

تمرین اول

UDP Socket Programming



محمد امین شهیدی نشرودکلی
۹۶۵۲۲۱۷۷

زبان برنامه نویسی استفاده شده: پایتون

روال کار

کد ارسال شده دارای دو تابع اصلی می‌باشد که هرکدام شامل یک سوکت UDP است. یک تابع برای دریافت پیام‌ها و یک تابع برای ارسال.

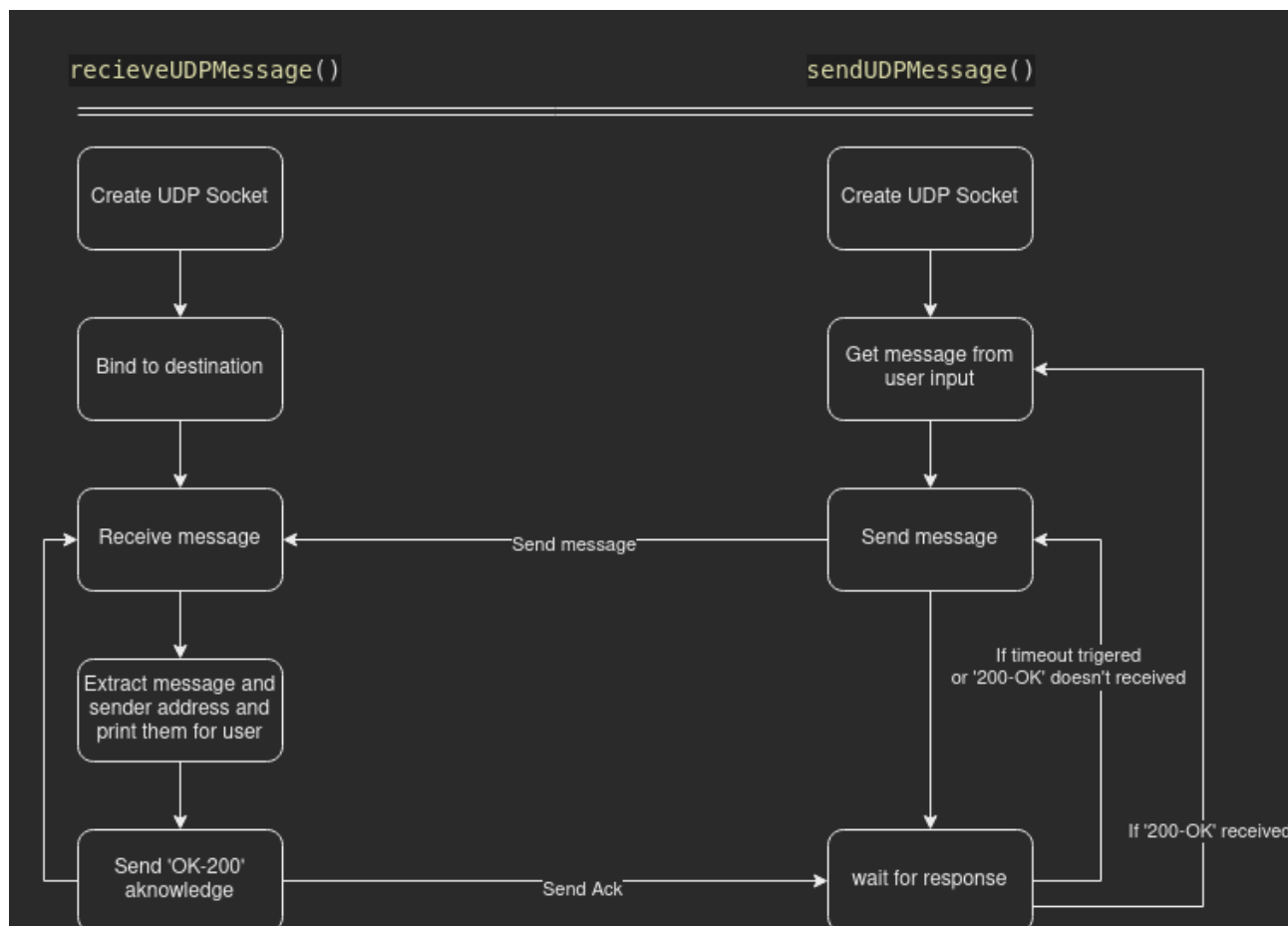
در انتهای کد دو رشته تعریف شده است که وظیفه هرکدام اجرای یکی از توابع (ارسال یا دریافت پیام) است. درواقع این برنامه هم یک برنامه سرور است و هم یک برنامه کلاینت زیرا قرار بر این است که بصورت p2p باشد. در ابتدا آدرس آی‌پی و شماره پورت کاربر از وی در ورودی گرفته می‌شود و سپس آدرس آی‌پی و شماره پورت مقصد از وی خواسته می‌شود.

