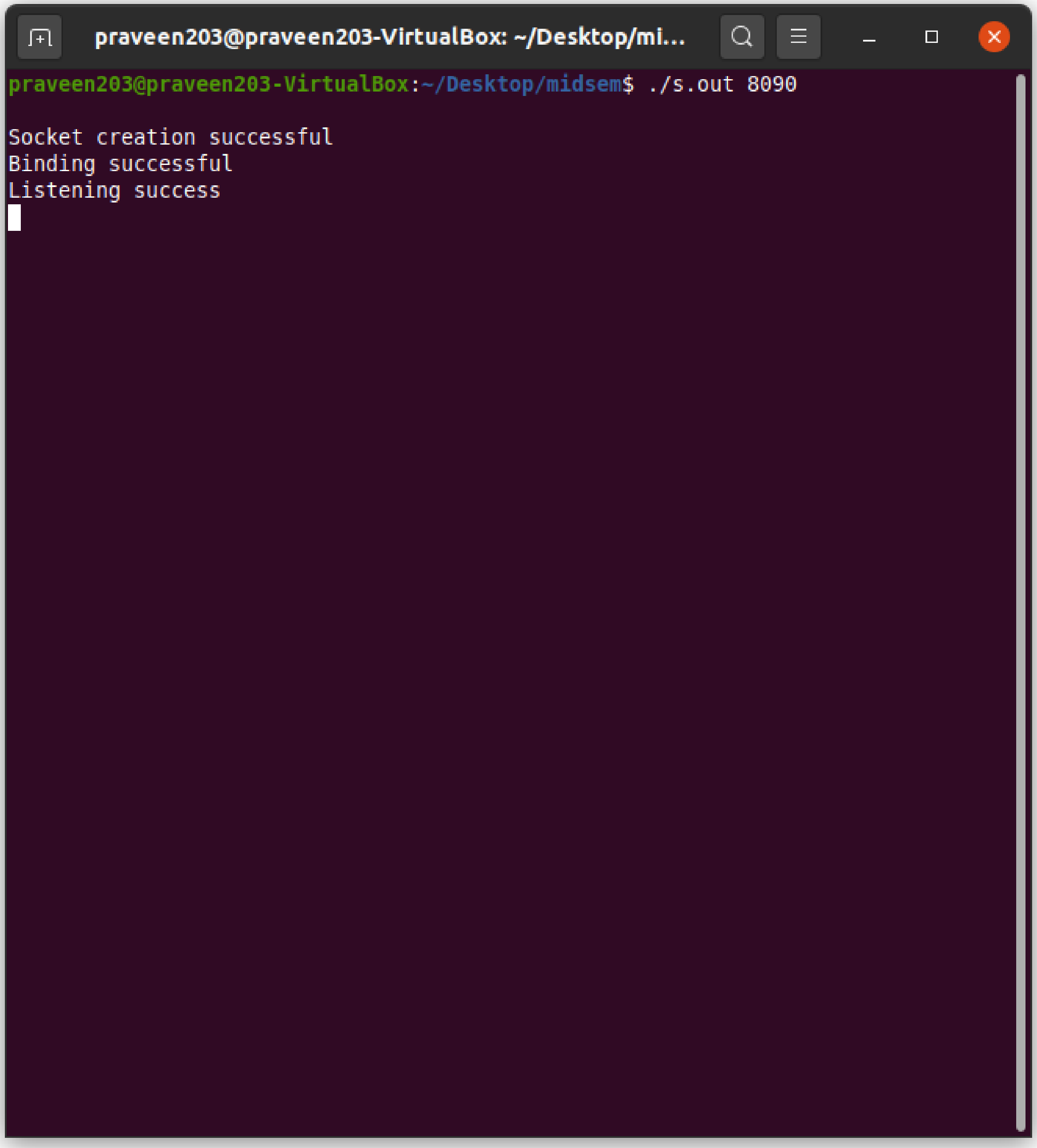
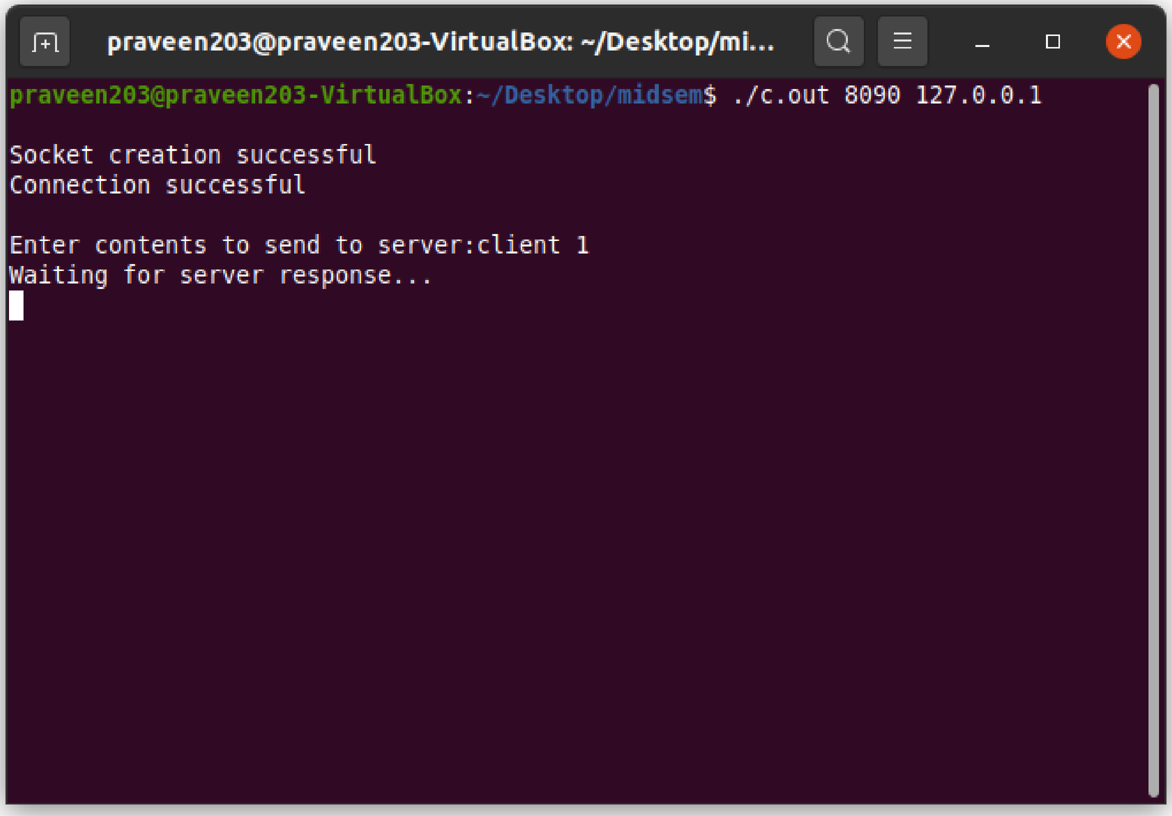
Computer networks midsem

