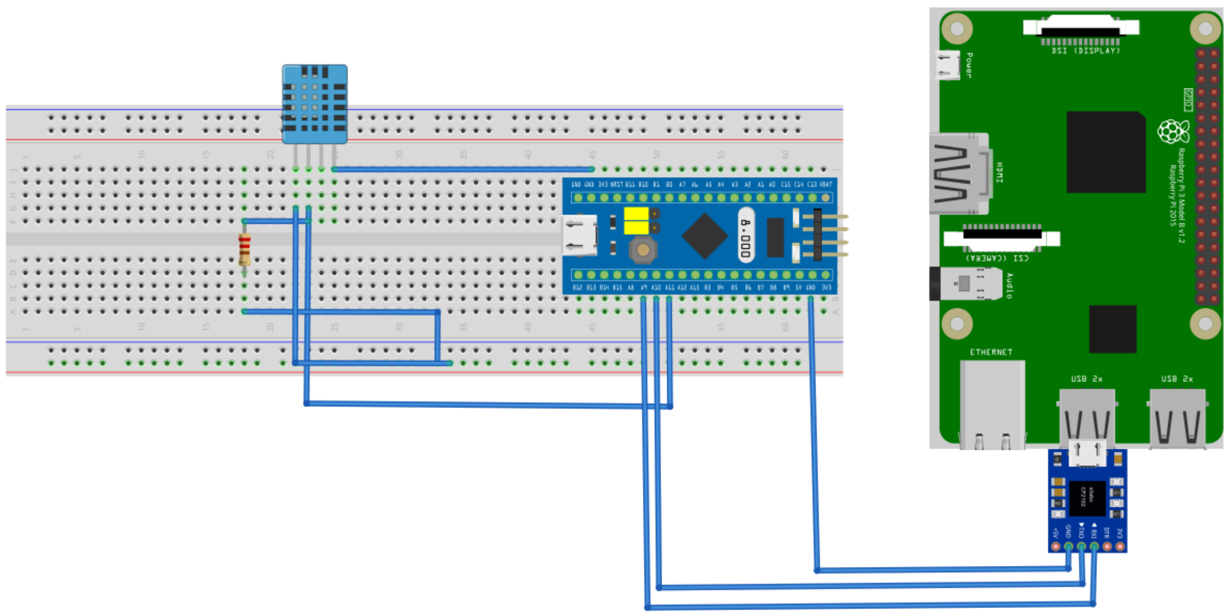
温湿度监控系统

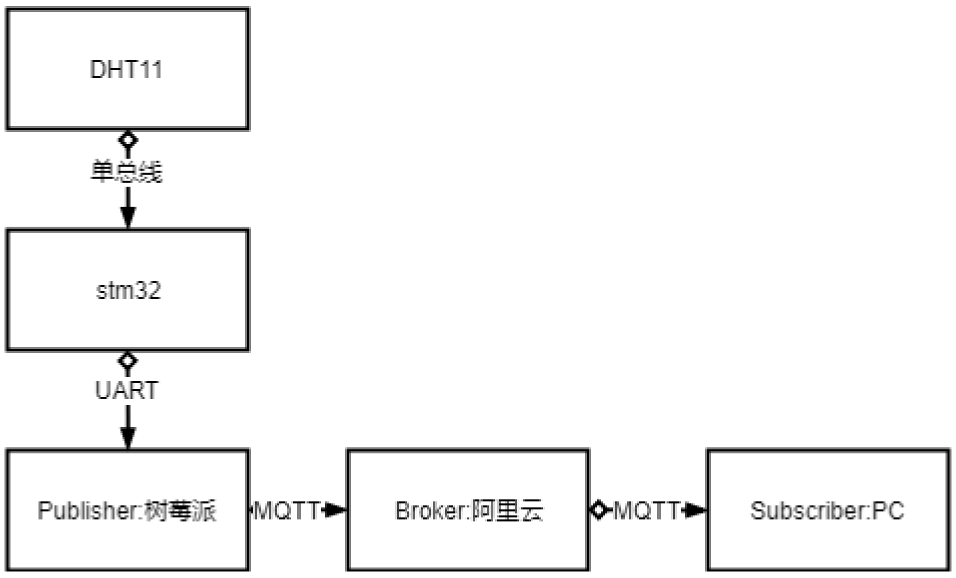
实验设计说明

1. 传感器数据来源为DHT11的温湿度数据
2. 在stm32端完成DHT11的数据读取并通过串口发送给树莓派
3. 在树莓派端将串口读取到的数据通过paho-mqtt上传至阿里云Broker
4. 在PC端订阅阿里云Broker的数据，存至本地的json文件