تضمین تحویل بسته‌ها

برای این منظور از روشی مشابه با TCP Acknowledge استفاده شده است. اگر رشته مربوط به ارسال پیام سوکت پیامی را ارسال کند اما پاسخ 'OK-200' را در جواب دریافت نکند، پس از ۵ ثانیه دوباره پیام را می‌فرستد. پیام تأیید که 'OK-200' می‌باشد از نشانه‌های وضعیت HTTP ایده گرفته شده است.

رشته دریافت‌کننده نیز اگر پیامی دریافت کند، در پاسخ پیام 'OK-200' را می‌فرستد تا فرستنده بداند پیامش به درستی دریافت شده‌است.

شکل زیر روال کار برنامه را نشان می‌دهد:



شرح قسمت‌های مختلف کد:

خط ۱ تا ۸

```
1 import socket
2 import threading
3
4 localIP = str(input('Enter your local IP: '))
5 localPort = int(input('Enter your desired port number(integer): '))
6
7 targetIP = str(input('Enter your destination IP: '))
8 targetPort = int(input('Enter your destination port number(integer): '))
9
```

در خط ۱ و ۲ کتابخانه‌های مربوط به Socket Programming و Multithreading را افزوده ایم. در خطوط ۴ تا ۸ به ترتیب ورودی‌های زیر را از کاربر می‌گیریم:

خط ۴) IP کاربر

خط ۵) شماره پورت کاربر

خط ۷) IP مقصد

خط ۸) شماره پورت مقصد

خط ۱۰ تا ۳۰

```
10 def recieveUDPMessage():
11     bufferSize = 1024
12     UDPReceiverSocket = socket.socket(family=socket.AF_INET, type=socket.SOCK_DGRAM)
13     UDPReceiverSocket.bind((localIP, localPort))
14     while(True):
15         messageAddressPair = UDPReceiverSocket.recvfrom(bufferSize)
16         message = messageAddressPair[0]
17         addressPair = messageAddressPair[1]
18
19         clientMsg = 'Message: {}'.format(str(message))
20         clientIP = 'Client IP Address & Port: {}'.format(addressPair)
21
22         #Show message details to user
23         print('---NEW MESSAGE!-----')
24         print(clientMsg)
25         print(clientIP)
26         print('-----\nEnter your message to send: \n')
27
28         #Sending a reply to client
29         if message != b'200-OK':
30             UDPReceiverSocket.sendto(str.encode('200-OK'), addressPair)
```

در این قسمت تابعی که پیام‌ها را دریافت می‌کند تعریف شده است. در خط ۱۲ ساین بافر دریافت را تعیین کرده‌ایم. در خط ۱۲ و ۱۳ به ترتیب سوکت UDP را تعریف کرده و آن را به آدرس IP و پورت Peer مقابل Bind کرده ایم.

در خط ۱۵ تا ۱۷، پیام دریافت شده و مشخصات سوکت فرستنده و محتوای پیام از آن استخراج شده است.

در خط ۲۳ تا ۲۶ محتوای پیام برای کاربر چاپ می‌شود.

