Network Lab Exam Question 3-B

Albin Antony
Roll No:10
TVE16CS010

25 April 2019

1 Calender Server

1.1 Problem Statement

The idea is to create a calendar server based on TCP. You need to define a protocol (i.e., define how messages are formatted and transmitted), implement it, and write a client and a server that communicate using your new connection-oriented protocol.
Your protocol should support the following functionalities:

- 1. Add a new calendar event.
- 2. Remove a calendar event.
- 3. Update an existing calendar event.
- 4. Get the events for a specific time or time range.

There are no other requirements regarding the protocol, i.e., you are free to decide the protocol details, e.g., the message structure and content for the client/server communication.

1.2 Theory

1.2.1 TCP

TCP (Transmission Control Protocol) is a standard that defines how to establish and maintain a network conversation via which application programs can exchange data. TCP works with the Internet Protocol (IP), which defines how computers send packets of data to each other. Together, TCP and IP are the basic rules defining the Internet. TCP is defined by the Internet Engineering Task Force (IETF) in the Request for Comment (RFC) standards document number 793.

1.2.2 Client, Server and Socket

• Server-A server is a software that waits for client requests and serves or processes them accordingly.
• Client- a client is requester of this service. A client program request for some resources to the server and server responds to that request.
• Socket- Socket is the endpoint of a bidirectional communications channel between server and client. Sockets may communicate within a process, between processes on the same machine, or between processes on different machines. For any communication with a remote program, we have to connect through a socket port.

1.3 Algorithm

Algorithm 1 Algorithm for Server

```
CREATE Socket
BIND to IP PORT
Listen on PORT
while True:
    RECIEVE data from Server
    split data into list tok
    if tok[1] is 'add'
        add the event to a calender list
        send 'Successfully added' RESPONSE
    if tok[1] is 'remove'
        remove that users events from calender list
        send 'Successfully removed' RESPONSE
    if tok[1] is 'update'
        remove that users event of that date from calender list
        add the new event to calender list
        send 'Successfully upadted' RESPONSE
    if tok[1] is 'get'
        find event with that user and event date
        send event as RESPONSE
```

The server receives arguments of client as a string from client.The string is split in a list.For add this list is appended to a calender list if does not exist in the list already. For deleting the calender list is searched to find the event if found it is removed.For update it is removed from calender list then the new venet is added.For get the event is searched based on the username and date, if found the even is send as response back to the client.

For add, remove, update response is send as succes if not error occurs else a Duplicate entry error is sent.

Algorithm 2 Algorithm for Client

```
CREATE Socket
Connect to addr,port in ARGV
send ARGS to SERVER
```

1.4 Program

1.4.1 Server

```
#################Calender-Server
#Python2.7

###AUTHOR: Albin Antony
###Roll No:TVE16CS010

import socket

s=socket.socket()

s.bind(('',8080))

s.listen(5)
cal=[]
flag=0
while True:
        c,addr=s.accept()
        print "Connected"
        str=c.recv(1024)
        tok=str.split()


        if(tok[1]=='add'):   #Checking which method is used
                print "Adding Event "+tok[5]
                for event in cal:
                        if tok[2] in event and tok[0] in
                            event :  #checking for duplicate
                            entry
                                flag=1
                                break
                if(flag==1):
                        c.send("Duplicate entry")
                else:
                        cal.append(tok)                #adding
                             event to the list
                        c.send(tok[5]+" added to calender")

                flag=0
                print cal
        elif(tok[1]=='remove'):
                print "Removing Event "+tok[2]
                for event in cal:
```

```
                        if tok[2] in event and tok[0] in
                            event: #remove only if username
                            and date match
                                cal.remove(event)
                                c.send(tok[2]+" removed")
                                break
                print cal
        elif(tok[1]=='update'):
                print "Updating Event "+tok[5]
                for event in cal:
                        if tok[2] in event and tok[0] in
                            event: #update only if username
                            and date match
                                flag=1
                                cal.remove(event)
                                break
                if(flag==1):
                        cal.append(tok)
                        c.send(tok[5]+" updated")
                else:
                        print "No found"

                flag=0
                print cal
        elif(tok[1]=='get'):
                print "Getting Event "+tok[2]
                for event in cal:
                        if tok[2] in event and tok[0] in
                            event:  #get only if username and
                             date match
                                str=''
                                for i in event:
                                        str=str+ " "+i
                                c.send(str)
                                break
                print cal

        if(str=='quit'):
                s.close()
                break
        c.close()
```

