## 《Linux 操作系统设计实践》实验六：综合应用

实验环境：Ubuntu 14.04.5 LTS

### 程序运行演示说明

程序包含client.c server,c

编译运行：

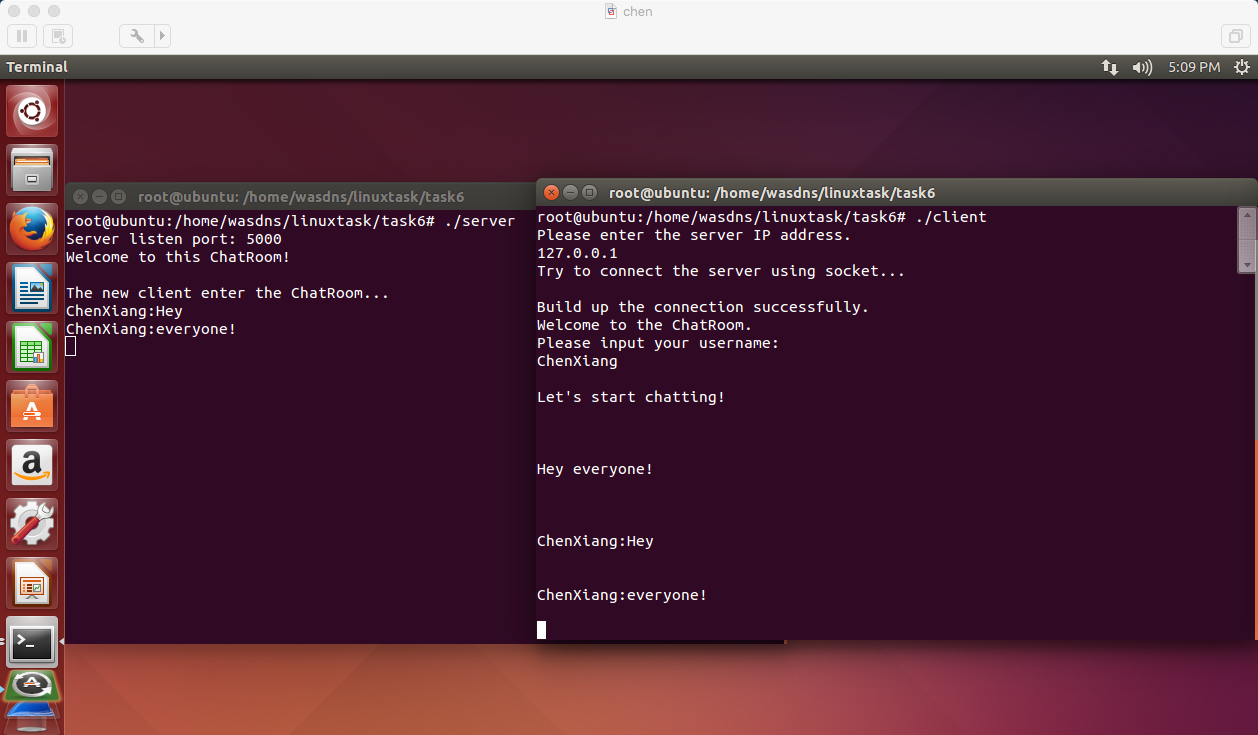
将所有文件置于统一文件夹下，使用

gcc server.c -lpthread –o server

gcc client.c –lpthread –o client

进行编译。

主要界面截图：



### 实验结果总结

本实验将实验代码进行细分，将代码分为多个模块分工合作，最后再进行数据和函数的对接。

### 编程工作总结

程序的编写是非常劳累的，虽然有先前实验的积累和经验，但需要有一个完整的规划，一点细小的偏差就容易导致实现效果与预想大不相同。各个模块之间的联系也是需要非常仔细处理的事情。这次实验也在代码组织和管理上给我上了很好的一课。

### 源程序代码

client.c

/\* gcc client.c -o client -lpthread \*/

#include <stdlib.h>

#include <stdio.h>

#include <string.h>

#include <errno.h>

#include <netdb.h>

#include <sys/types.h>

#include <netinet/in.h>

#include <sys/socket.h>

#include <sys/wait.h>

#include <unistd.h>

#include <pthread.h>

#define TRUE 1

#define PORT 5000

int quit = 0;

void \*get\_server(void\* sockfd) {

char buf[1024];

int rev;

if (((int)sockfd) < 0)

printf("\nCannot get the server information.\n");

else {

printf("\n\007\n");

for (;;) {

if (!quit) {

if ((rev = recv((int)sockfd, buf, 1024, 0)) > 0)

printf("\n\007%s\n", buf);

if (rev == 0) {

printf("\nThe server ends up the connection.\n");

quit = 1;

continue;

}

printf("\n");

