

ELL 402 Assignment 1

Network Programming Using Internet Sockets

Ritwik Chakravarti, 2016EE10432

Nalawade Sarvesh Bharat, 2016EE10442

25.01.2019

OBJECTIVE

Writing a client-server program where clients (instructor or students of a class) access the server (storing students' marks in 5 subjects out of 100) for information about marks in a semester examination.

1. Client will connect to server and login through username and password pre-stored on server. Server will refuse connection without proper authentication.
2. If client is logged on using 'instructor' as username, it will have access to marks of all the students in the class.
3. If client is any other user '<username>' (i.e. client is student) it will have access to his/her marks only.
4. Client (student) should be able to get information about:
 - a. His/her marks in each subject
 - b. Aggregate percentage
 - c. Subjects with maximum and minimum marks
5. Client (instructor) should be able to get information about:
 - a. Marks (individual and aggregate percentage) of each student
 - b. Class average
 - c. Number of students failed (passing percentage 33.33%) in each subject
 - d. Name of best and worst performing students
 - e. Instructor can update the marks of any student if he/she finds a bug (or need for correction). Therefore, create a menu having option 'Update' for 'Instructor' login to update marks of a particular student in a subject.

6. Create 'student_marks' file that contains marks of each student and is accessed by server for responding to client queries.
7. Create 'user_pass' file to hold data for usernames and passwords (with at least 20 users). This file is accessed by server for authentication
8. Create menu to select required information from client, either at client side or server side.
9. Using Wireshark, analyze packet size and frame size in different TCP/IP layers. Also trace the communication path between client and server machines, and find the number of hops used for communications. Comment on all the observations.

INTRODUCTION

For the purpose of this assignment, python was used due the large number of existing libraries and a high level of abstraction. The Twisted Library was central to the project. It is an event-driven framework which supports TCP, IP. The separation between the logical protocols and stream-based semantics was appreciated since it allowed for an easier programming experience.

The code for both the client and server are fairly simple, and follow a basic idea: create certain tags, similar to a "header" for each message, and using the uniqueness of the tags, create a large nested if-else structure.

The usage of the framework classes are:

1. Echofactory : This class helps us to build the protocol. It hides all the details of the data streams, sockets, etc. in a class. The reactor uses this class to respond to events.
2. Echoclient : Allows for branching, sending messages, etc. from the Client side. While there are other methods, for a project of this simplicity, adding a few functions to this class was a simple way to establish an event-driven client.
3. Reactor : Reactor is simply an object, which waits for some event to occur, and responds accordingly.
4. Echo : At the server side, this class was wherein the backend code based on the client's input was returned. It serves a similar purpose as Echoclient.

Wireshark was used to analyse the data exchange between the client and the server. At the node on which it is running, Wireshark gives a detailed view of the packets sent and

received; showing their protocol, destination IP, data length, TCP/IP flags, etc.

