Chat Application using C-Programming

Table of Contents

I. Introduction

A chat application that aims to provide a fundamental understanding of networking concepts, specifically Transmission Control Protocol. It serves as a foundation to complex chat applications and provides a peek into the Networking aspect of C-Programming.

Socket programming provides us with a way to allow two nodes on a network to communicate with each other. One socket(node) listens on a particular port at an IP, while another socket reaches out to the other to form a connection. Server forms the listener socket while the client reaches out to the server. TCP is designed to send the data packets over the network. It ensures that data is delivered to the correct destination. TCP creates a connection between the source and destination node before transmitting the data and keep the connection alive until the communication is active.

TCP is suited for applications that require high reliability, and transmission time is relatively less critical. It is used by other protocols like HTTP, HTTPs, FTP, SMTP, Telnet. TCP rearranges data packets in the order specified. There is absolute guarantee that the data transferred remains intact and arrives in the same order in which it was sent. TCP does Flow

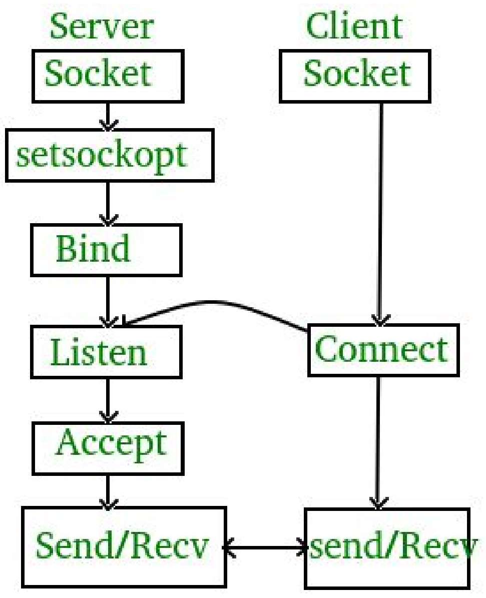
Control and requires three packets to set up a socket connection, before any user data can be sent. TCP handles reliability and congestion control. It also does error checking and error recovery. Erroneous packets are retransmitted from the source to the destination.
This chat application provides a server and client connection that will run on the command line. The two nodes can communicate with each other with the help of the command line interface of their host machine.

II. Requirements

- OS - Windows 7 +, MAC OS, Any GNU/Linux Distros
- CPU - 2GHz or faster processor (x86-compatible), multi-CPU configuration recommended
- RAM - 4GB
- Supported C/C++ Compiler

III. Design

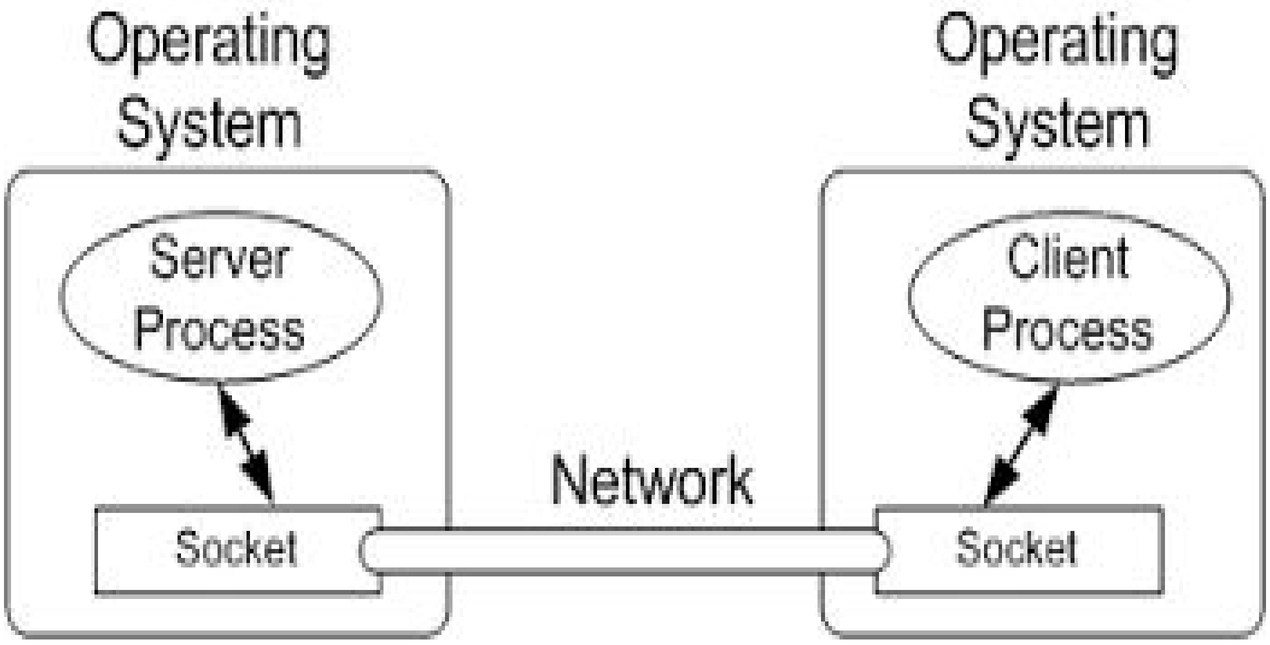
If we are creating a connection between client and server using TCP then it has few functionality like, TCP is suited for applications that require high reliability, and transmission time is relatively less critical.



Chat Application using C-Programming

A. TCP Server
1. **create()** - TCP socket creation.
2. **bind()**, Bind the socket to server address.
3. **listen()**, put the server socket in a passive mode, where it waits for the client to approach the server to make a connection
4. **accept()**, At this point, connection is established between client and server, and they are ready to transfer data.
5. Go back to **Step 3.**

B. TCP Client
1. Create TCP socket.
2. Connect the newly created client socket to the server.



IV. Design

1. Successful start and running of both server and client files.
2. Perform connection requests from client to server in order to check if the server is listening.
3. By 2. Check if the client is able to send connection requests.
4. Verify whether the connection persists between the client and the server.
5. Send data from client to server and check whether the output is received.
6. Send data from server to client and check whether the output is received.
7. Check for any data losses during transmission.
8. Check for any disconnections during transmission.

V. Conclusion

A chat application that aims to provide a fundamental understanding of networking concepts, specifically Transmission Control Protocol. It serves as a foundation to complex chat applications and provides a peek into the Networking aspect of C-Programming.

TCP does Flow Control and requires three packets to set up a socket connection, before any user data can be sent. TCP handles reliability and congestion control. It also does error checking and error recovery. Erroneous packets are retransmitted from the source to the destination. This chat application provides a server and client connection that will run on the command line. The two nodes can communicate with each other with the help of the command line interface of their host machine.