连线示意图



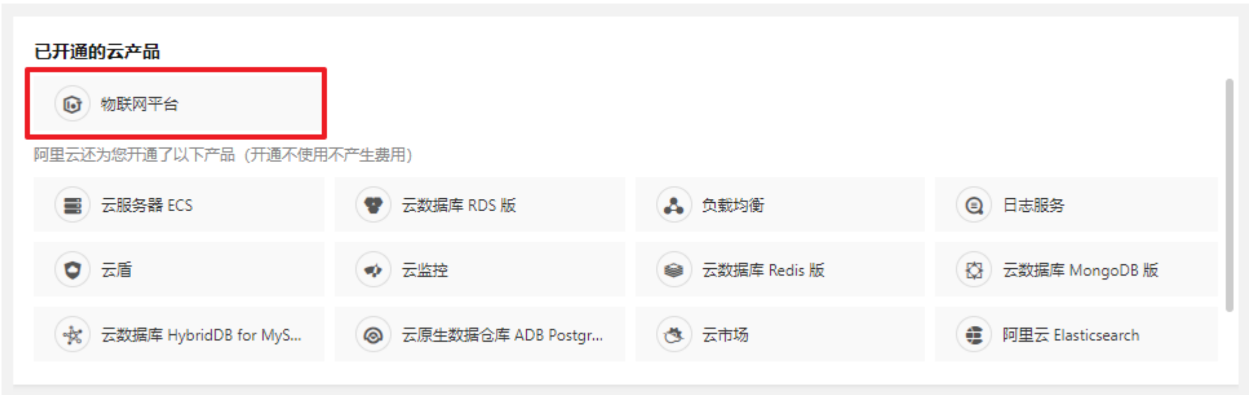
数据流转说明



103和树莓派上的代码说明

阿里云服务器注册与配置

1. 创建阿里云账号并进行实名认证
2. 在控制台中找到物联网平台



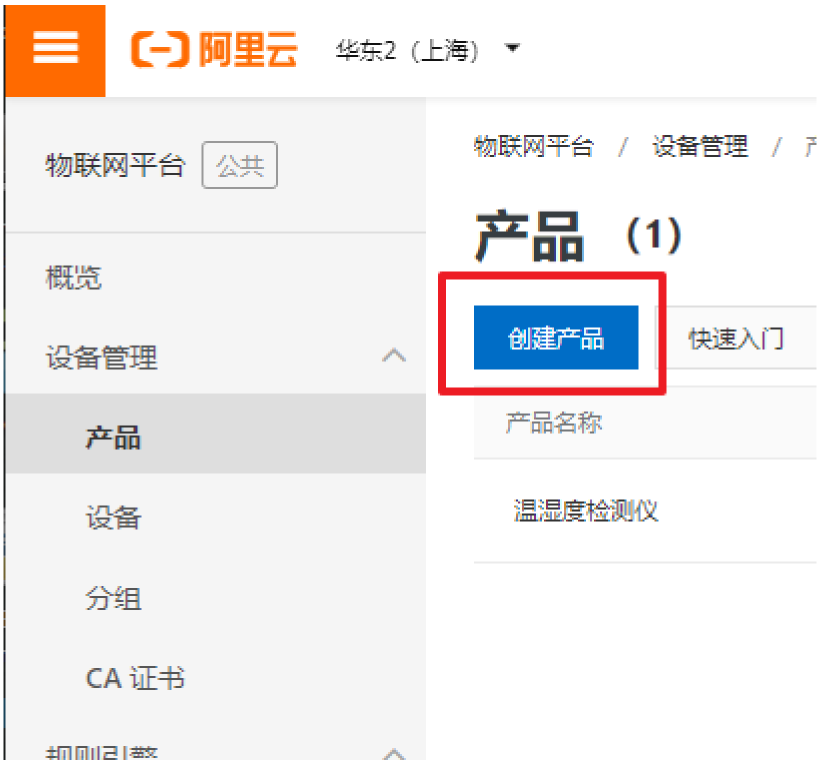
已开通的云产品

物联网平台

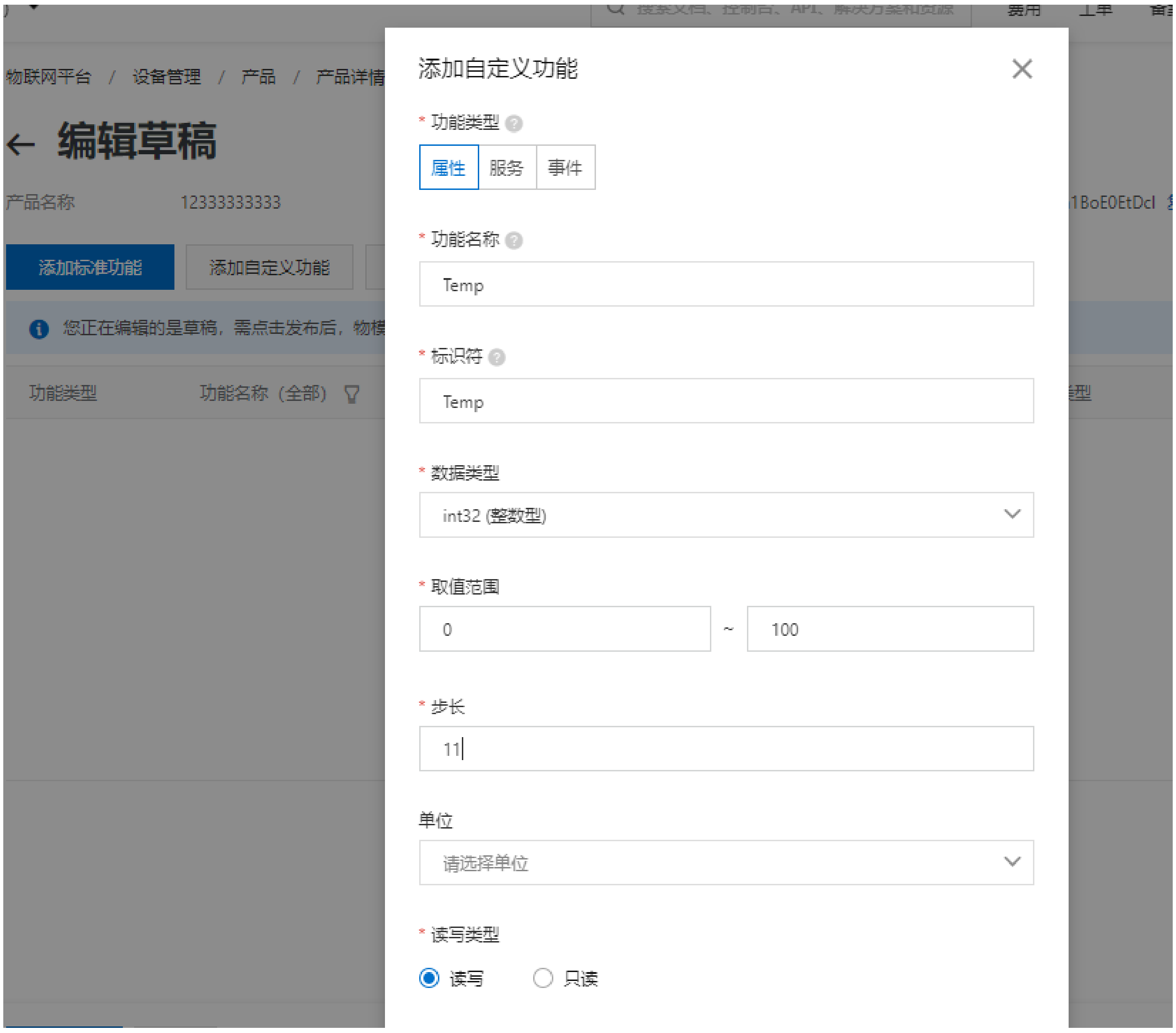
阿里云还为您开通了以下产品（开通不使用不产生费用）

云服务器 ECS	云数据库 RDS 版	负载均衡	日志服务
云盾	云监控	云数据库 Redis 版	云数据库 MongoDB 版
云数据库 HybridDB for MyS...	云原生数据仓库 ADB Postgr...	云市场	阿里云 Elasticsearch

3. 按照需求创建产品



4. 添加物模型


物联网平台 / 设备管理 / 产品 / 产品详情
编辑草稿
添加标准功能
添加自定义功能
您正在编辑的是草稿，需点击发布后，物模
功能类型
功能名称（全部）

添加自定义功能

* 功能类型 [✓ 属性] 服务 事件

* 功能名称
Temp

* 标识符
Temp

* 数据类型
int32 (整数型)

