# 概述

Python测试工具包括2个部分：测试执行工具和测试用例生成工具。

## 测试执行工具

* 测试框架

选择nose为测试框架。对于基于python和nose的测试系统，其具体介绍可以参看《Python测试框架nose介绍.pptx》。

* 接口驱动

目前实现的接口驱动有消息总线驱动和管理客户端消息接口驱动。

消息总线驱动封装了pika和google.protobuf的编解码，为测试用例提供了友好的接口。

* google.protobuf

测试用例直接使用google.protobuf进行测试数据的赋值。

QueryMemberRsp\_test.py是一个极端复杂数据的赋值例子。

## 测试用例生成工具

用wireshark或者tcpdump对消息总线或者客户端接口进行消息抓包并保存为pcap格式文件，测试用例生成工具能够从pcap文件生成消息总线测试用例和管理客户端消息接口两种测试用例。

# 软件

## 需安装软件

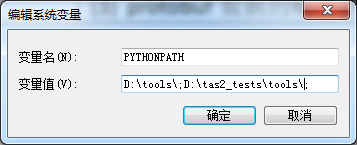
|  |  |
| --- | --- |
| 软件 | 说明 |
| Python 2.7.10 | google.protobuf要求用2.7，不能用python3 |
| nose-1.3.7 | python测试框架 |
| dpkt-1.8.6 | 解析pcap文件的python包 |
| haigha-0.8.0 | AMQP消息总线的python client |
| protobuf | protobuf编译器prptoc.exe  用于protobuf消息编解码的python库(google.protobuf) |
| wireshark | 在Windows计算机上进行消息抓包  查看抓包文件的内容  Linux上抓包用tcpdump即可 |

## 从svn获取版本

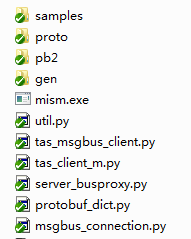
http://192.168.30.132/svn/doc/testDept/平台组/chen.xiaobo/tools/new

## PYTHONPATH设置

需要设置2个路径：测试工具文件夹的路径，pb2文件夹的路径。



## 测试工具的文件组织



|  |  |  |  |
| --- | --- | --- | --- |
| 文件夹 | | 文件 | 说明 |
| 测试  工具  目录 | 根目录 | protobuf.py | protobuf 消息的Codec类  protobuf数据的比对函数  protobuf数据的赋值语句的生成函数 |
| client.py | 用于模拟客户端的测试驱动 |
| msgbus.py | 总线消息驱动，在haigha库之上封装而成 |
| util.py | 杂项工具箱 |
| gen | gen\_pb2.py | 为proto文件夹里的所有proto文件生成对应的pb2\_.py文件 |
| gen\_protobuf\_dict.py | 从tas\_funcodes.h生成funcodes到protobuf数据类的映射表 |
| gen\_test.py | 根据消息总线或管理客户端的.pcap抓包文件，生成对应的测试用例 |
| mism |  | mism加密服务器的Visual Studio C++工程，生成加密服务器程序mism.exe。 |
| 测试  用例  目录 | tools | pb2/\*.proto | 放置所有的.proto文件 |
| \*\_pb2.py | 所有从.proto文件生成的\*\_pb2.py文件 |
| funcode.py | funcodes到protobuf数据类的映射表 |
| x\_tests | \*\_tests.py | 测试用例 |
|  |  |  |

