# python自动化测试说明

## 1 简述

本测试框架主要测试iotlink相关API接口，可以覆盖mqtt、lwm2m、coap等协议，也可以用来测试文件系统、驱动框架、AT命令等。测试框架使用一款python自动化框架pytest，关于pytest的介绍，可以自行百度，这里不再详述。

### 测试组网

IoT Hub

Win/Linux主机运行Python脚本

嵌入式开发板（如STM32）

https应用通道

Mqtt/Coap/LwM2m

私有协议

测试消息

图1 测试组网说明

在图1中，运行在win或者linux上的python脚本通过消息接口将参数发给嵌入式开发板（使用了pytest框架），嵌入式开发板执行相应的用例，过程中也可能通过Mqtt/Coap/LwM2M和OC云交互，然后运行完成后，将结果返回给win或者linux主机，主机解析运行结果，判断用例执行情况，如果需要通过OC云查询或者下发命令的话，需要通过应用API直接和OC云通信（例如，查询设备影子是否在线）。

### 测试代码说明

测试代码分为2部分，第一部分是运行在win或者linux主机上的python脚本，第2部分为运行在嵌入式开发板上C编写的测试用例，这部分代码主要完成接受主机的指令，完成相应的接口调用，然后返回执行结果给主机，执行结果判断都放在主机上进行。

## 2 测试框架原理

## 2.1私有测试消息格式

主机和嵌入式单板通过UDP消息通信，其中，嵌入式开发板为服务器端，绑定了端口号5999，测试过程中，主机作为发起方需要将请求消息发给嵌入式单板，请求消息的格式为固定头格式的变长字符串消息，以mqtt\_config为例，请求消息为，

131074|test\_linux\_oc\_mqtt\_config\_static|mqtt\_sdk05|2348d831419f04b39512|49.4.93.24|8883|YES|0|10|1|0|1|2|2|1|

其中，第一个字段固定为用例的ID，第二个字段固定为用例的名字，后面的均为变长的参数，各个参数以|隔开。后面变长的参数只有单板上对应的用例才可以理解，每一个用例，对应的参数都是不同的。

应答消息为

131074,|0,|

其中，131074为测试用例ID，0表示成功且应答中不携带数据，如果应答中需要返回透明数据，那么第二个字段为1，如果失败，第二个字段为-1。

如下为用户返回了透明数据的例子，

Request：131080|test\_linux\_oc\_mqtt\_sendcmd\_static|

Response：131080,|1,|57

2.2 测试用例ID的用途

在上面的2.1节中，测试用例ID为131074，按照16进制表示为0x20002，

其中高16位为协议类型，低16位为具体功能实体，在test\_case.h中可以看到

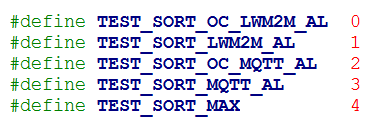


图2 协议类型

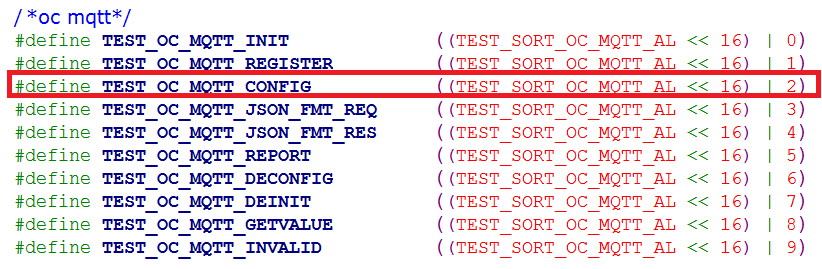


图3 功能实体

用例ID为0x20002，可以看出具体的用例为OC\_MQTT\_CONFIG，可以找到对应的执行的C代码函数为图4所示，

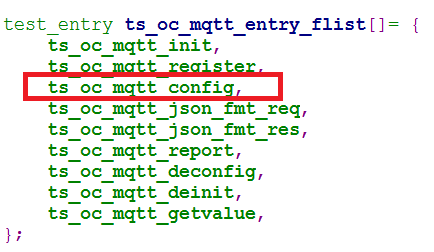


图4 功能实体对应的入口函数

该函数ts\_oc\_mqtt\_config，首先解析来自主机传递过来的参数mqtt\_sdk05|2348d831419f04b39512|49.4.93.24|8883|YES|0|10|1|0|1|2|2|1|，然后执行API接口oc\_mqtt\_config，最后将API接口oc\_mqtt\_config执行结果通过socket返回给主机，主机发现oc\_mqtt\_config执行结果为成功，那么此时可以通过北向接口查询一下对应的设备是否在线。