* 取值范围
0 ~ 100

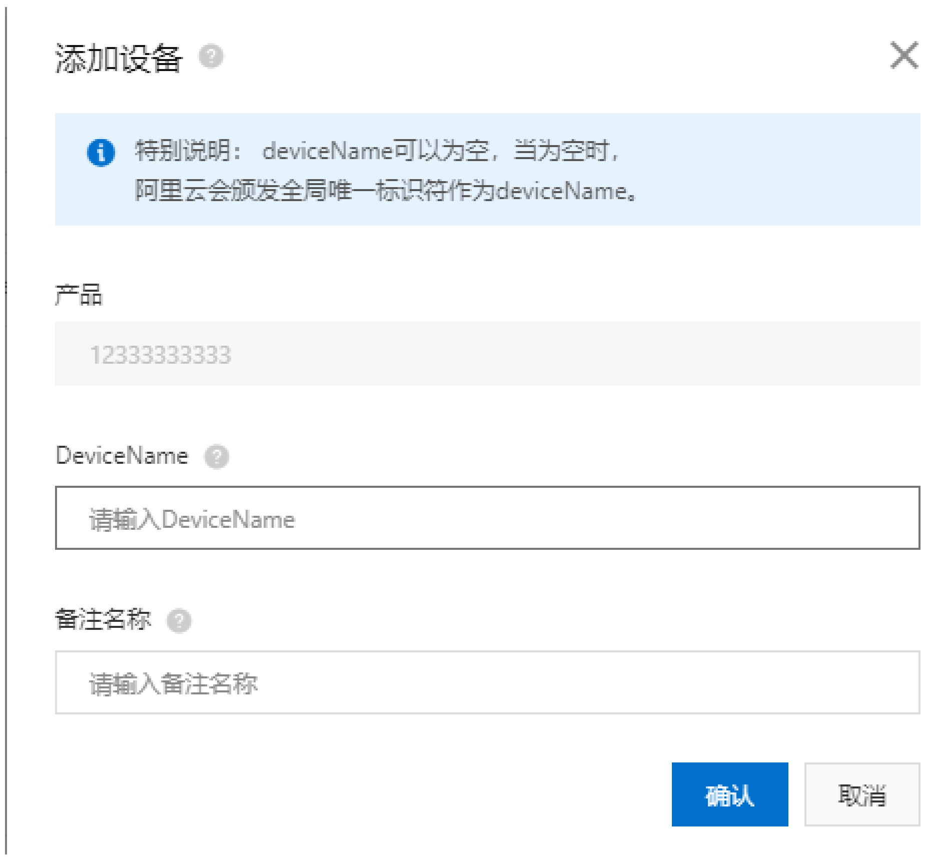
* 步长
11

单位
请选择单位

* 读写类型
[✓ 读写] 只读

产品名称 12333333333

5. 添加设备


添加设备

特别说明： deviceName可以为空，当为空时，
阿里云会颁发全局唯一标识符作为deviceName。

产品

12333333333

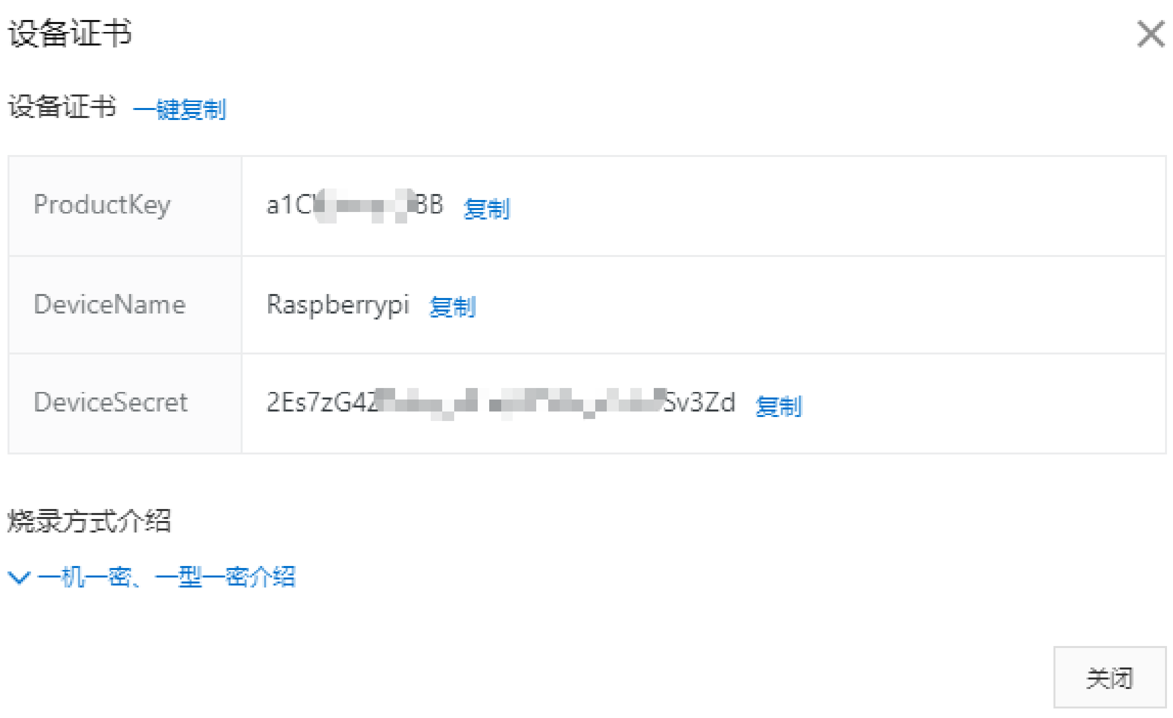
DeviceName

请输入DeviceName

备注名称

请输入备注名称

6. 查看设备证书，并在代码中相应地修改


设备证书

设备证书 一键复制

ProductKey	a1C[redacted]BB 复制
DeviceName	Raspberrypi 复制
DeviceSecret	2Es7zG4[redacted]Sv3Zd 复制

烧录方式介绍

一机一密、一型一密介绍

树莓派代码

树莓派端的代码主要包含两部分功能，一部分用于和stm32进行串口通讯，另一部分用于订阅TOPIC

- `option` 中存的是设备证书中的内容，需要对应

- 随后更据官方给出的demo以及option中的数据得到 `HOST` , `PORT` , `TOPIC` 等参数
- 主程序中先初始化串口
- `getAliyunIoTClient` 同样根据官方demo确定用户名和密码
- 随后注册 `on_connect` 和 `on_message`
- 不停从串口读取数据并上传至服务器

```python
import paho.mqtt.client as mqtt
import time
import hashlib
import hmac
import random
import json
import serial
import struct

# 设备证书中的内容
options = {
    'productKey':'a1C********',
    'deviceName':'Raspberrypi',
    'deviceSecret':'********************************',
    'regionId':'cn-shanghai'
}

HOST = options['productKey'] + '.iot-as-mqtt.' + options['regionId'] +
'.aliyuncs.com'
PORT = 1883
PUB_TOPIC = "/sys/" + options['productKey'] + "/" + options['deviceName'] +
"/thing/event/property/post"

# 连接后事件
def on_connect(client, userdata, flags, respons_code):
    if respons_code == 0:
        # 连接成功
        print('Connection Succeed!')
    else:
        # 连接失败并显示错误代码
        print('Connect Error status {0}'.format(respons_code))

# 收到数据后事件
def on_message(client, userdata, msg):
    print(msg.topic+" "+str(msg.payload))

def hmacsha1(key, msg):
    return hmac.new(key.encode(), msg.encode(), hashlib.sha1).hexdigest()

def getAliyunIoTClient():
    timestamp = str(int(time.time()))
    CLIENT_ID =
"paho.py|securemode=3,signmethod=hmacsha1,timestamp="+timestamp+"|"
    CONTENT_STR_FORMAT =
"clientIdpaho.pydeviceName"+options['deviceName']+"productKey"+options['productKey