1. Server creation:



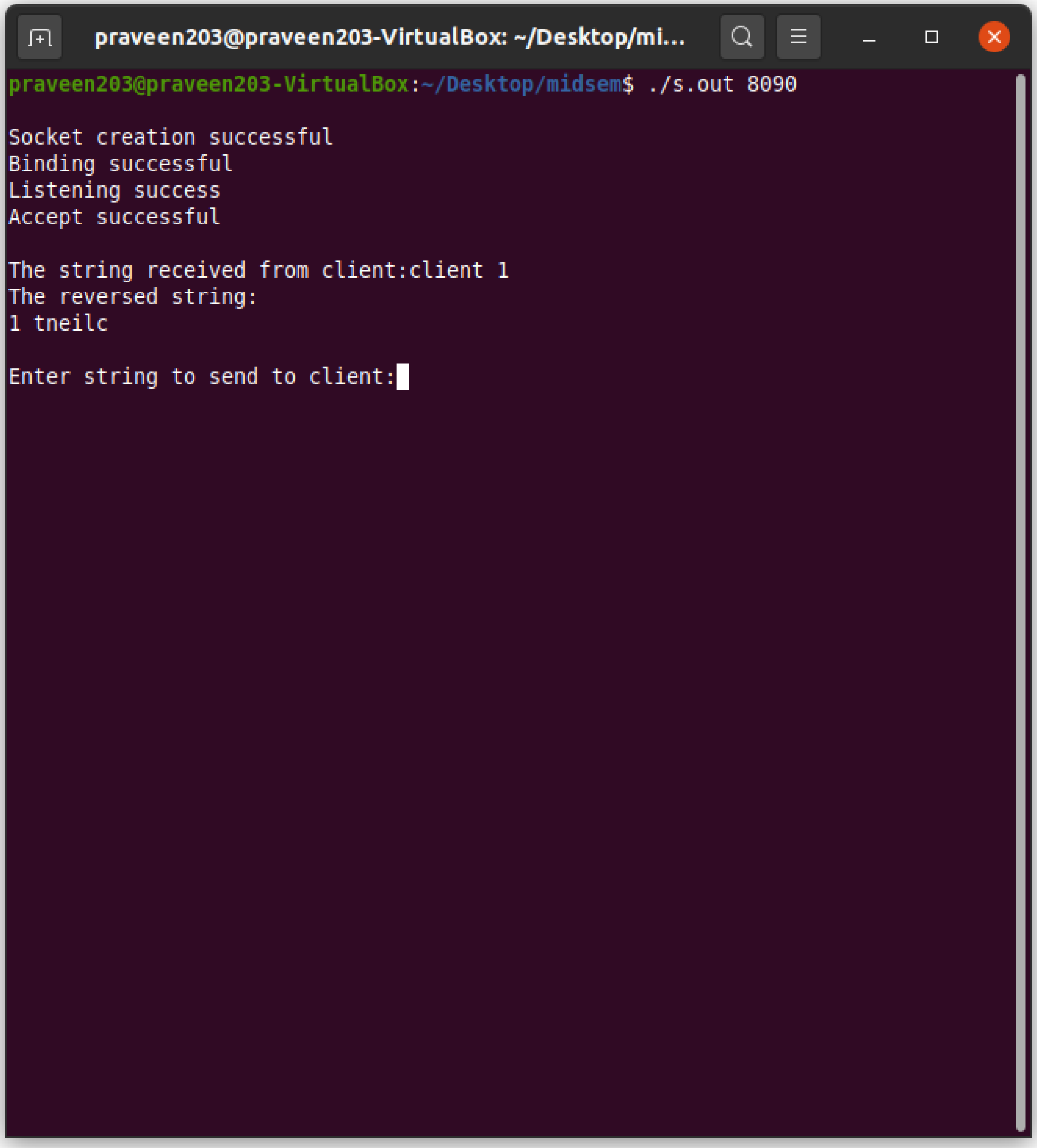
```
praveen203@praveen203-VirtualBox:~/Desktop/midsem$ ./s.out 8090

Socket creation successful
Binding successful
Listening success
```

