1. 服务器程序

这里使用Python3，需要import socket库和select库。

首先确定，需要用一个socket作为服务器的监听socket，用来监听是否有新的连接请求。一旦有连接，就把连接上的socket加入到一个list之中。同时把socket的ip地址作为Value，socket作为Key放入一个字典中。这是为了可以查询对应的socket的ip地址。

def \_\_init\_\_(self):

self.ConnectionList = []

self.SocketDict = {} #python初始化一个字典的方法

self.maxConnection = 10

self.receiveBufferSize = 4096

self.address = ("127.0.0.1“”)

self.sock = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

#

self.sock.setsockopt(socket.SOL\_SOCKET, socket.SO\_REUSEADDR, 1)

def start(self):

self.sock.bind(self.address) #绑定地址和端口

self.ConnectionList.append(self.sock) #加入socket列表

self.sock.listen(self.maxConnection) #开始监听

#...后面是死循环

由于每个socket都有可能接收到新的信息，所以这里我使用select方法，在每个循环里进行轮询。

while True:

readSockets, writeSockets, errorSockets = select.select(self.ConnectionList, [], [])

这样子会把该循环时收到消息的socket从list里面选择出来，然后对消息进行处理

for sock in readSockets:

if sock == self.sock: #如果是监听的socket收到消息，则建立一个新的连接

newClient, addr = self.sock.accept()

self.SocketDict[newClient] = addr

self.ConnectionList.append(newClient)

print("New connection from: " + addr[0])

self.broadcast("client: " + str(addr[0]) + " comes")

else:

try:

data = sock.recv(self.receiveBufferSize)

if data: #如果数据不为空

self.broadcast(self.SocketDict[sock][0] + ": \n" + data.decode())

except:

sock.close() #接受失败，则关闭连接。

self.ConnectionList.remove(sock)

self.broadcast("client: " + self.SocketDict[sock][0] + " leaves")

self.SocketDict.pop(sock)

2在本机上测试服务器

为了测试服务器，需要再写个客户端程序。为了方便，我还是使用Python。由于是Windows平台，所以不能用select([socket, stdin],[],[])的方法，所以只能采用多线程。

class MessageSender(threading.Thread):

def run(self):

while True:

message = input()

sock.send(message.encode())

这个线程用来监听输入， 主线程则监听收到的消息。

with socket.socket(socket.AF\_INET,socket.SOCK\_STREAM) as sock:

sock.connect((HOST,PORT))

readList = [sock]

MessageSender().start()

while True:

read, write, error = select.select(readList, [], []) #其实只有一个socket

for socket in read:

if socket == sock:

data = socket.recv(bufferSize).decode()

print(data)

开启一个服务器，开启一个客户端：

服务器Console：

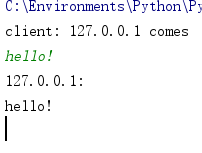
D:/MyDoc/Python/ZFSocketServer/TCPServer.py

New connection from: 127.0.0.1

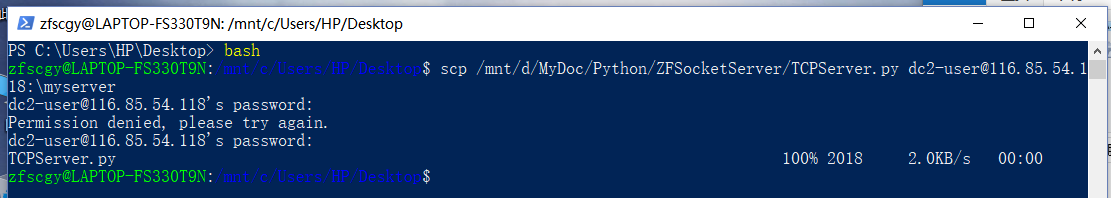
客户端Console：

D:/MyDoc/Python/ZFSocketServer/TestClient.py

client: 127.0.0.1 comes



如图，绿色的字是客户端输入，客户端输入后发送给服务器，服务器再广播发送回客户端，此时客户端才把收到的消息打印到屏幕上



使用Windows10 Powershell 里面的Linux子系统，用scp把本机的Py脚本传送到云服务器的指定目录。