']+"timestamp"+timestamp
```

```python
    # 设置用户名和密码
    USER_NAME = options['deviceName']+"&"+options['productKey']
    PWD = hmacsha1(options['deviceSecret'],CONTENT_STR_FORMAT)
    client = mqtt.Client(client_id=CLIENT_ID, clean_session=False)
    client.username_pw_set(USER_NAME, PWD)
    return client

# 开串口
def open_ser(port='com3', baudrate=115200):
    try:
        global ser
        ser = serial.Serial(port, baudrate, timeout=2)
        if(ser.isOpen()==True):
            print("串口打开成功")
    except Exception as exc:
        print("串口打开异常", exc)
    return ser

# 关串口
def close_ser(ser):
    try:
        ser.close()
        if ser.isOpen():
            print("串口未关闭")
        else:
            print("串口已关闭")
    except Exception as exc:
        print("串口关闭异常", exc)

if __name__ == '__main__':

    # 初始化串口
    ser = open_ser(port='/dev/ttyUSB0', baudrate=115200)

    # 初始化客户端
    client = getAliyunIoTClient()

    # 注册事件
    client.on_connect = on_connect
    client.on_message = on_message

    try:
        # 循环发送数据
        while 1:
            # 循环读取温湿度
            while 1:
                data = ser.read(2)
                if data != b'' and len(data) == 2:
                    break
            client.connect(HOST, PORT, 300)
            time.sleep(1)
            payload_json = {
                'id': int(time.time()),
                'params': {
                    'CurrentTemperature': data[0],
                    'CurrentHumidity': data[1]
                },
                'method': "thing.event.property.post"
            }
            print('send data to iot server: ' + str(payload_json))

            client.publish(PUB_TOPIC, payload = str(payload_json), qos=1)
            time.sleep(1)

        client.loop_forever()

    except KeyboardInterrupt:
        # 关闭串口
        close_ser(ser)
```

