



Assignment 1: netbridge

TCP-UDP bridge
UDP-TCP bridge



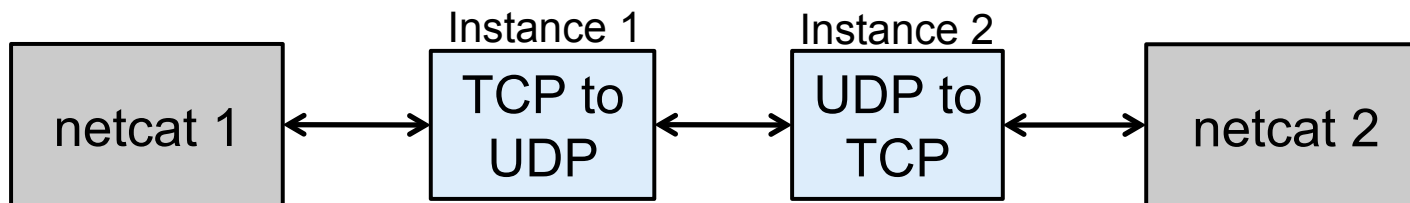
Goal of the assignment

- ▶ To get familiar with network socket programming.
- ▶ To develop a network application (netbridge) which can transport TCP data over UDP and vice versa.
- ▶ To test the application use netcat (nc), a unix utility.
- ▶ Furthermore, test the application by tunneling HTTP over netbridge to connect to a server.



Step 1/3

Scenario-1

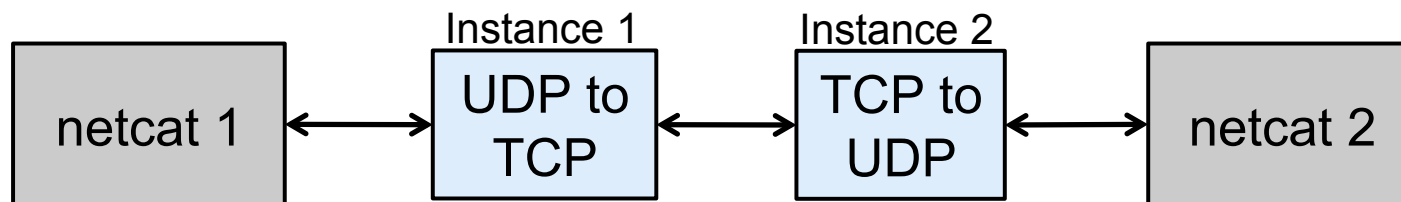


- ▶ the first instance of netbridge listen on a TCP port and accepts request from netcat 1. All the received TCP data is sent as UDP datagrams.
- ▶ the second instance of netbridge receives UDP datagrams, and translates them to TCP streams and sends to connects to netcat 2 which is in listening mode.
- ▶ Remember both instances bind to different UDP ports.
- ▶ Moreover netbridge should be able to handle multiple requests.



Step 2/3

Scenario-2

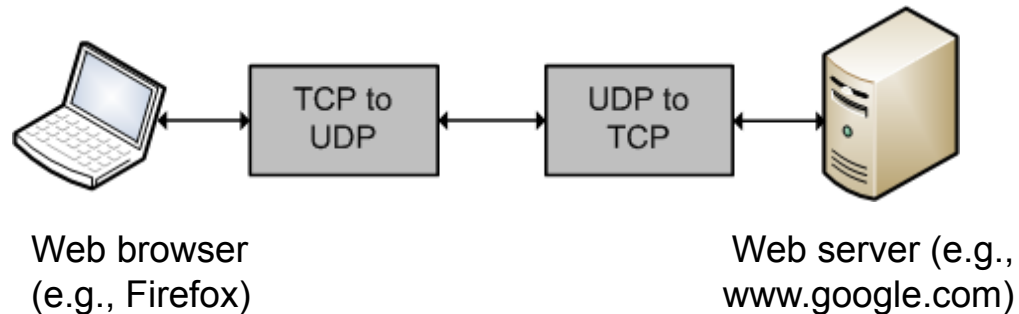


- ▶ the first instance receives UDP packets from netcat 1, connects with instance 2 over TCP and translates UDP packets into TCP data streams. A two bytes header, which carries the length of the UDP packet, needs to be added in the TCP data as TCP does not preserve message boundaries.
- ▶ The second instance parses the two byte TCP header to reconstruct the original UDP packet before forwarding to netcat 2 (which is in listening mode).



Step 3/3

- ▶ HTTP tunneling using the setup in scenario-1



Notes:

- ▶ Need to Parse HTTP GET requests.
- ▶ Example HTTP GET request:

```
GET http://www.google.com/index.html HTTP/1.1
```

```
Host: www.google.com
```

.....

- ▶ To retrieve the resource directly from the Google server, client would create a TCP connection to port 80 of the host "www.google.com" and send the request.
- ▶ The browser may issue multiple back-to-back requests.



Command line arguments

- ▶ `./netbridge <mode -TU or -UT> -l <Local_TCP_Port to listen>`
`-b <Local_UDP_Port to bind> -dt <TCP_Destination_Address to connect>`
`-du <UDP_Destination_Address to send>`
- ▶ `./netbridge -h` dumps meaningful help explaining all command line arguments
- ▶ Example for Scenario-1: (for Scenario-2 commands will be interchanged)
- ▶ Instance 1: `./netbridge -TU -l 2048 -b 2050 -du 130.233.100.100:2052`
- ▶ Instance 2: `./netbridge -UT -b 2052 -du 130.233.50.50:2050 -dt 130.233.150.150:2060`
- ▶ The application should be able to handle both hostname and IP Address



Program Output

- ▶ Each of the instances dumps information in the following format as soon as data is received either over TCP or UDP.

```
Receiving time   Src Addr -> Dest Addr   no. of bytes received  
Individual bytes in hexadecimal form
```

Example:

```
14:09:00   130.233.50.50:2048 -> 130.233.100.100:2050   70 bytes  
00 01 02 00 41 42 43 09 00 64 00 00 00 00 30 39 1a 3f 00 00 34  
00 02 67
```

- ▶ The program will handle Ctrl-C interrupt. After termination, it dumps the uptime and the total no. of bytes received so far.

Example:

```
10 minutes   423 bytes
```



Example netcat commands

- ▶ **% 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
- ▶ Note: To enable UDP, just add `-u` option to each of the above cases