyun platform (LinkDevelop Platform) ---> (a1dSQSGZ77X) Config Product
          Key
```

```
                    (RGB-LED-DEV-1) Config Device Name
                    (Ghuiyd9nmGowdZzjPqFtxhm3WUHEbIIl) Config Device Secret
                    -*- Enable MQTT [*]
                            Enable MQTT sample [*]
                            Enable MQTT direct connect [*] [
                            Enable SSL
                      ] Enable COAP
                    [*] Enable OTA
                                Select OTA channel (Use MQTT OTA channel)
                                    --->
                              version (latest) --->
```

– Increase mbedTLS frame size

During Alibaba's TLS authentication process, the data packet is large, so the TLS frame size needs to be increased. It needs to be at least **8K** in size during OTA .

Open the ENV tool provided by RT-Thread and use **menuconfig** to configure the **TLS** frame size.

```
RT-Thread online packages ---> security
    packages --->
      -*- mbedtls:An open source, portable, easy to use, readable and flexible
                  SSL library --->
      (8192) Maxium fragment length in bytes
```

– Download the package using the pkgs --update command

2.1.2 MQTT Example

This MQTT sample program uses RGB-LED as an example to demonstrate how to use MQTT + TLS/SSL on the device. The channel establishes two-way communication with the Alibaba Cloud platform.

Sample File

Sample program path	Verification Platform	illustrate
samples/mqtt/ mqtt-example.c	LinkDevelop, LinkPlatform	Device and cloud two-way communication routine based on MQTT channel

Command List

In the example, use the MSH command to start the MQTT example. The command is as follows:

Order	illustrate
ali_mqtt_test start	Start the MQTT example
ali_mqtt_test pub open	Turn on the light and synchronize the light status to the cloud
ali_mqtt_test pub close	Turn off the lights and synchronize the light-off status to the cloud
ali_mqtt_test stop	Stopping the MQTT Example

Start **MQTT**

Run the **ali_mqtt_test start** command to start the MQTT example. If successful, the device log will show that the subscription is successful.

The device log is as follows:

```
msh />ali_mqtt_test start
ali_mqtt_main|645 :: iotkit-embedded sdk version: V2.10 [inf]
iotx_device_info_init(40): device_info created successfully! [dbg] iotx_device_info_set(50):
start to set device info! [dbg] iotx_device_info_set(64): device_info set
successfully!
...
[inf] iotx_mc_init(1703): MQTT init success! [inf]
_ssl_client_init(175): Loading the CA root certificate ...
...
[inf] _TLSConnectNetwork(420): . Verifying peer X.509 certificate.. [inf] _real_confirm(92):
certificate verification result: 0x200 [inf] iotx_mc_connect(2035): mqtt connect success!

...
[inf] iotx_mc_subscribe(1388): mqtt subscribe success,topic = /sys/ a1HETlEuvri/RGB-LED-
    DEV-1/thing/service/property/set! [inf] iotx_mc_subscribe(1388): mqtt
subscribe success,topic = /sys/
    a1HETlEuvri/RGB-LED-DEV-1/thing/event/property/post_reply! [dbg]
iotx_mc_cycle(1269): SUBACK event_handle|
124 :: subscribe success, packet-id=0 [dbg] iotx_mc_cycle(1269):
SUBACK event_handle|124 :: subscribe
success, packet-id=0 [inf] iotx_mc_keepalive_sub(2226): send MQTT
ping... [inf] iotx_mc_cycle(1295): receive ping response!
```

Device release message

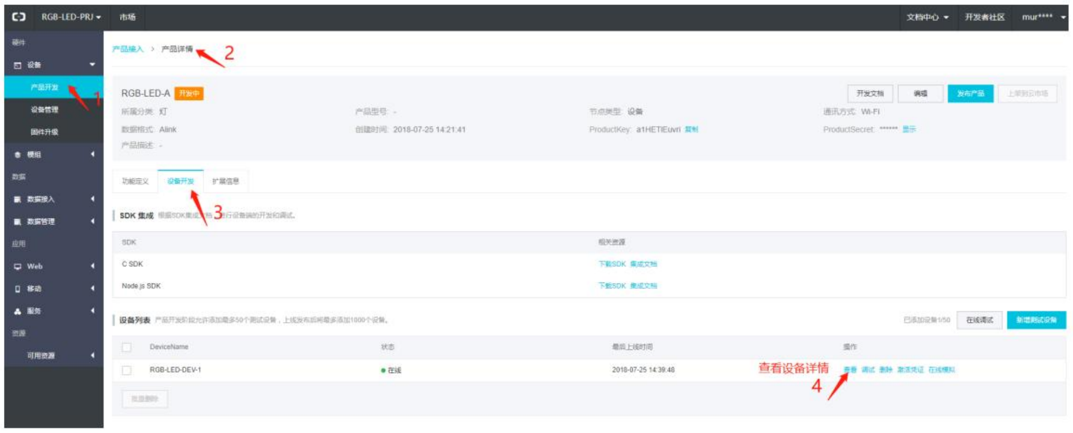
Use **the ali_mqtt_test pub open** command to send the LED status to the cloud. If successful, the device log will display a success code.
200ÿ

The device log is as follows:

```
msh />ali_mqtt_test pub open
...

[dbg] iotx_mc_cycle(1277): PUBLISH
[dbg] iotx_mc_handle_recv_PUBLISH(1091): [dbg]                Packet Ident : 00000000
iotx_mc_handle_recv_PUBLISH(1092): [dbg]                      Topic Length : 57
iotx_mc_handle_recv_PUBLISH(1096): Topic Name : /sys/
        a1HETlEuvri/RGB-LED-DEV-1/thing/service/property/set
[dbg] iotx_mc_handle_recv_PUBLISH(1099): [dbg]             Payload Len/Room : 100 / 962
iotx_mc_handle_recv_PUBLISH(1100): [dbg]                     Receive Buflen : 1024
iotx_mc_handle_recv_PUBLISH(1111): delivering msg ...
[dbg] iotx_mc_deliver_message(866): topic be matched
_demo_message_arrive|182 :: ----
_demo_message_arrive|183 :: packetId: 0
_demo_message_arrive|187 :: Topic: '/sys/a1HETlEuvri/RGB-LED-DEV-1/thing/
        service/property/set' (Length: 57)
_demo_message_arrive|191 :: Payload:
 '{"method": "thing.service.property.set","id": "36195462","params":{"
        LightSwitch":1},"version":"1.0.0"}' (Length: 100)
_demo_message_arrive|192 :: ----
```

View published messages in the cloud

In the operating status of the device details, you can view the message content reported to the cloud by the device.



RGB-LED-A

所属分类: 灯　产品型号: -　节点类型: 设备　通讯方式: Wi-Fi

数据格式: Alink　创建时间: 2018-07-25 14:21:41　ProductKey: a1HETlEuvri 复制　ProductSecret: ****** 显示

产品描述: -

SDK 集成 根据SDK集成文档进行设备端的开发和调试。

SDK	相关资源
C SDK	下载SDK 集成文档
Node.js SDK	下载SDK 集成文档

设备列表 产品开发阶段允许添加最多50个测试设备，上线发布后将最多添加1000个设备。

DeviceName	状态	最后上线时间	操作
RGB-LED-DEV-1	● 在线	2018-07-25 14:39:48	查看 调试 删除 激活凭证 在线模拟

Figure 2.10: View device details

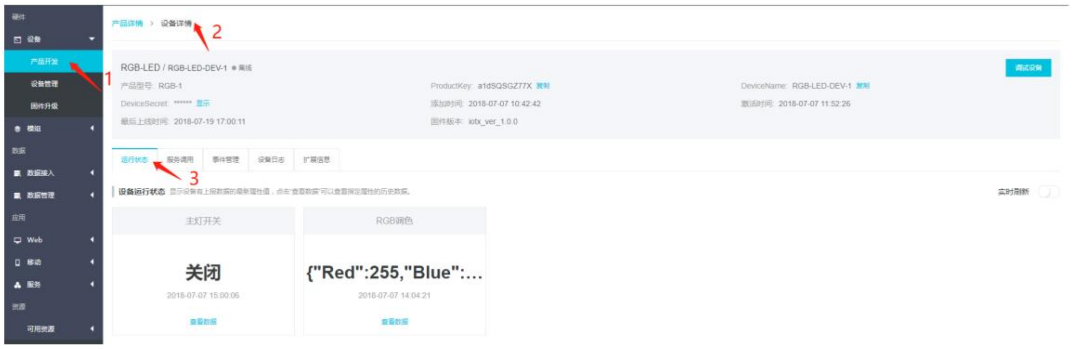


Figure 2.11: Check the device operating status

Push messages from the cloud to devices

Use the cloud-based debugging console to push messages to the device.

• Open the debug console

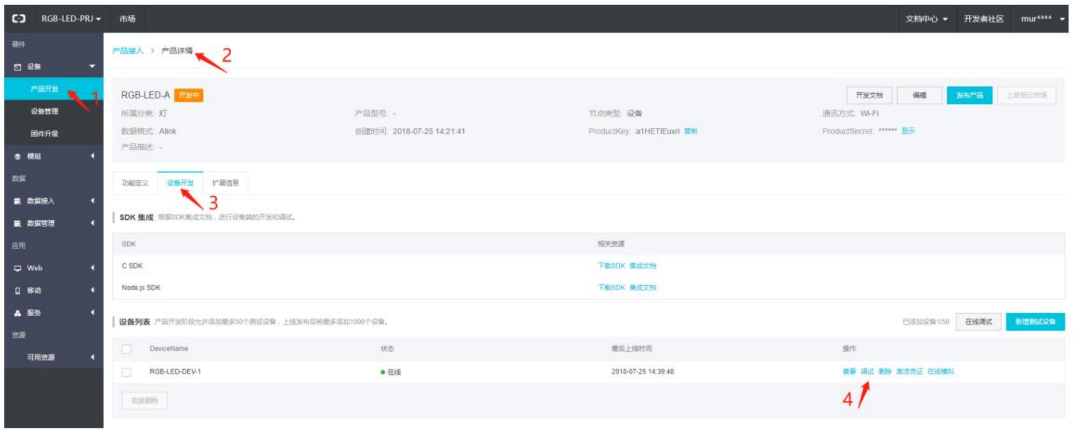
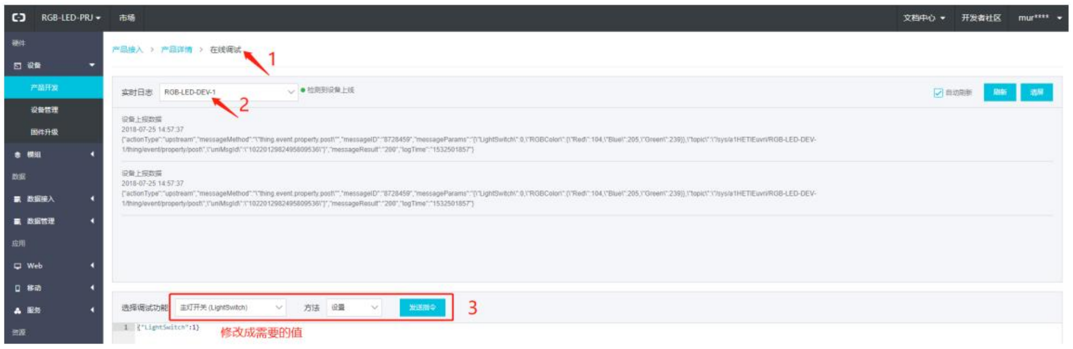


Figure 2.12: Open the debug console

• Send debug commands



```
设备上报数据
2018-07-25 14:57:37
{"actionType":"upstream","messageMethod":"\"thing.event.property.post\"","messageID":"8728459","messageParams":"{\"LightSwitch\":0,\"RGBColor\":{\"Red\":104,\"Blue\":205,\"Green\":239}},\"topic\":\"/sys/a1HETIEuvri/RGB-LED-DEV-1/thing/event/property/post\",\"uniMsgId\":\"1022012982495809536\"}","messageResult":"200","logTime":"1532501857"}

设备上报数据
2018-07-25 14:57:37
{"actionType":"upstream","messageMethod":"\"thing.event.property.post\"","messageID":"8728459","messageParams":"{\"LightSwitch\":0,\"RGBColor\":{\"Red\":104,\"Blue\":205,\"Green\":239}},\"topic\":\"/sys/a1HETIEuvri/RGB-LED-DEV-1/thing/event/property/post\",\"uniMsgId\":\"1022012982495809536\"}","messageResult":"200","logTime":"1532501857"}
```