1.4.2 Client

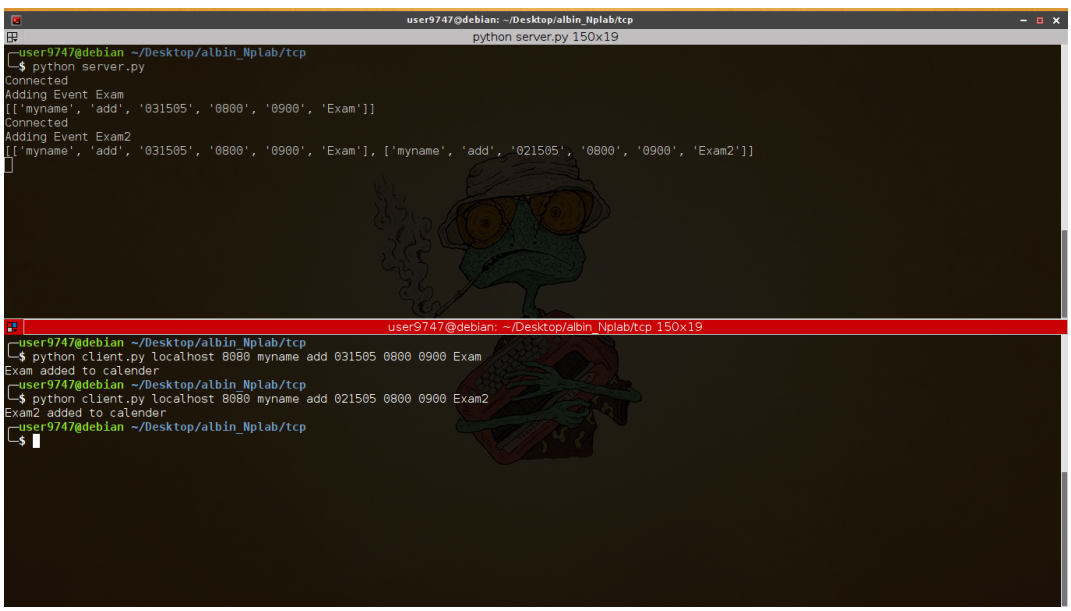
```
################Calender-Client
#Python2.7

###AUTHOR: Albin Antony
###Roll No:TVE16CS010

import socket
import sys
s=socket.socket()

s.connect((sys.argv[1],int(sys.argv[2])))



#Below code sends data to server based on the arguments got

if(sys.argv[4]== 'add'):
        s.send(sys.argv[3]+" "+sys.argv[4]+" "+sys.argv[5]+"
            "+sys.argv[6]+" "+sys.argv[7]+" "+sys.argv[8])
elif(sys.argv[4]== 'remove'):
        s.send(sys.argv[3]+" "+sys.argv[4]+" "+sys.argv[5])
if(sys.argv[4]== 'update'):
        s.send(sys.argv[3]+" "+sys.argv[4]+" "+sys.argv[5]+"
            "+sys.argv[6]+" "+sys.argv[7]+" "+sys.argv[8])
if(sys.argv[4]== 'get'):
        s.send(sys.argv[3]+" "+sys.argv[4]+" "+sys.argv[5])

print s.recv(1024)
s.close
```

1.4.3 To run the program:

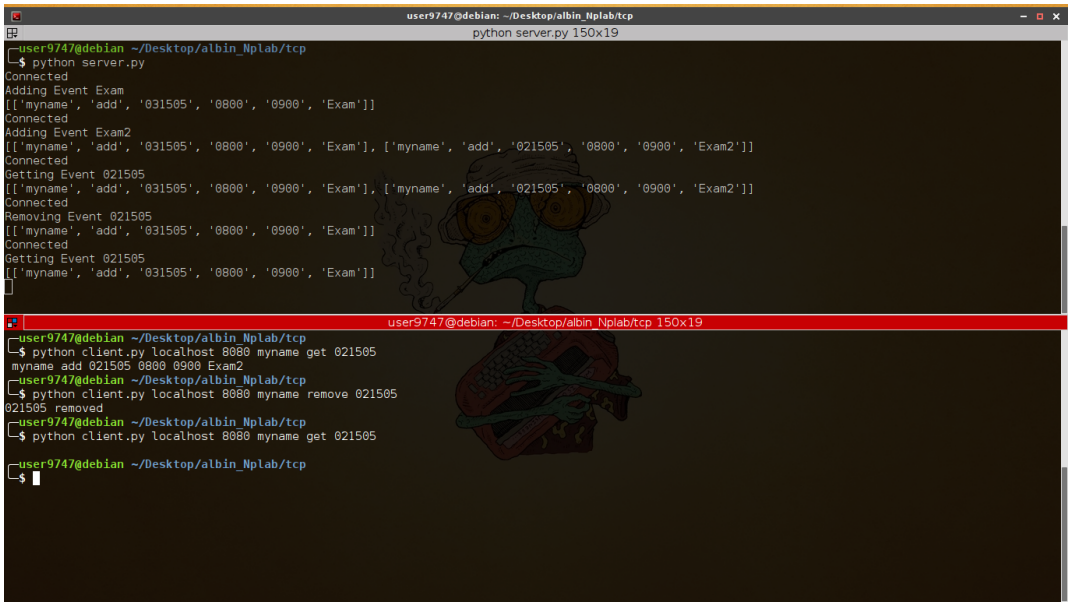
```
python server.py
python client.py hostname port [username] [action] [date] [time] [time] [Event]
```

1.5 Output

1.5.1 Test case 1



1.5.2 Test case 2



1.5.3 Test case 3