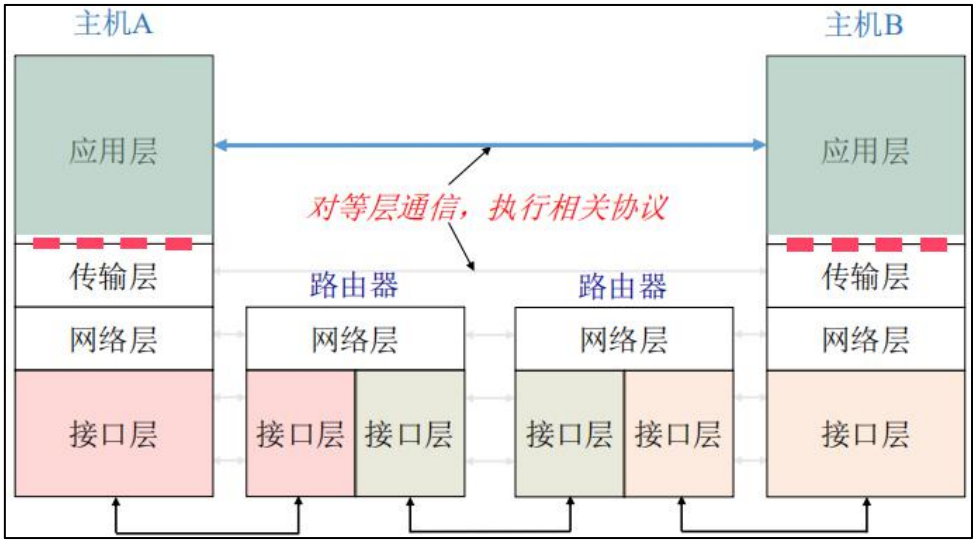
实验一 使用套接字实现聊天室

1813069 郭怡霏

一、实验目的：

1. 给出你聊天协议的完整说明。
2. 利用 C 或 C++语言，使用基本的 Socket 函数完成程序。不允许使用 CSocket 等封装后的类编写程序。
3. 使用流式 Socket 完成程序。
4. 程序应有基本的对话界面，但可以不是图形界面。程序应有正常的退出方式。
5. 完成的程序至少应能实现两个用户之间的英文和中文聊天。
6. 编写的程序应结构清晰，具有较好的可读性。

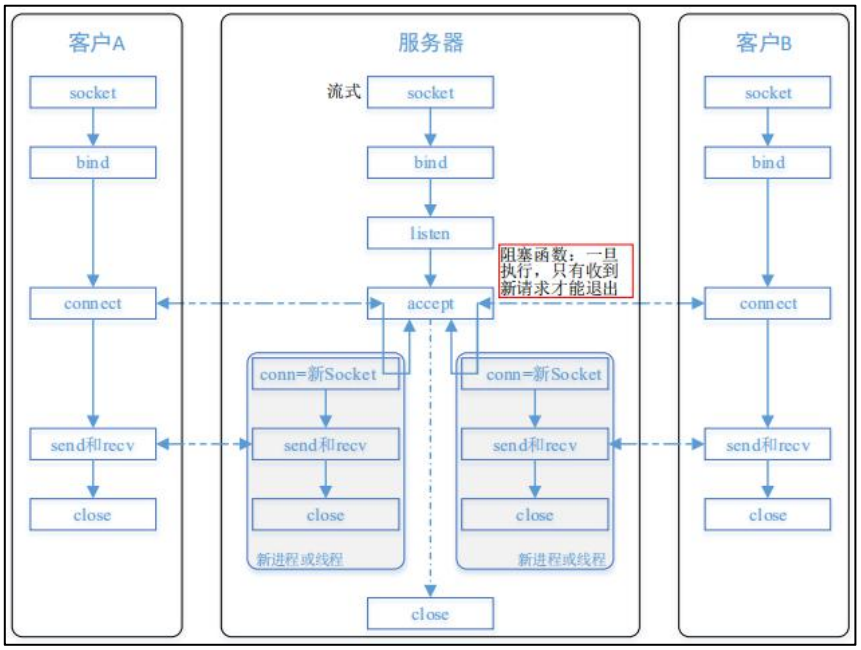
二、实验原理：



如上图所示，根据 TCP/IP 的体系结构，主机自顶向下分为应用层、传输层、网络层、接口层四层，当主机 A 向主机 B 发送信息时，信息由 A 自顶向下层层封装，经由路由器传递至 B 的底部接口层之后自底向上层层解封，B 的应用层获得数据。A 与 B 的信息通过以上的途径传输，但两台主机对等层的信息的识别与交流需执行相关的协议。进程是主机中运行的程序，不同主机上的进程通信通过交换信息来完成。本次实验中，采用较为常见的客户/服务器（C/S）模型，客