## gen（各生成器所在的目录）

### gen\_pb2.py

读取上述proto目录里所有的.proto文件，批量生成所有的pb2.py文件，放在proto\output目录里。

### gen\_protobuf\_dict.py

调用protobuf\_dict\_generator.py

输入：tas\_funcodes.h, tas\_inner\_funcodes.h, bank\_funcodes.cs, ……

输出：protobuf\_dict\_temp.py

protobuf\_dict\_temp.py需要手工添加如下2条记录，再拷贝到tools目录下作为正式文件：

336265514: bank\_pb2.QListBankParamReq,

336265515: bank\_pb2.QListBankParamRsp,

# 消息总线测试用例编写

## 需要的知识

### 消息总线

direct mode: connection, channel, exchange, queue

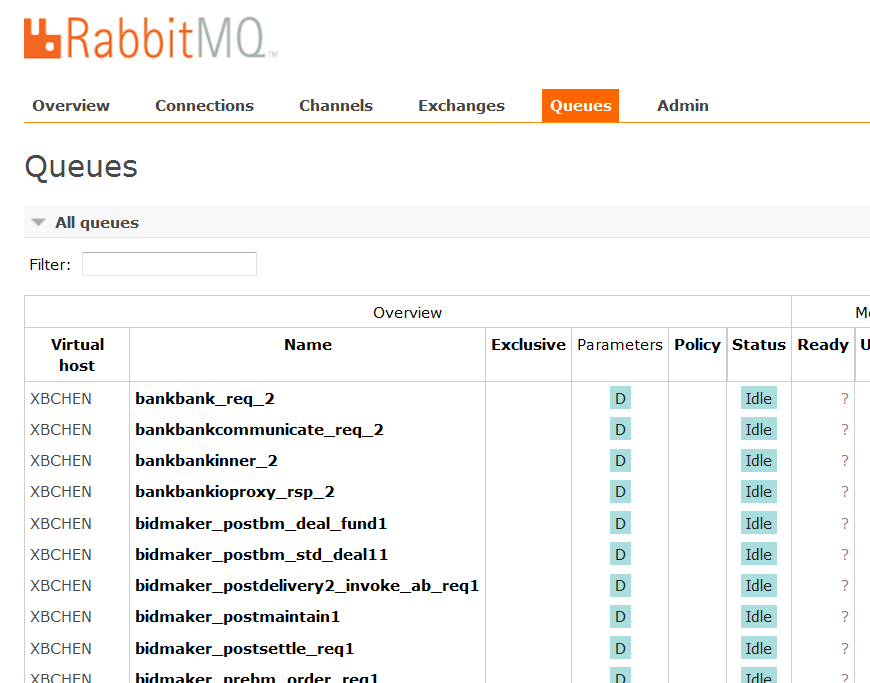
pika api

### protobuf

.proto

\_pb2.py

### 产品总线配置



## 消息总线的测试驱动(msgbus.py)

|  |  |  |  |
| --- | --- | --- | --- |
| 接口 |  | 参数 | 说明 |
| close\_all() | |  | 关闭所有消息总线连接 |
| Connection | Connection() |  |  |
| close() |  |  |
| queue\_bind() |  |  |
| clear\_queue() |  |  |
| publish() |  |  |
| get() |  |  |
| expect() |  | 只检查消息类型 |
| expect2() |  | 与预期数据进行比对 |

## 测试用例sample

见bank\_query\_test.py

测试用例可以只创建1个connection和 1个pulish channel, 然后对需要接收消息的各个queue进行绑定（订阅）。

## protobuf数据的赋值

方法1：创建数据对象时一并赋值。

方法2：创建数据对象后，对各个域分别赋值。

# 消息总线测试用例的自动生成

## 测试用例生成器

|  |  |
| --- | --- |
| 文件 | 说明 |
| gen\_test.py | 通过此程序来调用测试用例生成器  文件里包括需要的各种参数 |
| test\_generator.py | 测试用例生成器  支持多连接；支持2种pcap格式：ethernet(wireshark)和linux coked capture(tcpdump) |
| msgbus\_test\_generator.py | AMQP总线TCP消息的提取器 |
| 总线消息的测试代码生成器 |
| client\_m\_test\_generator.py | 管理客户端的TCP消息抽取器 |
| 管理客户端的测试代码生成器 |