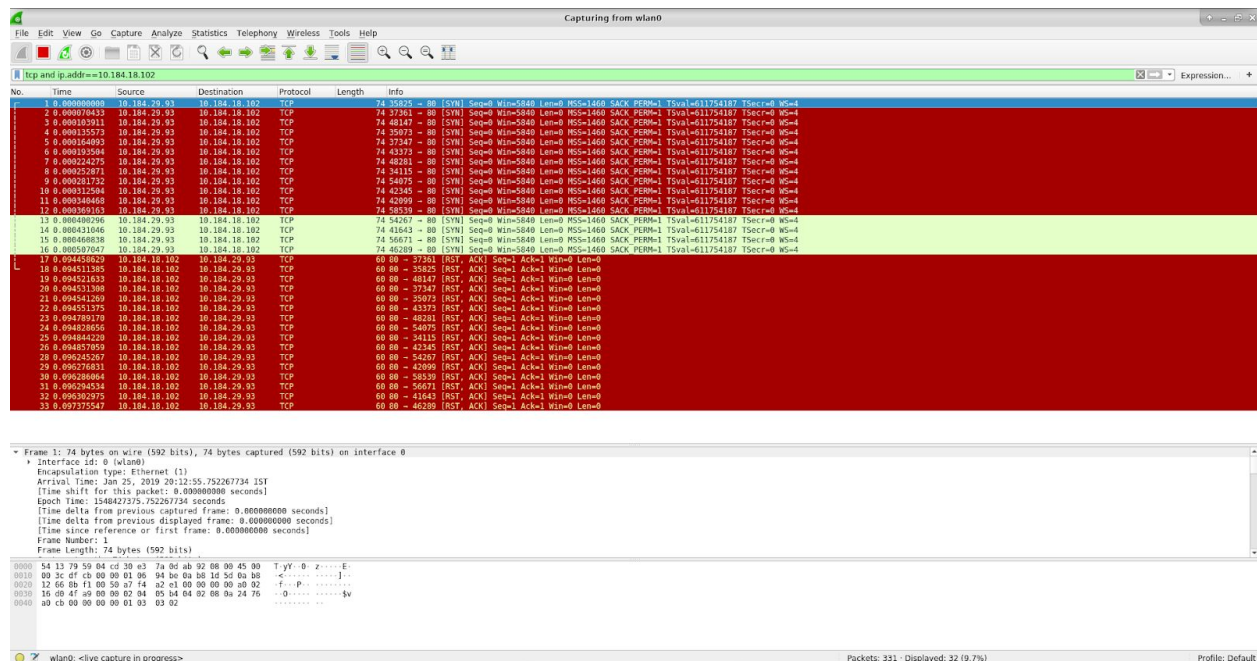
PROGRAM RUN COMMANDS

1. Install Libraries: `pip install Twisted numpy`
2. Run Server: `python3 Server.py`
3. Run Client: `python Client.py <server-ip-address>`

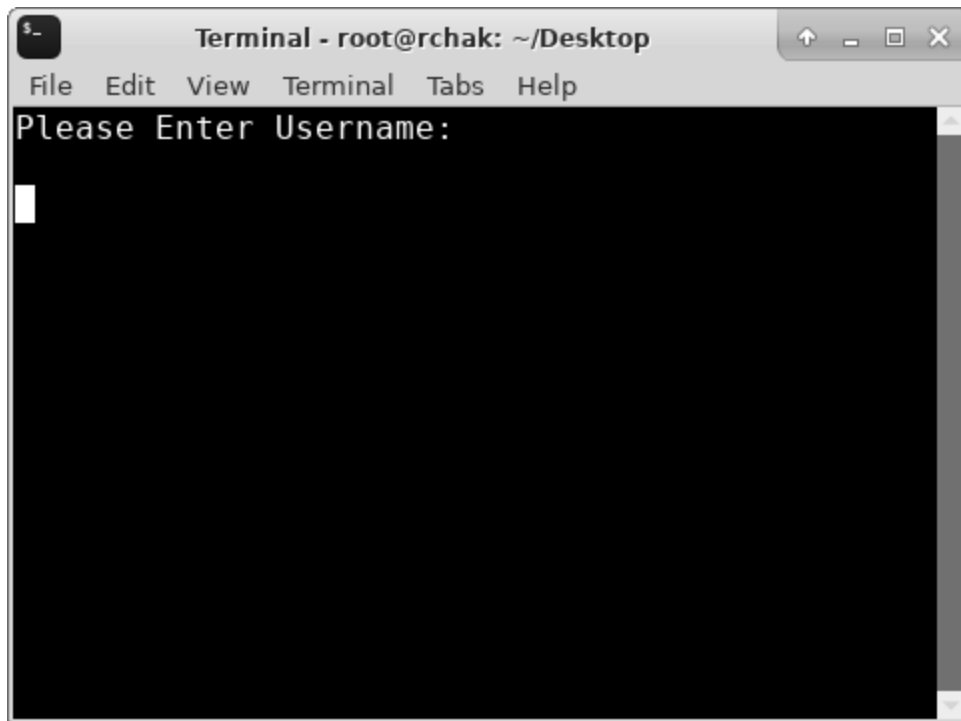
SCREENSHOTS of INTERFACE

```
root@rchak:~# traceroute --tcp 10.184.18.102
traceroute to 10.184.18.102 (10.184.18.102), 30 hops max, 60 byte packets
 1  10.184.18.102 (10.184.18.102)  94.549 ms  94.398 ms  94.426 ms
```