2. Client creation and sending message:



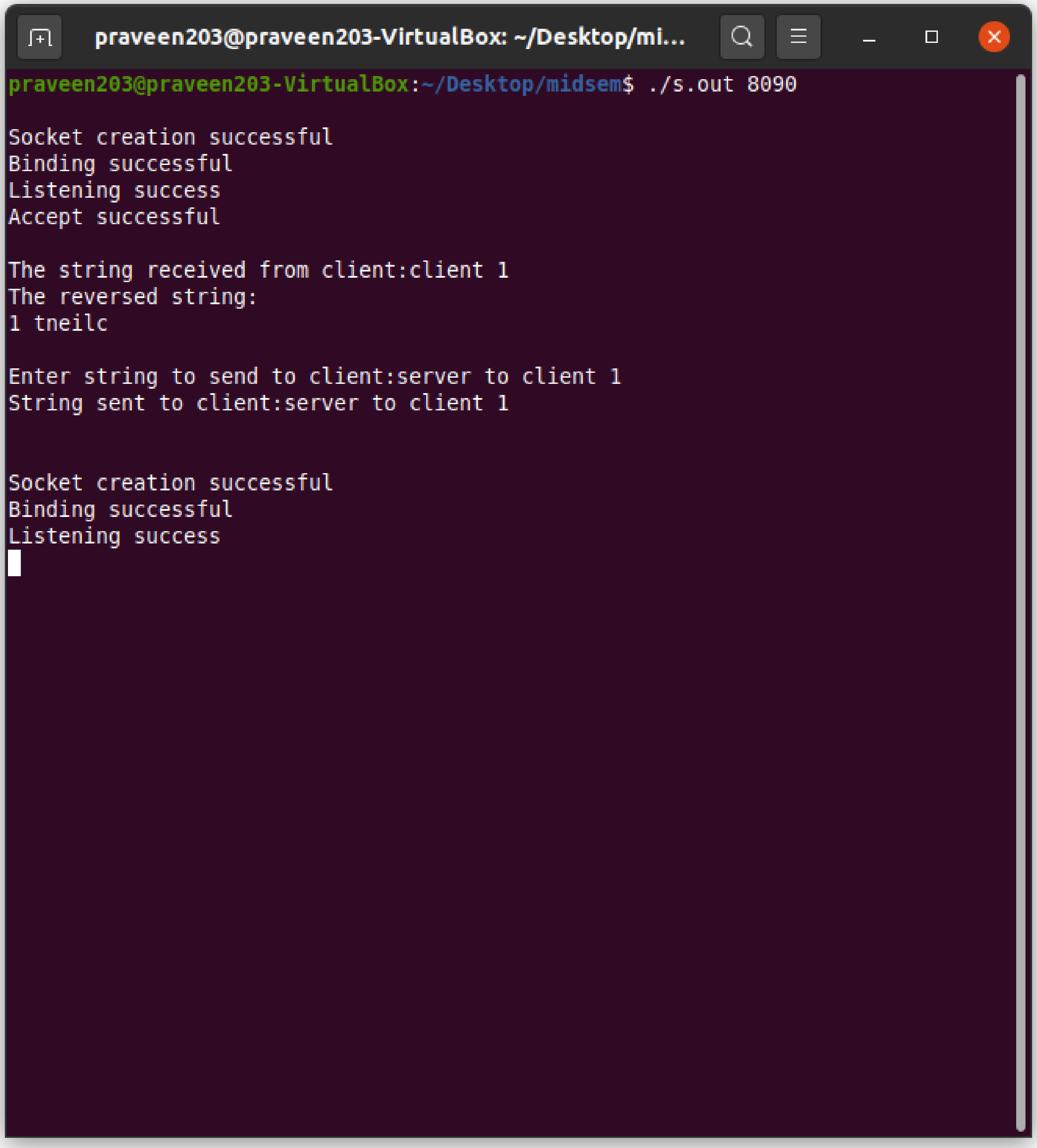
```
praveen203@praveen203-VirtualBox:~/Desktop/midsem$ ./c.out 8090 127.0.0.1

Socket creation successful
Connection successful

Enter contents to send to server:client 1
Waiting for server response...
```

3. Server receiving message and reversing it:



```
praveen203@praveen203-VirtualBox:~/Desktop/midsem$ ./s.out 8090

Socket creation successful
Binding successful
Listening success
Accept successful

The string received from client:client 1
The reversed string:
1 tneilc

Enter string to send to client:
```

