PART A: Dean Cohen & Roei Mesilaty

First of all, here you can see the client and server files, and the output:

udpServer.py | udpClient.py | Screenshot at 2024-11-13 17-08-13.png

Ex01ITCN > udpClient.py > ...

```python
import socket

s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
s.sendto(b'Dean Cohen & Roei Mesilaty', ('192.168.49.131', 12345))
# print("The data has been sent")

data, addr = s.recvfrom(1024)
print(str(data), addr)
# print("Received data from server: ", str(data), addr)

s.close()
# print("The socket has been closed")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\dean5\OneDrive\Desktop\Year3\Intro to Communication Networks\Ex01ITCN> python3 udpClient.py
b'DEAN COHEN & ROEI MESILATY' ('192.168.49.131', 12345)
PS C:\Users\dean5\OneDrive\Desktop\Year3\Intro to Communication Networks\Ex01ITCN>
```

udpServer.py - Ex01ITCN - Visual Studio Code

```python
import socket

s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
s.bind(('0.0.0.0', 12345))  # Binding to all available interfaces on port 12345

# print("Server is listening for incoming messages...")

while True:
    data, addr = s.recvfrom(1024)
    print(str(data), addr)
    # print("Received message:", str(data), "from", addr)
    s.sendto(data.upper(), addr)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
deanco100@deanco100-virtual-machine:~/Desktop/Advanced Programming/Ex01ITCN$ python3 udpServer.py
b'Dean Cohen & Roei Mesilaty' ('192.168.49.1', 62843)
```

Code explanation:

The code consists of a client and server that use a UDP socket to communicate via port 12345.

The client code creates a UDP socket and sends the message "Dean Cohen & Roei Mesilaty" in bytes (using the prefix b) to the server at address 192.168.49.131 on port 12345, receives a response from the server with a maximum size of 1024 bytes, and prints it.

The server code creates a UDP socket bound to address 0.0.0.0 on port 12345. The address 0.0.0.0 allows the server to listen to anyone connected to the network. It enters a loop to receive messages (while true), prints the received message and the sender's address, then sends back the message in uppercase to the client.

Relationship between Code and Traffic:

The client code sends a message to the server, and this message is captured in the first Wireshark packet (Image 1). The server receives this message, processes it, and sends the response back to the client. This response is captured in the second Wireshark packet (Image 2).

The code directly affected the network traffic by defining the contents of the UDP payload, the source and destination ports, and the overall flow of the communication between the client and server.

