**《Linux操作系统设计实践》**

**实验报告**

学号： 031502340

姓名： 易伟航

年级： 2015级

学院：数学与计算机科学学院

专业： 计算机类

本组其它成员：学号 姓名

学号 姓名

实验时间：2017－2018学年第一学期

# 《Linux操作系统设计实践》

## 实验三：网络编程

### 实验环境：

Ubuntu 17.04 64位

### 实验内容：

程序代码：

#include <sys/types.h>

#include <sys/socket.h>

#include <stdio.h>

#include <string.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <unistd.h>

int readPort()

{

int port;

while (port < 1024 || port > 65535)

{

printf("请输入端口（1024-65535）： ");

scanf("%d", &port);

}

return port;

}

int main()

{

int server\_fd;

int client\_fd;

int mesage\_length;

struct sockaddr\_in server\_addr;

struct sockaddr\_in client\_addr;

int port;

char sendBuf[256];

char receiveBuf[256];

char mode[7];

printf("选择运行模式（server/client）： ");

if (fgets(mode, sizeof(mode), stdin) == 0)

{

printf("can not read mode!\n");

return 1;

}

port = readPort();

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

server\_addr.sin\_family = AF\_INET;

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

server\_addr.sin\_port = port;

mesage\_length = sizeof(server\_addr);

if (!strcmp(mode, "server"))

{

bind(server\_fd, (struct sockaddr \*)&server\_addr, mesage\_length);

listen(server\_fd, 10);

client\_fd = accept(server\_fd, (struct sockaddr \*)&client\_addr, (socklen\_t \* \_\_restrict) & mesage\_length);

printf("请稍候，等待客户端上线……\n");

getchar();

while (1)

{

printf("\n等待客户端发送数据……\n");

if (recv(client\_fd, receiveBuf, 256, 0) > 0) //接收消息recv(c\_fd,buf,256,0)>0

{

if (receiveBuf[0] == '#')

{

printf("接受到退出信号，正在退出!\n");

break;

}

printf("收到客户端消息:\n %s\n\n", receiveBuf);

printf("请输入回信(输入 # 退出): ");

if (fgets(sendBuf, sizeof(sendBuf), stdin) == 0)

{

printf("read error!\n");

return 1;

}

send(client\_fd, sendBuf, sizeof(sendBuf), 0);

if (sendBuf[0] == '#')

{

printf("正在退出!\n");

break;

}

}

}

close(client\_fd);

}

else if (!strcmp(mode, "client"))

{

int newsockfd = connect(server\_fd, (struct sockaddr \*)&server\_addr, mesage\_length); //发送连接服务器的请求

if (newsockfd == -1)

{

perror("连接失败\n");

return 1;

}

getchar();

while (1)

{

printf("请输入要发送的数据(输入 # 退出): ");

if (fgets(sendBuf, sizeof(sendBuf), stdin) == 0)

{

printf("read error!\n");

return 1;

}

if (sendBuf[0] == '#')

{

printf("正在退出!\n");

break;

}

send(server\_fd, sendBuf, sizeof(sendBuf), 0);

printf("\n请稍候，等待服务端发送数据……\n");

if (recv(server\_fd, receiveBuf, 256, 0) > 0)

{ //rebuf[sizeof(rebuf)+1]='\0';

if (receiveBuf[0] == '#')

{

printf("接受到退出信号，正在退出!\n");

break;

}

printf("收到服务器消息:\n%s\n", receiveBuf);

}

}

close(server\_fd);

}

else

{

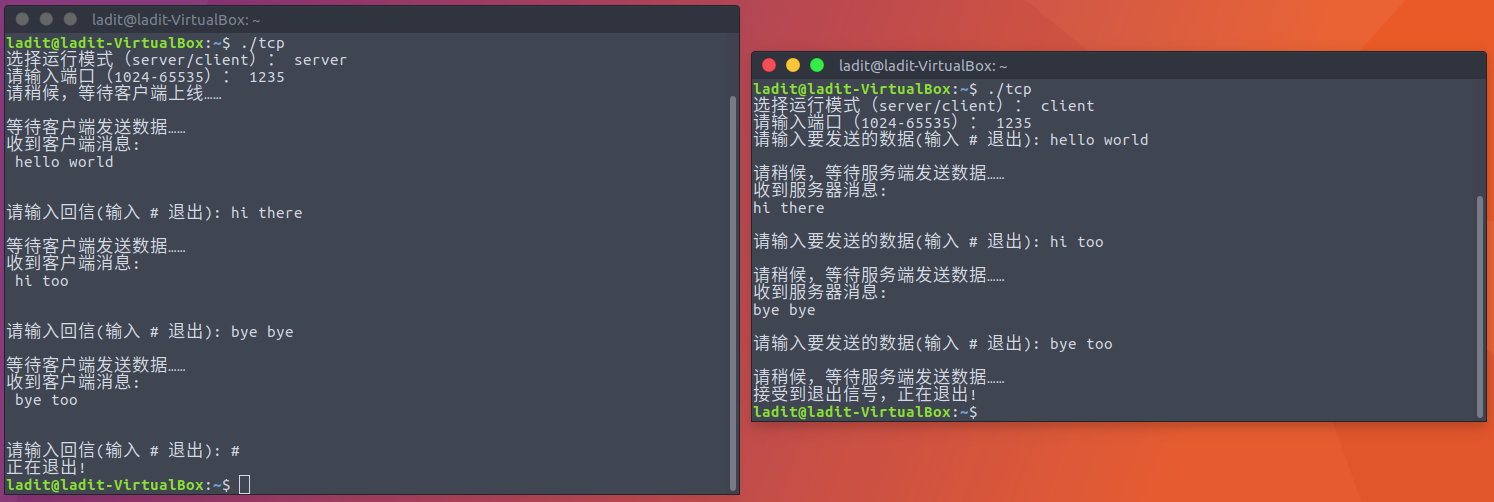
printf("输入错误！\n");

}

return 0;

}

运行截图：



### 实验总结：

实现服务端和客户端通过面向连接的套接口进行双向通信，基于示例代码改动，单文件单程序，支持选择端口，可以相互发送消息，服务端和客户端输入#时通知对方退出。