

# 实验报告

1611736 钟腾

本次计算机网络实验要求完成一次关于UDP连接的服务器socket的python编程，完成编程后运行代码，先执行服务器代码使服务器处于等待接收状态，然后执行客户端代码向服务器发送10个udp包，并接收服务器返回的数据，输出报文类型序号发送时间以及各项RTT及其相关统计。根据要求，完成代码如下：

实验给出的UDPPingerServer.py:

```
1  # UDPPingerServer.py
2  # We will need the following module to generate randomized lost packets
3  import random
4  from socket import *
5
6  # Create a UDP socket
7  # Notice the use of SOCK_DGRAM for UDP packets
8  serverSocket = socket(AF_INET, SOCK_DGRAM)
9  # Assign IP address and port number to socket
10 serverSocket.bind(('10.41.51.25', 12000))
11
12 while True:
13     print('listening..')
14     # Generate random number in the range of 0 to 10
15     rand = random.randint(0, 10)
16     # Receive the client packet along with the address it is coming from
17     message, address = serverSocket.recvfrom(1024)
18     # Capitalize the message from the client
19     message = message.upper()
20     # If rand is less is than 4, we consider the packet lost and do not
    respond
21     if rand < 4:
22         continue
23     # Otherwise, the server responds
24     serverSocket.sendto(message, address)
25
```

此代码创建了一个serverSocket来监听来自客户端的收到的信息，当收到后产生0-10间的随机数，随机数小于4的不再发送到客户端，以此模拟丢包情况。

UDPPingerClient.py:

```

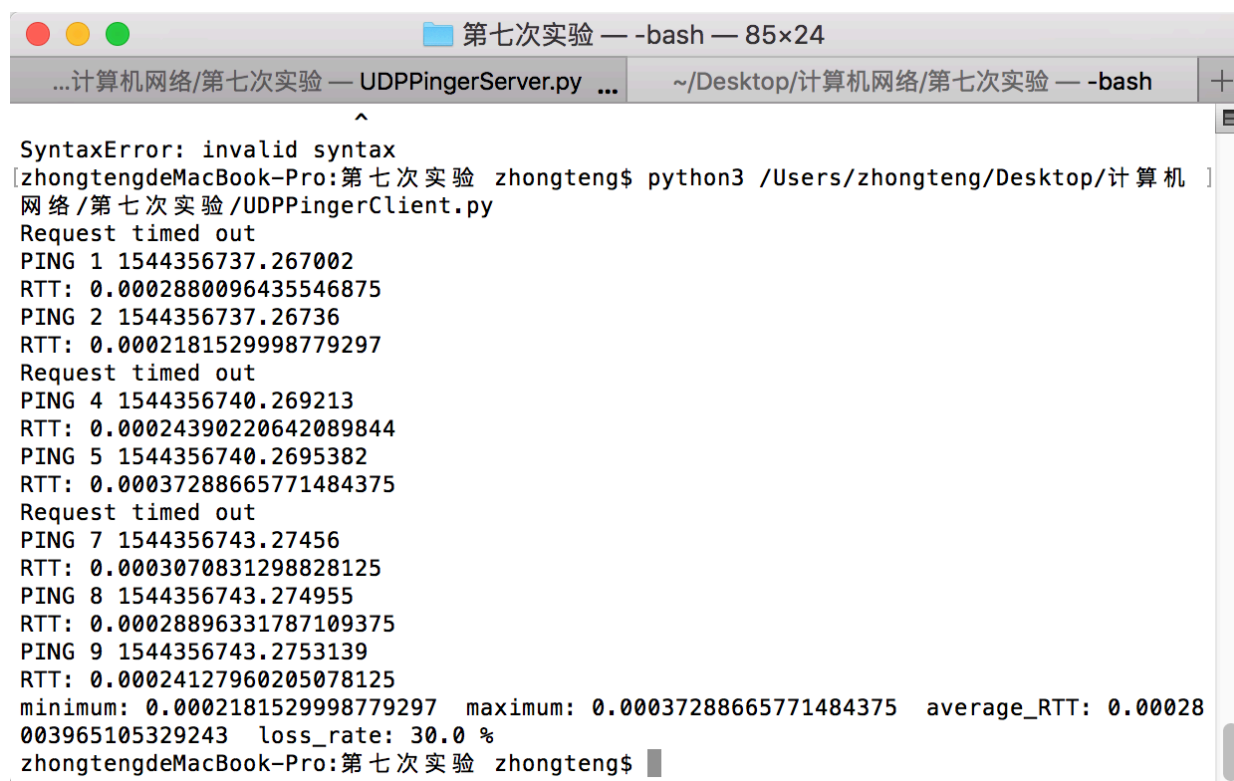
1  from socket import *
2  import time
3
4  clientSocket = socket(AF_INET, SOCK_DGRAM)
5  clientSocket.settimeout(3) #设置1秒的延迟
6
7  minimum = 10 #RTT最小值
8  maximum = 0 #RTT最大值
9  sum_RTT = 0 #RTT总值
10 average_RTT = 0 #平均RTT
11 loss_count = 0 #丢包数量
12
13 for i in range(0,10):
14     send_time = time.time() #记录发送时间
15     message = 'Ping '+str(i)+' '+str(send_time) #发送报文格式
16     clientSocket.sendto(message.encode('utf-8'),('10.41.51.25',12000)) #发送报
文
17
18     try:
19         count = 0
20         RTT = 0
21         received_message = clientSocket.recv(1024).decode('utf-8') #接收返回信
息
22         received_type,received_num= received_message.split()
23         [0],received_message.split()[1] #截取报文类型和报文序号
24         received_time = time.time() #记录接收时间
25         RTT = received_time - send_time #计算RTT
26         sum_RTT = sum_RTT+RTT #计算RTT总值
27         if maximum<RTT: #计算RTT最大值
28             maximum = RTT
29
30         if minimum>RTT: #计算RTT最小值
31             minimum = RTT
32         print(received_type,received_num,send_time) #打印报文
33         print("RTT:",RTT) #打印RTT
34
35     except:
36         print('Request timed out')
37         loss_count = loss_count+1 #统计丢包数量
38         continue
39
40 average_RTT = sum_RTT/(10-loss_count) #计算平均RTT
41 print("minimum:",minimum," maximum:",maximum," average_RTT:",average_RTT,"
42 loss_rate:",float(loss_count/10)*100,"%") #打印统计数据

