Assignment-1: udpXtcp
UDP <--> TCP Converter

Introduction

The task is prepare a program that could

(i) Receive data from a TCP stream and send it to a target machine as UDP packets.

(ii) Receive UDP packets from a particular interface and send it to the target machine as TCP streams

Input parameters to the program are

(i) Mode of operation (tcp2udp or udp2tcp)

(ii)IP address and port number that can be used in the local machine

(iii)IP address and port number of the target machine

Command line arguments

```
./udpxtcp <mode -U or -T>
        -l <Local_Interface_Address in IP format> or
        -h <Local_Interface_Address in host name format>
        -d <Destination_Interface_Address in IP format> or
        -t  <Destination_Interface_Address in host name format>
```

Example:

(i) **./udpxtcp -U -l 130.233.150.150/5000 -d 130.234.100.100/5500**

- Converts TCP streams to UDP packets
- It is listens for incoming TCP connection requests at 130.233.150.150/5000
- For every accepted TCP connection, a new UDP socket is created to transfer the received data to the destination.
- Received TCP streams are sent as UDP packets to 130.234.100.100/5500

Command line..contd..

Let xyz.hut.fi be the host name for 130.233.150.150/5000 and dest.hut.fi be the host name for 130.234.100.100/5500. Then it should also be possible to invoke as **./udpxtcp -u -l xyz.hut.fi/5000 -d dest.hut.fi/5500**

(ii) **./udpxtcp -T -l 130.233.150.150/5000 -d 130.234.100.100/5500**

- Converts UDP packets as TCP streams
- It receives UDP packets at the interface 130.233.150.150/5000
- For every new source address(IP address and port number), it makes a new TCP connection to the 130.234.100.100/5500

Note: (130.233.100.1/5000 and 130.233.100.1/5001) are considered as different source addresses. (source address here, also includes port number)

Two Byte Header need to be added

- When the program converts UDP packet to TCP stream, a two byte header need to be added to the stream. The two byte header carries the length of the UDP packet.

- Similarly, when the program converts TCP stream to UDP packet, it uses the two bye header to reconstruct the original UDP packet..