## 操作步骤

### 可执行测试用例的录制

如果想录制可执行的测试，那么每个服务都要从启动开始录制。否则测试用例就可能存在用不存在的连接来收发消息的情况。

### 建立TCP连接与进程的对照表

（1）Linux

执行以下程序，自动生成connection\_dict.py

$python gen\_connection\_service\_dict.py

（2）Windows

用netstat –b –n查询服务的连接信息，之后手工向connection\_dict.py添加内容

### 步骤2：消息抓包

Windows上的服务，可以用Wireshark抓包并存盘。

Linux上的服务，可以用tcpdump抓包并存盘。

### 步骤3：去掉被测服务的消息

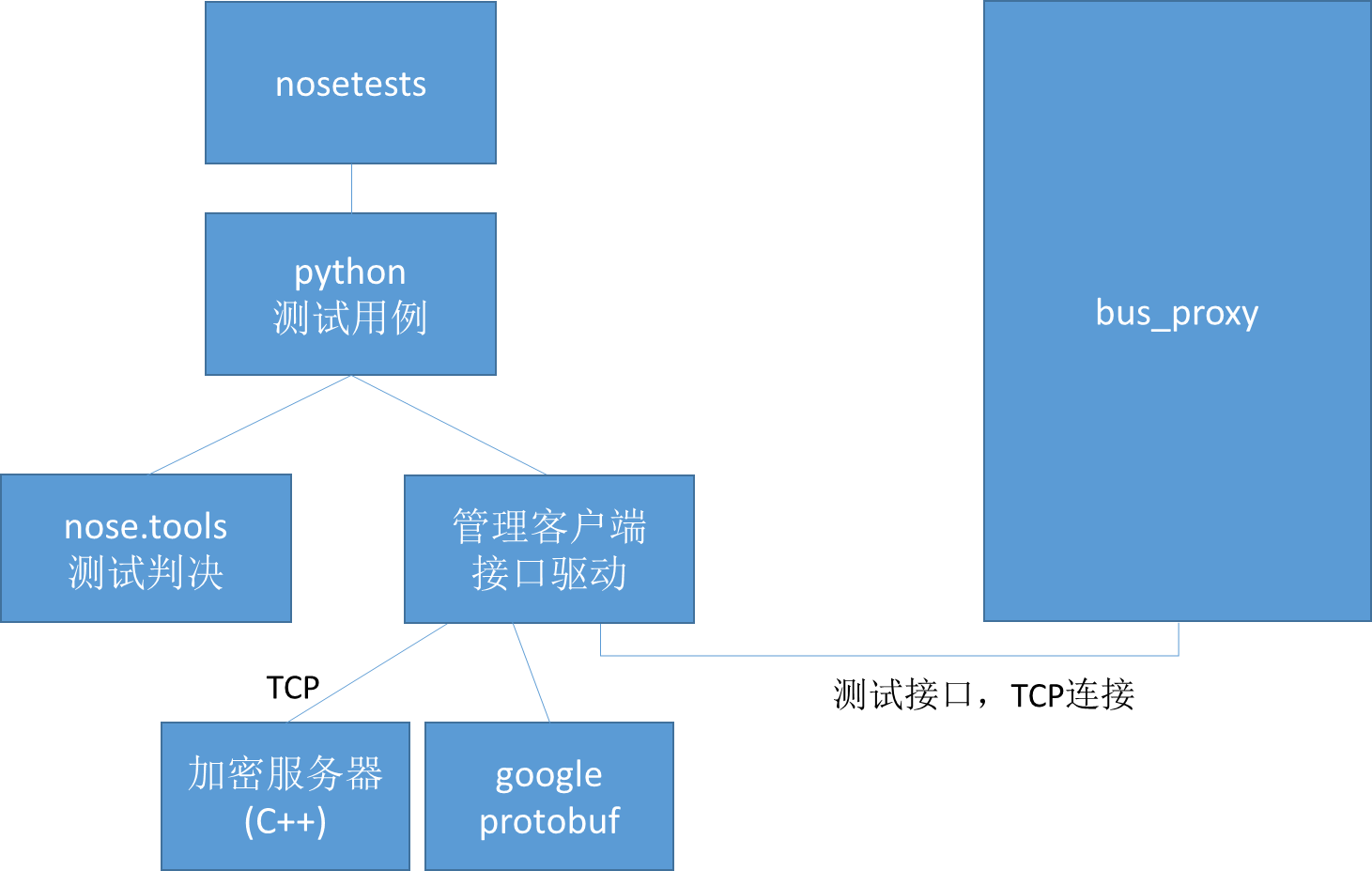
使用Wireshark里的消息过滤功能

### 步骤4：生成测试用例

### 步骤5：特殊处理

不认识的funcode

# 管理客户端接口的测试用例编写



tas\_client\_m.py

# 测试用例调试

## 被pika debug信息淹没

nosetests执行用例失败时，被pika输出的大量debug信息所淹没。解决办法：

pika.callback.LOGGER.setLevel('WARNING')

pika.channel.LOGGER.setLevel('WARNING')

pika.connection.LOGGER.setLevel('WARNING')

## 特定服务模拟器

生成某个服务启动过程的脚本，据此编写该服务的测试驱动。

## 对各个业务过程进行消息抓包

## connection\_service\_dict.txt

## 用测试用例生成器生成测试用例

# 常见问题

## 相关消息队列没有清空

测试开始前，检查相关消息队列是否清空。如果没有，手工清空之。

## queue\_declare/exchange\_declare

语句：

recv\_chan.queue\_declare(queue='heartbeat', durable=False) # durable应为True

错误：

ChannelClosed: (406, "PRECONDITION\_FAILED - parameters for queue 'heartbeat' in vhost 'XBCHEN' not equivalent")

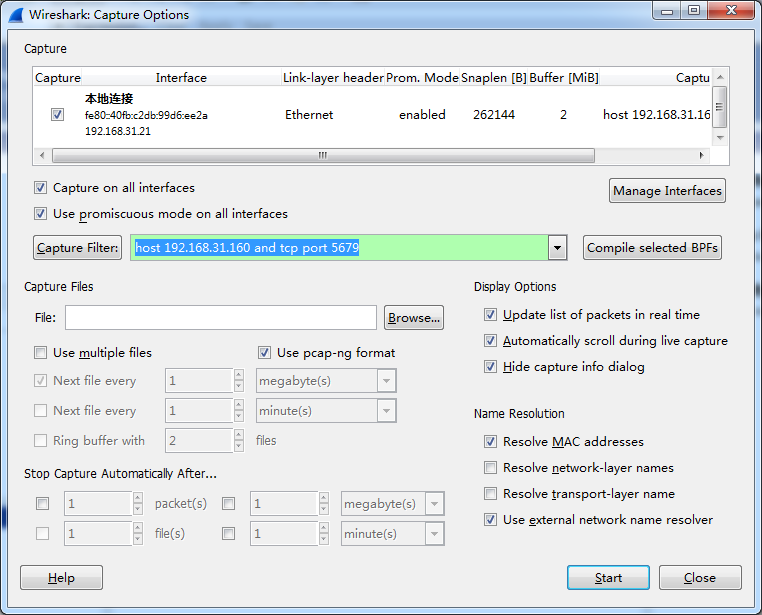
语句

recv\_chan.exchange\_declare(exchange='entry', exchange\_type='topic') #应为direct

错误

ChannelClosed: (406, "PRECONDITION\_FAILED - cannot redeclare exchange 'entry' in vhost 'XBCHEN' with different type, durable, internal or autodelete value")

## Wireshark抓包



host 192.168.31.160 and tcp port 5679