

# Data Communication and Computer Network Laboratory

## ASSIGNMENT

MCA 1st year 2nd Sem

---

Name	Roll-no
Dwip Shekhar Mondal	002210503037

---

## Assignment - III

### Problem statement →

The objective of this laboratory exercise is to look at the details of the User Datagram Protocol (UDP). UDP is a transport layer protocol. It is used by many application protocols like DNS, DHCP, SNMP etc., where reliability is not a concern.

To do this exercise you need to install the Wireshark tool, which is widely used to capture and examine a packet trace. Wireshark can be downloaded from [www.wireshark.org](http://www.wireshark.org).

### Step 1: Capture a Trace

- (i) Launch Wireshark
- (ii) From Capture→Options select Loopback interface
- (iii) Start a capture with a filter of “ip.addr==127.0.0.1 and udp.port==xxxx”, where xxxx is the port number used by the UDP server.
- (iv) Run the UDP server program on a terminal.
- (v) Run multiple instances of the UDP client program on separate terminals and send requests to the server.
- (vi) Stop Wireshark capture

### Step 2: Inspect the Trace

Select different packets in the trace and browse the expanded UDP header and record the following fields:

- Source Port: the port from which the udp segment is sent.
  - Destination Port: the port to which the udp segment is sent.
  - Length: the length of the UDP segment.
-

## Step 1: Capture a Trace →

For this task, the client/server program of question 3 from assignment-1 is used, in which the UDP client program sends requests to the server with a student name to get the requested student data (name, address).

The server returns the student info if the student exists in the local student database (a csv file stored in the server) otherwise it sends a 'student does not exist message'.

Using wireshark the packets sent/received by the client/server will be observed.

**Note:** there will be two instances of clients, both will request the server for student info, one will request for a student name that exists in the database the other client will request for a student that does not exist in the database.

- Wireshark capture with Loopback interface as input and with filter of "ip.addr==127.0.0.1 and udp.port==9999" (Port 9999 is used by the UDP server)
- Two instances of the client connect to the server.

ip.addr == 127.0.0.1 and udp.port == 9999						
No.	Time	Source	Destination	Protocol	Length	Info
12	52.623827	127.0.0.1	127.0.0.1	UDP	51	58783 → 9999 Len=19
13	52.625007	127.0.0.1	127.0.0.1	UDP	100	9999 → 58783 Len=68
22	77.584493	127.0.0.1	127.0.0.1	UDP	44	55881 → 9999 Len=12
23	77.585189	127.0.0.1	127.0.0.1	UDP	49	9999 → 55881 Len=17