103代码

main：

- 不断读取`DHT11`的数据，如果读取成功则向树莓派发送数据

```c
if(DHT11Read(&temp, &humi) == DHT11_OK)
{
    HAL_UART_Transmit(&huart1, (uint8_t *)&temp, 1, 0xff);
    HAL_UART_Transmit(&huart1, (uint8_t *)&humi, 1, 0xff);
    HAL_Delay(5000);
}
```

dht11驱动：

- 与之前的实验类似，不再赘述

```c
#include <dht.h>

static void DHT11InMode();
static void DHT11OutMode();
static uint8_t DHT11Rst();
static uint8_t DHT11ReadByte(uint8_t *data);

static void DHT11InMode()
{
    GPIO_InitTypeDef GPIO_InitStruct = {0};
    GPIO_InitStruct.Pin = DHT_Pin;
    GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
    GPIO_InitStruct.Pull = GPIO_NOPULL;
    HAL_GPIO_Init(DHT_GPIO_Port, &GPIO_InitStruct);
}

static void DHT11OutMode()
{
    GPIO_InitTypeDef GPIO_InitStruct = {0};
    GPIO_InitStruct.Pin = DHT_Pin;
    GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
    GPIO_InitStruct.Pull = GPIO_NOPULL;
    GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_HIGH;
    HAL_GPIO_Init(DHT_GPIO_Port, &GPIO_InitStruct);
}

static uint8_t DHT11Rst()
{
    uint32_t cnt = 0;
    // 主机发开始信号
    DHT11OutMode();
    HAL_GPIO_WritePin(DHT_GPIO_Port, DHT_Pin, GPIO_PIN_RESET);
    HAL_Delay(25);
    // 拉高并延时等待
    HAL_GPIO_WritePin(DHT_GPIO_Port, DHT_Pin, GPIO_PIN_SET);
    DHT11InMode();
    // 等待DHT响应
    while(HAL_GPIO_ReadPin(DHT_GPIO_Port, DHT_Pin) == GPIO_PIN_SET)
    {
        if(++cnt > TIMEOUT_THRESH)
            return DHT11_TIMEOUT;
    }
    cnt = 0;
    // 等待DHT拉高延时
    while(HAL_GPIO_ReadPin(DHT_GPIO_Port, DHT_Pin) == GPIO_PIN_RESET)
    {
        if(++cnt > TIMEOUT_THRESH)
            return DHT11_TIMEOUT;
    }
    cnt = 0;
    // 等待低电平
    while(HAL_GPIO_ReadPin(DHT_GPIO_Port, DHT_Pin) == GPIO_PIN_SET)
    {
        if(++cnt > TIMEOUT_THRESH)
            return DHT11_TIMEOUT;
    }
    return DHT11_OK;
}

/*
 * @brief  读取一个字节的数据
 * @param  数据存储的位置
 * @retval 读取成功与否
 */
static uint8_t DHT11ReadByte(uint8_t *data)
{
    uint16_t cnt = 0;
    for(uint8_t i = 0; i < 8; ++i)
    {
        // 等待数据
        while(HAL_GPIO_ReadPin(DHT_GPIO_Port, DHT_Pin) == GPIO_PIN_RESET)
        {
            ++cnt;
            if(cnt > TIMEOUT_THRESH) return DHT11_TIMEOUT;
        }
        cnt = 0;
        // 等待下一数据开始位
        while(HAL_GPIO_ReadPin(DHT_GPIO_Port, DHT_Pin) == GPIO_PIN_SET)
        {
            ++cnt;
            if(cnt > TIMEOUT_THRESH) return DHT11_TIMEOUT;
        }
        if(cnt > HIGH_LOW_THRESH)
        {
            // 高电平
            (*data) <<= 1;
            (*data) |= 0x01;
        }
        else
        {
            // 低电平
            (*data) <<= 1;
            (*data) &= 0xfe;
        }
    }
    return DHT11_OK;
}

/*
 * @brief  读取一次数据
 * @param  数据存储的位置
 * @retval 读取成功与否
 */
uint8_t DHT11Read(uint8_t *temp, uint8_t *humi)
{
    pDHT11_data data = (pDHT11_data)malloc(sizeof(DHT11_data));
    uint8_t timeout_flag = 0;
    timeout_flag += DHT11Rst();
    // 接收40位数据
    timeout_flag += DHT11ReadByte(&(data->humi_integer));
    timeout_flag += DHT11ReadByte(&(data->humi_decimal));
    timeout_flag += DHT11ReadByte(&(data->temp_integer));
    timeout_flag += DHT11ReadByte(&(data->temp_decimal));
    timeout_flag += DHT11ReadByte(&(data->checksum));
    if(timeout_flag != DHT11_OK)
    {
        free(data);
        return DHT11_TIMEOUT;
    }
    if(data->checksum != data->humi_integer + data->humi_decimal + data-
>temp_integer + data->temp_decimal)
    {
        free(data);
        return DHT11_CHECKERROR;
    }
    *temp = data->temp_integer;
    *humi = data->humi_integer;
    free(data);
    return DHT11_OK;
}
```

