np-nmb

A network message bus implemented for the course IS F462 Network Programming at BITS Pilani

Run

```
    gcc -o server server.c

    ./server

    gcc -o driver driver.c nmb.c

    ./driver <port number>

    gcc -o error error.c nmb.c

    sudo ./error (Error process creates raw socket and hence root
privileges are required)
```

To generate ICMP traffic

```
    nping --icmp --icmp-type 3 --icmp-code 0 --data
"450000292c21400080065a10ac112b8dd83ac4c4"  <ip address>

    nping --icmp --icmp-type 3 --icmp-code 1 --data
"450000292c21400080065a10ac112b8dd83ac4c4"  <ip address>
    host unreachable
```

Design Features

The Network Message Bus (NMB) has following characteristics (in brief)

- The message from process A on OS 1 to process B on OS 2 is routed as follows: Process A → Unix Domain Datagram Socket of OS 1 → UDP Socket of OS 1 → UDP Socket of OS 2 (via multicast) → Unix Domain Datagram Socket / Message queue of OS 2 → Process B.
- Each OS (local server) has a UDP socket (for receiving multicast messages), a Unix Domain Datagram Socket (for communication with its own processes), a message queue (which stores messages for processes which are down) and an error process.
- During multicast, if the destination IP Address provided is loop back address, then that message isn't multicasted
- When a receiving process is down, the local server buffers its messages in message queue and the process later reads those messages from the queue itself. Otherwise the message is directly sent to the Unix Domain Datagram Socket of the process.

NMB API

- **int msget_nmb(short port)**: It takes port number as a parameter and returns a Unix Domain Datagram socket fd for the client process. It sets socket permission as 0777 if port is zero (error process).
- **int msgsnd_nmb(int clientsockfd, char *ip, short port, void *msg, size_t msgsz)**: It is used by both client and error process to send a message to its local server (on Unix Domain Socket). It accepts the client socket fd, destination ip and port and message to be sent and its size as parameters and sends the message to the corresponding local server (on Unix Domain Socket). If error process uses this API, it passes NULL and 0 as destination ip and port respectively. It also sets the mtype of the message using destination ip and port number (mtype = 0 for error processes). It returns number of bytes sent to the local server.
- **int msgrcv_nmb(int clientsockfd, void *msg, size_t msgsz)**: It accepts client socket fd, a message buffer to read into and its size as parameters. It tries to fetch messages from message queue (if message queue is non empty) into the buffer. If the message queue is empty, it fetches the message from its socket (clientsockfd). It returns number of bytes read into the buffer.

Local server

- Each local server has a UDP socket, a Unix Domain Datagram Socket and a message queue. UDP Socket is used for receiving multicasted messages and Unix Domain Datagram Socket is used for relaying messages to its processes.
- The local server multicasts the messages which are sent by its processes and these messages are received by other servers through UDP socket.
- When a local server receives a multicasted message at its UDP socket, it calculates the port of the destination client by calculating its port number from the mtype of the message. If that client process is running, this message is directly sent to its UDS. Else the message is buffered in the message queue.