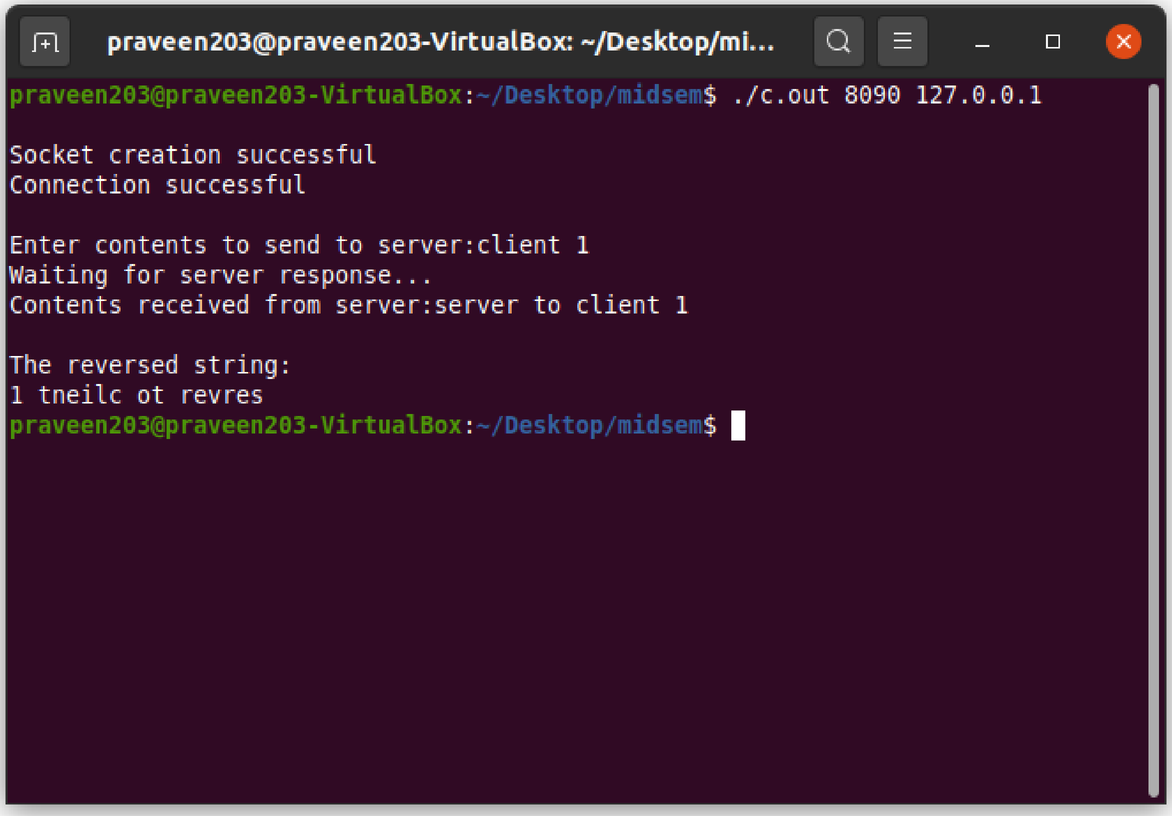
4. Server sending message:



praveen203@praveen203-VirtualBox: ~/Desktop/mi...

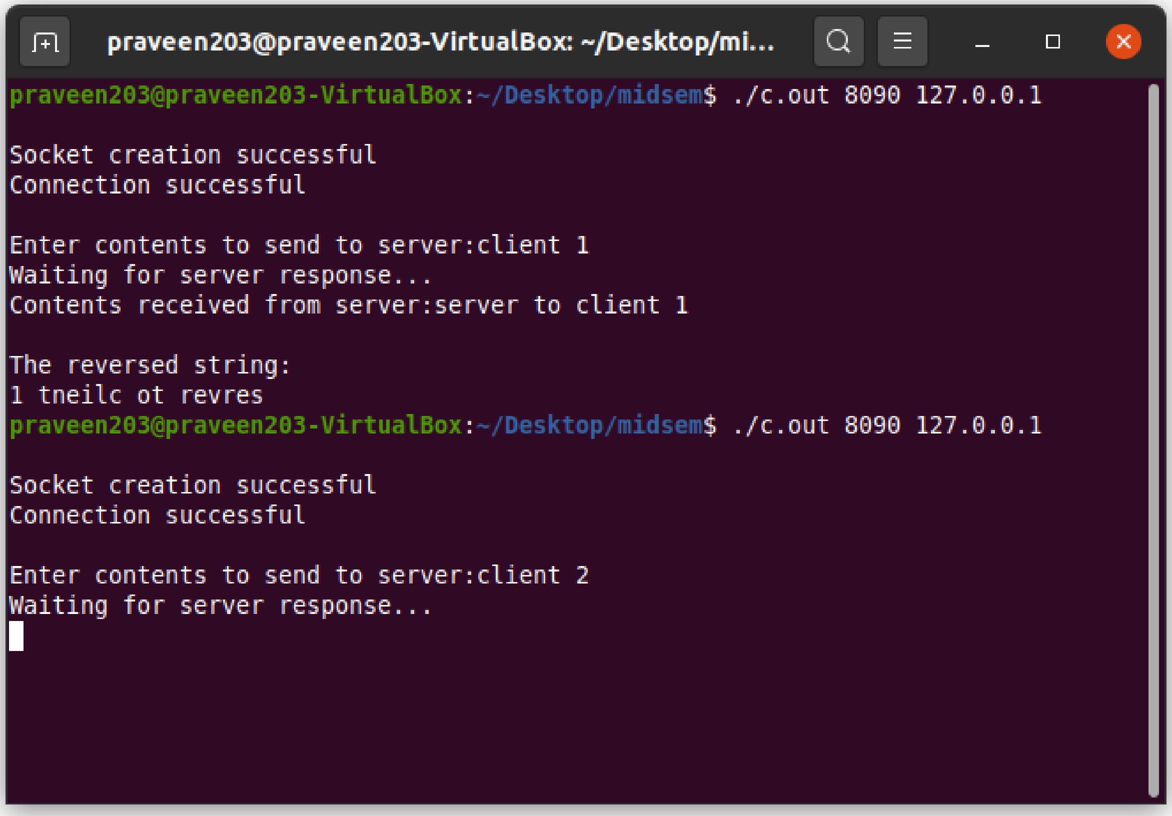
```
praveen203@praveen203-VirtualBox:~/Desktop/midsem$ ./s.out 8090

Socket creation successful
Binding successful
Listening success
Accept successful

The string received from client:client 1
The reversed string:
1 tneilc

Enter string to send to client:server to client 1
String sent to client:server to client 1


Socket creation successful
Binding successful
Listening success
```