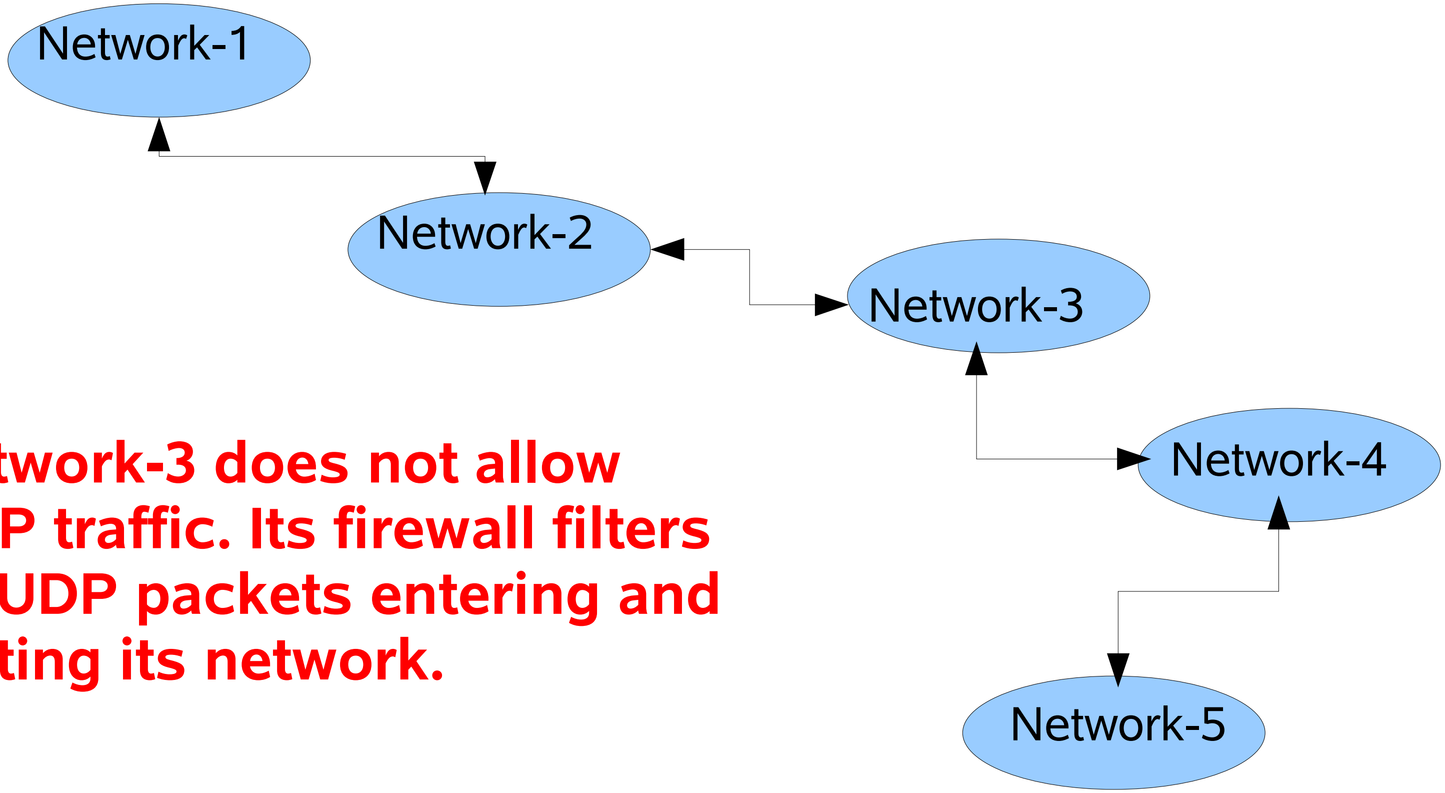
- Let an UDP packet be of length 'x' bytes. When this packet is sent using TCP, there is no packet boundary. TCP operates as streams and not as packets. So, When this UDP packet is received by udp2tcp converter, it has to add a two byte header carrying a the length value.

- When using a test-sender and a test-receiver, make sure that they conform to this requirement.

Sample Scenarios

- In the next few slides, some sample scenarios are explained.
- The purpose of the scenarios are to help in understanding the assignment requirements.
- If observed closely, the converters(udpXtcp) receive data from many sources and direct them to the same destination.
- Though the data are received at the same destination, they are segregated using the source address as key.

Sample scenario-1 explaining the assignment



Network-3 does not allow UDP traffic. Its firewall filters all UDP packets entering and exiting its network.

Scenario-1 contd...

('x' represents IP address of a machine and 'y' represent the port number)