Error Process

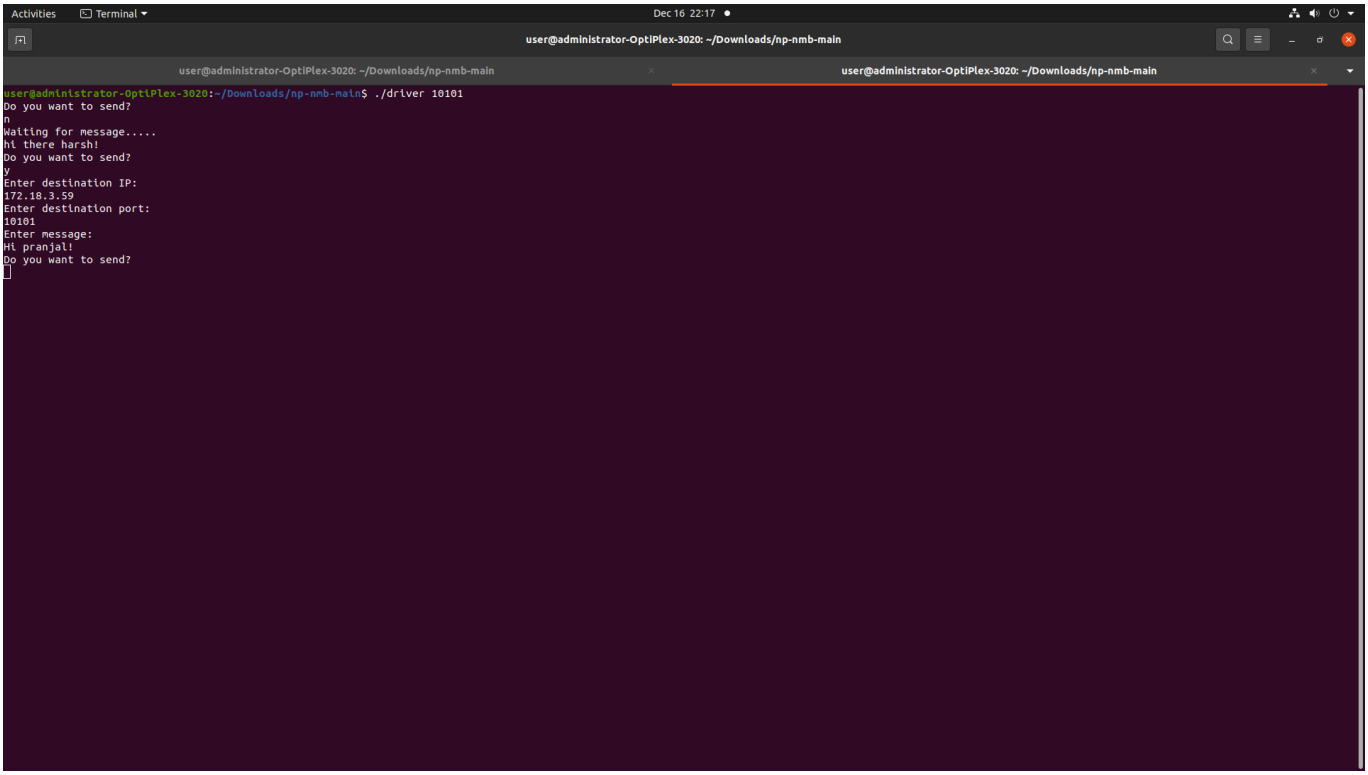
- Error Process uses the NMB API to communicate with the local server.
- Each error process has a raw socket (for reading ICMP messages) and a Unix Domain Datagram socket (for sending the received ICMP error to its local server).
- When raw socket receives a ICMP message, it extracts the type of ICMP message from its header and sends the error message to local server only if the type of the message is "ICMP Network Unreachable" or "ICMP Host Unreachable".