Image1 **(NOTE: len("Dean Cohen & Roei Mesilaty) = 26):**

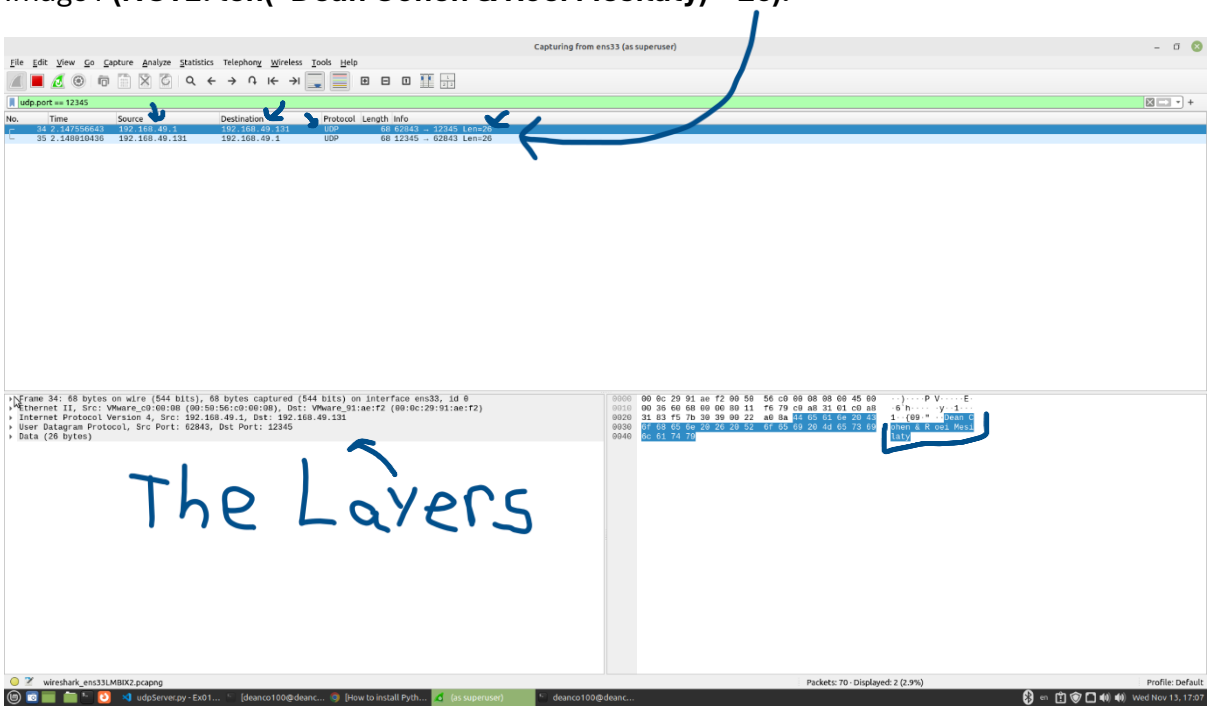
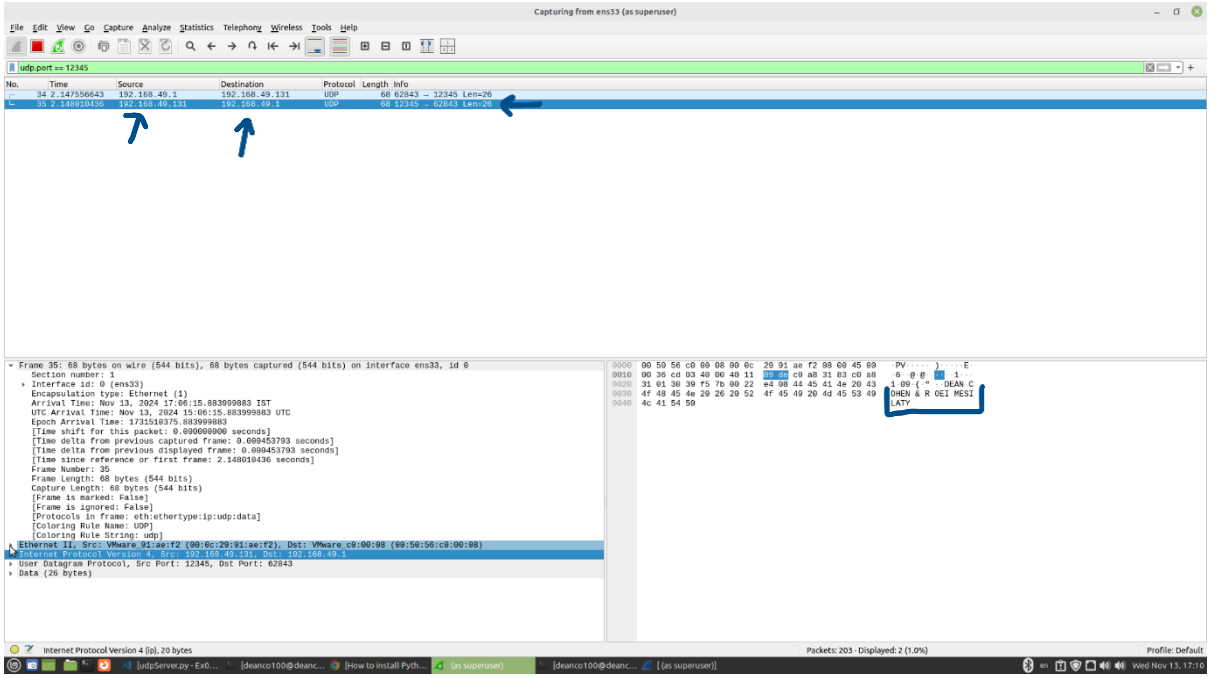


Image2:



Traffic Analysis in Wireshark –

Ethernet Layer – In the **Frame Details**, the **MAC Source Address** appears, which is the MAC address of VMware (appears as 00:0c:29:91:ae:f2).