5. Client receiving message, reversing it and terminating:



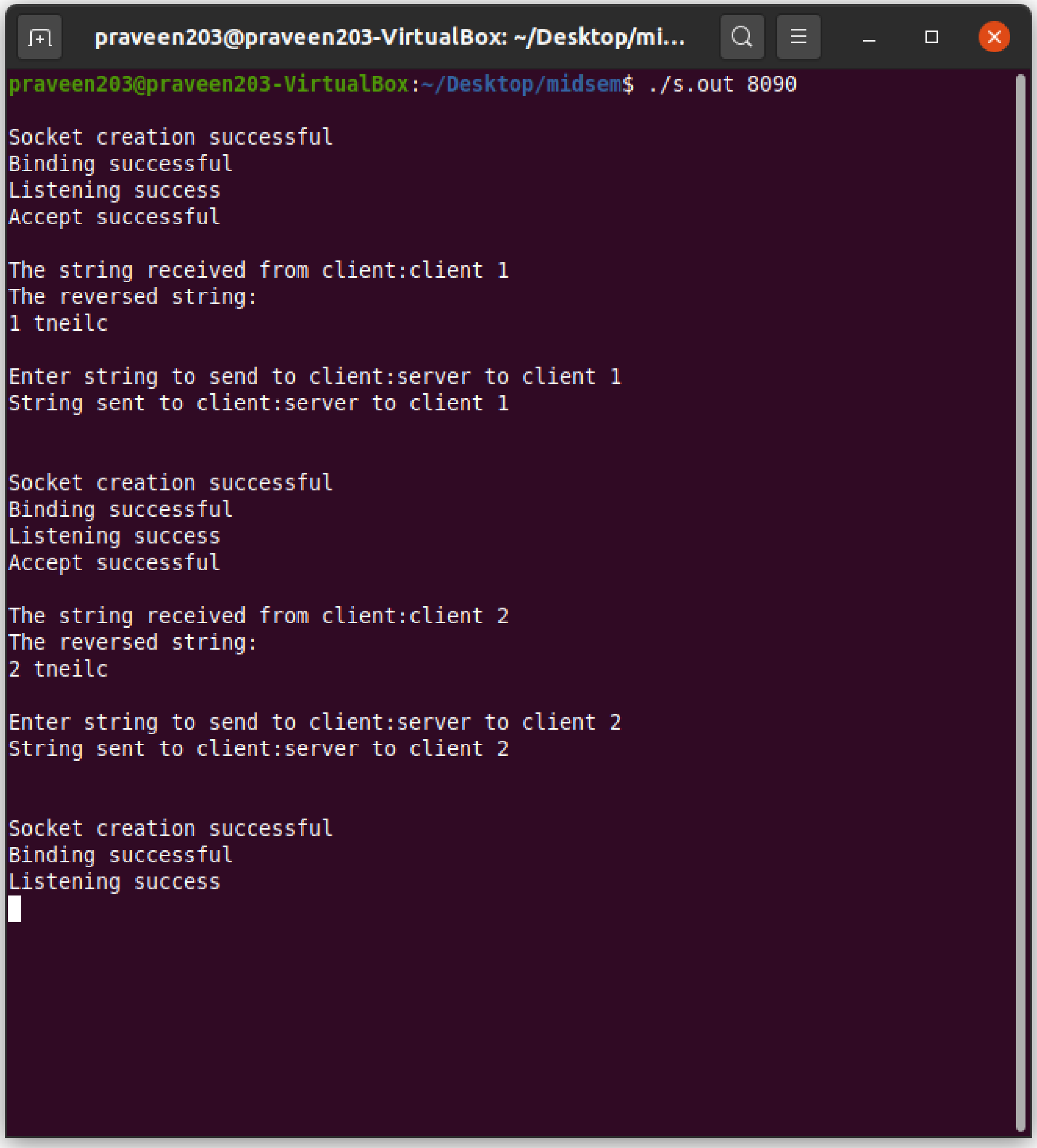
```
praveen203@praveen203-VirtualBox:~/Desktop/midsem$ ./c.out 8090 127.0.0.1

Socket creation successful
Connection successful

Enter contents to send to server:client 1
Waiting for server response...
Contents received from server:server to client 1

The reversed string:
1 tneilc ot revres
praveen203@praveen203-VirtualBox:~/Desktop/midsem$
```

6. New client creation and sending message:


praveen203@praveen203-VirtualBox: ~/Desktop/mi...

