Project for CCN

TCP/IP Server

Without error checking.

```
int listenFd, connectFd;

struct sockaddr_in serverAddr;


listenFd = socket(AF_INET, SOCK_STREAM, 0); // get a tcp/ip socket


bzero(&serverAddr, sizeof(serverAddr));

serverAddr.sin_family = AF_INET;

serverAddr.sin_addr.s_addr = htonl(INADDR_ANY); // any iternet interface

                                                // on this server.

serverAddr.sin_port = htons(13);


bind(listenFd, (struct sockaddr_in *) &serverAddr, sizeof(serverAddr));


listen(listenFd, 5);


for ( ; ; ) {

      connectFd = accept(listenFd, (struct sockaddr_in *) NULL, NULL);

      // .. read and write operations on connectFd ..

      shutdown(connectFd, 2);

      close(connectFd);

}
```

Note that the above is an iterative server, which means that it serves one connection at a time.

To build a concurrent server, a fork is performed after the accept. The child process closes listenFd, and communicates using connectFd. The parent process closses connectFd, and then loops back to the accept to wait for another connection request.

TCP/IP Client code

```
int sockFd;

struct sockaddr_in serverAddr;


sockFd = socket(AF_INET, SOCK_STREAM, 0); // get a tcp/ip socket


bzero(&serverAddr, sizeof(serverAddr));

serverAddr.sin_family = AF_INET;

inet_pton(AF_INET, serverName, serverAddr.sin_addr); // host IP #

serverAddr.sin_port = htons(13);


connect(sockFd, (struct sockaddr_in *) serverAddr, sizeof(serverAddr));

      // .. read and write operations on sockFd ..

shutdown(sockFd, 2);

close(sockFd);
```