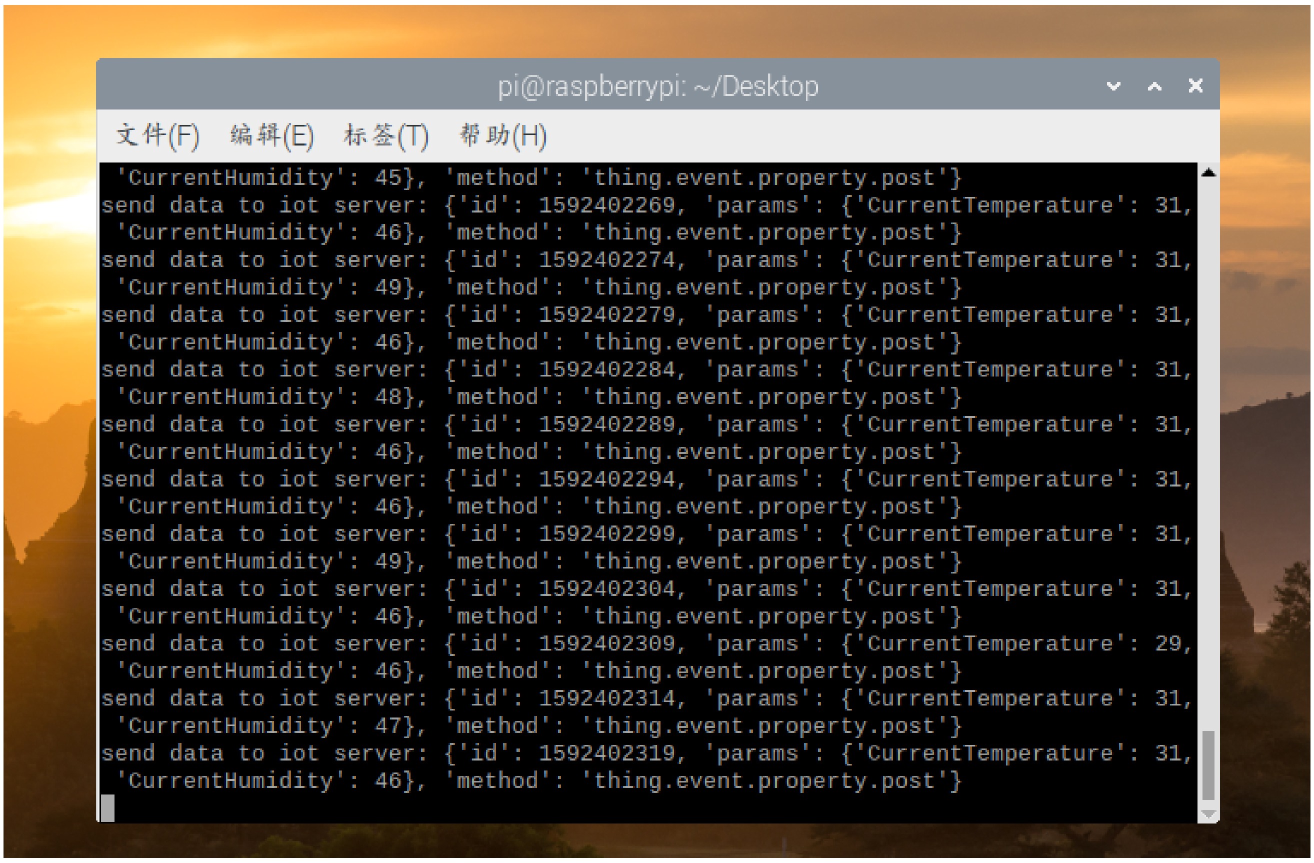
pc端订阅代码

从服务器订阅TOPIC

```python
import paho.mqtt.client as mqtt
import time
import hashlib
import hmac
import json

options = {
    'productKey':'a1C********',
    'deviceName':'Raspberrypi',
    'deviceSecret':'********************************',
    'regionId':'cn-shanghai'
}

HOST = options['productKey'] + '.iot-as-mqtt.' + options['regionId'] + '.aliyuncs.com'
PORT = 1883
SUB_TOPIC = "/sys/" + options['productKey'] + "/" + options['deviceName'] + "/thing/event/property/post"

# 连接后事件
def on_connect(client, userdata, flags, respons_code):
    if respons_code == 0:
        # 连接成功
        print('\r','Connection Succeed!', end='', flush=True)
    else:
        # 连接失败并显示错误代码
        print('Connect Error status {0}'.format(respons_code))

    # 订阅信息
    client.subscribe(SUB_TOPIC, qos=1)

# 接收到数据后事件
def on_message(client, userdata, msg):
    # 打印订阅消息主题
    # print("topic", msg.topic)
    # 打印消息数据
    # print("msg payload", str(msg.payload))
    str1 = str(msg.payload, encoding = "utf-8")
    msg_dict = eval(str1)
    print(' CurrentTemperature:', msg_dict['params']['CurrentTemperature'], '°C', end='', flush=True)
    print(' CurrentHumidity:', msg_dict['params']['CurrentHumidity'], '%', end='', flush=True)
    dict[msg_dict['id']] = {'CurrentTemperature':msg_dict['params']['CurrentTemperature'],
                            'CurrentHumidity':msg_dict['params']['CurrentHumidity']}

def hmacsha1(key, msg):
    return hmac.new(key.encode(), msg.encode(), hashlib.sha1).hexdigest()

def getAliyunIoTClient():
    timestamp = str(int(time.time()))
    CLIENT_ID = "paho.py|securemode=3,signmethod=hmacsha1,timestamp="+timestamp+"|"
    CONTENT_STR_FORMAT = "clientIdpaho.pydeviceName"+options['deviceName']+"productKey"+options['productKey']+"timestamp"+timestamp
    # 设置用户名和密码
    USER_NAME = options['deviceName']+"&"+options['productKey']
    PWD = hmacsha1(options['deviceSecret'],CONTENT_STR_FORMAT)
    client = mqtt.Client(client_id=CLIENT_ID, clean_session=False)
    client.username_pw_set(USER_NAME, PWD)
    return client

if __name__=="__main__":

    dict = {}
    # 初始化客户端，选择MQTT版本
    client = getAliyunIoTClient()

    # 注册事件
    client.on_connect = on_connect
    client.on_message = on_message

    # 连接到服务器
    client.connect(HOST, port = PORT, keepalive=60)

    # 守护连接状态
    try:
        client.loop_forever()
    except:
        with open('record.json','w') as f:
            json.dump(dict, f, sort_keys=True, indent=4, separators=(',', ':'))
```