MAC Destination Address – 00:50:56:c0:00:08.

EtherType – indicates that IPv4 addresses are being used.

34	2.147556643	192.168.49.1	192.168.49.131	UDP	68 62843 → 12345 Len=26
35	2.148010436	192.168.49.131	192.168.49.1	UDP	68 12345 → 62843 Len=26

```
Frame 34: 68 bytes on wire (544 bits), 68 bytes captured (544 bits) on interface ens33, id 0
    Section number: 1
    Interface id: 0 (ens33)
    Encapsulation type: Ethernet (1)
    Arrival Time: Nov 13, 2024 17:06:15.883546090 IST
    UTC Arrival Time: Nov 13, 2024 15:06:15.883546090 UTC
    Epoch Arrival Time: 1731510375.883546090
    [Time shift for this packet: 0.000000000 seconds]
    [Time delta from previous captured frame: 0.705450400 seconds]
    [Time delta from previous displayed frame: 0.000000000 seconds]
    [Time since reference or first frame: 2.147556643 seconds]
    Frame Number: 34
    Frame Length: 68 bytes (544 bits)
    Capture Length: 68 bytes (544 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:ip:udp:data]
    [Coloring Rule Name: UDP]
    [Coloring Rule String: udp]
Ethernet II, Src: VMware_c0:00:08 (00:50:56:c0:00:08), Dst: VMware_91:ae:f2 (00:0c:29:91:ae:f2)
    Destination: VMware_91:ae:f2 (00:0c:29:91:ae:f2)
    Source: VMware_c0:00:08 (00:50:56:c0:00:08)
    Type: IPv4 (0x0800)
    [Stream index: 1]
```

info

Internet Protocol (IP) Layer –

Source IP Address – 192.168.49.131 – the IP address of the client on the local network.

Destination IP Address – 192.168.49.1 – the IP address of the server.

Protocol – UDP.

User Datagram Protocol Layer – shows that the source port is 12345 (the port the client uses), and the destination port is 62843 – a randomly assigned port because the client doesn't specify a port, as explained in the lecture.

Length – indicates the length of the UDP header and data.

Application Layer – the data sent (Dean Cohen & Roei Mesilaty) is sent as text. The server receives it, converts it to uppercase, and sends the response back to the client (DEAN COHEN & ROEI MESILATY), as shown in the attached images.

Data Layer – In this section, we see the **UDP payload** containing the actual data sent from the client to the server.

34	2.147556643	192.168.49.1	192.168.49.131	UDP	68 62843 → 12345 Len=26
35	2.148010436	192.168.49.131	192.168.49.1	UDP	68 12345 → 62843 Len=26

```
▸ Frame 34: 68 bytes on wire (544 bits), 68 bytes captured (544 bits) on interface ens33, id 0
▸ Ethernet II, Src: VMware_c0:00:08 (00:50:56:c0:00:08), Dst: VMware_91:ae:f2 (00:0c:29:91:ae:f2)
▾ Internet Protocol Version 4, Src: 192.168.49.1, Dst: 192.168.49.131
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 54
    Identification: 0x6068 (24680)
  ▸ 000. .... = Flags: 0x0
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 128
    Protocol: UDP (17)
    Header Checksum: 0xf679 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.49.1
    Destination Address: 192.168.49.131
    [Stream index: 3]
▾ User Datagram Protocol, Src Port: 62843, Dst Port: 12345
    Source Port: 62843
    Destination Port: 12345
    Length: 34
    Checksum: 0xa08a [unverified]
    [Checksum Status: Unverified]
    [Stream index: 3]
    [Stream Packet Number: 1]
  ▸ [Timestamps]
    UDP payload (26 bytes)
▾ Data (26 bytes)
    Data: 4465616e20436f68656e202620526f6569204d6573696c617479
    [Length: 26]
```

Data

NOTE:

We have another packet (the response from the server to the client), that we can see in Image2. This packet has a similiar layers like the first packet, but it differs in

the src&dst IP's & MAC's & ports (swapped) and the message is now in Upper case.