اگر پیامی که دریافت می‌شود محتوای آن رشته 'OK-200' باشد، یعنی این پیام ACK پیامی بوده که خودمان در

گذشته فرستاده‌ایم و نیازی نیست به آن جواب دهیم. در غیر این صورت در خط ۳۰، رشته 'OK-200' را فرستاده

ایم تا فرستنده بداند پیامش به دست گیرنده رسیده است. اینکه به چه کسی این پیام را بفرستیم در خط ۱۵ تا ۱۷ مشخص شد زیرا در آنجا مشخصات فرستنده پیام (شماره پورت و آی‌پی) استخراج شد.

```

32 def sendUDPMessage():
33     bufferSize = 1024
34     destinationAddressPair = (targetIP, targetPort)
35     UDPSenderSocket = socket.socket(family=socket.AF_INET, type=socket.SOCK_DGRAM)
36     while(True):
37         msg = str(input('Enter your message to send: \n'))
38         encodedMsg = str.encode(msg)
39         #Send message and recieve ack
40         while(True):
41             UDPSenderSocket.settimeout(5)
42             try:
43                 UDPSenderSocket.sendto(encodedMsg, destinationAddressPair)
44                 msgFromDestination = UDPSenderSocket.recvfrom(bufferSize)
45                 msg = msgFromDestination[0]
46                 if msg != b'200-OK':