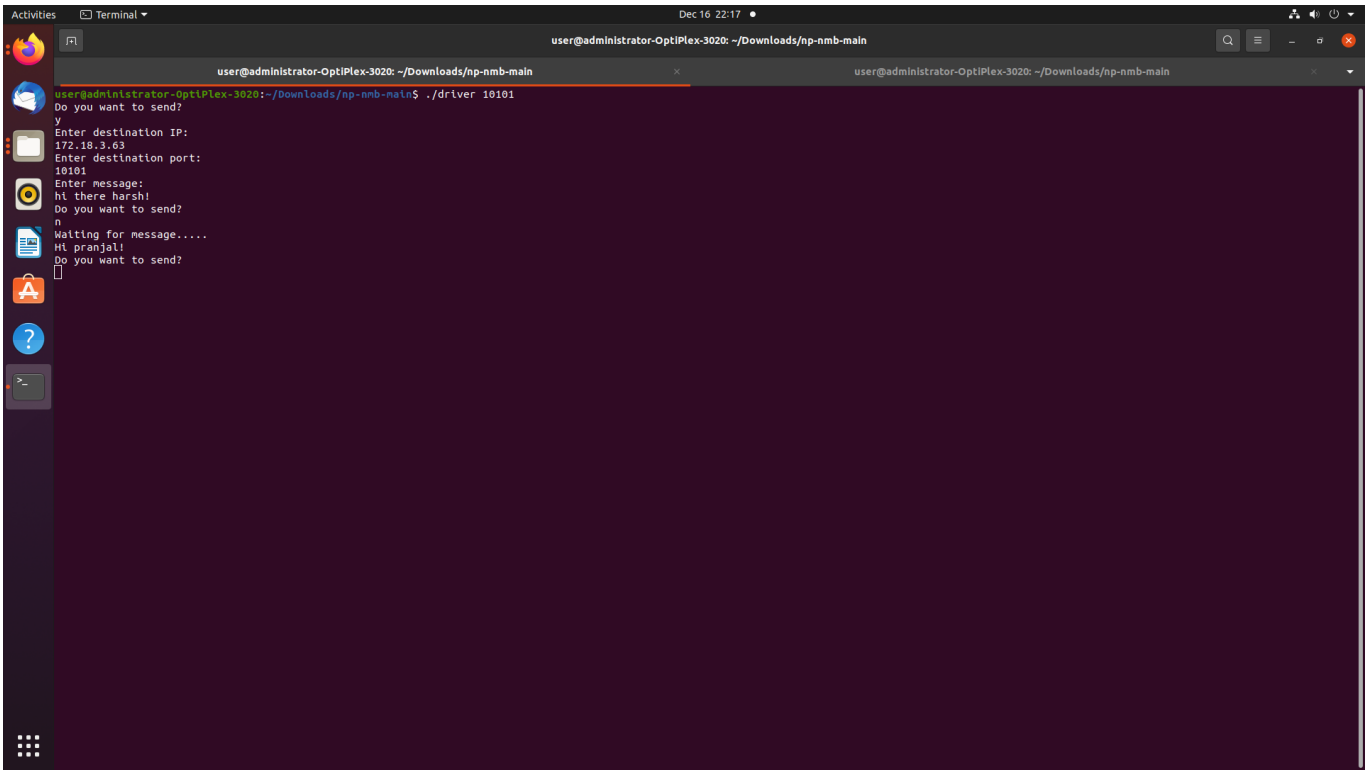
Assumptions

- The local Unix Domain Server runs at /tmp/1111.
- Unix Domain Sockets for clients run at /tmp/nmb.port_number
- Message queue's key is "."