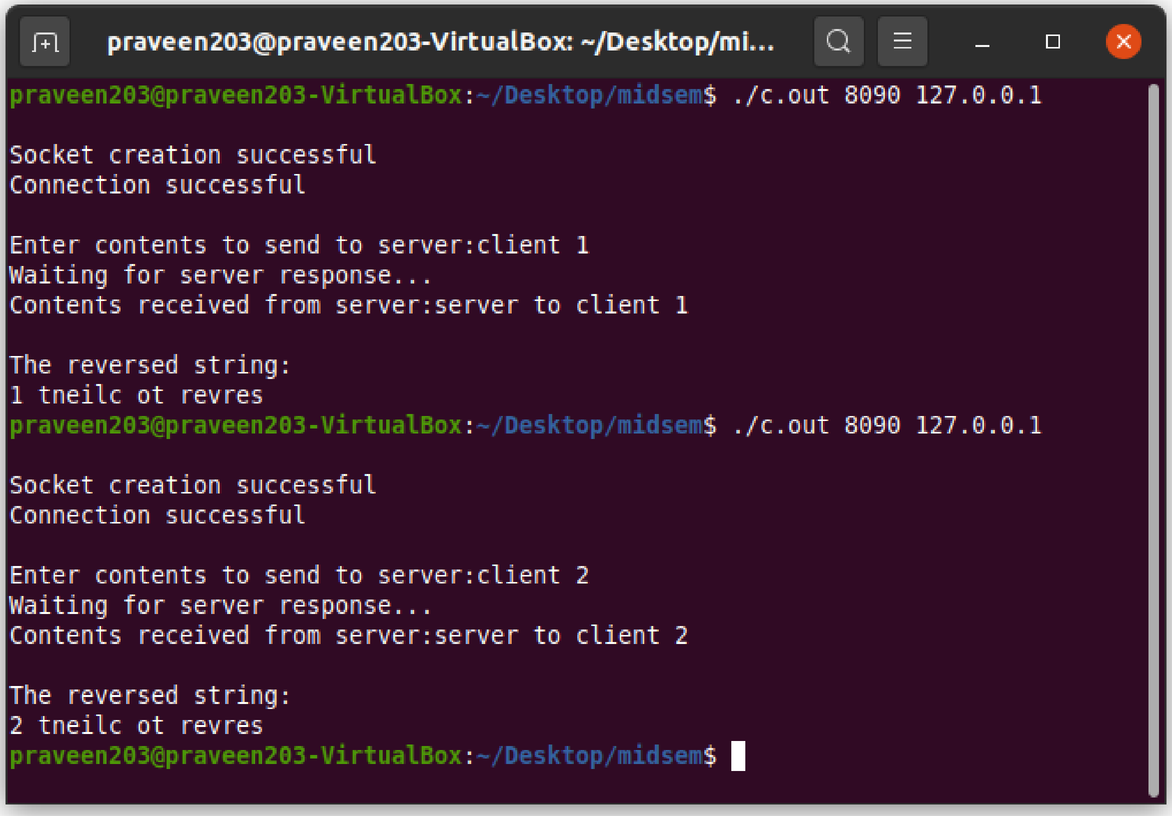
```
praveen203@praveen203-VirtualBox:~/Desktop/midsem$ ./c.out 8090 127.0.0.1

Socket creation successful
Connection successful

Enter contents to send to server:client 1
Waiting for server response...
Contents received from server:server to client 1

The reversed string:
1 tneilc ot revres
praveen203@praveen203-VirtualBox:~/Desktop/midsem$ ./c.out 8090 127.0.0.1

Socket creation successful
Connection successful

Enter contents to send to server:client 2
Waiting for server response...
```

7. Server connecting to new client, receiving and reversing message; then sending new message:



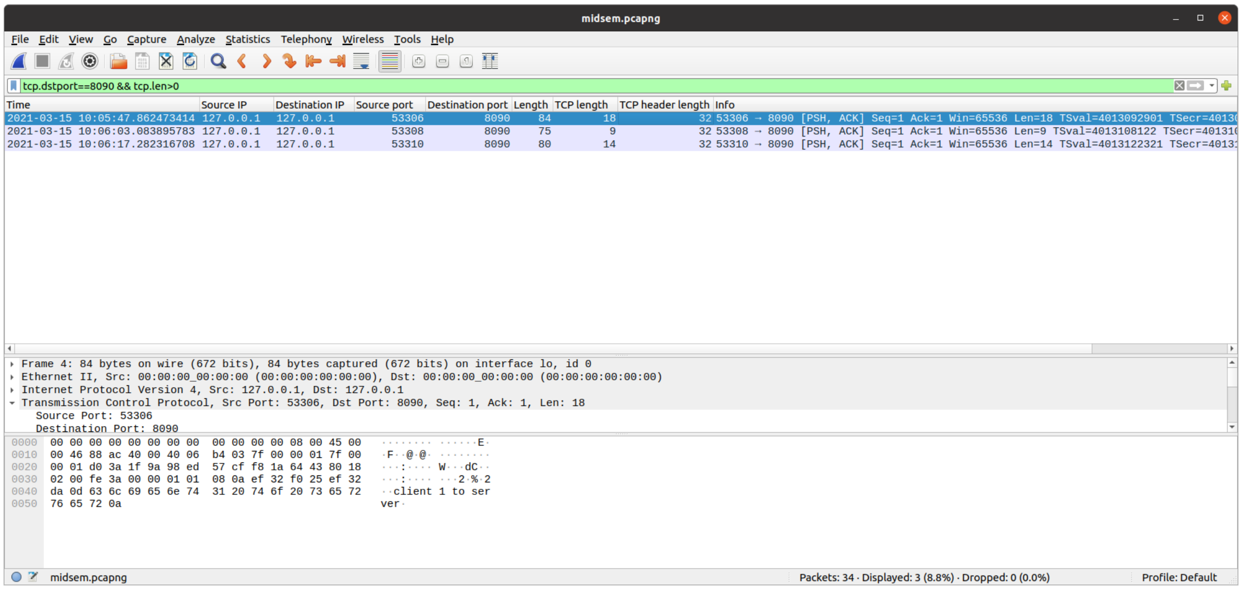
```
praveen203@praveen203-VirtualBox:~/Desktop/midsem$ ./s.out 8090

Socket creation successful
Binding successful
Listening success
Accept successful

The string received from client:client 1
The reversed string:
1 tneilc

Enter string to send to client:server to client 1
String sent to client:server to client 1


Socket creation successful
Binding successful
Listening success
Accept successful

The string received from client:client 2
The reversed string:
2 tneilc

Enter string to send to client:server to client 2
String sent to client:server to client 2


Socket creation successful
Binding successful
Listening success
```

