CS359 Computer Networks

Indian Institute of Technology, Patna

Assignment 2

Simple Communication using Datagram Socket using POSIX C

January 25, 2021

Instructions:

<u>**POSIX**</u>**:** **POSIX** stands for Portable Operating System Interface, and is an IEEE standard designed to facilitate application portability. **POSIX** is an attempt by a consortium of vendors to create a single standard version of UNIX. If they are successful, it will make it easier to port applications between hardware platforms.

Socket Programming: **Socket programming** is a way of connecting two nodes on a network to communicate with each other. One **socket**(node) listens on a particular port at an IP, while other **socket** reaches out to the other to form a connection. The server forms the listener **socket** while the client reaches out to the server.

UDP: **UDP** (User Datagram Protocol) is an alternative communications protocol to Transmission Control Protocol (TCP) used primarily for establishing low-latency and loss-tolerating connections between applications on the internet.

Objective:

The objective of this assignment is to get familiar with datagram sockets using POSIX C programming. The target is to establish a communication between two computers (processes) using datagram socket. A datagram socket uses a simple communication paradigm to transfer short messages between two computers (processes) without ensuring any reliability.

Problem Statement:

Your task will be to write two programs - one program corresponding to the server process and another program corresponding to the client process. The client process requests for the content of a file (by providing the file name) and the server process sends the contents of that file to the client.

For simplicity, we assume that the file is a simple text file that contains a set of words, with the first work being HELLO and the last word being END. We assume that HELLO and END are two special keywords that do not come anywhere except at the first line (HELLO) and the last line (END). The content of a sample file looks as follows.

HELLO
CAT DOG

```
TIGER
LION
HORSE ZEBRA COW
END
```

The transfer of the contents of the file works using a communication protocol as follows.

- The client first sends the file name to the server.
- The server looks for the file in the local directory, if the file is not there it sends back with a message NOTFOUND. By receiving this message, the client prints an error message “File Not Found” and exits.
- If the file is present, the server reads the first line of the file, which contains HELLO, and sends this message to the client.
- After receiving HELLO, the client creates a local file (a different file name from the requested one) and sends a message WORD1 to the server. This message indicates that the client is requesting for the first word.
- After receiving the message WORD1, the server sends the first word (after HELLO) to the client. The client writes this word to the local file after receiving it and sends the message WORD2 to request for the next word. This procedure continues for each word.
- On receiving WORDi, the server sends the i-th word to the client. This process continues until the client receives the keyword END.
- Once the client receives the keyword END, it closes the local file after writing the last word to the file.

Submission Instruction:
You should write two C programs - wordserver.c (contains the server program) and wordclient.c (contains the client program). Keep these two files in a single compressed folder (zip or tar.gz) having the name <roll number>_Assignment2.zip or <roll number>_Assignment2.tar.gz.