```c
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <string.h>

#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>

#include <pthread.h>
#include <sys/types.h>

#define PORT 3500
#define MAXLINE 1024

pthread_mutex_t t_lock;
pthread_cond_t t_cond;

int client_index = 0;

struct data_r
{
        char buf[MAXLINE];
        int num;
};

void * thread_func(void *data)
{
        int sockfd = *((int *)data);
        char buf[MAXLINE];
        int num;

        pid_t pid = getpid();
        pthread_t tid = pthread_self();

        printf("pid:%u, tid:%x\n", (unsigned int)pid, (unsigned int)tid);

        struct data_r rdata;

        read(sockfd, (char*)&rdata, sizeof(rdata));

        printf("read data : %s and %d\n", rdata.buf, ntohl(rdata.num));
        strcpy(buf,rdata.buf);
        int len = strlen(buf);

        while(1)
        {
                pthread_mutex_lock(&t_lock);
//              pthread_cond_wait(&t_cond, &t_lock);

                num = ntohl(rdata.num);
                num ++;
                rdata.num = htonl(num);
                strcpy(buf, rdata.buf);
                for(int i=0; i<strlen(buf); i++)
                {
                        rdata.buf[i] = buf[i+1];
                }
                rdata.buf[len-1] = buf[0];
                rdata.buf[len] = 0;

                write(sockfd, (char*)&rdata, sizeof(rdata));

                pthread_mutex_unlock(&t_lock);
        }

        close(sockfd);
        return NULL;
};
```



```c
int main(int argc, char **argv)
{
        int listen_sockfd, client_sockfd;
        int addr_len;
        struct data_r rdata;
        int num;
        char buf[MAXLINE];
        pthread_t thread_id[3];

        struct sockaddr_in server_addr, client_addr;

        if( (listen_sockfd  = socket(AF_INET, SOCK_STREAM, 0)) == -1)
        {
                perror("Error ");
                return 1;
        }

        memset((void *)&server_addr, 0x00, sizeof(server_addr));
        server_addr.sin_family = AF_INET;
        server_addr.sin_addr.s_addr = htonl(INADDR_ANY);
        server_addr.sin_port = htons(PORT);

        if( bind(listen_sockfd, (struct sockaddr *)&server_addr, sizeof(server_a
ddr)) == -1)
        {
                perror("bind error ");
                return 1;
        }

        if(listen(listen_sockfd, 5) == -1)
        {
                perror("listen error ");
                return 1;
        }

        while(1)
        {
                addr_len = sizeof(client_addr);
                client_sockfd = accept(listen_sockfd,
                        (struct sockaddr *)&client_addr, &addr_len);
                if(client_sockfd == -1)
                {
                        perror("accept error : ");
                        return 1;
                }

                if(client_index == 3)
                {
                        close(client_sockfd);
                        continue;
                }
                printf("\n New Client Connect : %s\n", inet_ntoa(client_addr.sin_addr));

                pthread_mutex_init(&t_lock, NULL);
                pthread_cond_init(&t_cond, NULL);

                pthread_create(&thread_id[client_index], NULL, thread_func, (voi
d*)&client_sockfd);

                pthread_cond_broadcast(&t_cond);

                client_index++;


        }

        return 0;
}
```