```

首先引入socket包，建立clientSocket，确定好ip地址和端口号，然后开始发送十条消息到服务器，记录好发送消息的时间。

接收返回消息，收到对应序号的消息计算好RTT，把类型和序号及发送时间输出，丢包的输出Request timed out提示丢包。最后输出RTT的最小值、最大值、平均值和丢包率。

实验截图：



```
SyntaxError: invalid syntax
[zhongtengdeMacBook-Pro:第七次实验 zhongteng$ python3 /Users/zhongteng/Desktop/计算机
网络/第七次实验/UDPPingerClient.py
Request timed out
PING 1 1544356737.267002
RTT: 0.0002880096435546875
PING 2 1544356737.26736
RTT: 0.0002181529998779297
Request timed out
PING 4 1544356740.269213
RTT: 0.00024390220642089844
PING 5 1544356740.2695382
RTT: 0.00037288665771484375
Request timed out
PING 7 1544356743.27456
RTT: 0.0003070831298828125
PING 8 1544356743.274955
RTT: 0.00028896331787109375
PING 9 1544356743.2753139
RTT: 0.00024127960205078125
minimum: 0.0002181529998779297 maximum: 0.00037288665771484375 average_RTT: 0.00028
003965105329243 loss_rate: 30.0 %
zhongtengdeMacBook-Pro:第七次实验 zhongteng$
```

至此顺利完成本次实验要求，通过这次实验学会了python的socket网络编程和服务端客户端利用socket完成的UDP传输数据包的步骤，受益匪浅。