8. Client accepting message, reversing and terminating:



```
praveen203@praveen203-VirtualBox:~/Desktop/mi...

praveen203@praveen203-VirtualBox:~/Desktop/midsem$ ./c.out 8090 127.0.0.1

Socket creation successful
Connection successful

Enter contents to send to server:client 1
Waiting for server response...
Contents received from server:server to client 1

The reversed string:
1 tneilc ot revres
praveen203@praveen203-VirtualBox:~/Desktop/midsem$ ./c.out 8090 127.0.0.1

Socket creation successful
Connection successful

Enter contents to send to server:client 2
Waiting for server response...
Contents received from server:server to client 2

The reversed string:
2 tneilc ot revres
praveen203@praveen203-VirtualBox:~/Desktop/midsem$
```

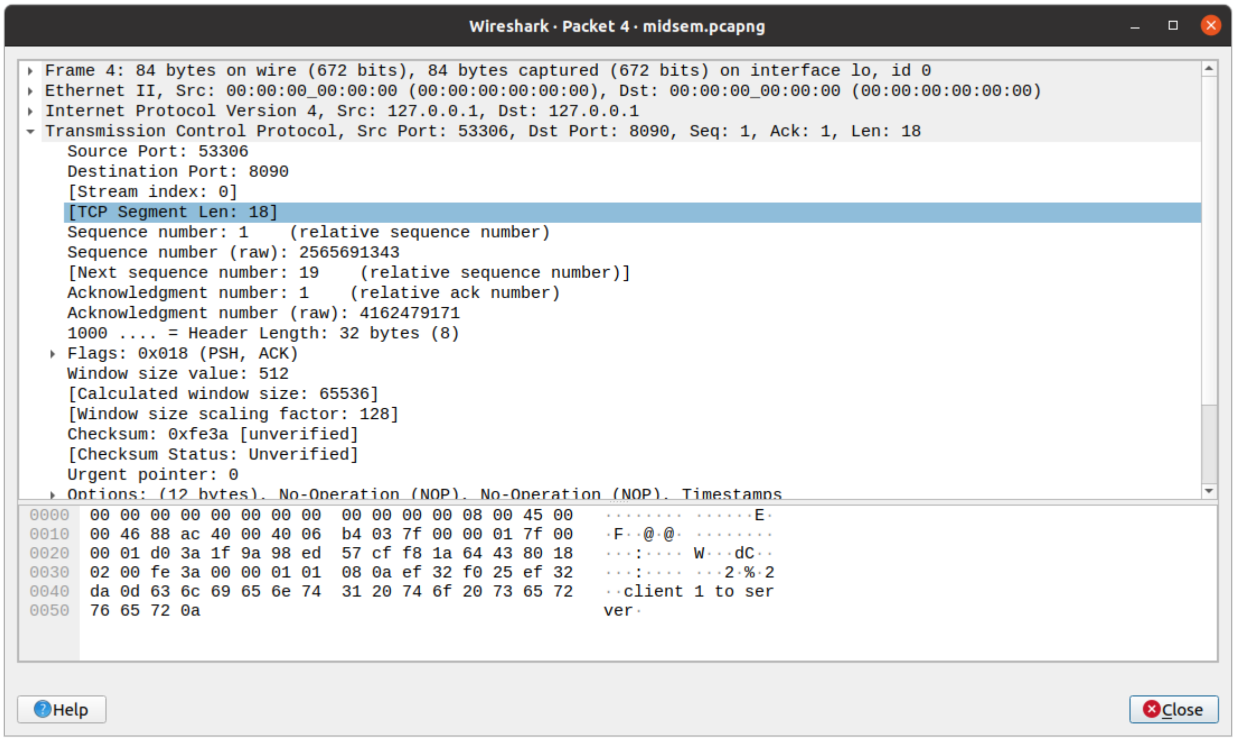
9. Wire shark client to server messages: Filter used: tcp.dstport==8090 && tcp.len>0:



midsem.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

```
tcp.dstport==8090 && tcp.len>0

Time                          Source IP  Destination IP  Source port  Destination port  Length  TCP length  TCP header length  Info
2021-03-15 10:05:47.862473414 127.0.0.1  127.0.0.1       53306        8090              84      18          32 53306 → 8090 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=18 TSval=4013092901 TSecr=4013
2021-03-15 10:06:03.083895783 127.0.0.1  127.0.0.1       53308        8090              75       9          32 53308 → 8090 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=9 TSval=4013108122 TSecr=40131
2021-03-15 10:06:17.282316708 127.0.0.1  127.0.0.1       53310        8090              80      14          32 53310 → 8090 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=14 TSval=4013122321 TSecr=4013

› Frame 4: 84 bytes on wire (672 bits), 84 bytes captured (672 bits) on interface lo, id 0
› Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
› Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
˅ Transmission Control Protocol, Src Port: 53306, Dst Port: 8090, Seq: 1, Ack: 1, Len: 18
    Source Port: 53306
    Destination Port: 8090

0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 00   ········ ······E·
0010  00 46 88 ac 40 00 40 06  b4 03 7f 00 00 01 7f 00   ·F··@·@· ········
0020  00 01 d0 3a 1f 9a 98 ed  57 cf f8 1a 64 43 80 18   ···:···· W···dC··
0030  02 00 fe 3a 00 00 01 01  08 0a ef 32 f0 25 ef 32   ···:···· ···2·%·2
0040  da 0d 63 6c 69 65 6e 74  31 20 74 6f 20 73 65 72   ··client 1 to ser
0050  76 65 72 0a                                        ver·

midsem.pcapng          Packets: 34 · Displayed: 3 (8.8%) · Dropped: 0 (0.0%)          Profile: Default
```

