[Linux下基于socket和多线程的聊天室小程序](http://blog.csdn.net/robot__man/article/details/52460733)

标签： [socket](http://www.csdn.net/tag/socket)[多线程](http://www.csdn.net/tag/%e5%a4%9a%e7%ba%bf%e7%a8%8b)[聊天室](http://www.csdn.net/tag/%e8%81%8a%e5%a4%a9%e5%ae%a4)

2016-09-07 16:23 91人阅读 [评论](http://blog.csdn.net/robot__man/article/details/52460733#comments)(0) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/robot__man/article/details/52460733#report)

http://static.blog.csdn.net/images/category_icon.jpg 分类：

Linux下C应用编程（32） http://static.blog.csdn.net/images/arrow_triangle%20_down.jpg

版权声明：本文为博主原创文章，遵循GPL协议精神，转载请注明出处。

要求：基于TCP编写，一个聊天室最多100人。   
客户端：   
　　1、用户需要登录，登录时只需要输入一个昵称即可无需判断昵称是否重复（如果其他功能都ok考虑）   
　　2、用户登录后连接服务器端，进入聊天室   
　　3、用户可以输入聊天信息，也可以收到别人的聊天信息。   
　　4、用户可以用某个特殊单词代表退出聊天室。

服务器端：   
　　1、启动服务器，开放端口   
　　2、等待客户端的连接，每连接上一个客户端，启动一个线程   
　　3、在线程中与客户端交互，交互过程：如果有客户端登录、退出、提交聊天，都应该发给所有的客户端。需要　　　保存所有客户端。

额外功能：可以考虑实现TCP的文件传输。

server.c

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <unistd.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <sys/socket.h>

#include <pthread.h>

int sockfd;//服务器socket

int fds[100];//客户端的socketfd,100个元素，fds[0]~fds[99]

int size =100 ;//用来控制进入聊天室的人数为100以内

char\* IP = "192.168.10.143";

short PORT = 10222;

typedef struct sockaddr SA;

void init(){

sockfd = socket(PF\_INET,SOCK\_STREAM,0);

if (sockfd == -1){

perror("创建socket失败");

exit(-1);

}

struct sockaddr\_in addr;

addr.sin\_family = PF\_INET;

addr.sin\_port = htons(PORT);

addr.sin\_addr.s\_addr = inet\_addr(IP);

if (bind(sockfd,(SA\*)&addr,sizeof(addr)) == -1){

perror("绑定失败");

exit(-1);

}

if (listen(sockfd,100) == -1){

perror("设置监听失败");

exit(-1);

}

}

void SendMsgToAll(char\* msg){

int i;

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

if (fds[i] != 0){

printf("sendto%d\n",fds[i]);

send(fds[i],msg,strlen(msg),0);

}

}

}

void\* service\_thread(void\* p){

int fd = \*(int\*)p;

printf("pthread = %d\n",fd);

while(1){

char buf[100] = {};

if (recv(fd,buf,sizeof(buf),0) <= 0){

int i;

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

if (fd == fds[i]){

fds[i] = 0;

break;

}

}

printf("退出：fd = %dquit\n",fd);

pthread\_exit((void\*)i);

}

//把服务器接受到的信息发给所有的客户端

SendMsgToAll(buf);

}

}

void service(){

printf("服务器启动\n");

while(1){

struct sockaddr\_in fromaddr;

socklen\_t len = sizeof(fromaddr);

int fd = accept(sockfd,(SA\*)&fromaddr,&len);

if (fd == -1){

printf("客户端连接出错...\n");

continue;

}

int i = 0;

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

if (fds[i] == 0){

//记录客户端的socket

fds[i] = fd;

printf("fd = %d\n",fd);

//有客户端连接之后，启动线程给此客户服务

pthread\_t tid;

pthread\_create(&tid,0,service\_thread,&fd);

break;

}

if (size == i){

//发送给客户端说聊天室满了

char\* str = "对不起，聊天室已经满了!";

send(fd,str,strlen(str),0);

close(fd);

}

}

}

}

int main(){

init();

service();

}

client.c

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <pthread.h>

int sockfd;//客户端socket

char\* IP = "192.168.10.143";//服务器的IP

short PORT = 10222;//服务器服务端口

typedef struct sockaddr SA;

char name[30];

void init(){

sockfd = socket(PF\_INET,SOCK\_STREAM,0);

struct sockaddr\_in addr;

addr.sin\_family = PF\_INET;

addr.sin\_port = htons(PORT);

addr.sin\_addr.s\_addr = inet\_addr(IP);

if (connect(sockfd,(SA\*)&addr,sizeof(addr)) == -1){

perror("无法连接到服务器");

exit(-1);

}

printf("客户端启动成功\n");

}

void start(){

pthread\_t id;

void\* recv\_thread(void\*);

pthread\_create(&id,0,recv\_thread,0);

char buf2[100] = {};

sprintf(buf2,"%s进入了聊天室",name);

send(sockfd,buf2,strlen(buf2),0);

while(1){

char buf[100] = {};

scanf("%s",buf);

char msg[131] = {};

sprintf(msg,"%s:%s",name,buf);

send(sockfd,msg,strlen(msg),0);

if (strcmp(buf,"bye") == 0){

memset(buf2,0,sizeof(buf2));

sprintf(buf2,"%s退出了聊天室",name);

send(sockfd,buf2,strlen(buf2),0);

break;

}

}

close(sockfd);

}

void\* recv\_thread(void\* p){

while(1){

char buf[100] = {};

if (recv(sockfd,buf,sizeof(buf),0) <= 0){

return;

}

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

}

}

int main(){

init();

printf("请输入您的名字：");

scanf("%s",name);

start();

return 0;

}

　　刚开始，我声明的那个全局数组int fds[99],它是有99个元素，fds[0]~fds[98],我以为是到fds[99]，结果数组越界，向fds[99]指向的未知文件描述符发了个数据，而fds[99]指向的是监视进程的，向它发任何数据，进程都会莫名奇妙的终止。于是，服务器莫名奇妙的挂掉了，朋友用gdb调试，才发现Program received signal SIGPIPE, Broken pipe.这个问题我自己看了一天半也没发现，最后还是在群里面，别人帮提出来的，感悟有以下几点：   
1、[**Linux**](http://lib.csdn.net/base/linux)下写程序，也有调试工具gdb，要学会使用，程序出错了，如果看不出来，就可以来调试；   
2、程序员不是一直埋头写程序就行了，还是要多和别人交流，一切以解决问题为根本。