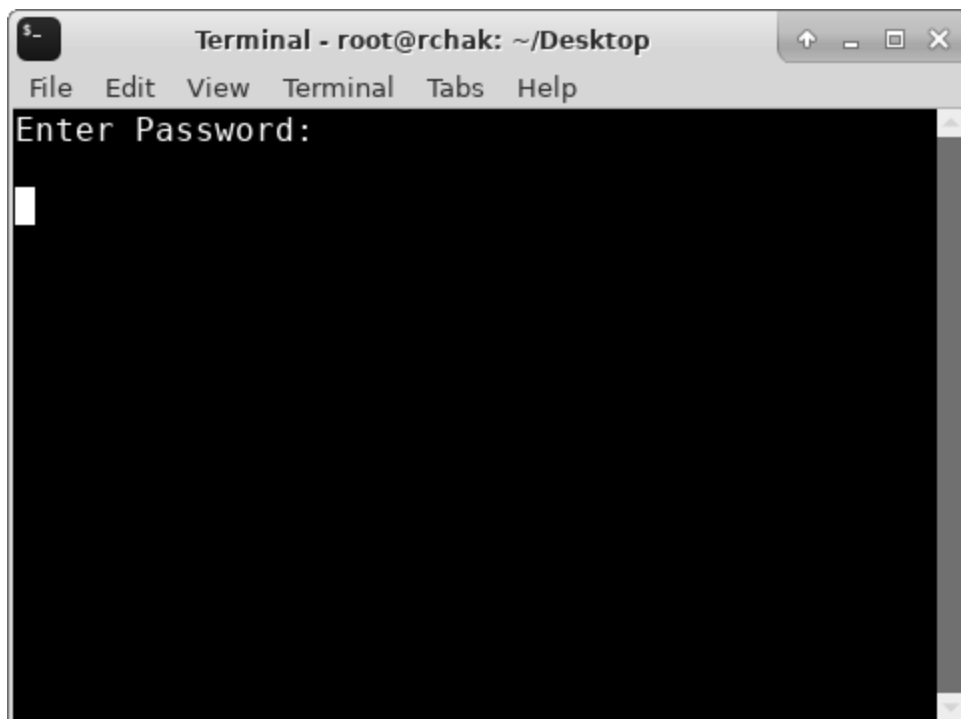
traceroute on the client's machine to the server's IP address.



Wireshark showing the TCP SYN packets used by the client for the traceroute call



Asking for username



After username, asking for password

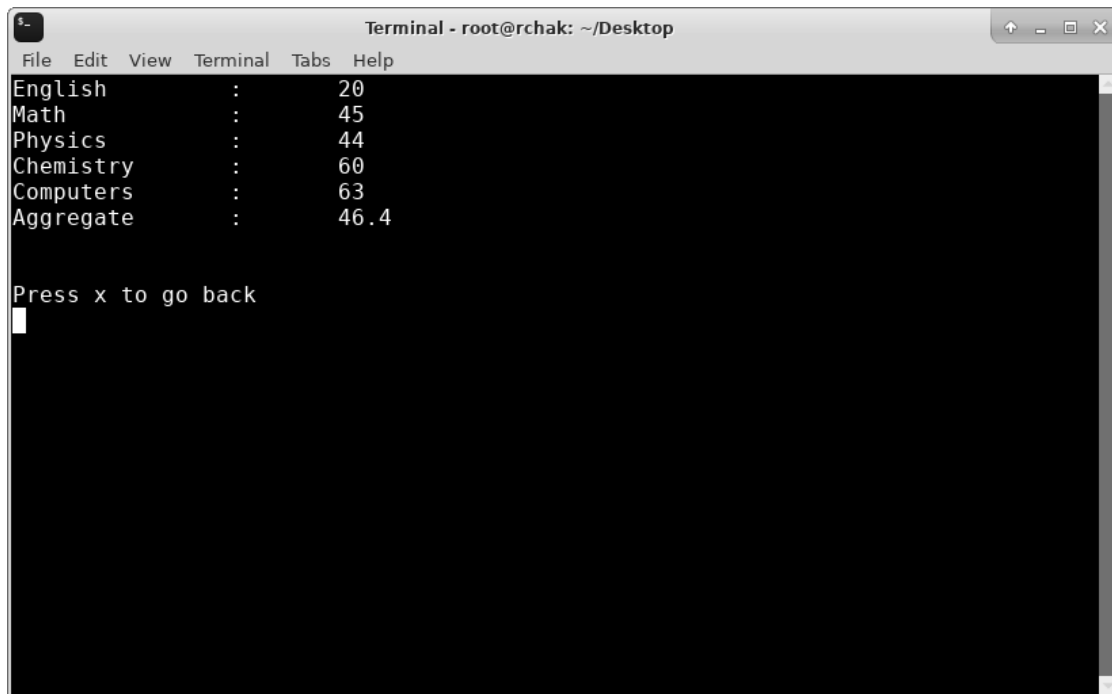
```
Terminal - root@rchak: ~/Desktop
File Edit View Terminal Tabs Help
Welcome Instructor
Choose Option:
1. Show Student Grades
2. Show Class Average
3. Number of Failing Students
4. Performance Wise Data
5. Update Marks

Press x to logout
█
```