### 2.3 测试注意事项

测试某一个模块时，对应的demo不能运行，而应该由测试代码来执行这个操作，例如，我们在测试OC\_MQTT\_AL的接口时，不能运行如下的demo代码，因为我们的测试代码会做demo中相同的功能。

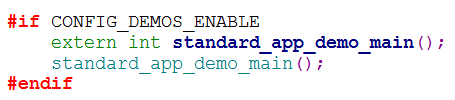


图5 link\_main.c中不能运行demo代码

同样，由于测试用例会调用oc\_lwm2m\_init()等函数，因此，下面图6的代码我们也不希望在主任务执行，

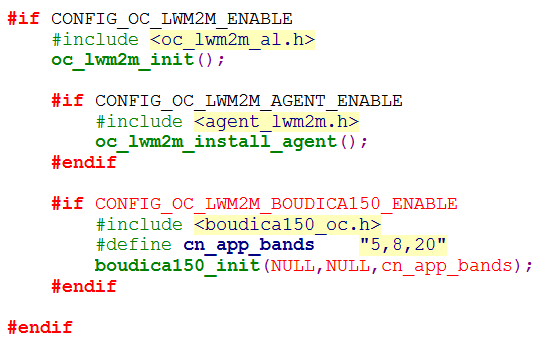
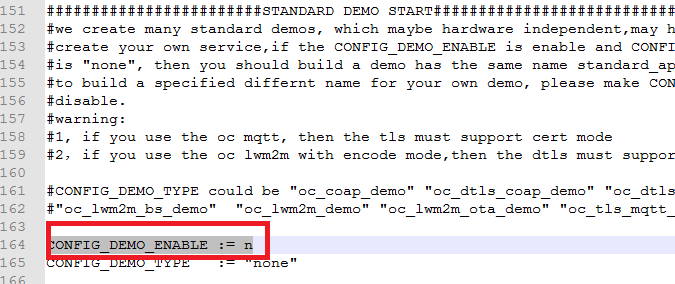


图6 link\_main.c中不要运行上面的代码

为了完成以上的功能，我们必须修改makefile，修改的地方如图7、图8。

图7 config.mk设置不启用demo

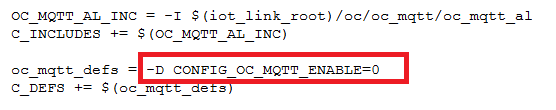


图8 修改oc\_mqtt.mk

## 3 多协议测试

根据开发环境（开发板）、协议类型组合起来，要测试的接口包括如下，

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | OC-MQTT | MQTT-AL | OC-LWM2M | LWM2M-AL | OC-COAP | COAP-AL | SOTA | FOTA |
| Linux | Y | Y | Y | Y | Y | Y | N | N |
| STM32F429 | Y | Y | Y | Y | Y | Y | Y | Y |
| F429\_NOVA | Y | Y | Y | Y | Y | Y | Y | Y |