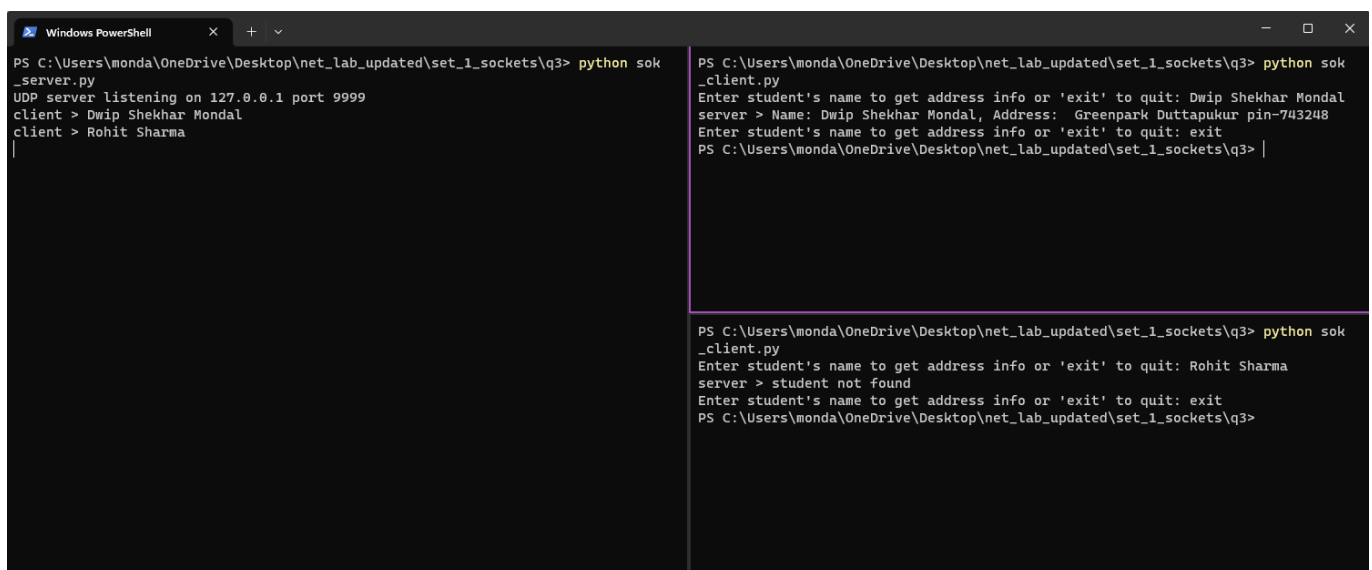
  

> Frame 12: 51 bytes on wire (408 bits), 51 bytes captured (408 bits) on interface \Device\NPF... > Null/Loopback > Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1 > User Datagram Protocol, Src Port: 58783, Dst Port: 9999 > Data (19 bytes) Data: 44776970205368656b686172204d6f6e64616c [Length: 19]		0000 02 00 00 00 45 00 00 2f a7 0b 00 00 00 11 00 00 .....E../ ..... 0010 7f 00 00 01 7f 00 00 01 e5 9f 27 0f 00 1b 91 6e .....n 0020 44 77 69 70 20 53 68 65 6b 68 61 72 20 4d 6f 6e Dwip She khar Mon 0030 64 61 6c dal
---	--	--

Figure 1: Capture of the packets from the client/server (UDP)

From the above capture it it's evident that →

- Two clients connects to the UDP server
- Server listens at port 9999
- 2 instances of client connects to the server
  - ◆ Client 1 connects from 58789 → 9999
  - ◆ Client 2 connects from 55881 → 9999
- Client 1 sends a request to get a student info “Dwip Shekhar Mondal” that exists in the student database stored at the server, and gets response “Name: Dwip Shekhar Mondal, Address: Greenpark Duttapukur pin-743248”
- Client 2 sends a request to get a student info “Rohit Sharma” that does not exist in the student database stored in the server, and gets a generic error message “Student does not exist”



```
PS C:\Users\monda\OneDrive\Desktop\net_lab_updated\set_1_sockets\q3> python sok_server.py
UDP server listening on 127.0.0.1 port 9999
client > Dwip Shekhar Mondal
client > Rohit Sharma
|

PS C:\Users\monda\OneDrive\Desktop\net_lab_updated\set_1_sockets\q3> python sok_client.py
Enter student's name to get address info or 'exit' to quit: Dwip Shekhar Mondal
server > Name: Dwip Shekhar Mondal, Address: Greenpark Duttapukur pin-743248
Enter student's name to get address info or 'exit' to quit: exit
PS C:\Users\monda\OneDrive\Desktop\net_lab_updated\set_1_sockets\q3> |

PS C:\Users\monda\OneDrive\Desktop\net_lab_updated\set_1_sockets\q3> python sok_client.py
Enter student's name to get address info or 'exit' to quit: Rohit Sharma
server > student not found
Enter student's name to get address info or 'exit' to quit: exit
PS C:\Users\monda\OneDrive\Desktop\net_lab_updated\set_1_sockets\q3>
```

Figure 2: Request/response from client/server console (UDP)

## Step 2: Inspect the Trace

## Client 1 - server →

→ Client 1 connects from port 58783 → 9999

- ◆ It sends the request with source port: 58783 and destination port: 9999
- ◆ It sends data with length 19 bytes i.e the UDP payload size is 19 bytes.  
Data = “Dwip Sekhar Mondal”

◆ Server Responds with appropriate message.

- Source port → 9999
- Destination port → 58783
- Payload size → 68 Bytes
- Data = “Name: Dwip Shekhar Mondal, Address: Greenpark Duttapukur pin-743248”

ip.addr == 127.0.0.1 and udp.port == 9999						
No.	Time	Source	Destination	Protocol	Length	Info
12	52.623827	127.0.0.1	127.0.0.1	UDP	51	58783 → 9999 Len=19
13	52.625007	127.0.0.1	127.0.0.1	UDP	100	9999 → 58783 Len=68
22	77.584493	127.0.0.1	127.0.0.1	UDP	44	55881 → 9999 Len=12
23	77.585189	127.0.0.1	127.0.0.1	UDP	49	9999 → 55881 Len=17

> Frame 12: 51 bytes on wire (408 bits), 51 bytes captured on interface Null/Loopback > Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1 > User Datagram Protocol, Src Port: 58783, Dst Port: 9999		0000	02 00 00 00 45 00 00 2f	a7 0b 00 00 80 11 00 00	....E../ .....
Source Port: 58783		0010	7f 00 00 01 7f 00 00 01	e5 9f 27 0f 00 1b 91 6e	.....'.....n
Destination Port: 9999		0020	44 77 69 70 20 53 68 65	6b 68 61 72 20 4d 6f 6e	Dwip She khar Mon
Length: 27		0030	64 61 6c		dal
Checksum: 0x916e [unverified]					
[Checksum Status: Unverified]					
[Stream index: 5]					
> [Timestamps]					
UDP payload (19 bytes)					
> Data (19 bytes)					
Data: 44776970205368656b686172204d6f6e64616c					
[Length: 19]					

Figure 3: Request from client 1 to server (UDP)

ip.addr == 127.0.0.1 and udp.port == 9999

No.	Time	Source	Destination	Protocol	Length	Info
12	52.623827	127.0.0.1	127.0.0.1	UDP	51	58783 → 9999 Len=19
13	52.625007	127.0.0.1	127.0.0.1	UDP	100	9999 → 58783 Len=68
22	77.584493	127.0.0.1	127.0.0.1	UDP	44	55881 → 9999 Len=12
23	77.585189	127.0.0.1	127.0.0.1	UDP	49	9999 → 55881 Len=17