Instructor Menu

```
Terminal - root@rchak: ~/Desktop
File Edit View Terminal Tabs Help
Enter Student Name
█
```

After asking to show student's grades, enter their name

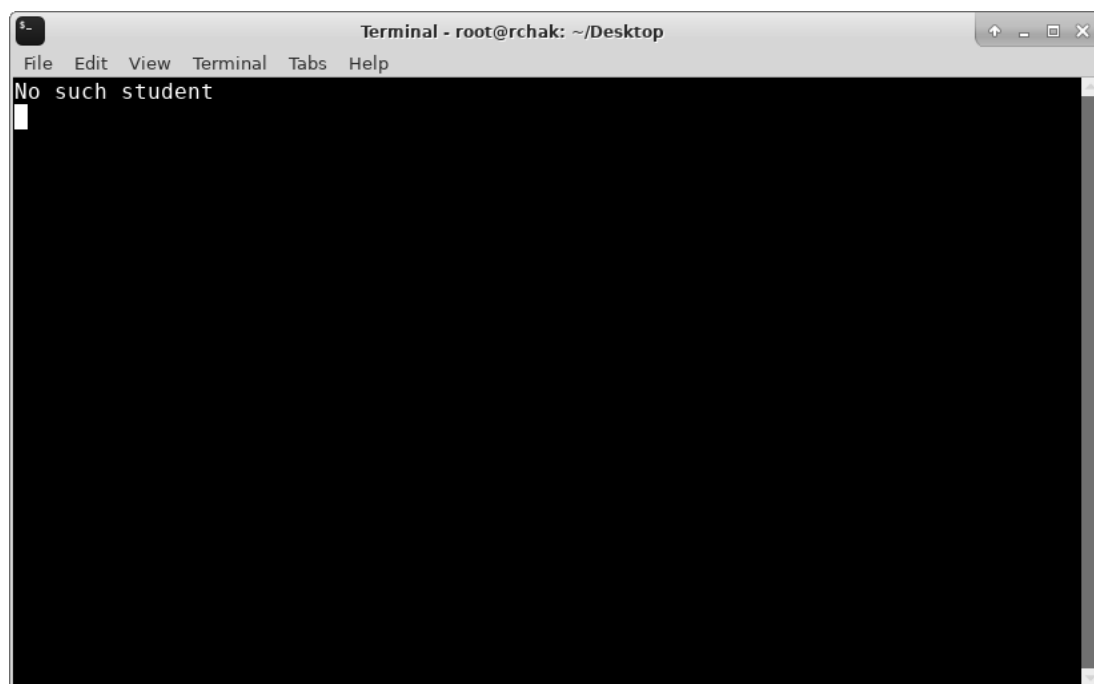


A terminal window titled "Terminal - root@rchak: ~/Desktop" with a menu bar (File, Edit, View, Terminal, Tabs, Help). The terminal displays a list of subjects and their marks, followed by an aggregate score. Below the list, it says "Press x to go back" with a cursor on the next line.

English	:	20
Math	:	45
Physics	:	44
Chemistry	:	60
Computers	:	63
Aggregate	:	46.4

Press x to go back
█

If name exists in the database, student's marks and aggregate score is displayed



A terminal window titled "Terminal - root@rchak: ~/Desktop" with a menu bar (File, Edit, View, Terminal, Tabs, Help). The terminal displays the message "No such student" followed by a cursor on the next line.