网页上的实时场景

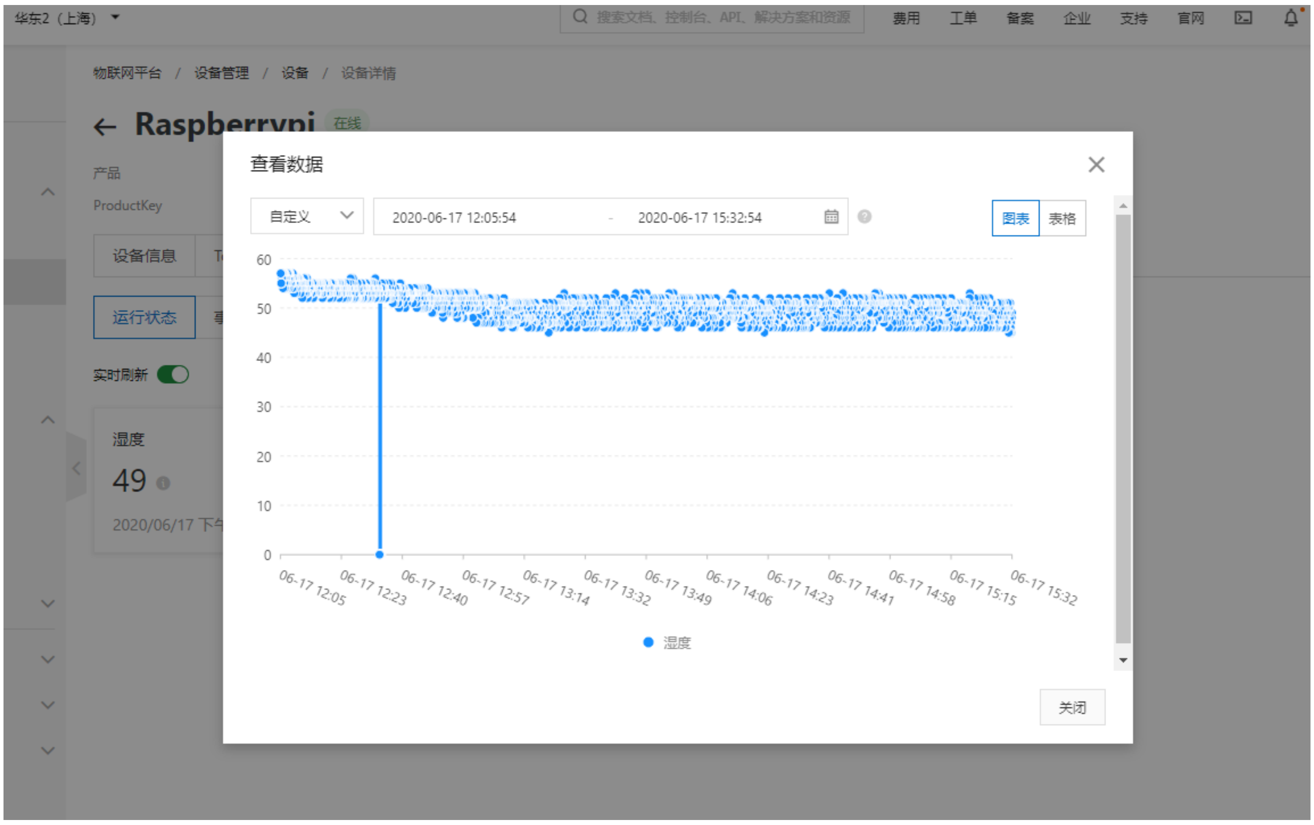
树莓派post端数据：



```
 'CurrentHumidity': 45}, 'method': 'thing.event.property.post'}
send data to iot server: {'id': 1592402269, 'params': {'CurrentTemperature': 31,
 'CurrentHumidity': 46}, 'method': 'thing.event.property.post'}
send data to iot server: {'id': 1592402274, 'params': {'CurrentTemperature': 31,
 'CurrentHumidity': 49}, 'method': 'thing.event.property.post'}
send data to iot server: {'id': 1592402279, 'params': {'CurrentTemperature': 31,
 'CurrentHumidity': 46}, 'method': 'thing.event.property.post'}
send data to iot server: {'id': 1592402284, 'params': {'CurrentTemperature': 31,
 'CurrentHumidity': 48}, 'method': 'thing.event.property.post'}
send data to iot server: {'id': 1592402289, 'params': {'CurrentTemperature': 31,
 'CurrentHumidity': 46}, 'method': 'thing.event.property.post'}
send data to iot server: {'id': 1592402294, 'params': {'CurrentTemperature': 31,
 'CurrentHumidity': 46}, 'method': 'thing.event.property.post'}
send data to iot server: {'id': 1592402299, 'params': {'CurrentTemperature': 31,
 'CurrentHumidity': 49}, 'method': 'thing.event.property.post'}
send data to iot server: {'id': 1592402304, 'params': {'CurrentTemperature': 31,
 'CurrentHumidity': 46}, 'method': 'thing.event.property.post'}
send data to iot server: {'id': 1592402309, 'params': {'CurrentTemperature': 29,
 'CurrentHumidity': 46}, 'method': 'thing.event.property.post'}
send data to iot server: {'id': 1592402314, 'params': {'CurrentTemperature': 31,
 'CurrentHumidity': 47}, 'method': 'thing.event.property.post'}
send data to iot server: {'id': 1592402319, 'params': {'CurrentTemperature': 31,
 'CurrentHumidity': 46}, 'method': 'thing.event.property.post'}
```