```
{"LightSwitch":1}
```

Figure 2.13: Online debugging page

View device subscription logs

After sending a command using the debug console, the device can receive the command and the log is as follows:

```
[dbg] iotx_mc_handle_recv_PUBLISH(1091): [dbg]                    Packet Ident : 00000000
iotx_mc_handle_recv_PUBLISH(1092): [dbg]                          Topic Length : 52
iotx_mc_handle_recv_PUBLISH(1096):                                  Topic Name : /sys/
        a1Ayv8xhoIl/RGB-DEV1/thing/service/property/set [dbg]
iotx_mc_handle_recv_PUBLISH(1099): [dbg]                     Payload Len/Room : 100 / 967
iotx_mc_handle_recv_PUBLISH(1100): [dbg]                        Receive Buflen : 1024
iotx_mc_handle_recv_PUBLISH(1111): delivering msg ... [dbg]
iotx_mc_deliver_message(866): topic be matched _demo_message_arrive|
178 :: ---- _demo_message_arrive|179 ::
packetId: 0 _demo_message_arrive|183 :: Topic: '/sys/
a1Ayv8xhoIl/RGB-DEV1/thing/
        service/property/set' (Length: 52)
_demo_message_arrive|187 :: Payload:
'{"method":"thing.service.property.set","id":"35974024","params":{"
        LightSwitch":0},"version":"1.0.0"}' (Length: 100) _demo_message_arrive|
188 :: ----
```

Exit **MQTT** Example

Use **the ali_mqtt_test stop** command to exit the MQTT example. The device log is as follows:

```
msh />ali_mqtt_test stop [inf]
iotx_mc_unsubscribe(1423): mqtt unsubscribe success,topic = /sys/ a1HETlEuvri/RGB-LED-DEV-1/
        thing/event/property/post_reply! [inf] iotx_mc_unsubscribe(1423): mqtt unsubscribe
success,topic = /sys/
        a1HETlEuvri/RGB-LED-DEV-1/thing/service/property/set!
event_handle|136 :: unsubscribe success, packet-id=0 event_handle|136 ::
unsubscribe success, packet-id=0 [dbg] iotx_mc_disconnect(2121): rc =
MQTTDisconnect() = 0 [inf] _network_ssl_disconnect(514): ssl_disconnect [inf]
iotx_mc_disconnect(2129): mqtt disconnect! [inf] iotx_mc_release(2175):
mqtt release! [err] LITE_dump_malloc_free_stats(594):
WITH_MEM_STATS = 0 mqtt_client|329 :: out of sample!
```

2.1.3 OTA Example

Firmware Upgrade supports remote over-the-air (OTA) firmware upgrades for devices, enabling remote maintenance, feature upgrades, and problem fixes. You can specify a new firmware for a specific product, verify the firmware, and initiate a batch upgrade after verification. You can also view the upgrade results in the firmware details.

Sample File

Sample program path	Verification Platform	illustrate
samples/ota/ ota_mqtt-example.c	LinkDevelop, LinkPlatform	Devices based on MQTT channels OTA routines

Command List

In the example, use the MSH command to start the OTA example. The command is as follows:

Order	illustrate
ali_ota_test start	Launch OTA Example
ali_ota_test stop	Manually exit OTA example

Running **the OTA** Example

Use the **ali_ota_test start** command to start the OTA routine, and then wait for the cloud to send OTA instructions.

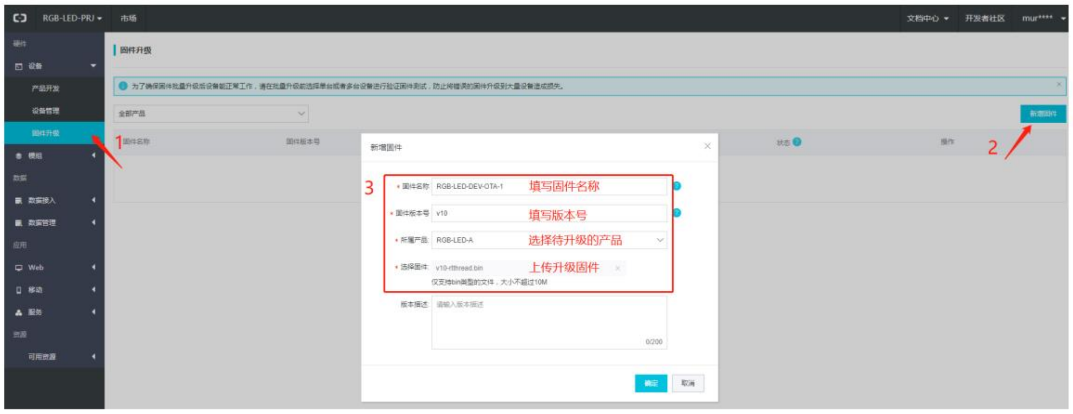
The device log is as follows:

```
msh />ali_ota_test start
ali_ota_main|372 :: iotkit-embedded sdk version: V2.10 [inf]
iotx_device_info_init(40): device_info created successfully! [dbg] iotx_device_info_set(50):
start to set device info! [dbg] iotx_device_info_set(64): device_info set
successfully!
...
[inf] iotx_mc_init(1703): MQTT init success! [inf]
_ssl_client_init(175): Loading the CA root certificate ...
...
[inf] _TLSConnectNetwork(420): . Verifying peer X.509 certificate.. [inf] _real_confirm(92):
certificate verification result: 0x200 [inf] iotx_mc_connect(2035): mqtt connect success!

...
[inf] iotx_mc_subscribe(1388): mqtt subscribe success,topic = /ota/device
        /upgrade/a1HETlEuvri/RGB-LED-DEV-1!
mqtt_client|241 :: wait ota upgrade command.... [dbg]
iotx_mc_cycle(1260): PUBACK event_handle|
130 :: publish success, packet-id=2 [dbg] iotx_mc_cycle(1269):
SUBACK event_handle|106 :: subscribe
success, packet-id=1 mqtt_client|241 :: wait ota upgrade command....
mqtt_client|241 :: wait ota upgrade command....
```

New firmware

Here, users are required to upload a test firmware of bin type. Any bin firmware can be used. The demo program only performs

The firmware is downloaded and verified, but not written to the Flash, so the firmware will not be actually moved or upgraded.


新增固件

* 固件名称: RGB-LED-DEV-OTA-1

* 固件板本号: v10

* 所属产品: RGB-LED-A

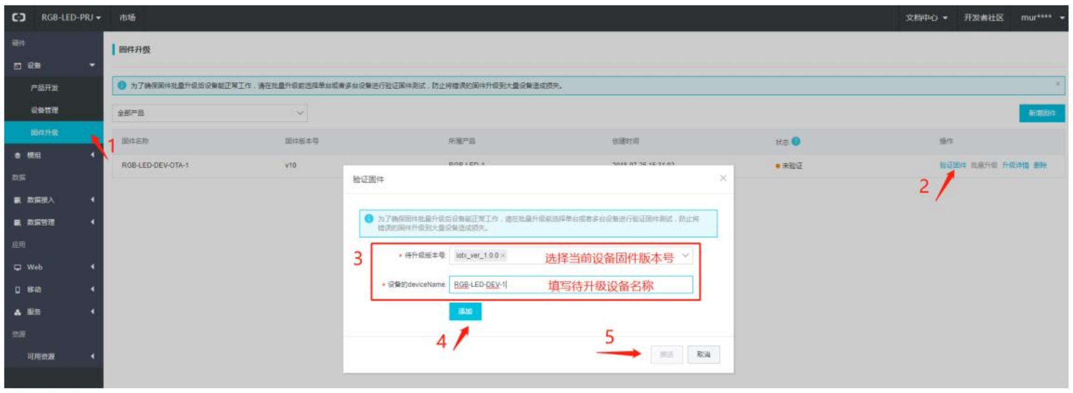
* 选择固件: v10-rtthread.bin

仅支持bin类型的文件，大小不超过10M

版本描述: 请输入版本描述 0/200

Figure 2.14: *LinkDevelop* New Platform *ORDER* firmware

Verify the firmware


验证固件

为了确保固件批量升级后设备能正常工作，请在批量升级前选择单台或者多台设备进行验证固件测试，防止将错误的固件升级到大量设备造成损失。

* 待升级板本号: iotx_ver_1.0.0

* 设备的deviceName: RGB-LED-DEV-1

Figure 2.15: *LinkDevelop* Platform Verification *ORDER* firmware

Device logs

After the push is successful, the device starts downloading the firmware. After the download is complete, the firmware integrity check is automatically performed.

The log is as follows:

```
...

mqtt_client|254 :: Here write OTA data to file....
 [dbg] IOT_OTA_Ioctl(457):
origin=e4e54df52a3b530c7e0544b2872f1305, now=
        e4e54df52a3b530c7e0544b2872f1305
mqtt_client|280 :: The firmware is valid! Download firmware successfully

mqtt_client|294 :: OTA FW version: v10
```

Cloud upgrade progress display

During the device upgrade process, the cloud will display the progress of the device downloading the firmware. When the firmware download is completed and the firmware verification is successful, the device

The SDK reports the new version number to the cloud, and the cloud will display a successful upgrade, as shown below:

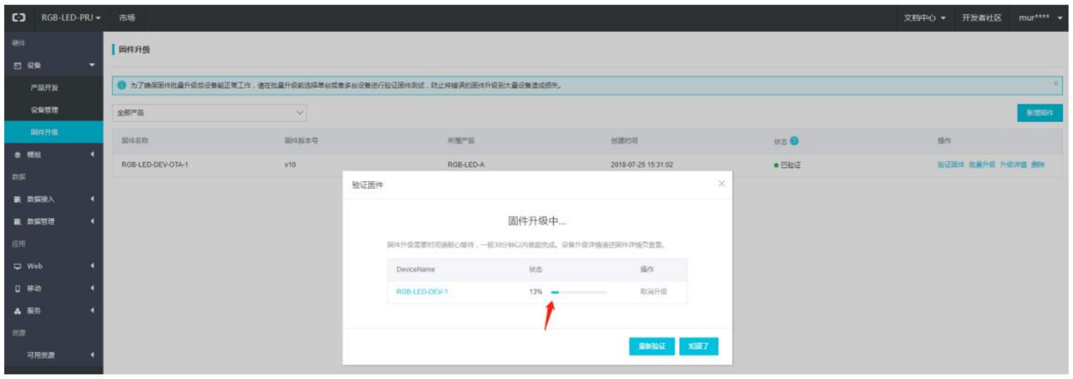


Figure 2.16: Upgrade progress

After the upgrade progress reaches 100%, run the **ali_ota_test start** command again to upload the latest version number to the cloud. After the version

numbers match successfully, the cloud displays the upgrade success, as shown below:

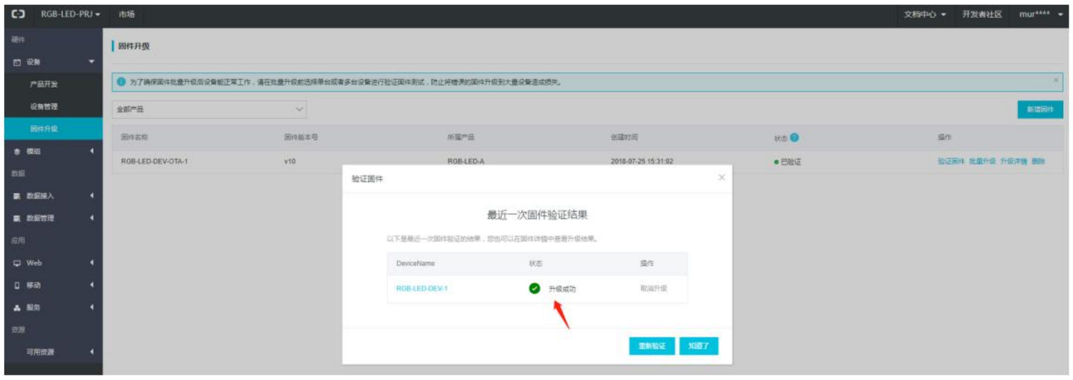


Figure 2.17: Upgrade successful

Exit **OTA** routine

The OTA routine will automatically exit if the upgrade is successful or failed. If you need to exit the OTA routine manually, use

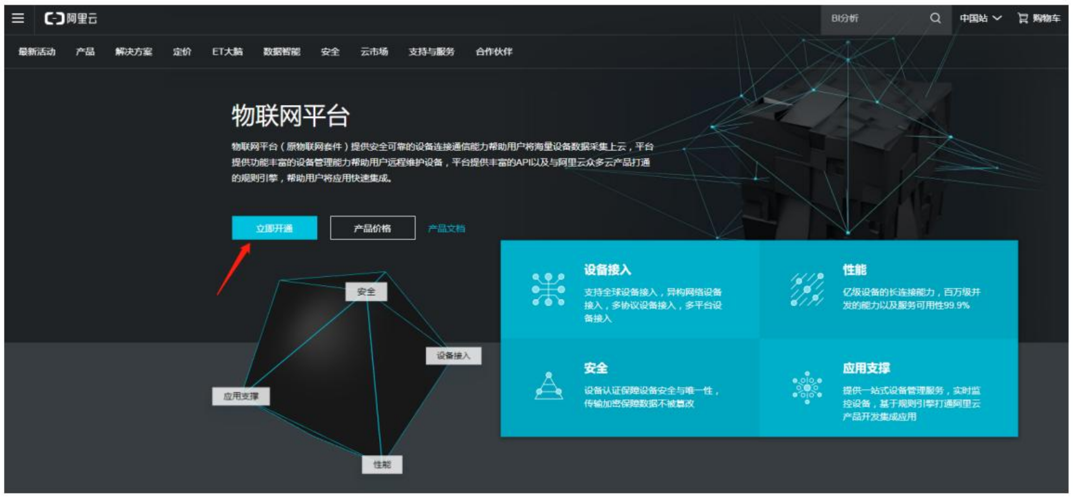
ali_ota_test stop command.