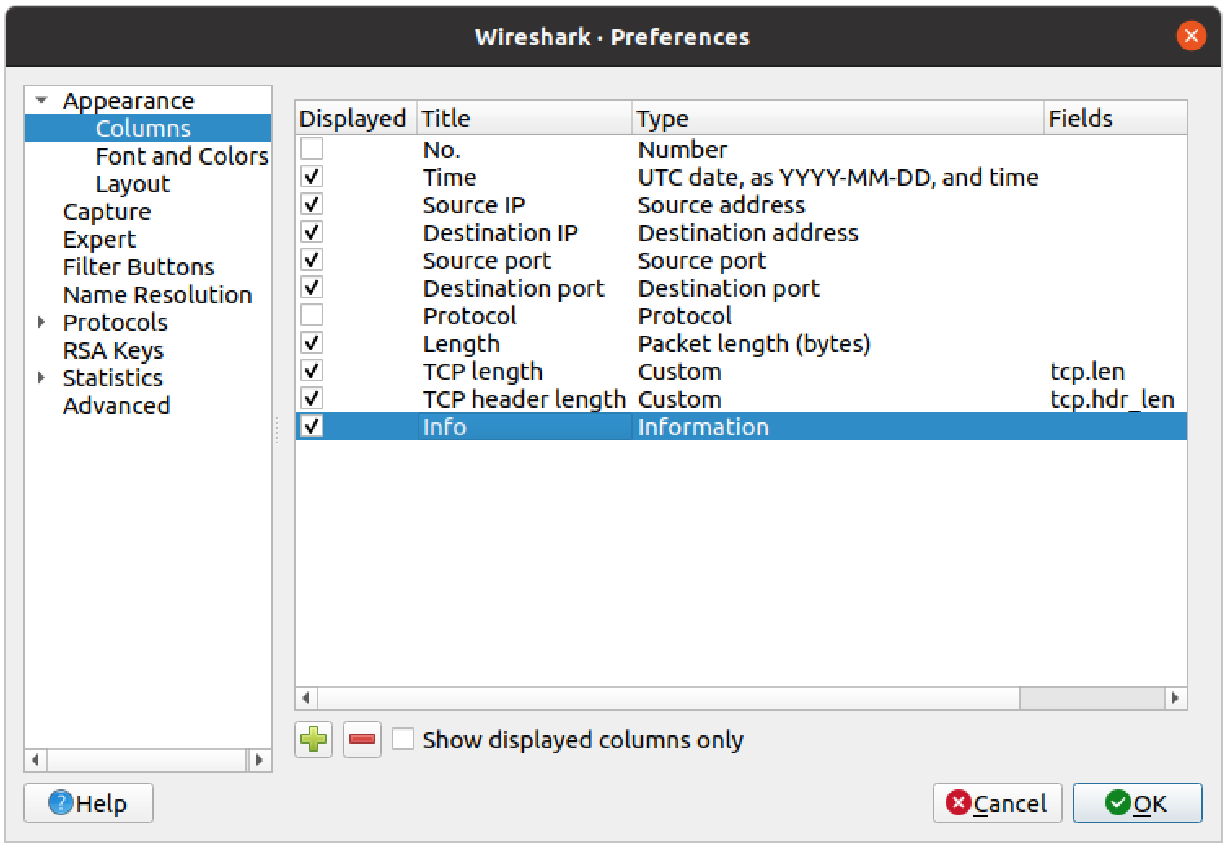
10. Wire shark packet 1 properties:



```
▸ Frame 4: 84 bytes on wire (672 bits), 84 bytes captured (672 bits) on interface lo, id 0
▸ Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
▸ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▾ Transmission Control Protocol, Src Port: 53306, Dst Port: 8090, Seq: 1, Ack: 1, Len: 18
    Source Port: 53306
    Destination Port: 8090
    [Stream index: 0]
    [TCP Segment Len: 18]
    Sequence number: 1    (relative sequence number)
    Sequence number (raw): 2565691343
    [Next sequence number: 19    (relative sequence number)]
    Acknowledgment number: 1    (relative ack number)
    Acknowledgment number (raw): 4162479171
    1000 .... = Header Length: 32 bytes (8)
  ▸ Flags: 0x018 (PSH, ACK)
    Window size value: 512
    [Calculated window size: 65536]
    [Window size scaling factor: 128]
    Checksum: 0xfe3a [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  ▸ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps

0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 00   ········ ······E·
0010  00 46 88 ac 40 00 40 06  b4 03 7f 00 00 01 7f 00   ·F··@·@· ········
0020  00 01 d0 3a 1f 9a 98 ed  57 cf f8 1a 64 43 80 18   ···:···· W···dC··
0030  02 00 fe 3a 00 00 01 01  08 0a ef 32 f0 25 ef 32   ···:···· ···2·%·2
0040  da 0d 63 6c 69 65 6e 74  31 20 74 6f 20 73 65 72   ··client 1 to ser
0050  76 65 72 0a                                        ver·
```

Here we can see the properties of packet 1 sent from the client to the server with dstport 8090. We use tcp.len>0 in the filter to filter out messages and packets which have application data payload. The TCP segment length is the sum of the tcp payload and the tcp header length = 18+32 =50 bytes. The payload is the message “client1 to server” and it is 17+1(new line character)= 18 bytes. This is the same value that is seen in the above image in [TCP Segment Len: 18].

Column preferences:



Displayed	Title	Type	Fields
☐	No.	Number	
✓	Time	UTC date, as YYYY-MM-DD, and time	
✓	Source IP	Source address	
✓	Destination IP	Destination address	
✓	Source port	Source port	
✓	Destination port	Destination port	
☐	Protocol	Protocol	
✓	Length	Packet length (bytes)	
✓	TCP length	Custom	tcp.len
✓	TCP header length	Custom	tcp.hdr_len
✓	Info	Information	

☐ Show displayed columns only

Praveen Sridhar
2018A7PS0166G