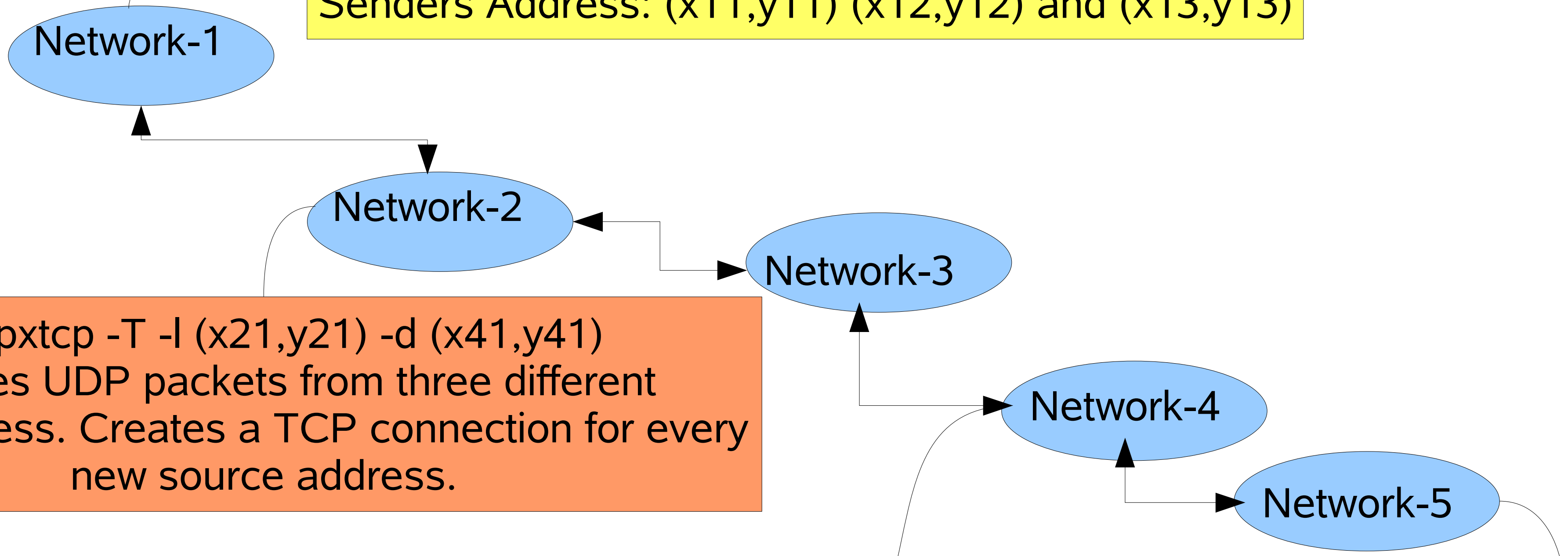
- There is a machine(IP address x11) in Network-1 which need to send data(from port number y11) to a machine in Network-5

(IP address->x51and port number->y51)

- It wants to send real time data(delay sensitive) using UDP. Assume that network-1 and 5 are geographically distant apart. So using TCP would have increased the delay.
- The UDP packets are not reaching Network-5, because Network-3's firewall filters UDP packets.
- So, we want to tunnel the packets using TCP while crossing Network-3.
- We have access to a system each in network-2(IP address-x21) and Network-4 (IP address-x41)

Using the assignment program, we can successfully tunnel the UDP packets.

Scenario-1 contd...



There are three senders in Network-1
who want to send data to (x51, y51)
Senders Address: (x11,y11) (x12,y12) and (x13,y13)

./udpxtcp -T -l (x21,y21) -d (x41,y41)
(receives UDP packets from three different source address. Creates a TCP connection for every new source address.

./udpxtcp -U -l (x41,y41) -d (x51,y51)
(Creates a UDP socket for each accepted TCP connection. The data received in the TCP connection is sent using the corresponding created UDP socket.

Destination machine receives UDP packet at the interface(x51,y51). The packets will be received from same IP address(x41) but from different port numbers

Sample Scenario-2

- Lets take a case in which few machines in network-1 wants to send UDP packets(that are delay sensitive) to a particular destination(in network-10). But network-1 and network-10(access networks) introduces high percentage of bit error(bits get corrupted). So, TCP is used for transport in the access networks. (the solution is not efficient, but at least we can get it retransmitted quickly)
- So, tcp2udp converter is run in Network-2 and a udp2tcp converter is run in Network-9.
- TCP streams arrive at (x21,y21) and it is then converted to UDP packets and sent to (x91, y91).
- At (x91,y91), the received UDP packets are converted back to TCP streams and sent to (x10, y10)

Test senders and receivers

- Write your own test sender and receiver to test the program

- Stress test with different packet sizes and with many number of concurrent flows.

- A unix utility 'nc' (installed in University computing machines) may help in debugging.

- Example:

```
% nc 130.233.x.y 5000
```

-> makes a TCP connection request to the specified address

```
% nc -l -p 5000
```

-> starts listening for TCP connections at the port number 5000