No such student
█

Screen when the entered name doesn't exist in the database

```
Terminal - root@rchak: ~/Desktop
File Edit View Terminal Tabs Help
Subject      :      Class Averages

English      :      63.4
Math         :      60.666666666666664
Physics      :      56.63333333333333
Chemistry     :      58.333333333333336
Computers    :      61.333333333333336

Press x to go back
█
```

Instructor's view of the class averages

```
Terminal - root@rchak: ~/Desktop
File Edit View Terminal Tabs Help
Subject      :      No. of Failures

English      :      2
Math         :      3
Physics      :      5
Chemistry     :      1
Computers    :      1

Press x to go back
█
```

Instructor's view of the subject-wise number of failing students

```
Terminal - root@rchak: ~/Desktop
File Edit View Terminal Tabs Help
Subject          Best          Worst

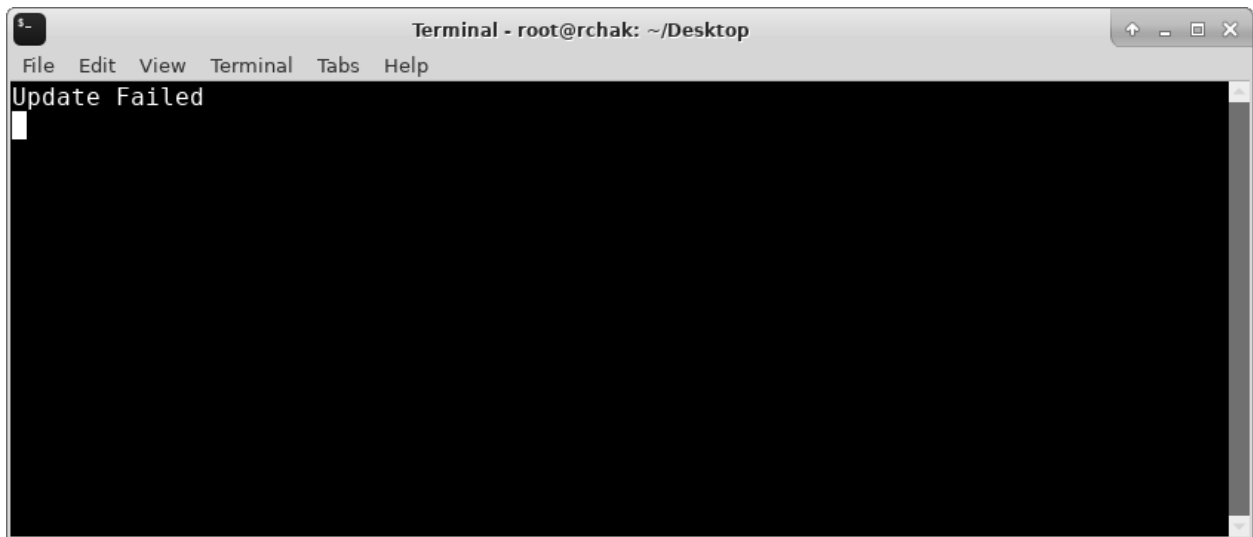
English          Eric Denmark      Abcd
Math              Aubree Friedman   Ernest Northam
Physics           Eric Denmark      Ernest Northam
Chemistry         Eric Denmark      Ernest Northam
Computers         Eric Denmark      Bo Fullwood
Aggregate         Eric Denmark      Ernest Northam

Press x to go back
█
```