经过排列组合，大约至少要22种，测试包括嵌入开发版本制作、烧结、重启、测试这几个步骤，因此，测试前，

需要把每一种测试的makfile提前准备好，如下图9所示，制作版本时，用脚本编写copy到对应的编译路径，就可以制作测试版本了。

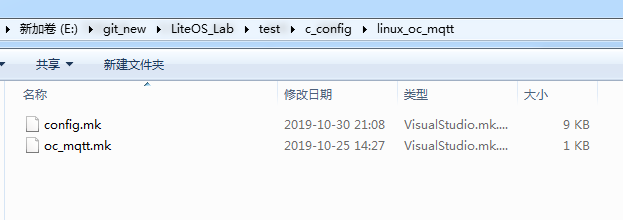


图9 makefile在测试目录中就绪

## 4. 主机测试环境搭建

## 4.1 windows下环境搭建

（1） 下载python

<https://www.python.org/downloads/windows/>

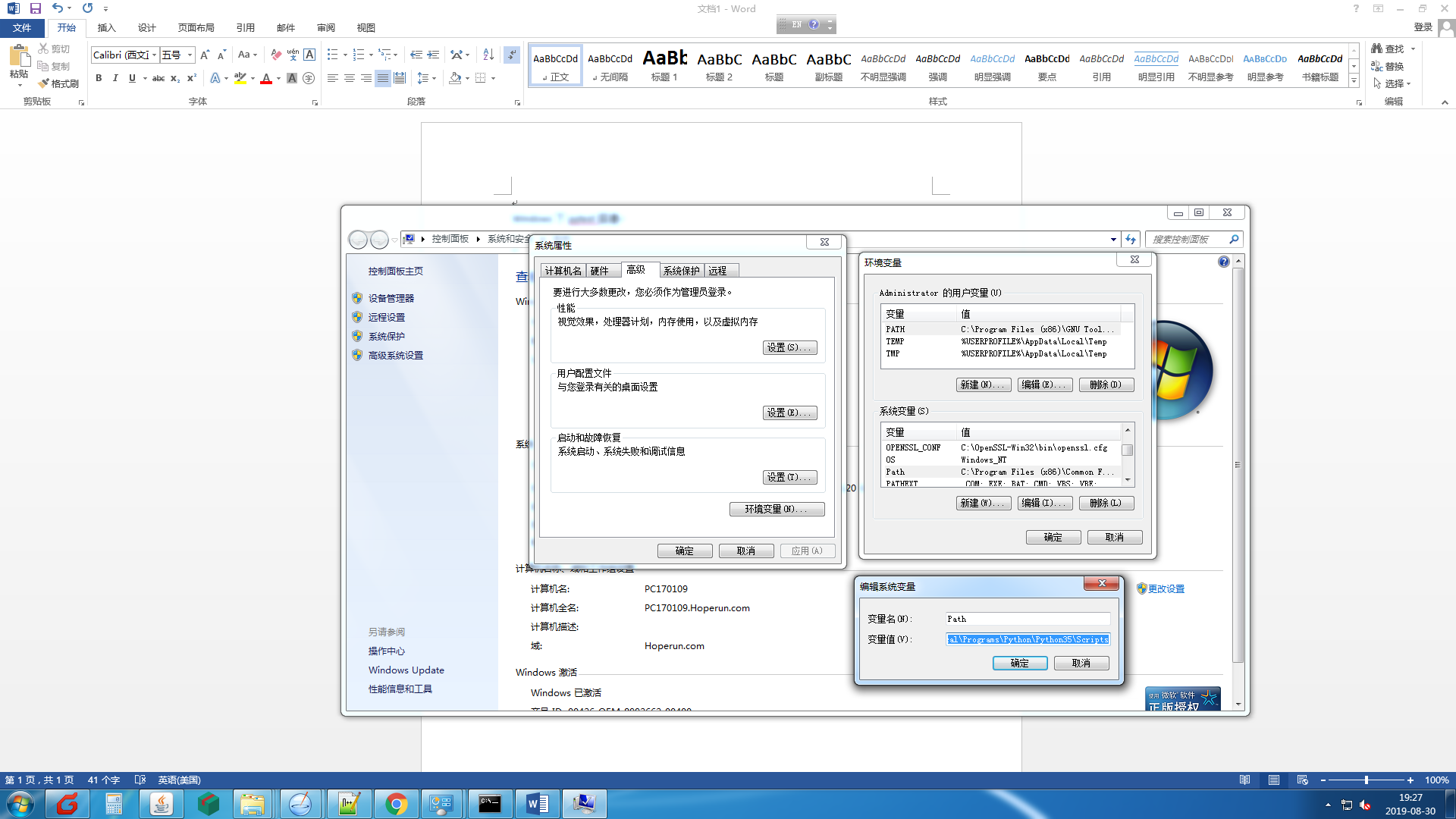
选择3.5.4版本的64位

（2） 设置环境变量

将C:\Users\Administrator\AppData\Local\Programs\Python\Python35;