```
msh />ali_ota_test stop msh />[dbg]
iotx_mc_disconnect(2121): rc = MQTTDisconnect() = 0 [inf] _network_ssl_disconnect(514): ssl_disconnect [inf]
 iotx_mc_disconnect(2129): mqtt disconnect! [inf] iotx_mc_release(2175): mqtt release!
 [err] LITE_dump_malloc_free_stats(594): WITH_MEM_STATS = 0 mqtt_client|340 ::
 out of sample!
```

2.2 LinkPlatform

2.2.1 Preparation

• Register LinkPlatform



物联网平台（原物联网套件）提供安全可靠的设备连接通信能力帮助用户将海量设备数据采集上云，平台提供功能丰富的设备管理能力帮助用户远程维护设备，平台提供丰富的API以及与阿里云众多云产品打通的规则引擎，帮助用户将应用快速集成。

设备接入：支持全球设备接入，异构网络设备接入，多协议设备接入，多平台设备接入

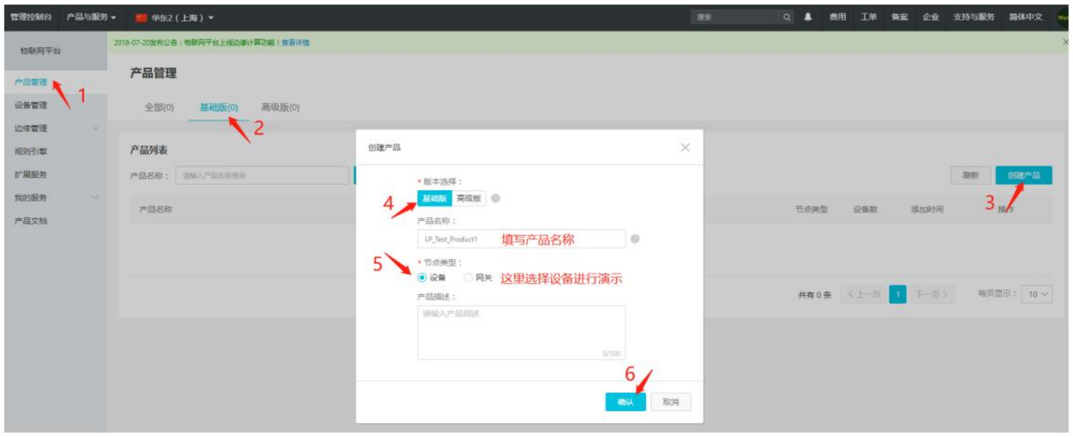
性能：亿级设备的长连接能力，百万级并发的能力以及服务可用性99.9%

安全：设备认证保障设备安全与唯一性，传输加密保障数据不被篡改

应用支撑：提供一站式设备管理服务，实时监控设备，基于规则引擎打通阿里云产品开发集成应用

Figure 2.18: register LinkPlatform platform

• Create products



创建产品

* 版本选择： [✓ 基础版] 高级版

产品名称： LP_Test_Product1 填写产品名称

* 节点类型： [✓ 设备] 网关 这里选择设备进行演示

产品描述： 请输入产品描述 0/100

Figure 2.19: Create a product

• Add a device

Under the Device Management menu, add a new test device and click View to enter the device details page.

After successfully creating the device, you can obtain the triplet required for device activation (ProductKey, DeviceName,

DeviceSecret), which needs to be configured in the device SDK using **menuconfig** later.

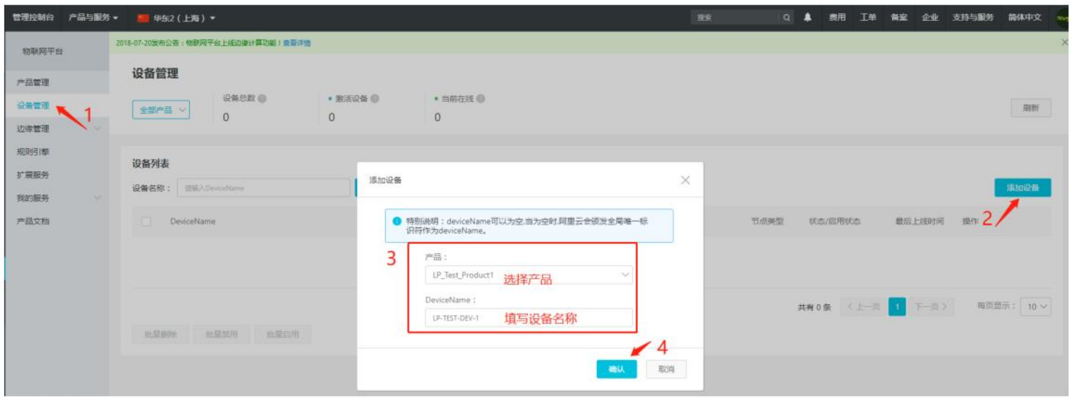


Figure 2.20: Add a device

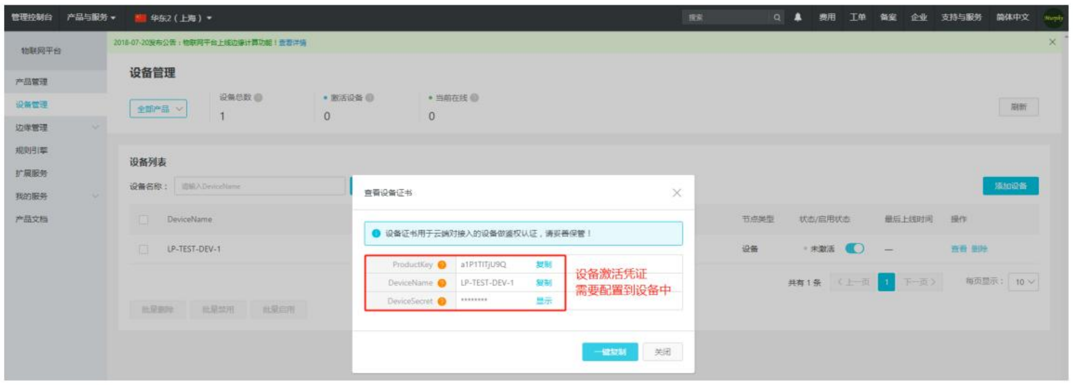

查看设备证书
设备证书用于云端对接入的设备做鉴权认证，请妥善保管！
ProductKey a1P1TITjU9Q 复制
DeviceName LP-TEST-DEV-1 复制
DeviceSecret ******** 显示
设备激活凭证
需要配置到设备中
一键复制
关闭

Figure 2.21: Device activation certificate

• View the message topic list

Go to the device details page, and then in **the Topic** list option, view the default topic list assigned to the created device.

and Topic permissions.

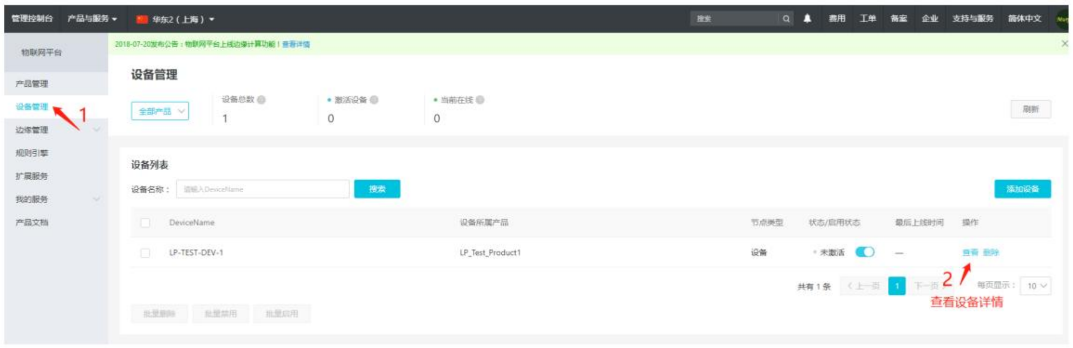
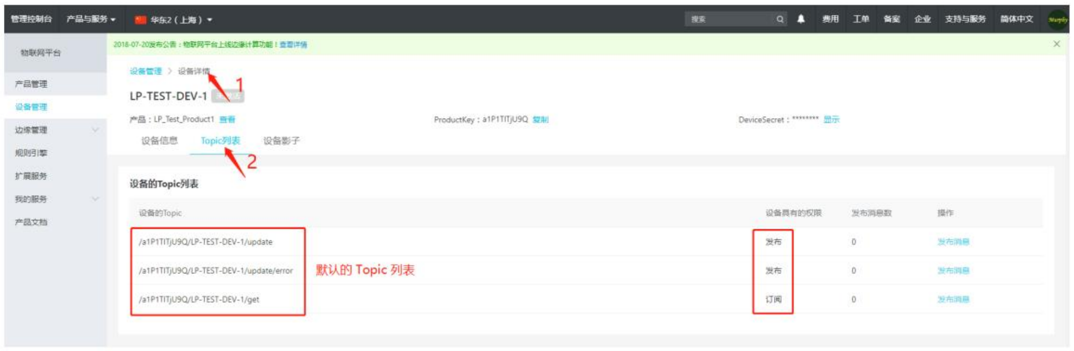


Figure 2.22: Go to the device details page



LP-TEST-DEV-1

产品：LP_Test_Product1 查看　ProductKey：a1P1TITjU9Q 复制　DeviceSecret：******** 显示

设备的Topic列表

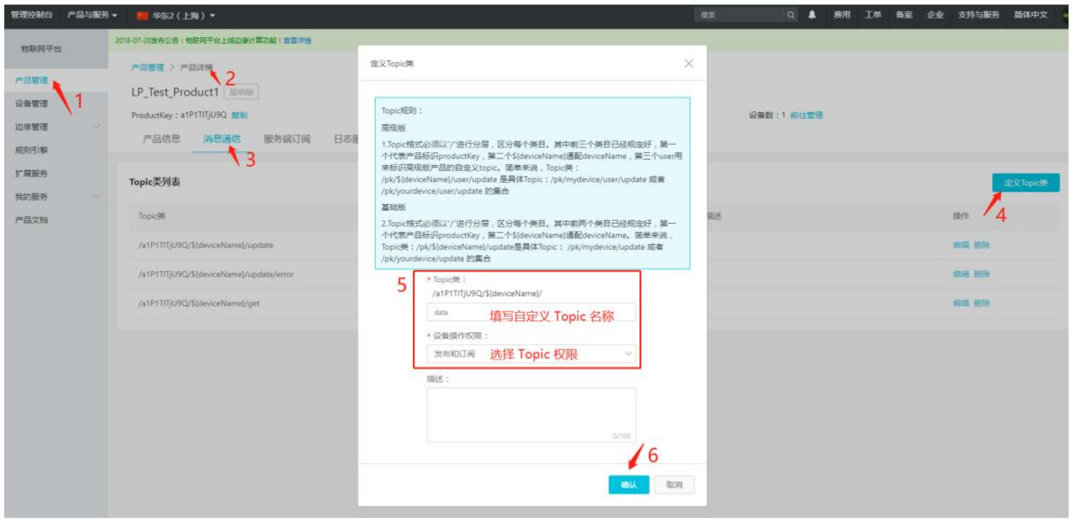
设备的Topic	设备具有的权限	发布消息数	操作
/a1P1TITjU9Q/LP-TEST-DEV-1/update	发布	0	发布消息
/a1P1TITjU9Q/LP-TEST-DEV-1/update/error	发布	0	发布消息
/a1P1TITjU9Q/LP-TEST-DEV-1/get	订阅	0	发布消息

Figure **2.23:** Check *MQTT Topic* List

• Custom **Topics**

The MQTT sample program uses a topic named **data .** The topic permissions are publish and subscribe, so it is necessary to

You need to customize a **data** topic, as shown below:



LP_Test_Product1 基础版

ProductKey：a1P1TITjU9Q 复制　设备数：1 前往管理

Topic类列表

Topic类	描述	操作
/a1P1TITjU9Q/${deviceName}/update		编辑 删除
/a1P1TITjU9Q/${deviceName}/update/error		编辑 删除
/a1P1TITjU9Q/${deviceName}/get		编辑 删除

定义Topic类

Topic规则：

高级版

1.Topic格式必须以"/"进行分层，区分每个类目。其中前三个类目已经规定好，第一个代表产品标识productKey，第二个${deviceName}通配deviceName，第三个user用来标识高级版产品的自定义topic。简单来说，Topic类：/pk/${deviceName}/user/update 是具体Topic：/pk/mydevice/user/update 或者 /pk/yourdevice/user/update 的集合

基础版

2.Topic格式必须以"/"进行分层，区分每个类目。其中前两个类目已经规定好，第一个代表产品标识productKey，第二个${deviceName}通配deviceName。简单来说，Topic类：/pk/${deviceName}/update是具体Topic：/pk/mydevice/update 或者 /pk/yourdevice/update 的集合

* Topic类：/a1P1TITjU9Q/${deviceName}/ data

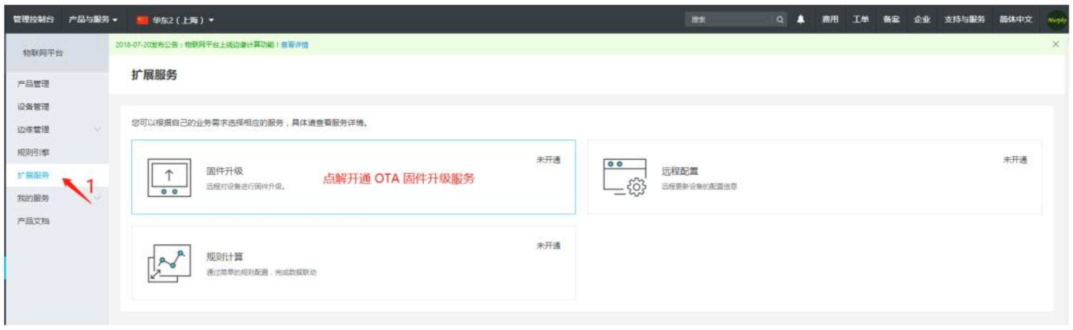
* 设备操作权限：发布和订阅

描述：

0/100

Figure **2.24:** Custom *Topic*

• Activate firmware upgrade service



扩展服务

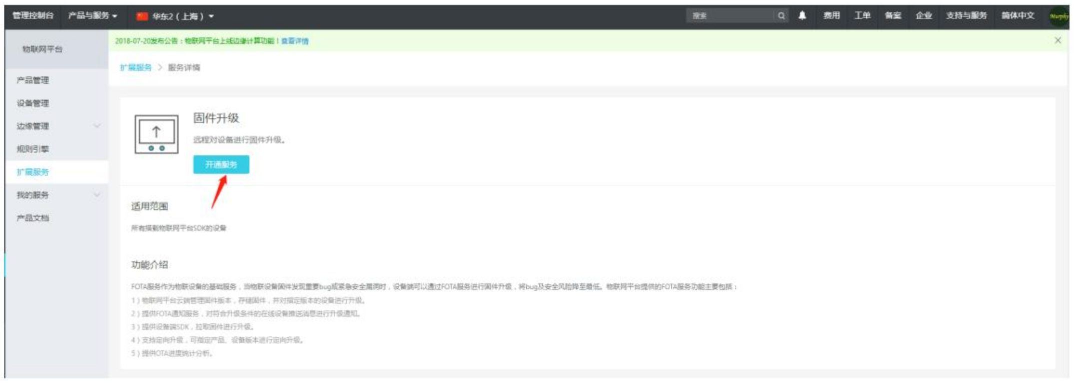
您可以根据自己的业务需求选择相应的服务，具体请查看服务详情。

固件升级　远程对设备进行固件升级。　未开通

远程配置　远程更新设备的配置信息　未开通

规则计算　通过简单的规则配置，完成数据联动　未开通

Figure **2.25:** Activate firmware upgrade service



2018-07-20发布公告：物联网平台上线边缘计算功能！查看详情

扩展服务 > 服务详情

固件升级

远程对设备进行固件升级。

适用范围

所有搭载物联网平台SDK的设备

功能介绍

FOTA服务作为物联设备的基础服务，当物联设备固件发现重要bug或紧急安全漏洞时，设备端可以通过FOTA服务进行固件升级，将bug及安全风险降至最低。物联网平台提供的FOTA服务功能主要包括：

1）物联网平台云端管理固件版本，存储固件，并对指定版本的设备进行升级。

2）提供FOTA通知服务，对符合升级条件的在线设备推送消息进行升级通知。

3）提供设备端SDK，拉取固件进行升级。

4）支持定向升级，可指定产品、设备版本进行定向升级。

5）提供OTA进度统计分析。

Figure **2.26:** Activate firmware upgrade service

• Get the software package

Open the ENV tool provided by RT-Thread and use **menuconfig** to configure the software package.

–**Configure** iotkit software package

Enable the iotkit software package and fill in the device activation

credentials. In menuconfig , select Alibaba Cloud Platform as **LinkPlatform and** MQTT as the OTA channel **(using** MQTT as an

example). The detailed configuration is as follows:

```
RT-Thread online packages --->
    IoT - internet of things --->
        IoT Cloud --->
          [*] Ali-iotkit: Ali Cloud SDK for IoT platform --->
                Select Aliyun platform (LinkPlatform Platform) ---> (a1dSQSGZ77X) Config
          Product Key
          (RGB-LED-DEV-1) Config Device Name
          (Ghuiyd9nmGowdZzjPqFtxhm3WUHEblII) Config Device Secret
          -*- Enable MQTT [*]
                  Enable MQTT sample [*]
          [*]     Enable MQTT direct connect
          [       Enable SSL
            ] Enable COAP
          [*] Enable OTA
                      Select OTA channel (Use MQTT OTA channel)
                          --->
                    version (latest) --->
```

– Increase mbedTLS frame size

During Alibaba's TLS authentication process, the data packet is large, so the TLS frame size needs to be increased. It needs to be at

least **8K** in size during OTA .

Open the ENV tool provided by RT-Thread and use **menuconfig** to configure the **TLS** frame size.

```
RT-Thread online packages --->
    security packages --->
        -*- mbedtls:An open source, portable, easy to use,
                    readable and flexible SSL library --->
        (8192) Maxium fragment length in bytes
```

– Download the package using the pkgs --update command

2.2.2 MQTT Example

This MQTT sample program uses data **topic** as an example to demonstrate how to use MQTT + TLS/SSL on a device.

The channel establishes two-way communication with the Alibaba Cloud platform.

Sample File

Sample program path	Verification Platform	illustrate
samples/mqtt/ mqtt-example.c	LinkDevelop, LinkPlatform	Devices and cloud based on MQTT channels Two-way communication routines

Command List

In the example, use the MSH command to start the MQTT example. The command is as follows:

Order	illustrate
ali_mqtt_test start	Start the MQTT example
ali_mqtt_test pub open	Turn on the light and synchronize the light status to the cloud
ali_mqtt_test pub close	Turn off the lights and synchronize the light-off status to the cloud
ali_mqtt_test stop	Stopping the MQTT Example

Start **MQTT**

Run the **ali_mqtt_test start** command to start the MQTT example. If successful, the device log will show that the subscription is successful.

The device log is as follows:

```
msh />ali_mqtt_test start
ali_mqtt_main|645 :: iotkit-embedded sdk version: V2.10
[inf] iotx_device_info_init(40): device_info created successfully!
[dbg] iotx_device_info_set(50): start to set device info!
```

```
[dbg] iotx_device_info_set(64): device_info set successfully!
...

[inf] iotx_mc_init(1703): MQTT init success! [inf]
_ssl_client_init(175): Loading the CA root certificate ...
...

[inf] _TLSConnectNetwork(420): . Verifying peer X.509 certificate.. [inf] _real_confirm(92): certificate
verification result: 0x200 [inf] iotx_mc_connect(2035): mqtt connect success!

...

[inf] iotx_mc_subscribe(1388): mqtt subscribe success,topic = /
        a1P1TlTjU9Q/LP-TEST-DEV-1/data!
[dbg] iotx_mc_cycle(1269): SUBACK
event_handle|124 :: subscribe success, packet-id=0 [inf]
iotx_mc_keepalive_sub(2226): send MQTT ping... [inf] iotx_mc_cycle(1295):
receive ping response!
```

Device release message

Run **the ali_mqtt_test pub open** command to send the message **data** topic.

The device log is as follows:

```
msh />ali_mqtt_test pub open
ali_mqtt_test_pub|583 ::
  publish message:
  topic: /a1P1TlTjU9Q/LP-TEST-DEV-1/data payload:
  {"id" : "1","version":"1.0","params" : {"RGBColor" : {"Red"
          :247,"Green":60,"Blue":74},"LightSwitch" : 1},"method":"thing.event. property.post"}

  rc = 3
msh />[dbg] iotx_mc_cycle(1260): PUBACK event_handle|
148 :: publish success, packet-id=0 [dbg] iotx_mc_cycle(1260):
PUBACK event_handle|148 :: publish success,
packet-id=0 [dbg] iotx_mc_cycle(1277): PUBLISH

...

[dbg] iotx_mc_handle_recv_PUBLISH(1100): [dbg]                Receive Buflen : 1024
iotx_mc_handle_recv_PUBLISH(1111): delivering msg ... [dbg]
iotx_mc_deliver_message(866): topic be matched _demo_message_arrive|
182 :: ---- _demo_message_arrive|183 ::
packetId: 19324 _demo_message_arrive|187 :: Topic: '/
a1P1TlTjU9Q/LP-TEST-DEV-1/data' (
        Length: 31)
_demo_message_arrive|191 :: Payload: '{"id" :
 "1","version":"1.0","params" : {"RGBColor" : {"Red":247,"Green
        ":60,"Blue":74},"LightSwitch" : 1},
```

Machine Translated by Google

```
"method":"thing.event.property.post"}' (Length: 142)
_demo_message_arrive|192 :: ----
```

View cloud logs

In the operating status of the device details, you can view the message content reported to the cloud by the device.

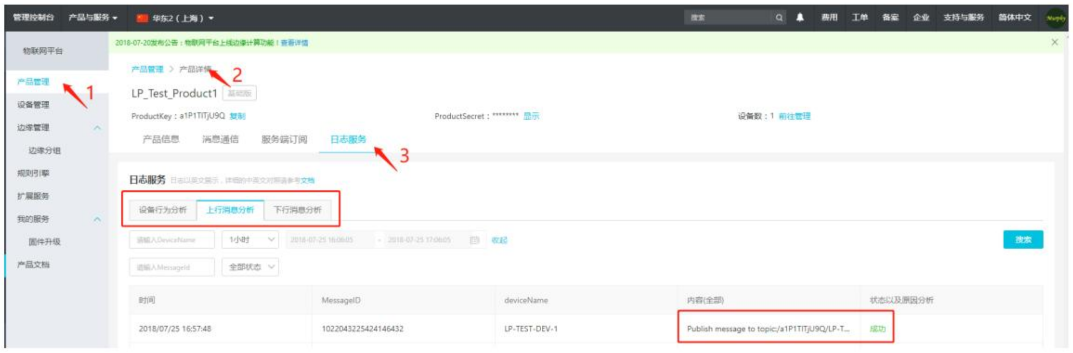


Figure 2.27: View cloud logs

Push messages from the cloud to devices

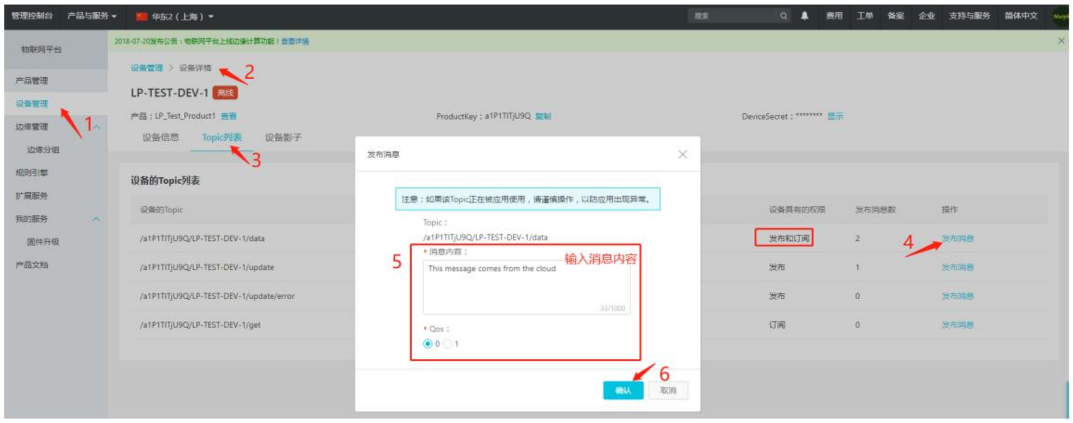


Figure 2.28: Publish messages from the cloud to devices

View device subscription logs

After sending a command using the debug console, the device can receive the command and the log is as follows:

```
msh />[dbg] iotx_mc_cycle(1277): PUBLISH
[dbg] iotx_mc_handle_recv_PUBLISH(1091): [dbg]          Packet Ident : 00000000
iotx_mc_handle_recv_PUBLISH(1092): [dbg]                Topic Length : 31
iotx_mc_handle_recv_PUBLISH(1096):                        Topic Name : /
    a1P1TITjU9Q/LP-TEST-DEV-1/data
[dbg] iotx_mc_handle_recv_PUBLISH(1099): [dbg]       Payload Len/Room : 33 / 989
iotx_mc_handle_recv_PUBLISH(1100):                      Receive Buflen : 1024
```

Machine Translated by Google

```
[dbg] iotx_mc_handle_recv_PUBLISH(1111): delivering msg ... [dbg]
 iotx_mc_deliver_message(866): topic be matched _demo_message_arrive|
182 :: ---- _demo_message_arrive|183 ::
packetId: 0 _demo_message_arrive|187 :: Topic: '/
a1P1TITjU9Q/LP-TEST-DEV-1/data' (
        Length: 31)
_demo_message_arrive|191 :: Payload: 'This message comes from the cloud'
        (Length: 33)
_demo_message_arrive|192 :: ----
```

Exit **MQTT** Example

Use **the ali_mqtt_test stop** command to exit the MQTT example. The device log is as follows:

```
msh />ali_mqtt_test stop msh /
>[inf] iotx_mc_unsubscribe(1423): mqtt unsubscribe success,topic = /
        a1P1TITjU9Q/LP-TEST-DEV-1/data!
event_handle|136 :: unsubscribe success, packet-id=0 [dbg]
 iotx_mc_disconnect(2121): rc = MQTTDisconnect() = 0 [inf]
 _network_ssl_disconnect(514): ssl_disconnect [inf]
 iotx_mc_disconnect(2129): mqtt disconnect! [inf]
 iotx_mc_release(2175): mqtt release! [err]
 LITE_dump_malloc_free_stats(594): WITH_MEM_STATS = 0 mqtt_client|329 ::
out of sample!
```

2.2.3 OTA Example

Firmware Upgrade supports remote over-the-air (OTA) firmware upgrades for devices, enabling remote maintenance, feature upgrades, and problem fixes. You can specify a new firmware for a specific product, verify the firmware, and initiate a batch upgrade after verification. You can also view the upgrade results in the firmware details.

Sample File

Sample program path	Verification Platform	illustrate
samples/ota/ ota_mqtt-example.c	LinkDevelop, LinkPlatform	Devices based on MQTT channels OTA routines

Command List

In the example, use the MSH command to start the OTA example. The command is as follows:

Order	illustrate
ali_ota_test start	Launch OTA Example
ali_ota_test stop	Manually exit OTA example

Running **the OTA** Example

Use the **ali_ota_test start** command to start the OTA routine, and then wait for the cloud to send OTA instructions.

The device log is as follows:

```
msh />ali_ota_test start
ali_ota_main|372 :: iotkit-embedded sdk version: V2.10 [inf]
iotx_device_info_init(40): device_info created successfully! [dbg] iotx_device_info_set(50):
start to set device info! [dbg] iotx_device_info_set(64): device_info set
successfully!
...

[inf] iotx_mc_init(1703): MQTT init success! [inf]
_ssl_client_init(175): Loading the CA root certificate ...
...

[inf] _TLSConnectNetwork(420): . Verifying peer X.509 certificate.. [inf] _real_confirm(92):
certificate verification result: 0x200 [inf] iotx_mc_connect(2035): mqtt connect success!

...

[dbg] iotx_mc_report_mid(2292): MID Report: topic name = '/sys/
      a1P1TlTjU9Q/LP-TEST-DEV-1/thing/status/update'
[dbg] iotx_mc_report_mid(2309): MID Report: finished, IOT_MQTT_Publish()
      = 0
[inf] iotx_mc_subscribe(1388): mqtt subscribe success,topic = /ota/device
      /upgrade/a1P1TlTjU9Q/LP-TEST-DEV-1!
mqtt_client|241 :: wait ota upgrade command.... mqtt_client|241 ::
wait ota upgrade command....
```

New firmware

Here, users are required to upload a test firmware of bin type. Any bin firmware can be used. The demo program only performs

The firmware is downloaded and verified, but not written to the Flash, so the firmware will not be actually moved or upgraded.

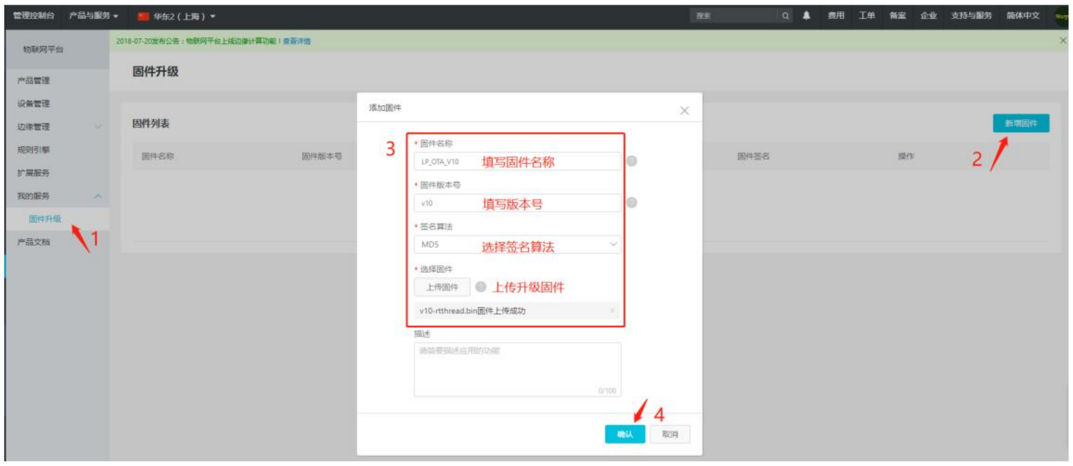


Figure 2.29: *LinkPlatform* New Platform *ORDER* firmware

Verify the firmware

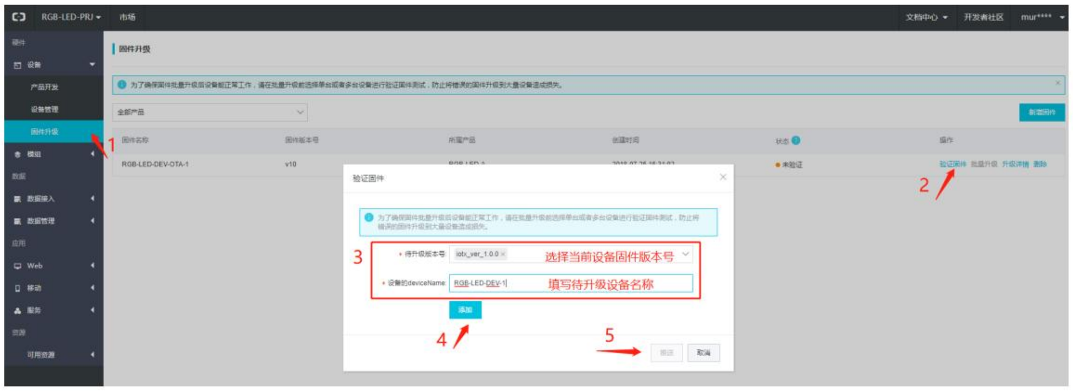


Figure 2.30: *LinkPlatform* verify *ORDER* firmware

Device logs

After the push is successful, the device starts downloading the firmware. After the download is complete, the firmware integrity check is automatically performed.

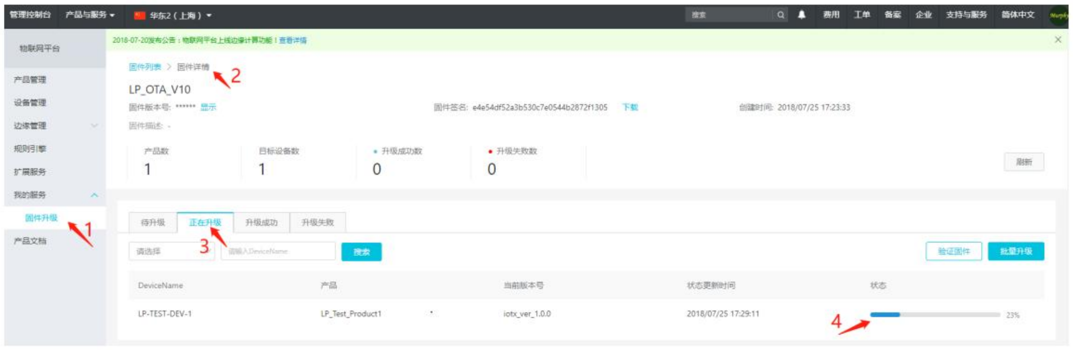
The log is as follows:

```
...

mqtt_client|254 :: Here write OTA data to file....
 [dbg] IOT_OTA_Ioctl(457):
origin=e4e54df52a3b530c7e0544b2872f1305, now=
        e4e54df52a3b530c7e0544b2872f1305
mqtt_client|280 :: The firmware is valid! Download firmware successfully

mqtt_client|294 :: OTA FW version: v10
```

Cloud upgrade progress display

During the device upgrade process, the cloud will display the progress of the device downloading the firmware. When the firmware download is completed and the firmware verification is successful, the device

The SDK reports the new version number to the cloud, and the cloud will display a successful upgrade, as shown below:



LP_OTA_V10

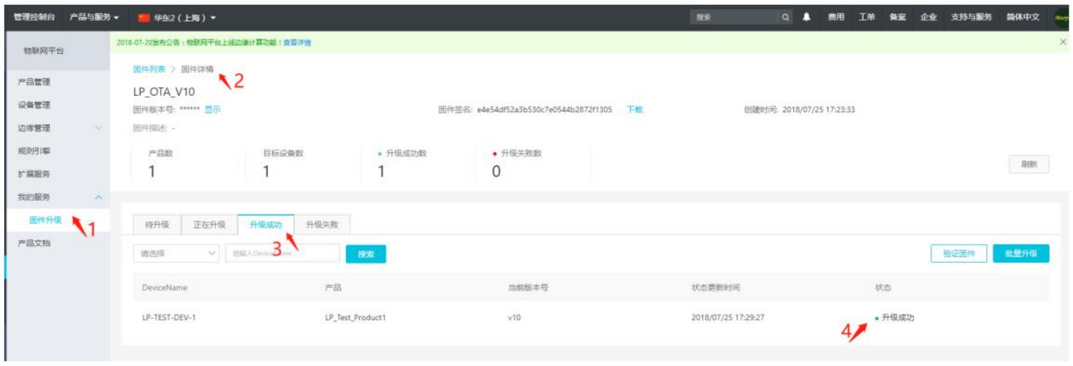
固件版本号: ****** 显示　固件签名: e4e54df52a3b530c7e0544b2872f1305 下载　创建时间: 2018/07/25 17:23:33

固件描述: -

产品数	目标设备数	升级成功数	升级失败数
1	1	0	0

DeviceName	产品	当前版本号	状态更新时间	状态
LP-TEST-DEV-1	LP_Test_Product1	iotx_ver_1.0.0	2018/07/25 17:29:11	23%

Figure 2.31: Upgrade progress



LP_OTA_V10

固件版本号: ****** 显示　固件签名: e4e54df52a3b530c7e0544b2872f1305 下载　创建时间: 2018/07/25 17:23:33

固件描述: -

产品数	目标设备数	升级成功数	升级失败数
1	1	1	0

DeviceName	产品	当前版本号	状态更新时间	状态
LP-TEST-DEV-1	LP_Test_Product1	v10	2018/07/25 17:29:27	升级成功

Figure 2.32: Upgrade successful

Exit **OTA** routine

The OTA routine will automatically exit if the upgrade is successful or failed. If you need to exit the OTA routine manually, use **ali_ota_test stop** command.