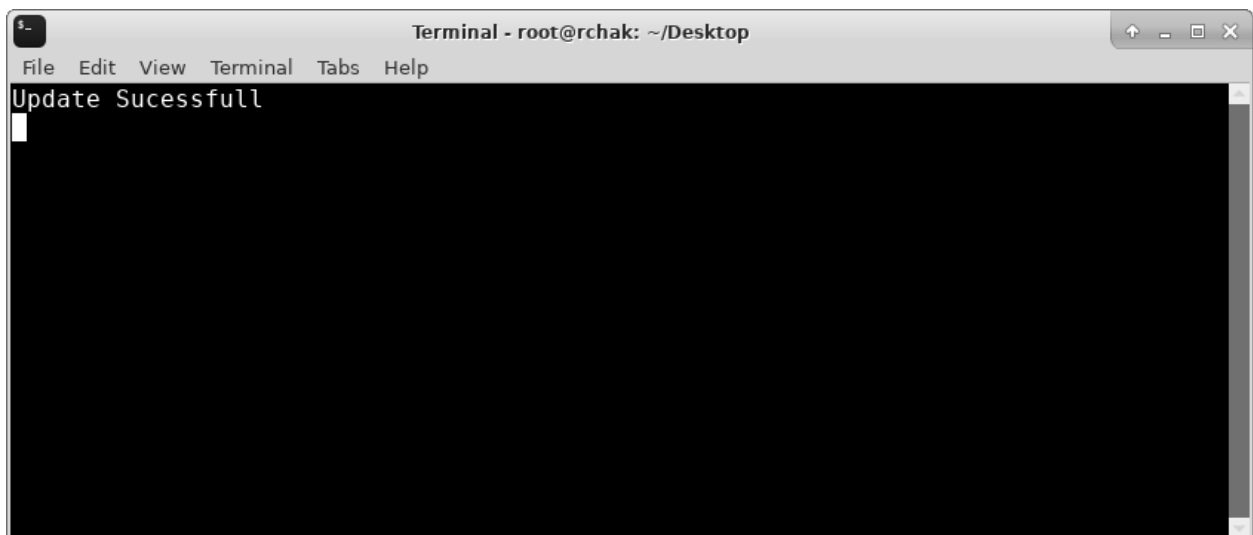
Instructor's view of the subject-wise best performers

```
Terminal - root@rchak: ~/Desktop
File Edit View Terminal Tabs Help
Enter Student Name
Eric Denmark
Enter Subject to change
1. English
2. Math
3. Physics
4. Chemistry
5. Computers
3
Enter New Marks
89█
```

Instructor's view while updating marks



Instructor's view when the update has failed due to wrong input



Instructor's view when the update is a success.

```
Terminal - root@rchak: ~/Desktop
File Edit View Terminal Tabs Help
Welcome Abcd

Choose Option:
1. Grades
2. Aggregate
3. Min/Max

Press x to logout
█
```

Student Menu

```
Terminal - root@rchak: ~/Desktop
File Edit View Terminal Tabs Help
English - 20
Math - 45
Physics - 44
Chemistry - 60
Computers - 63

Press x to go back
█
```

Student's marks in each subject

```
Terminal - root@rchak: ~/Desktop
File Edit View Terminal Tabs Help
Aggregate :46.4

Press x to go back
█
```