} else {

close((int)sockfd);

break;

}

}

return(NULL);

}

}

int main() {

int connfd,snd,slenth;

struct sockaddr\_in server;

struct hostent \*hp;

char honame[20], msg2[1024], msg1[1024], cln[102], qstr[] = {"Quit"};

pthread\_t tid;

printf("Please enter the server IP address.\n");

scanf("%s\*", honame);

printf("Try to connect the server using socket...\n");

if ((connfd = socket(AF\_INET, SOCK\_STREAM, 0)) < 0)

printf("Error: Cannot create the socket.\n");

if ((hp = gethostbyname(honame)) == NULL) {

printf("Error: Cannot get the server IP address\n");

exit(1);

} else printf(" \n");

memcpy(&server.sin\_addr, hp->h\_addr, hp->h\_length);

server.sin\_family = AF\_INET;

server.sin\_port = htons(PORT);

if (connect(connfd, (struct sockaddr\*)&server, sizeof(server)) < 0) {

printf("Error: Cannot connect to the server.\n");

exit(1);

}

printf("Build up the connection successfully.\n");

printf("Welcome to the ChatRoom.\n");

printf("Please input your username:\n");

scanf("%s",msg1);

slenth = strlen(msg1);

msg1[slenth] = ':';

msg1[slenth+1] = '\0';

strcpy(cln, msg1);

pthread\_create(&tid, NULL, &get\_server, (void\*)connfd);

printf("\nLet's start chatting!\n");

while (TRUE) {

printf("\n");

scanf("%s", msg2);

if (strcmp(msg2, qstr) == 0) {

close(connfd);

quit = 1;

} else {

strcat(msg1, msg2);

snd = send(connfd, msg1, strlen(msg1)+1, 0);

strcpy(msg1, cln);

if (snd < 0) printf("\n \n");

}

}

return 0;

}

server.c:

/\* gcc server.c -o server -lpthread \*/

#include <stdlib.h>

#include <stdio.h>

#include <string.h>

#include <errno.h>

#include <netdb.h>

#include <sys/types.h>

#include <netinet/in.h>

#include <sys/socket.h>

#include <sys/wait.h>

#include <unistd.h>

#include <pthread.h>

#define MAXLINE 1000

#define LISTENQ 20

#define PORT 5000

#define MAXFD 20

FILE \*fp;

int i, maxi = -1;

int client[MAXFD];

void \*get\_client(void \*sockfd) {

char buf[MAXLINE];

int rev;

if ((int)sockfd < 0) printf("\nThe new client cannot enter the ChatRoom\n");

else {

printf("\nThe new client enter the ChatRoom...\n");

do {

memset(buf, 0, sizeof(buf));

if ((rev = recv((int)sockfd, buf, 1024, 0)) < 0)

printf("\nCannot read the user information.\n");

if (rev == 0)

printf("\nThe user ends up the connection.\n");

else {

printf("%s\n", buf);

for(i = 0;i <= maxi; i++)

send(client[i],buf,strlen(buf)+1,0);

fputs(buf,fp);

}

} while (rev != 0);

fclose(fp);

}

close((int)sockfd);

return(NULL);

}

int main() {

int connfd,listenfd,sockfd;

socklen\_t length;

fp = fopen("student.txt","w");

struct sockaddr\_in server;

struct sockaddr tcpaddr;

pthread\_t tid;

listenfd = socket(AF\_INET,SOCK\_STREAM,0);

if (listenfd < 0) {

printf("Error: Cannot build up the socket.\n");

exit(1);

}

memset(&server, 0, sizeof(server));

server.sin\_family = AF\_INET;

server.sin\_port = htons(PORT);

server.sin\_addr.s\_addr = htonl(INADDR\_ANY);

if (bind(listenfd, (struct sockaddr\*)&server, sizeof(server))<0) {

printf("Error: Cannot bind the socket.\n");

exit(1);

}

length = sizeof(server);

if (getsockname(listenfd,(struct sockaddr\*)&server,&length)<0) {

printf("Error: Cannot get the server listen port.\n");

exit(1);

}

for(i = 0; i < MAXFD; i++) {

client[i] = -1; //initialize the client column

}

listen(listenfd, LISTENQ);

printf("Server listen port: %d\n", ntohs(server.sin\_port));

printf("Welcome to this ChatRoom!\n");

for (;;) {

connfd = accept(listenfd, &tcpaddr, &length);

for (i = 0; i < MAXFD; i++) {

if (client[i] < 0) {

client[i] = connfd;

break;

}

}

if (i == MAXFD-1) {

printf("The user number has exceeded the threshold.\n");

exit(0);

}

if (i > maxi) maxi = i;

pthread\_create(&tid, NULL, &get\_client, (void\*)connfd);

}

return 0;

}