> Frame 13: 100 bytes on wire (800 bits), 100 bytes captured (800 bits) on interface 0  
 > Null/Loopback  
 > Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1  
 > User Datagram Protocol, Src Port: 9999, Dst Port: 58783  
 Source Port: 9999  
 Destination Port: 58783  
 Length: 76  
 Checksum: 0x4618 [unverified]  
 [Checksum Status: Unverified]  
 [Stream index: 5]  
 > [Timestamps]  
 UDP payload (68 bytes)  
 Data (68 bytes)  
 Data: 4e616d653a2044776970205368656b686172204d66e64  
 [Length: 68]

0000 02 00 00 00 45 00 00 60 a7 0c 00 00 80 11 00 00 ....E..  
 0010 7f 00 00 01 7f 00 00 01 27 0f e5 9f 00 4c 46 18 ..... '...LF.  
 0020 4e 61 6d 65 3a 20 44 77 69 70 20 53 68 65 6b 68 Name: Dw ip Shekh  
 0030 61 72 20 4d 6f 6e 64 61 6c 2c 20 41 64 64 72 65 ar Monda l, Addre  
 0040 73 73 3a 20 20 47 72 65 65 6e 70 61 72 6b 20 44 ss: Gre enpark D  
 0050 75 74 74 61 70 75 6b 75 72 20 70 69 6e 2d 37 34 uttapuku r pin-74  
 0060 33 32 34 38 3248

Figure 4: Response from server to client 1 (UDP)

Client 2 - server  $\rightarrow$

→ Client 2 connects from port 55881 → 9999

- ◆ It sends the request with source port: 55881 and destination port: 9999
- ◆ It sends data with length 12 bytes i.e the UDP payload size is 12 bytes.  
Data = “Rohit Sharma”
- ◆ Server Responds with appropriate message.
  - Source port → 9999
  - Destination port → 55881
  - Payload size → 17 Bytes
  - Data = “student not found”

ip.addr == 127.0.0.1 and udp.port == 9999						
No.	Time	Source	Destination	Protocol	Length	Info
12	52.623827	127.0.0.1	127.0.0.1	UDP	51	58783 → 9999 Len=19
13	52.625007	127.0.0.1	127.0.0.1	UDP	100	9999 → 58783 Len=68
22	77.584493	127.0.0.1	127.0.0.1	UDP	44	55881 → 9999 Len=12
23	77.585189	127.0.0.1	127.0.0.1	UDP	49	9999 → 55881 Len=17

> Frame 22: 44 bytes on wire (352 bits), 44 bytes captured on interface Null/Loopback	0000 02 00 00 00 45 00 00 28 a7 0d 00 00 80 11 00 00	.....E... ( .....
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1	0010 7f 00 00 01 7f 00 00 01 da 49 27 0f 00 14 af 35	......I'....5
> User Datagram Protocol, Src Port: 55881, Dst Port: 9999	0020 52 6f 68 69 74 20 53 68 61 72 6d 61	Rohit Sh arma
Source Port: 55881		
Destination Port: 9999		
Length: 20		
Checksum: 0xaf35 [unverified]		
[Checksum Status: Unverified]		
[Stream index: 7]		
> [Timestamps]		
UDP payload (12 bytes)		
> Data (12 bytes)		
Data: 526f68697420536861726d61		
[Length: 12]		

Figure 5: Request from client 2 to server (UDP)

ip.addr == 127.0.0.1 and udp.port == 9999						
No.	Time	Source	Destination	Protocol	Length	Info
12	52.623827	127.0.0.1	127.0.0.1	UDP	51	58783 → 9999 Len=19
13	52.625007	127.0.0.1	127.0.0.1	UDP	100	9999 → 58783 Len=68
22	77.584493	127.0.0.1	127.0.0.1	UDP	44	55881 → 9999 Len=12
23	77.585189	127.0.0.1	127.0.0.1	UDP	49	9999 → 55881 Len=17

> Frame 23: 49 bytes on wire (392 bits), 49 bytes captured on interface Null/Loopback	0000 02 00 00 00 45 00 00 2d a7 0e 00 00 80 11 00 00	.....E... ( .....
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1	0010 7f 00 00 01 7f 00 00 01 27 0f da 49 00 19 1b 8b	......I'....
> User Datagram Protocol, Src Port: 9999, Dst Port: 55881	0020 73 74 75 64 65 6e 74 20 6e 6f 74 20 66 6f 75 6e	student not found
Source Port: 9999	0030 64	d
Destination Port: 55881		
Length: 25		
Checksum: 0x1b8b [unverified]		
[Checksum Status: Unverified]		
[Stream index: 7]		
> [Timestamps]		
UDP payload (17 bytes)		
> Data (17 bytes)		
Data: 73747564656e7420666f7420666f756e64		
[Length: 17]		

Figure 6: Response from server to client 2 (UDP)