47                     print('Failed to send. Retrying...')
48                     continue
49                 print('-----Successfully received!-----\n')
50                 break
51             except:
52                 print('Failed to send. Retrying...')

```

در این قسمت تابع ارسال کننده تعریف شده است.
 در خط ۳۳ اندازه بافر ارسال را تعریف کرده ایم.
 در خط ۳۴ مشخصات Peer مقصد را به صورت یک Tuple مشخص کرده ایم تا برای ارسال از آن استفاده کنیم.
 در خط ۳۵ سوکت ارسال را مشخص کرده ایم.
 حلقه ای که از خط ۴۰ تا ۵۲ تعریف شده است کاربرد زیر را دارد:
 با تعریف یک تایمر ۵ ثانیه ای برای سوکت مشخص کرده ایم که اگر بعد از ارسال پیام (که در خط ۳۷ از کاربر به عنوان ورودی گرفته شده است) هیچ پاسخی به شکل 'OK-200' دریافت نشد عبارت خط ۴۷ را چاپ کن و برای ارسال پیام دوباره تلاش کن.
 اگر پاسخ 'OK-200' دریافت شد، عبارت خط ۴۹ را چاپ کن. همچنین اگر پاسخی دریافت شد اما محتوای آن برابر با 'OK-200' نبود دوباره تلاش کن.

```

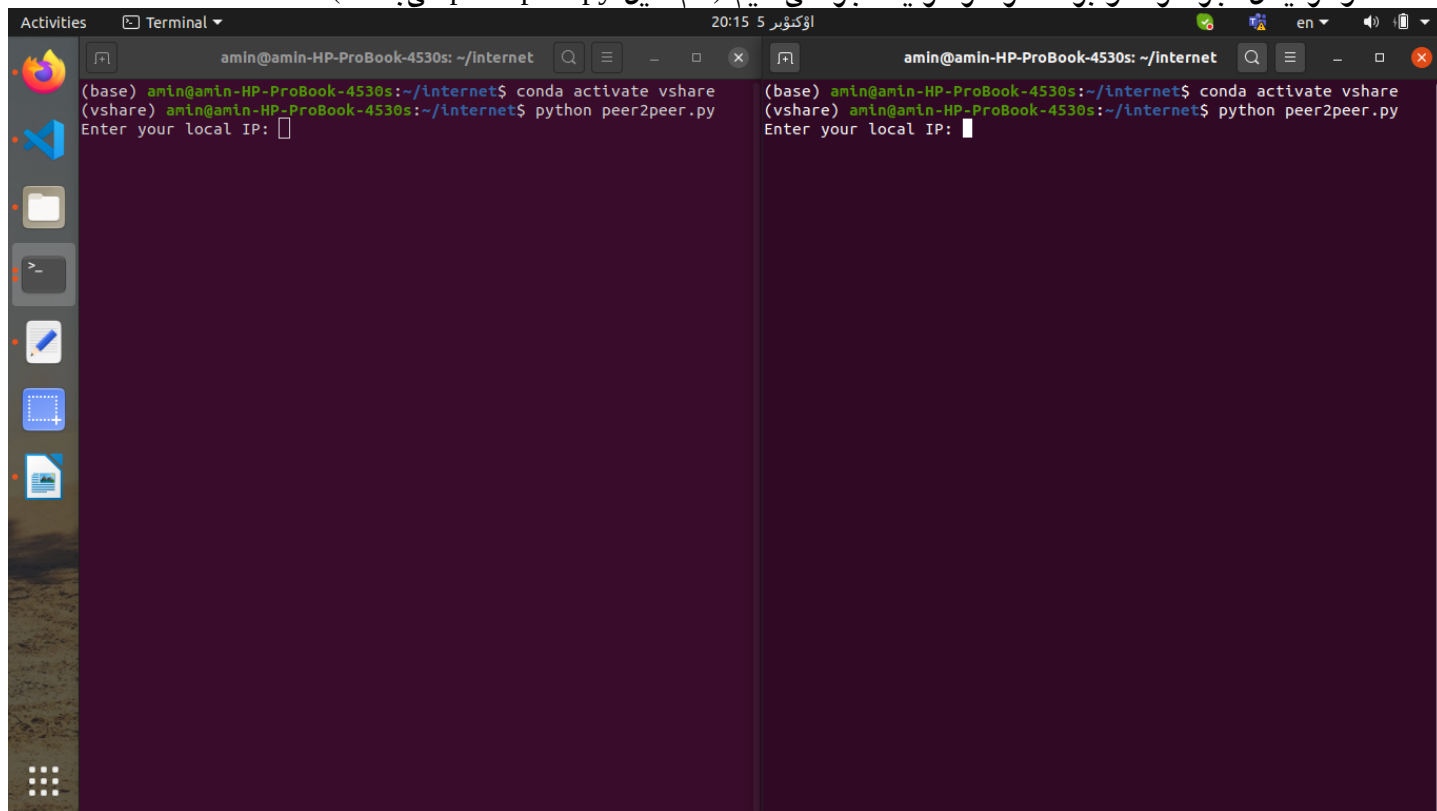
54 receive = threading.Thread(target = recieveUDPMessage)
55 send = threading.Thread(target = sendUDPMessage)
56
57 receive.start()
58 send.start()