C:\Users\Administrator\AppData\Local\Programs\Python\Python35\Scripts

加入系统的Path路径



（3） 安装pytest

在cmd下执行pip install –U pytest

（4） 查看pytest版本

在cmd下执行pytest –version

（5）安装集成开发环境pycharm（可选）

下载地址：<https://www.jetbrains.com/pycharm/>

### 4.2 Linux环境下搭建

### 5 **测试样例**

测试样例给出的是MQTT Linux版本+windows主机测试方法，完全自动化方案待后续补充。

5.1制作MQTT Linux版本

将test\c\_config\linux\_oc\_mqtt\config.mk拷贝并覆盖targets\LINUX\GCC\config.mk，将test\c\_config\linux\_oc\_mqtt\oc\_mqtt.mk拷贝并覆盖iot\_link\oc\oc\_mqtt\oc\_mqtt.mk，

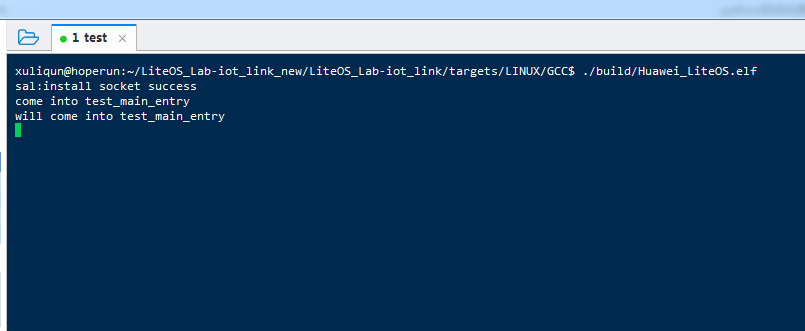
编译并运行，得到如下的图10，

图10 MQTT Linux版本运行

### 5.2在windows主机下执行脚本

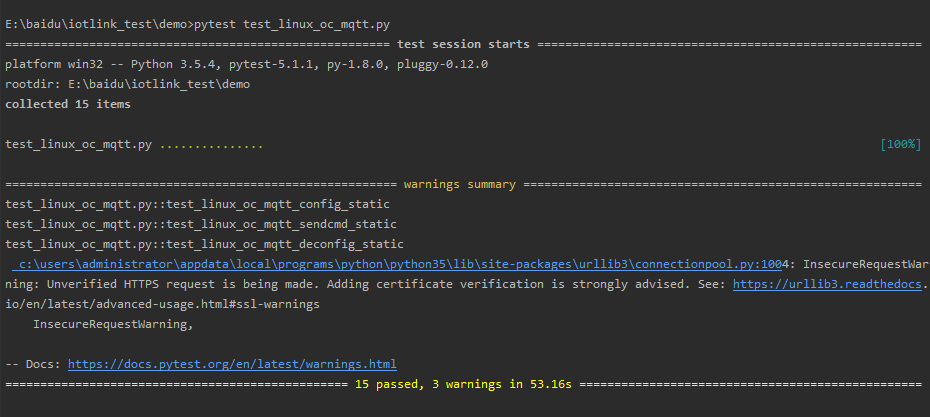
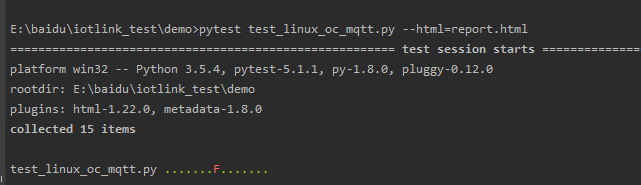


图11 在windows主机执行测试脚本

### **5.3 生成html报告**

安装插件，执行pip install pytest-html

运行



执行完成后，可以看到当前目录下生成了html文件。

（未完待续）