```
msh />ali_ota_test stop
msh />[dbg] iotx_mc_disconnect(2121): rc = MQTTDisconnect() = 0
 [inf] _network_ssl_disconnect(514): ssl_disconnect
 [inf] iotx_mc_disconnect(2129): mqtt disconnect!
 [inf] iotx_mc_release(2175): mqtt release!
 [err] LITE_dump_malloc_free_stats(594): WITH_MEM_STATS = 0
mqtt_client|340 :: out of sample!
```

2.3 Notes

• Please configure your device activation credentials (PRODUCT_KEY, DE-
VICE_NAME and DEVICE_SECRET)

• Use menuconfig **to select the platform to connect to (LinkDevelop** or **LinkPlatform)** • Encrypted connection must be enabled to enable OTA function, which is selected by default (because OTA upgrades must use **HTTPS** download)

Download Firmware)

2.4 Frequently Asked Questions

• MbedTLS returns 0x7200 error

This is usually because the MbedTLS frame length is too small. Please increase the MbedTLS frame length (it needs to be at least **8K).**

Chapter 3

How it works

In order to facilitate the connection between devices and the cloud, the iotkit SDK encapsulates a variety of connection protocols, such as MQTT, CoAP, HTTP, and TLS. The hardware platform is abstracted to make it more flexible without being restricted by specific hardware platforms.

Typically, users don't need to worry about the underlying SDK implementation mechanism. Instead, they only need to understand how devices interact with the cloud through the SDK. This helps them understand how to use the application-layer API to write business logic. Here's an example showing the data interaction process between MQTT and OTA applications.

3.1 MQTT data interaction process

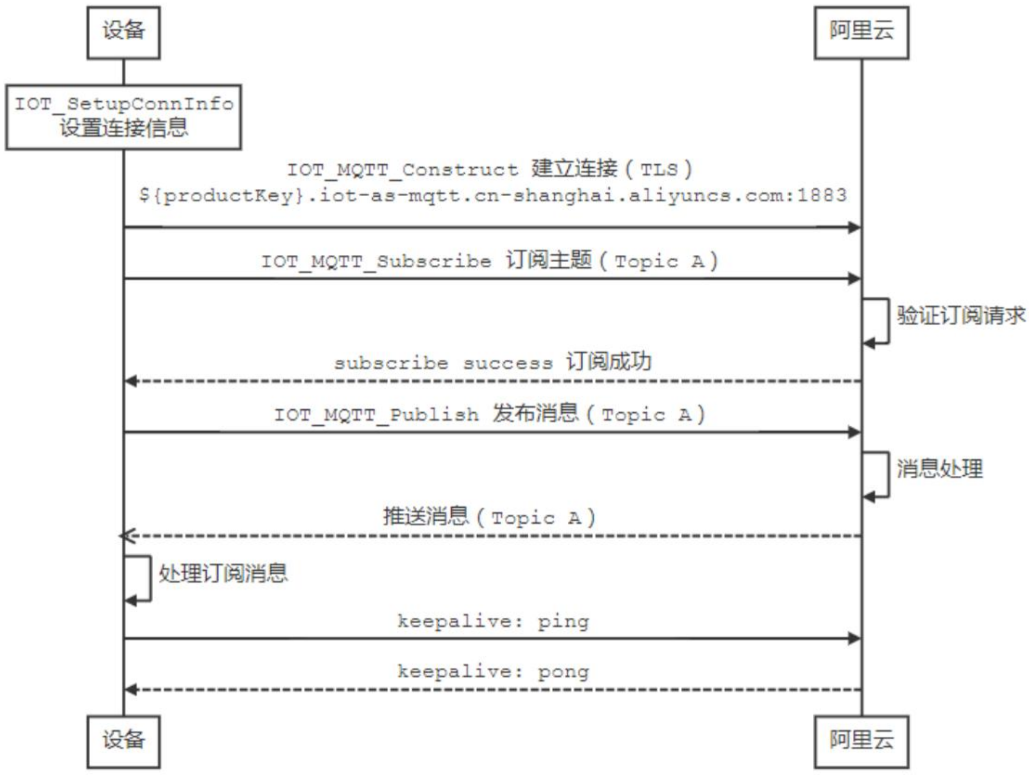

设备
阿里云
IOT_SetupConnInfo 设置连接信息
IOT_MQTT_Construct 建立连接(TLS)
${productKey}.iot-as-mqtt.cn-shanghai.aliyuncs.com:1883
IOT_MQTT_Subscribe 订阅主题(Topic A)
验证订阅请求
subscribe success 订阅成功
IOT_MQTT_Publish 发布消息(Topic A)
消息处理
推送消息(Topic A)
处理订阅消息
keepalive: ping
keepalive: pong
设备
阿里云

Figure 3.1: *MQTT* Data interaction process

3.2 OTA Data Interaction Process

Taking the MQTT channel as an example, the firmware upgrade process is shown in the figure below:

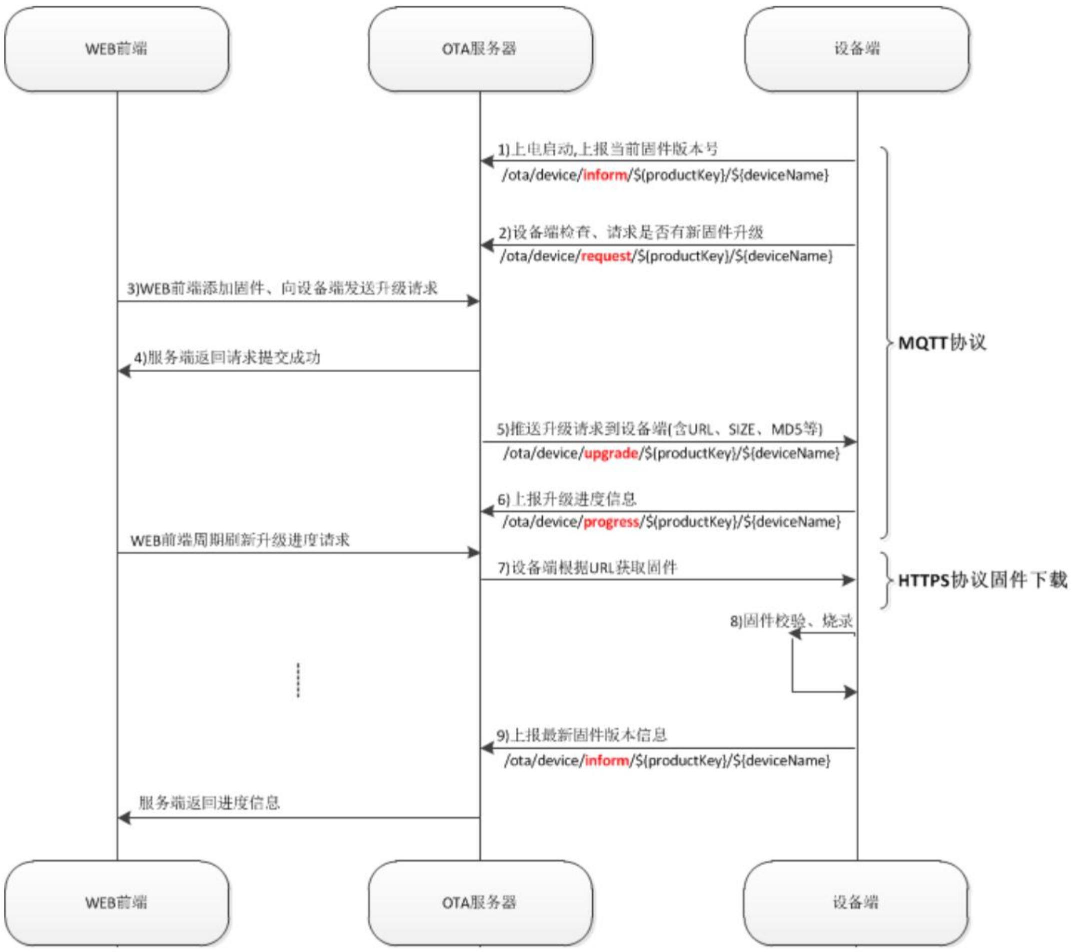

WEB前端
OTA服务器
设备端
1)上电启动,上报当前固件版本号
/ota/device/inform/${productKey}/${deviceName}
2)设备端检查、请求是否有新固件升级
/ota/device/request/${productKey}/${deviceName}
3)WEB前端添加固件、向设备端发送升级请求
4)服务端返回请求提交成功
5)推送升级请求到设备端(含URL、SIZE、MD5等)
/ota/device/upgrade/${productKey}/${deviceName}
6)上报升级进度信息
/ota/device/progress/${productKey}/${deviceName}
WEB前端周期刷新升级进度请求
7)设备端根据URL获取固件
8)固件校验、烧录
9)上报最新固件版本信息
/ota/device/inform/${productKey}/${deviceName}
服务端返回进度信息
MQTT协议
HTTPS协议固件下载
WEB前端
OTA服务器
设备端

Figure **3.2:** *OTA* Firmware Upgrade Process

Chapter 4

Usage Guidelines

The ali-iotkit software package encapsulates application layer protocols such as HTTP, MQTT, CoAP and OTA, making it convenient for users

The device is connected to the cloud platform. Here is a brief introduction of some parts.

4.1 MQTT Connection

Currently, Alibaba Cloud supports MQTT standard protocol access and is compatible with protocol versions 3.1.1 and 3.1. For specific protocols, please refer to

MQTT 3.1.1 and MQTT 3.1 Protocol documentation.

4.1.1 Features

• Support MQTT PUB, SUB, PING, PONG, CONNECT, DISCONNECT and UNSUB

• Support

cleanSession • Do not support

will, retain msg • Do not support QOS2 •

Support RRPC

synchronization mode based on native MQTT topic, the server can synchronously call the device and obtain

Device receipt results

4.1.2 Security Level

Supports TLSV1, TLSV1.1, and TLSV1.2 protocols to establish secure connections.

• TCP channel basic + chip-level encryption (ID2 hardware integration): high security level • TCP

channel basic + symmetric encryption (using the device private key for symmetric encryption): medium security level

• TCP method (data is not encrypted): low security level

4.1.3 Connecting Domain Name

• East China 2 node: productKey.iot-as-mqtt.cn-shanghai.aliyuncs.com:1883 • US West Coast node: productKey.iot-as-mqtt.us-west-1.aliyuncs.com:1883 • Singapore node: productKey.iot-as-mqtt.ap-southeast-1.aliyuncs.com:1883

4.1.4 Topic Specifications

By default, after a product is created, all devices under the product have the following Topic permissions:

• /***productKey/ deviceName/ update*** pub • /***productKey/ deviceName/ update/ error*** pub • /***productKey/ deviceName/ get*** sub • /***sys/ productKey/ deviceName/ thing/ #*** pub&sub • /***sys/ productKey/ deviceName/ rrpc/ #*** pub&sub • /***broadcast/ productKey/ #*** pub&sub

Each Topic rule is called a topic class, and the topic class implements device dimension isolation. When each device sends a message, Replace deviceName with the deviceName of your own device to prevent the topic from being overridden across devices. The topic description is as follows:

• pub: indicates the permission to report data to the topic • sub: indicates the permission to subscribe to the topic

• /***productKey/ deviceName/ xxx*** type topic class: can be expanded in the IoT platform console and Custom

• Topic categories starting with /sys: These are application protocol communication standards agreed upon by the system and are not user-defined. The specified topic must comply with Alibaba Cloud ALink data standards

• /***sys/ productKey/ deviceName/ thing/ xxx*** type topic class: used by gateway master and slave devices Topic class, used in gateway scenarios

• Topic types starting with /broadcast: Broadcast-specific topics • /sys/productKey/deviceName/rrpc/request/${messageId}: For synchronous requests, server The topic will be dynamically generated for the message ID, and the device can subscribe to the wildcard

• /***sys/ productKey/ deviceName/ rrpc/ request/ +: After receiving the message, send*** a pub message to / sys/productKey/deviceName/rrpc/response/${messageId}. The server can receive the result synchronously when sending the request.

4.1.5 Establishing an MQTT connection

Use the IOT_MQTT_Construct interface to establish an MQTT connection with the cloud.

If you want to keep the device online for a long time, you need to remove IOT_MQTT_Unregister and In the IOT_MQTT_Destroy part, use the while statement to maintain a long connection state.

The sample code is as follows:

```
while(1) {

        IOT_MQTT_Yield(pclient, 200);
        HAL_SleepMs(100);
}
```

4.1.6 Subscribing to a Topic

Use the IOT_MQTT_Subscribe interface to subscribe to a topic.

The code is as follows:

```
/* Subscribe the specific topic */ rc =
IOT_MQTT_Subscribe(pclient, TOPIC_DATA, IOTX_MQTT_QOS1, _demo_message_arrive,
                                                  NULL);
if (rc < 0) {
        IOT_MQTT_Destroy(&pclient);
        EXAMPLE_TRACE("IOT_MQTT_Subscribe() failed, rc = %d", rc); rc = -1; goto do_exit;


}
```

4.1.7 Publishing Messages

Use the IOT_MQTT_Publish interface to publish information to the cloud.

The code is as follows:

```
/* Initialize topic information */ memset(&topic_msg,
0x0, sizeof(iotx_mqtt_topic_info_t)); strcpy(msg_pub, "message: hello! start!");

topic_msg.qos = IOTX_MQTT_QOS1;
topic_msg.retain = 0;
topic_msg.dup = 0;
topic_msg.payload = (void *)msg_pub;
topic_msg.payload_len = strlen(msg_pub); rc =
IOT_MQTT_Publish(pclient, TOPIC_DATA, &topic_msg); EXAMPLE_TRACE("rc =
IOT_MQTT_Publish() = %d", rc);
```

4.1.8 Cancelling a Subscription

Use the IOT_MQTT_Unsubscribe interface to unsubscribe from cloud messages

4.1.9 Downlink Data Reception

Use the IOT_MQTT_Yield data receiving function to receive messages from the cloud.

Please call this API wherever you need to receive data. If the system allows, please start a separate thread to execute this API.

The code is as follows:

```
/* handle the MQTT packet received from TCP or SSL connection */
IOT_MQTT_Yield(pclient, 200);
```

4.1.10 Destroy MQTT connection

Use the IOT_MQTT_Destroy interface to destroy the MQTT connection and release memory.

The code is as follows:

```
IOT_MQTT_Destroy(&pclient);
```

4.1.11 Checking the connection status

Use the IOT_MQTT_CheckStateNormal interface to check the current connection status.

This interface is used to query the MQTT connection status. However, this interface cannot immediately detect device disconnection. It can only detect disconnection when there is data being sent or keepalive.

4.1.12 MQTT Keep-Alive

The device needs to send a message at least once within the keepalive_interval_ms interval, including a ping request.

If the server fails to receive any messages within the keepalive_interval_ms time, the IoT platform will disconnect and the device will need to reconnect.

In the IOT_MQTT_Construct function, you can set the value of keepalive_interval_ms.
Use this value as the heartbeat interval. The value range of keepalive_interval_ms is 60000~300000.

Sample code:

```
iotx_mqtt_param_t mqtt_params;

memset(&mqtt_params, 0x0, sizeof(mqtt_params));
mqtt_params.keepalive_interval_ms = 60000;
mqtt_params.request_timeout_ms = 2000;

/* Construct a MQTT client with specify parameter */ pclient =
IOT_MQTT_Construct(&mqtt_params);
```

4.2 CoAP Connection

- Support RFC 7252 Constrained Application Protocol. For details, please refer to: RFC 7252
- Use DTLS v1.2 to ensure channel security. For details, please refer to: DTLS v1.2
- Server address endpoint = productKey.iot-as-coap.cn-shanghai.aliyuncs.com:5684

  Please replace productKey with the product key you applied for.

4.2.1 CoAP Conventions

• Not supported? Pass parameters in
the form of numbers • Resource
discovery is not supported yet • Only UDP protocol is supported, and
currently it must be through DTLS • URI specification, CoAP URI resources are consistent with MQTT TOPIC, refer to MQTT specification

4.2.2 Application Scenarios

The CoAP protocol is suitable for low-power devices with limited resources, especially NB-IoT devices.

The process of connecting NB-IoT devices to the IoT platform is shown in the following figure:

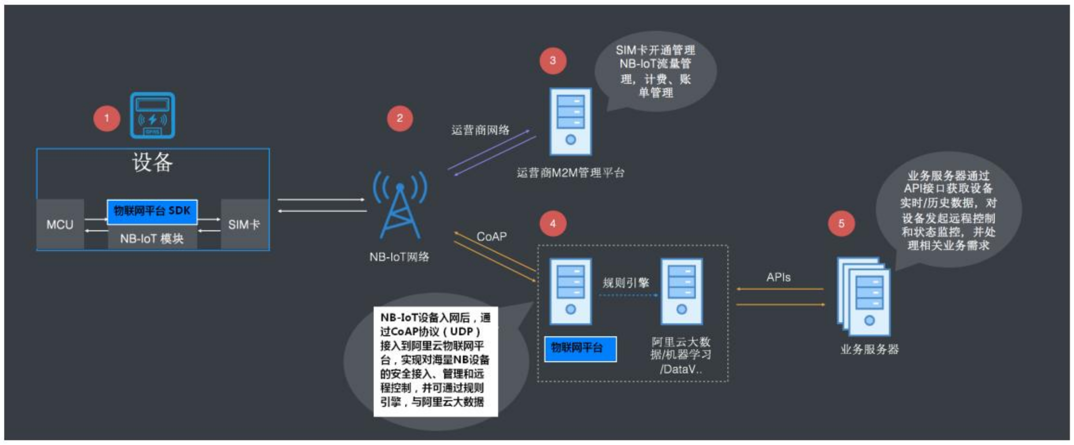

1
设备
MCU
物联网平台 SDK
NB-IoT 模块
SIM卡
2
NB-IoT网络
运营商网络
3
运营商M2M管理平台
SIM卡开通管理 NB-IoT流量管理，计费、账单管理
CoAP
4
规则引擎
物联网平台
阿里云大数据/机器学习 /DataV..
NB-IoT设备入网后，通过CoAP协议（UDP）接入到阿里云物联网平台，实现对海量NB设备的安全接入、管理和远程控制，并可通过规则引擎，与阿里云大数据
APIs
5
业务服务器
业务服务器通过API接口获取设备实时/历史数据，对设备发起远程控制和状态监控，并处理相关业务需求

Figure 4.1: *CoAP* Application Scenario

Machine Translated by Google

4.2.3 Establishing a Connection

Use the IOT_CoAP_Init and IOT_CoAP_DeviceNameAuth interfaces to establish a CoAP authentication connection with the cloud.

Sample code:

```
iotx_coap_context_t *p_ctx = NULL; p_ctx =
IOT_CoAP_Init(&config); if (NULL != p_ctx)

{ IOT_CoAP_DeviceNameAuth(p_ctx); do { count ++;
      if
         (count == 11) {

                    count = 1;
               }
      IOT_CoAP_Yield(p_ctx); } while
      (m_coap_client_running);
      IOT_CoAP_Deinit(&p_ctx); } else {

      HAL_Printf("IoTx CoAP init failed\r\n");
}
```

4.2.4 Sending and Receiving Data

SDK uses the interface IOT_CoAP_SendMessage to send data and IOT_CoAP_GetMessagePayload and IOT_CoAP_GetMessageCode to receive data.

Sample code:

```
/* send data */
static void iotx_post_data_to_server(void *param) {

      char path[IOTX_URI_MAX_LEN + 1] = {0}; iotx_message_t
      message; iotx_deviceinfo_t devinfo;
      message.p_payload = (unsigned char
      *)"{\"name\":\"hello world\"}"; message.payload_len = strlen("{\"name\":\"hello world\"}");
      message.resp_callback = iotx_response_handler;

      message.msg_type = IOTX_MESSAGE_CON;
      message.content_type = IOTX_CONTENT_TYPE_JSON;
      iotx_coap_context_t *p_ctx = (iotx_coap_context_t *)param; iotx_set_devinfo(&devinfo);
```

```
        snprintf(path, IOTX_URI_MAX_LEN, "/topic/%s/%s/update/",
                        (char *)devinfo.product_key, (char
                        *)devinfo.device_name);
        IOT_CoAP_SendMessage(p_ctx, path, &message);
}

/* receive data */
static void iotx_response_handler(void *arg, void *p_response) {

        int len = 0;
        unsigned char *p_payload = NULL;
        iotx_coap_resp_code_t resp_code;
        IOT_CoAP_GetMessageCode(p_response, &resp_code);
        IOT_CoAP_GetMessagePayload(p_response, &p_payload, &len);
        HAL_Printf("[APPL]: Message response code: 0x%x\r\n", resp_code);
        HAL_Printf("[APPL]: Len: %d, Payload: %s, \r\n", len, p_payload);
}
```

4.2.5 Downlink Data Reception

Use the IOT_CoAP_Yield interface to receive downlink data from the cloud.

Please call this API wherever you need to receive data. If the system allows, please start a separate thread to execute this interface.

4.2.6 Destroying a CoAP Connection

Use the IOT_CoAP_Deinit interface to destroy the CoAP connection and release the memory.

4.3 OTA Upgrade

4.3.1 Firmware Upgrade Topic

• The device reports the firmware version to the cloud

/ ota/ device/ inform/ productKey/ deviceName

• The device subscribes to this topic to receive cloud firmware upgrade notifications

/ ota/ device/ upgrade/ productKey/ deviceName

• Device reports firmware upgrade progress

/ota/device/progress/productKey/deviceName

• The device requests a firmware upgrade

/ota/device/request/productKey/deviceName

4.3.2 Firmware Upgrade Instructions

• The device firmware version number only needs to be reported once during system startup, and does not need to be reported periodically.

• Determine whether the device OTA upgrade is successful based on the version number

• Initiate a batch upgrade from the OTA server console, and the device upgrade operation record status is Pending Upgrade

The actual upgrade begins when the OTA system receives the upgrade progress reported by the device, and the device upgrade operation record status is Upgrading.

• When the device is offline, it cannot receive upgrade messages pushed by the server

When the device goes online, it actively notifies the server of the online message. The OTA server receives the device online message and verifies whether the device needs to be upgraded. If so, it pushes the upgrade message to the device again. Otherwise, no message is pushed.

4.3.3 OTA Code Description

initialization

The initialization of the OTA module depends on the MQTT connection, that is, the MQTT client handle pclient must be obtained first.

```
h_ota = IOT_OTA_Init(PRODUCT_KEY, DEVICE_NAME, pclient); if (NULL == h_ota)
{ rc = -1; printf("initialize OTA
        failed\n");

}
```

Report version number

After the OTA module is initialized, the IOT_OTA_ReportVersion interface is called to report the current firmware version number. After the upgrade is successful, the new firmware is restarted and the new firmware version number is reported using this interface. After the version number of the cloud and the OTA upgrade task are successfully compared, the OTA upgrade is successful.

The sample code is as follows:

```
if (0 != IOT_OTA_ReportVersion(h_ota, "version2.0")) {
        rc = -1;
        printf("report OTA version failed\n");
}
```

Machine Translated by Google

Download firmware

After the MQTT channel obtains the URL for OTA firmware download, it uses HTTPS to download the firmware and stores it in the Flash OTA partition while downloading.

• IOT_OTA_IsFetching() interface: used to determine whether there is firmware to download • IOT_OTA_FetchYield() interface: used to download a firmware block • IOT_OTA_IsFetchFinish() interface: used to determine whether the download is complete

Sample code:

```
// Determine whether there is firmware available for download
if (IOT_OTA_IsFetching(h_ota)) {
        unsigned char buf_ota[OTA_BUF_LEN]; uint32_t len,
        size_downloaded, size_file; do { // Loop download firmware len =

        IOT_OTA_FetchYield(h_ota,
                buf_ota, OTA_BUF_LEN, 1); if (len > 0) { // Write to Flash or other storage (download and store
                at the same time)

                }
        } while (!IOT_OTA_IsFetchFinish(h_ota)); // Check if the firmware is downloaded
}
```

Report download status

Use the IOT_OTA_ReportProgress interface to report the firmware download progress.