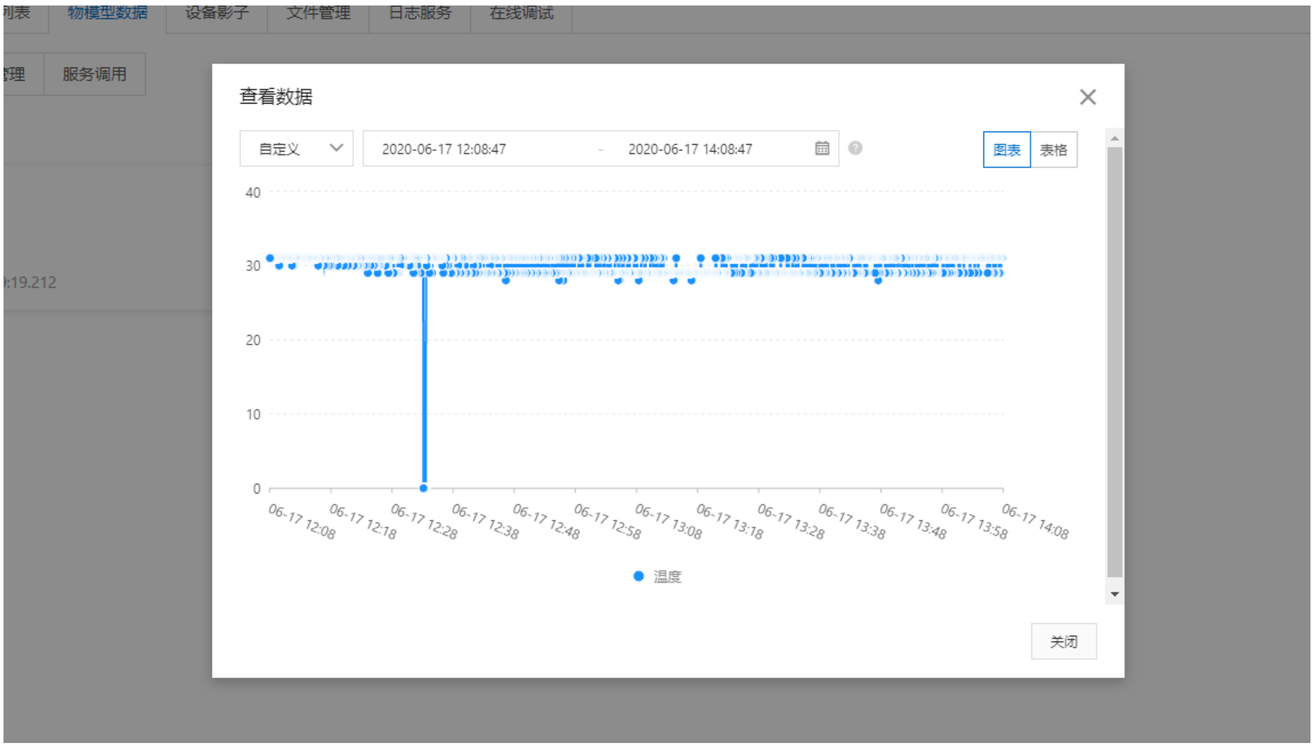
从12：00-15：30每隔5s就传一次数据，结果如下：

湿度数据：



- 可以看到在12：30的时候数据出过一些错误，由于数据全0，没有被DHT11的checksum滤去。在异常数据的处理上还是有一些瑕疵。由于选择的传输质量`qos=1`，因此数据只会冗余不会缺失。

温度数据



- 可以看到，和湿度相同，在12：30的时候出现了一些问题，其余均正常工作。

订阅端显示:

```
PS C:\Users\Rookie\Desktop\大三下\嵌入式\project> python -u "c:\Users\Rookie\Desktop\大三下\嵌入式\project\subscribe.py"
 Connection Succeed! CurrentTemperature: 31 °C CurrentHumidity: 49 %
```

保存的json文件（部分）：

```
{
    "1592379400":{
        "CurrentHumidity":50,
        "CurrentTemperature":29
    },
    "1592379405":{
        "CurrentHumidity":50,
        "CurrentTemperature":31
    },
    "1592379410":{
        "CurrentHumidity":50,
        "CurrentTemperature":31
    },
    "1592379415":{
        "CurrentHumidity":48,
        "CurrentTemperature":31
    },
    "1592379420":{
        "CurrentHumidity":48,
        "CurrentTemperature":31
    },
    "1592379425":{
        "CurrentHumidity":49,
        "CurrentTemperature":31
    }
}
```

总结

跑通了数据的整个流程，阿里云的服务做得很好，用起来还是很方便的。paho的库也相对简单易用。

参考

https://www.yuque.com/cloud-dev/iot-tech/rz6fpl

https://liaocy.net/2018/20180620-mqttclient/