Student Aggregate Percentage

```
Terminal - root@rchak: ~/Desktop
File Edit View Terminal Tabs Help
Personal Record

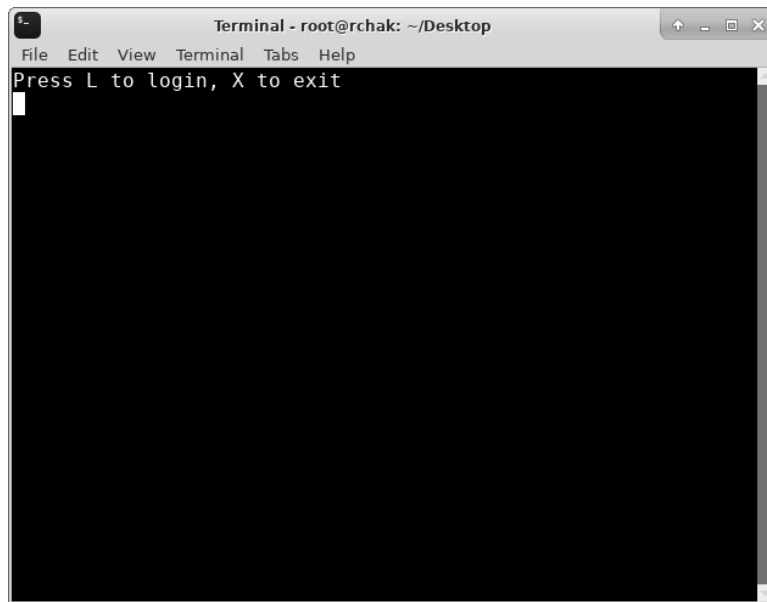
      Minimum Marks in English : 20
      Maximum Marks in Computers : 63

Overall Record

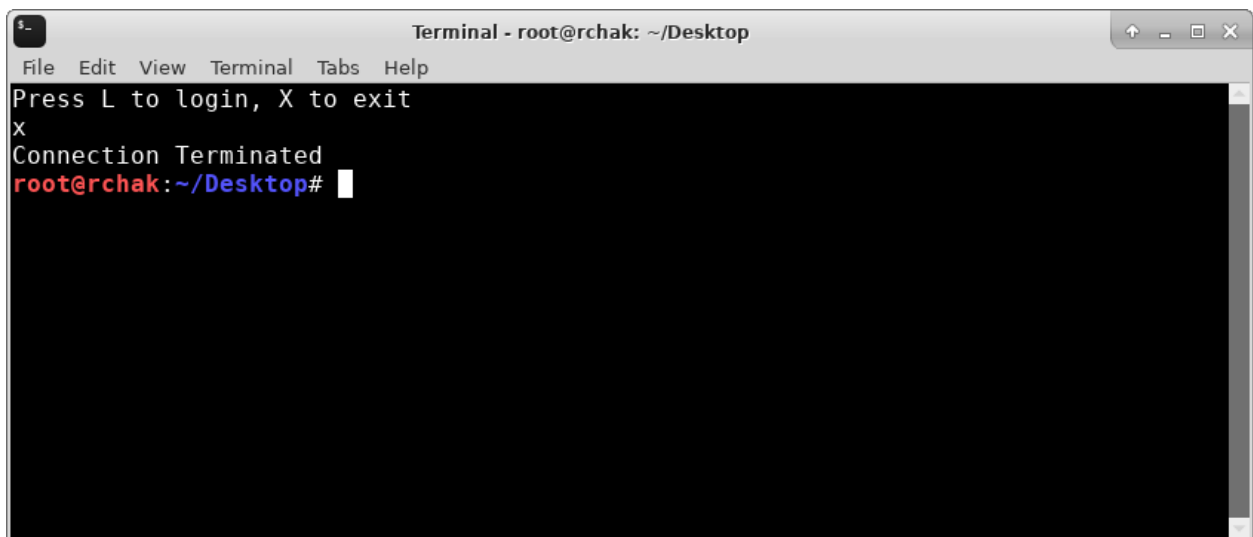
Subject      Max    Min
English      :    100.0  20.0
Math         :    108.0  22.0
Physics      :     98.0  19.0
Chemistry    :    100.0  22.0
Computers    :    100.0  32.0

Press x to go back
█
```

Subjects with maximum and minimum marks



After logging out



After the user has terminated the connection

DISCUSSION

1. **Routing** : As can be clearly seen by the traceroute command, there is a single hop from the client to the server. This is due to the fact that both are on Local Area Networks, either via a ethernet connection or via IITD_WIFI.
2. **Packet Analysis using Wireshark** : Wireshark was used to look at the various packets being transmitted or received at a node. For this assignment, Wireshark was run at the client side.

Basic Observations:

When the connection is established, three packets are exchanged to confirm and establish the pipeline(TCP is a connection-oriented protocol). The client sends a SYN request, initiating the conversation, server side responds with an [SYN ACK], and finally an ACK is sent by the client. During transmission of data from either side, the sender uses a [PSH ACK] packet, and the receiver responds with an ACK packet containing the appropriate flags.

The connection is terminated by exchange of the [FIN ACK] packets.

In the packets which contain data, the data is appended to the end of the packet(screenshot at the end of the section).

Analysis of Frame

The SYN frame sent for establishing the connection is 74 bytes in length. The MAC information is 14 bytes long, 12 bytes for two addresses and 2 bytes for type of IP. The IPv4 part is 20 bytes long. TCP part is 40 bytes long. [SYN ACK] is also 74 bytes long.

Finally, ACK is 66 bytes long. The TCP part has 32 bytes, rest have same sizes. The difference in TCP part is due to the fact that ACK has 12 bytes of TCP Options and SYN or [SYN ACK] has 20 bytes.

The [PSH ACK] frame size depends on the length of the packet sent. The size of ACK is 66 bytes. Therefore, the segment size 66 + Data Bytes. For example, to start the login page, the server sends “us” to the client. Hence, the size of [PSH ACK] in this case is 68 bytes.