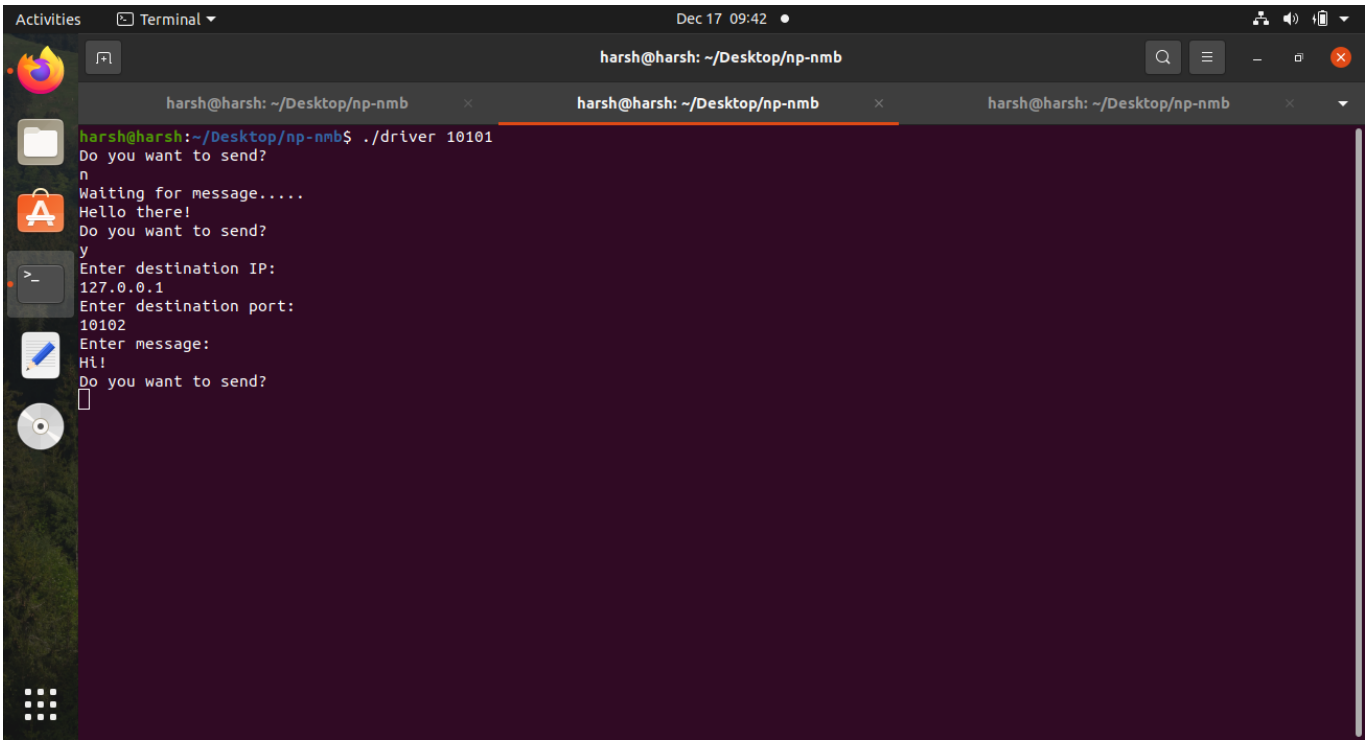
Screenshots

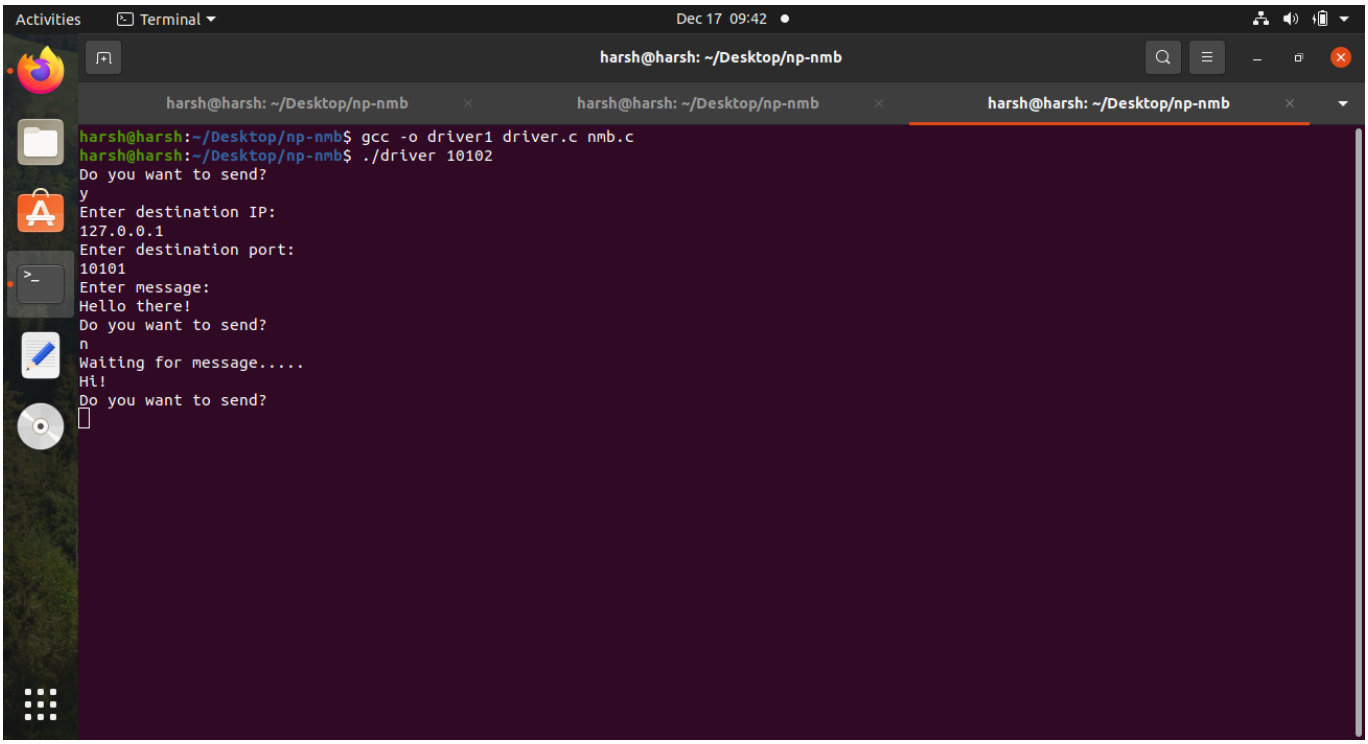
Example of communication between two processes on different systems





Example of communication between two processes on same system


Activities
Terminal
Dec 17 09:42
harsh@harsh: ~/Desktop/np-nmb
harsh@harsh: ~/Desktop/np-nmb
harsh@harsh: ~/Desktop/np-nmb
harsh@harsh: ~/Desktop/np-nmb
harsh@harsh:~/Desktop/np-nmb$ ./driver 10101
Do you want to send?
n
Waiting for message.....
Hello there!
Do you want to send?
y
Enter destination IP:
127.0.0.1
Enter destination port:
10102
Enter message:
Hi!
Do you want to send?


Activities
Terminal
Dec 17 09:42
harsh@harsh: ~/Desktop/np-nmb
harsh@harsh: ~/Desktop/np-nmb
harsh@harsh: ~/Desktop/np-nmb
harsh@harsh: ~/Desktop/np-nmb
harsh@harsh:~/Desktop/np-nmb$ gcc -o driver1 driver.c nmb.c
harsh@harsh:~/Desktop/np-nmb$ ./driver 10102
Do you want to send?
y
Enter destination IP:
127.0.0.1
Enter destination port:
10101
Enter message:
Hello there!
Do you want to send?
n
Waiting for message.....
Hi!
Do you want to send?