```

در خط ۵۴ و ۵۵ دو رشته مجزا به تعریف کرده ایم که اولی تابع دریافت و دومی تابع ارسال را اجرا می کند.
 در خط ۵۷ و ۵۸ نیز این دو رشته را به اجرا در می آوریم.

اجرا و خروجی

برای اینکه یک ارتباط p2p نشان بدهیم، برنامه را در دو ترمینال مختلف و به موازات هم اجرا می‌کنیم. هر دو ترمینال قادر خواهند بود که با یکدیگر به تبادل پیام بپردازند. برای این کار از سیستم عامل اوبونتو و پایتون ۳.۶ استفاده می‌کنیم. دو ترمینال اجرا کرده و برنامه را در هر یک اجرا می‌کنیم. (نام فایل peer2peer.py می‌باشد).



```
(base) amin@amin-HP-ProBook-4530s: ~/internet
(vshare) amin@amin-HP-ProBook-4530s:~/internet$ python peer2peer.py
Enter your local IP: 
```

```
(base) amin@amin-HP-ProBook-4530s: ~/internet
(vshare) amin@amin-HP-ProBook-4530s:~/internet$ python peer2peer.py
Enter your local IP: 
```

همانطور که مشاهده می‌شود پس اجرا برنامه مشخصات مبدأ و مقصد (پورت و آی‌پی) را درخواست می‌کند. از آنجایی که هر دو peer روی یک سیستم در حال اجرا هستند پس آی‌پی هر دو ۱۲۷.۰.۰.۱ می‌باشد اما شماره پورت‌های مختلفی باید داشته باشند. به peer در ترمینال سمت چپ شماره پورت ۲۰۰۰۱ و سمت راست ۲۰۰۰۲ را اختصاص می‌دهیم و آدرس آی‌پی هر دو را ۱۲۷.۰.۰.۱ وارد می‌کنیم. سپس می‌توانیم به تبادل پیام بپردازیم. همانطور که در شکل زیر مشخص است می‌توانیم هم‌اکنون به نوشتن و ارسال پیام بپردازیم. ابتدا از ترمینال سمت چپ پیامی ارسال می‌کنیم.

```
(base) amin@amin-HP-ProBook-4530s: ~/internet
(vshare) amin@amin-HP-ProBook-4530s:~/internet$ python peer2peer.py
Enter your local IP: 127.0.0.1
Enter your desired port number(integer): 20001
Enter your destination IP: 127.0.0.1
Enter your destination port number(integer): 20002
Enter your message to send:

(base) amin@amin-HP-ProBook-4530s: ~/internet
(vshare) amin@amin-HP-ProBook-4530s:~/internet$ python peer2peer.py
Enter your local IP: 127.0.0.1
Enter your desired port number(integer): 20002
Enter your destination IP: 127.0.0.1
Enter your destination port number(integer): 20001
Enter your message to send:
```

همانطور که در عکس زیر مشاهده می‌شود پیام به درستی در سمت راست دریافت و نمایش داده شد. در سمت چپ نیز پیامی مبنی بر موفقیت آمیز بودن ارسال دیده می‌شود.

```
(base) amin@amin-HP-ProBook-4530s: ~/internet
(vshare) amin@amin-HP-ProBook-4530s:~/internet$ python peer2peer.py
Enter your local IP: 127.0.0.1
Enter your desired port number(integer): 20001
Enter your destination IP: 127.0.0.1
Enter your destination port number(integer): 20002
Enter your message to send:
Hello!!! This is a message from (127.0.0.1, 20002).
-----Successfully received!-----
Enter your message to send:

(base) amin@amin-HP-ProBook-4530s: ~/internet
(vshare) amin@amin-HP-ProBook-4530s:~/internet$ python peer2peer.py
Enter your local IP: 127.0.0.1
Enter your desired port number(integer): 20002
Enter your destination IP: 127.0.0.1
Enter your destination port number(integer): 20001
Enter your message to send:
---NEW MESSAGE!-----
Message: b'Hello!!! This is a message from (127.0.0.1, 20002).'
Client IP Address & Port: ('127.0.0.1', 48450)
Enter your message to send:
```

حال همین عمل را از راست به چپ تکرار می‌کنیم اما اینبار، قبل از ارسال پیام، برنامه سمت چپ را با `ctrl+c` متوقف می‌کنیم. همانطور که مشاهده می‌شود، برنامه سمت راست متوجه مشکل پیش آمده می‌شود و دوباره تلاش به ارسال پیام می‌کند.

The image displays two side-by-side terminal windows from a Linux system, showing a Python script for peer-to-peer communication using the `peer2peer.py` module. The user is `amin` on a machine named `amin@amin-HP-ProBook-4530s`.

Left Terminal Window (Sender):

```
(base) amin@amin-HP-ProBook-4530s:~/internet$ conda activate vshare
(vshare) amin@amin-HP-ProBook-4530s:~/internet$ python peer2peer.py
Enter your local IP: 127.0.0.1
Enter your destination IP: 127.0.0.1
Enter your destination port number(integer): 20002
Enter your message to send:
Hello!!! This is a message from (127.0.0.1, 20002).
-----Successfully received!-----

Enter your message to send:
^CException ignored in: <module 'threading' from '/home/amin/anaconda3
/envs/vshare/lib/python3.6/threading.py'>
Traceback (most recent call last):
  File "/home/amin/anaconda3/envs/vshare/lib/python3.6/threading.py",
line 1294, in _shutdown
    t.join()
  File "/home/amin/anaconda3/envs/vshare/lib/python3.6/threading.py",
line 1056, in join
    self._wait_for_tstate_lock()
  File "/home/amin/anaconda3/envs/vshare/lib/python3.6/threading.py",
line 1072, in _wait_for_tstate_lock
    elif lock.acquire(block, timeout):
KeyboardInterrupt
(vshare) amin@amin-HP-ProBook-4530s:~/internet$
```