```
if (percent - last_percent > 0) {
        IOT_OTA_ReportProgress(h_ota, percent, NULL);
}
IOT_MQTT_Yield(pclient, 100);
```

Determine whether the downloaded firmware is complete

After the firmware download is complete, use the IOT_OTA_Ioctl interface to verify the integrity of the firmware.

```
int32_t firmware_valid;
IOT_OTA_Ioctl(h_ota, IOT_OTAG_CHECK_FIRMWARE, &firmware_valid, 4); if (0 == firmware_valid)
{ printf("The firmware is invalid\n"); } else
        { printf("The firmware is valid\n");



}
```

Destroy **OTA** connection

Use IOT_OTA_Deinit to destroy the OTA connection and release the memory.

4.4 Reference

• The above content is quoted from the Alibaba Cloud IoT Platform documentation. For more information, please visit the Alibaba Cloud IoT Platform documentation.

Heart to check

• For more API usage instructions, please refer to the API usage documentation. •

For more sample codes, please refer to the sample programs and sample usage instructions.

Chapter **5**

API Description

ali-iotkit is a software package ported from RT-Thread for connecting to Alibaba Cloud IoT platform.

Provided **iotkit-embedded C-SDK.**

Here we quote the instructions for using Alibaba's **iotkit-embedded** API, which are as follows.

Note: The following API description information comes from Alibaba Cloud. For more details, please refer to the iotkit-embedded **wiki.**

5.1 Required **APIs**

Serial number	Function name	illustrate
1	IOT_OpenLog	Start printing log information (log), accept a const char * is the input parameter, indicating the module name
2	IOT_CloseLog	Stop printing log information (log), the input parameter is empty
3	IOT_SetLogLevel	Set the log level to print, accept input parameters from 1 to 5, the larger the number, the more detailed the printing
4	IOT_DumpMemoryStats	Debug function, print memory usage statistics The input parameter is 1-5, the larger the number, the more detailed

5.2 MQTT Function-Related **APIs**

Machine Translated by Google

Serial number	Function name	illustrate
1	IOT_SetupConnInfo	Preparation before MQTT connection, basic ÿDeviceName + DeviceSecret + ProductKey generates MQTT connection Username and password, etc.
2	IOT_SetupConnInfoSecure	Preparation before MQTT connection, based on ID2 + DeviceSecret + ProductKey generated Username and password for the MQTT connection etc., ID2 mode enabled
3	IOT_MQTT_CheckStateNormal	After MQTT is connected, call this function to check the long Is the connection normal?
4	IOT_MQTT_Construct	Constructor of MQTT instance, input parameters For the iotx_mqtt_param_t structure, connect MQTT server, and returns the created handle
5	IOT_MQTT_ConstructSecure	MQTT instance constructor, input parameters For the iotx_mqtt_param_t structure, connect MQTT server, and returns the created sentence handle, ID2 mode enabled
6	IOT_MQTT_Destroy	The destruction function of the MQTT instance, input parameter Statement created for IOT_MQTT_Construct() Pattern
7	IOT_MQTT_Publish	MQTT session stage, organize a complete MQTT Publish message to the server Send message release message
8	IOT_MQTT_Subscribe	MQTT session stage, organize a complete MQTT Subscribe message to the server Send a subscription request
9	IOT_MQTT_Unsubscribe	MQTT session stage, organize a complete MQTT UnSubscribe message to the service The client sends an unsubscribe request
10	IOT_MQTT_Yield	MQTT session phase, MQTT main loop Function, which contains the maintenance of heartbeat, server Receiving of messages, etc.

5.3 CoAP Function-Related APIs

Machine Translated by Google

Serial number	Function name	illustrate
1	IOT_CoAP_Init	Constructor of CoAP instance, input parameters For the iotx_coap_config_t structure, return Returns the created CoAP session handle
2	IOT_CoAP_Deinit	The destruction function of the CoAP instance, input parameter Handle created by IOT_CoAP_Init()
3	IOT_CoAP_DeviceNameAuth	Based on the DeviceName requested by the console, DeviceSecret, ProductKey for devices Certification
4	IOT_CoAP_GetMessageCode	CoAP session phase, from the server The CoAP Response message is obtained ÿRespond Code
5	IOT_CoAP_GetMessagePayload	CoAP session phase, from the server Get the message from the CoAP Response message load
6	IOT_CoAP_SendMessage	CoAP session phase, the connection has been successfully established After calling, organize a complete CoAP message Send the document to the server
7	IOT_CoAP_Yield	CoAP session phase, the connection has been successfully established After calling, checking and collecting server Response message to CoAP Request

5.4 HTTP Function-Related APIs

Serial number	Function name	illustrate
1	IOT_HTTP_Init	Https instance constructor, create a HTTP session handle and returns
2	IOT_HTTP_DeInit	Https instance destruction function, destroy all Related data structures
3	IOT_HTTP_DeviceNameAuth	is based on the DeviceName requested by the console. DeviceSecret, ProductKey for devices Certification
4	IOT_HTTP_SendMessage	Https session stage, organize a complete HTTP message is sent to the server and synchronized Get HTTP response message

Serial number	Function name	illustrate
5	IOT_HTTP_Disconnect	Https session phase, close the HTTP layer connection, but still maintain TLS level connect

5.5 OTA Function Related API

Serial number	Function name	illustrate
1	IOT_OTA_Init	The constructor of the OTA instance creates an The handle of the OTA session is returned
2	IOT_OTA_Deinit	The destruction function of the OTA instance destroys all related Related data structures
3	IOT_OTA_Ioctl	The input and output functions of the OTA instance are based on different The same command word can set the OTA session Properties, or get the status of the OTA session
4	IOT_OTA_GetLastError	During the OTA session, if The IOT_OTA_*() function returns an error, calling This interface can be used to obtain the most recent detailed error code
5	IOT_OTA_ReportVersion	During the OTA session, report the current status to the server. Firmware version number
6	IOT_OTA_FetchYield	During the OTA download phase, Download the firmware from the server within the timeout period. Download a piece of firmware content and save it in the input parameter In the buffer
7	IOT_OTA_IsFetchFinish	During the OTA download phase, determine the iterative adjustment Has IOT_OTA_FetchYield() been used? Download all firmware contents
8	IOT_OTA_IsFetching	During the OTA download phase, determine whether the firmware download is Still in progress, not all firmware has been completed Download
9	IOT_OTA_ReportProgress	Optional API, OTA download phase, call this The function reports to the server that all What percentage of firmware content
10	IOT_OTA_RequestImage	Optional API to request firmware download from the server
11	IOT_OTA_GetConfig	Optional API to request remote configuration from the server

5.6 Cloud Connection API

Serial number	Function name	illustrate
1	IOT_Cloud_Connection_Init	Constructor of cloud connection instance, input parameters ÿiotx_cloud_connection_param_pt Structure, returns the created cloud connection session Handle
2	IOT_Cloud_Connection_Deinit	Cloud connection instance destruction function, input parameters ÿIOT_Cloud_Connection_Init() The sentence created Pattern
3	IOT_Cloud_Connection_Send_Message	sends data to the cloud
4	IOT_Cloud_Connection_Yield	After the cloud connection is successfully established, the server receives Sent messages

5.7 CMP Function-Related APIs

Serial number	Function name	illustrate
1	IOT_CMP_Init	CMP instance constructor, input parameters It is the iotx_cmp_init_param_pt structure. There is only one CMP instance
2	IOT_CMP_Register	Subscription Services through CMP
3	IOT_CMP_Unregister	Cancelling a service subscription through CMP
4	IOT_CMP_Send	Send data through CMP and send it to the cloud can also be sent to local devices
5	IOT_CMP_Send_Sync	Send data synchronously via CMP, not supported yet hold
6	IOT_CMP_Yield	Receiving data through CMP, single thread situation Support
7	IOT_CMP_Deinit	The destroy function of the CMP example
8	IOT_CMP_OTA_Start	Initialize the ota function and report the version
9	IOT_CMP_OTA_Set_Callback	sets the OTA callback function
10	IOT_CMP_OTA_Get_Config	Get remote configuration
11	IOT_CMP_OTA_Request_Image	Get firmware

Serial number	Function name	illustrate
12	IOT_CMP_OTA_Yield	Complete OTA function through CMP

5.8 Device Shadow Related (Optional) API

Serial number	Function name	illustrate
1	IOT_Shadow_Construct	Establish an MQTT connection to a device shadow, And returns the created session handle
2	IOT_Shadow_Destroy	Destroy the MQTT connection of a device shadow, Destroy all related data structures and release the memory Save, Disconnect
3	IOT_Shadow_Pull	The JSON data cached on the server Pull down to local and update local data attributes
4	IOT_Shadow_Push	Push local data attributes to the server cache Stored JSON data, update the server data According to the attributes
5	IOT_Shadow_Push_Async	Similar to IOT_Shadow_Push() interface, but It is asynchronous and returns immediately after pushing, without waiting for the server to Server response
6	IOT_Shadow_PushFormat_Add	adds a format to the created data type format. Member Attributes
7	IOT_Shadow_PushFormat_Finalize	completes the construction process of a data type format
8	IOT_Shadow_PushFormat_Init	starts the construction process of a data type format
9	IOT_Shadow_RegisterAttribute	creates a data type and registers it to the server. When registering, you need to create the *PushFormat*() interface Created data type format
10	IOT_Shadow_DeleteAttribute	Deletes a successfully registered data attribute
11	IOT_Shadow_Yield	The main loop function of MQTT, which receives The server pushes down the message to update the local data According to the attributes

5.9 Master and slave device related (optional) API

Serial number	Function name	illustrate
1	IOT_Gateway_Construct	Create a master device and establish an MQTT connection Connect and return the created session handle
2	IOT_Gateway_Destroy	Destroy a master's MQTT connection, Destroy all related data structures and release the memory Save, Disconnect
3	IOT_Subdevice_Login	The sub-device is online, notifying the cloud to create the sub-device session
4	IOT_Subdevice_Logout	The sub-device is offline, the cloud is destroyed and the sub-device is created session and all related data structures, Memory
5	IOT_Gateway_Yield	MQTT's main loop function, called to receive Push message from the server
6	IOT_Gateway_Subscribe	Send subscription to the server via MQTT connection Read Request
7	IOT_Gateway_Unsubscribe	Send the query to the server via MQTT connection Unsubscribe request
8	IOT_Gateway_Publish	Send messages via MQTT connection server Publish message
9	IOT_Gateway_RRPC_Register	Registers the RRPC callback function of the device, Receive RRPC requests initiated by the cloud
10	IOT_Gateway_RRPC_Response	responds to the RRPC request from the cloud
11	IOT_Gateway_Generate_Message_ID	generates message id
12	IOT_Gateway_Get_TOPO	Send a packet to topo/get topic and wait for a response Reply (TOPIC_GET_REPLY reply)
13	IOT_Gateway_Get_Config	Send a packet to the conifg/get topic and wait Reply (TOPIC_CONFIG_REPLY reply)
14	IOT_Gateway_Publish_Found_List	reports the device list

5.10 LinkKit Function-Related APIs

Serial number	Function name	illustrate
1	links_start	Start the linkkit service and establish a connection with the cloud Connect and install the callback function

Machine Translated by Google

Serial number	Function name	illustrate
2	links_end	Stop the linkkit service and disconnect from the cloud Recycling
3	links_dispatch	Event dispatch function, triggering linkkit_start Installed callbacks
4	links_yield	Linkkit main loop function, including heartbeat Maintenance, receiving of downlink messages from the server, etc. If multithreading is enabled, do not call this function.
5	links_set_value	Set the TSL of the object according to the identifier Attribute, if the identifier is a struct type, event output type or service Output type, use dot '.' to separate fields; For example, "identifier1.identifier2" points to Specific items
6	links_get_value	Get the TSL of the object according to the identifier property
7	links_set_tsl	Read TSL files from local machine, generate Image and add it to linkkit
8	links_answer_service	Respond to cloud service requests
9	linkkit_invoke_raw_service	Sending raw data to the cloud
10	links_trigger_event	Report device events to the cloud
11	links_photo_init	Initialize OTA-fota service and install it back Call function (need to compile and set macro SERVICE_OTA_ENABLED)
12	links_invoke_fota_service	Execute fota service
13	links_photo_init	Initialize OTA-cota service and install it back Call function (need to compile and set macro SERVICE_OTA_ENABLED SERVICE_COTA_ENABLED)
14	linkkit_invoke_cota_get_config Device requests remote configuration	
15	links_invoke_cota_service	Execute cota service