[FIN ACK] statements are also 66 bytes long.

Wireshark interface showing packet capture data for a TCP session. The top pane displays a list of packets, with packet 84 selected. The middle pane shows the details of the selected packet, including the TCP header and options. The bottom pane shows the raw packet data in hexadecimal and ASCII.

Packet 84: 11.15635493.10.184.29.93 → 10.184.21.106 [RST] Seq=61680 Win=0 Len=0

Details:

- Frame 84: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
- Ethernet II, Src: IntelCor, Dst: 08:00:00:00:00:00, Bti: HomePr, 59:04:cd (54:13:79:59:04:cd)
- Internet Protocol Version 4, Src: 10.184.29.93, Dst: 10.184.21.106
- Transmission Control Protocol, Src Port: 41680, Dst Port: 8080, Seq. 0, Len: 0

Raw packet data (hex): 54 13 79 59 04 cd 30 e3 7a 0d ab 92 06 00 45 09 00 3c 5f 7c 40 00 40 06 92 89 0a b8 1c 5d 0a b8 15 6a 92 00 1f 40 06 93 86 09 00 00 00 00 00 02 72 10 c3 0a 00 00 02 04 05 b4 02 00 0a 27 03 0d ab 00 00 00 00 01 03 03 07

All packets for a server session for instructor

Wireshark interface showing packet capture data for a TCP session. The top pane displays a list of packets, with packet 84 selected. The middle pane shows the details of the selected packet, including the TCP header and options. The bottom pane shows the raw packet data in hexadecimal and ASCII.

Packet 84: 11.15635493.10.184.29.93 → 10.184.21.106 [RST] Seq=61680 Win=0 Len=0

Details:

- Frame 84: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
- Ethernet II, Src: IntelCor, Dst: 08:00:00:00:00:00, Bti: HomePr, 59:04:cd (54:13:79:59:04:cd)
- Internet Protocol Version 4, Src: 10.184.29.93, Dst: 10.184.21.106
- Transmission Control Protocol, Src Port: 41680, Dst Port: 8080, Seq. 0, Len: 0

Raw packet data (hex): 54 13 79 59 04 cd 30 e3 7a 0d ab 92 06 00 45 09 00 3c 5f 7c 40 00 40 06 92 89 0a b8 1c 5d 0a b8 15 6a 92 00 1f 40 06 93 86 09 00 00 00 00 00 02 72 10 c3 0a 00 00 02 04 05 b4 02 00 0a 27 03 0d ab 00 00 00 00 01 03 03 07

SYN and [SYN ACK] size

Wireshark interface showing a packet capture of an ACK packet (No. 84) from 10.184.21.106. The packet details pane shows the Transmission Control Protocol (TCP) section with the following information:

- Frame 8: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0
- Ethernet II, Src: IntelCor 8d:ab:92 (18:03:7a:8d:ab:92), Dst: HondaIPr 59:04:cd (54:13:79:59:04:cd)
- Internet Protocol Version 4, Src: 10.184.29.93, Dst: 10.184.21.106
- Transmission Control Protocol, Src Port: 41680, Dst Port: 8000, Seq: 1, Ack: 1, Len: 0

The packet bytes pane shows the raw data of the ACK packet, which is a single byte (0x00) representing the acknowledgment number.

ACK size

Wireshark interface showing a packet capture of an ACK packet (No. 84) from 10.184.21.106. The packet details pane shows the Transmission Control Protocol (TCP) section with the following information:

- Frame 11: 68 bytes on wire (544 bits), 68 bytes captured (544 bits) on interface 0
- Ethernet II, Src: HondaIPr 59:04:cd (54:13:79:59:04:cd), Dst: IntelCor 8d:ab:92 (18:03:7a:8d:ab:92)
- Internet Protocol Version 4, Src: 10.184.21.106, Dst: 10.184.29.93
- Transmission Control Protocol, Src Port: 8000, Dst Port: 41680, Seq: 1, Ack: 2, Len: 2

The packet bytes pane shows the raw data of the ACK packet, which is a single byte (0x00) representing the acknowledgment number.

[PSH ACK] packet.

REFERENCES

1. Python Twisted Library : <https://twistedmatrix.com/trac/>
2. <http://briancaffey.github.io/2017/12/22/getting-started-with-twisted.html>