Right Terminal Window (Receiver):

```
(base) amin@amin-HP-ProBook-4530s:~/internet$ conda activate vshare
(vshare) amin@amin-HP-ProBook-4530s:~/internet$ python peer2peer.py
Enter your local IP: 127.0.0.1
Enter your destination IP: 127.0.0.1
Enter your destination port number(integer): 20001
Enter your message to send:
---NEW MESSAGE!-----
Message: b'Hello!!! This is a message from (127.0.0.1, 20002).'
Client IP Address & Port: ('127.0.0.1', 48450)
-----

Enter your message to send:

this is a test message!!
Failed to send. Retrying...
Failed to send. Retrying...
Failed to send. Retrying...
Failed to send. Retrying...
```

حال دوباره در ترمینال سمت چپ برنامه را اجرا می‌کنیم. مشاهده می‌شود که در این صورت نتیجه تلاش‌های مکرر برنامه در ترمینال سمت راست موفقیت آمیز می‌شود زیرا در سمت راست پیام دریافت می‌شود و برای آن ack ارسال می‌شود. بعد از دریافت ack برنامه سمت راست دیگر پیام را نمی‌فرستد زیرا می‌داند پیام به مقصد رسیده است.

The image displays two side-by-side terminal windows from a Linux environment. Both windows show the execution of a Python script named `peer2peer.py` within a virtual environment (`base`) on a system named `amin@amin-HP-ProBook-4530s`.

Left Terminal Window:

```
(base) amin@amin-HP-ProBook-4530s:~/Internet$ conda activate vshare
(vshare) amin@amin-HP-ProBook-4530s:~/Internet$ python peer2peer.py
Enter your local IP: 127.0.0.1
Enter your desired port number(integer): 20001
Enter your destination IP: 127.0.0.1
Enter your destination port number(integer): 20002
Enter your message to send:
Hello!!! This is a message from (127.0.0.1, 20002).
-----Successfully received!-----

Enter your message to send:
^CException ignored in: <module 'threading' from '/home/amin/anaconda3
/envs/vshare/lib/python3.6/threading.py'>
Traceback (most recent call last):
  File "/home/amin/anaconda3/envs/vshare/lib/python3.6/threading.py",
line 1294, in _shutdown
    t.join()
  File "/home/amin/anaconda3/envs/vshare/lib/python3.6/threading.py",
line 1056, in join
    self._wait_for_tstate_lock()
  File "/home/amin/anaconda3/envs/vshare/lib/python3.6/threading.py",
line 1072, in _wait_for_tstate_lock
    elif lock.acquire(block, timeout):
KeyboardInterrupt
(vshare) amin@amin-HP-ProBook-4530s:~/Internet$ python peer2peer.py
Enter your local IP: 127.0.0.1
Enter your desired port number(integer): 20001
Enter your destination IP: 127.0.0.1
Enter your destination port number(integer): 20002
Enter your message to send:
-----NEW MESSAGE!-----
Message: b'this is a test message!!'
Client IP Address & Port: ('127.0.0.1', 39776)
-----
Enter your message to send:
```

Right Terminal Window:

```
(base) amin@amin-HP-ProBook-4530s:~/Internet$ conda activate vshare
(vshare) amin@amin-HP-ProBook-4530s:~/Internet$ python peer2peer.py
Enter your local IP: 127.0.0.1
Enter your desired port number(integer): 20002
Enter your destination IP: 127.0.0.1
Enter your destination port number(integer): 20001
Enter your message to send:
-----NEW MESSAGE!-----
Message: b'Hello!!! This is a message from (127.0.0.1, 20002).'
Client IP Address & Port: ('127.0.0.1', 48450)
-----
Enter your message to send:

this is a test message!!
Failed to send. Retrying...
Failed to send. Retrying...
Failed to send. Retrying...
Failed to send. Retrying...
Failed to send. Retrying...
Failed to send. Retrying...
Failed to send. Retrying...
Failed to send. Retrying...
Failed to send. Retrying...
Failed to send. Retrying...
Failed to send. Retrying...
Failed to send. Retrying...
-----Successfully received!-----

Enter your message to send:
]
```