户向服务器发出服务请求并接收服务器的响应，服务器等待客户的请求并为客户提供服务。应用层的协议包括信息的类型、语法、语义、处理方式等内容，需自行定义；传输层可提供 TCP 和 UDP 服务，TCP 服务具有面向连接、可靠传输、流量控制、拥塞控制等特性，而 UDP 服务则不可靠，不能提供流量控制、拥塞控制等。

socket（套接字）是 TCP/IP 网络操作系统为网络程序开发提供的典型网络编程界面，分为数据报套接字和流式套接字。本次实验使用流式套接字，使用 TCP 协议，支持主机之间面向连接的、顺序的、可靠的、全双工字节流传输（通讯可双向同时传输）。流式套接字采用 TCP 服务，流程图如下图所示：



三、实验重点：

1. 自定义应用层协议；
2. 使用基本 socket 函数，流式套接字；
3. 支持中英文聊天。

四、实验环境：

1. 操作系统：Oracle VM VirtualBox 6.1.14 r140239 (Qt 5.6.2)
   Ubuntu 18.04.5 LTS (64 bit)
2. IDE：Visual Studio Code 1.50.1
3. 使用语言：C
4. 使用库：stdio.h string.h stdlib.h unistd.h netinet/in.h arpa/inet.h sys/socket.h
   pthread.h time.h

五、实验步骤：

自定义协议：

1. 统一使用 1207 号服务端口；
2. 单次发送消息长度不得超过 100 个字；
3. 客户端只输入“exit”时断开连接。

本实验由服务器和客户端两个部分构成，下面分别对其进行介绍：

1. server.c：

本文件为服务器源文件，由以下几部分构成：

（1）预设部分：此处包含了九个头文件，用宏定义限定了聊天室的人数上限，设置了服务器的 socket、可连接的客户端空数组、主机 ip 地址、所使用的端口号、socket 地址（来自 socket.h）、用户名长度限制、时间。回送地址（127.x.x.x）是本机回送地址（Loopback Address），即主机 IP 堆栈内部的 IP 地址，主要用于网络软件测试以及本地机进程间通信，无论什么程序，一旦使用回送地址发送数据，协议软件立即返回，不进行任何网络传输;127.0.0.1 分配给 loopback 接口,loopback 是一个特殊的网络接口(可理解成虚拟网卡)，用于本机中各个应用之间的网络交互。只要操作系统的网络组件是正常的，loopback 就能工作。

（2）init 函数：创建了服务器的流式套接字 serverSocket（来自 socket.h），更新并存储地址信息（来自 in.h），完成 socket 和地址的 bind 操作（来自 socket.h），开始监听客户端的 socket 连接请求情况（来自 socket.h）。

（3）sendAll 函数：在客户端连接数组中进行遍历，如果有连接的客户端，就将发出操作写入监听日志，并将参数 msg 用 send 函数发送给该客户端（来自 socket.h）。

（4）server_thread：被用作线程的参数。实时检查，需要退出时，从客户端连接数组中找到该客户端并清除连接，在监听日志中记录退出信息，终止线程；否则把服务器接收到的信息进行发送（来自 socket2.h、pthread.h）。

（5）server 函数：提示已经启动服务器，在一个大循环中不断地设置套接字地址存放（来自 in.h），使用 accept 函数接收来自客户端的连接请求（来自 socket.h），生成一个对应的小 socket，分配一个线程并启动（来自 pthread.h）。

（6）main 函数：调用 init 函数和 server 函数。

2. client.c：

本文件为客户端源文件，由以下几部分构成：

（1）预设部分：同 1（1）。

（2）init 函数：创建了客户端的流式套接字 clientSocket（来自 socket.h），更新并存储地址信息（来自 in.h），向客户端发起 connect 连接请求（来自 socket.h）。

（3）recv_thread：被用作线程的参数。实时接收从服务器发来的信息并显示。

（4）start 函数：创建一个用于接收消息的线程，send 该用户进入聊天室的消息，持续接收输入并发送至服务器，直到用户输入“exit”四个字母，最后发送一条退出聊天室的消息（来自 socket.h）。

（5）main 函数：调用 init 函数以初始化套接字，提示用户输入用户名并规范化，进入聊天室，调用 start 函数，退出聊天室时以文字提示。

编译：

在 vscode 命令行打开当前文件夹，输入如下代码：

```
idaguo@idaguo-VirtualBox:~/codes/network/lab1$ gcc -o client client.c -lpthread
idaguo@idaguo-VirtualBox:~/codes/network/lab1$ gcc -o server server.c -lpthread
```

运行：

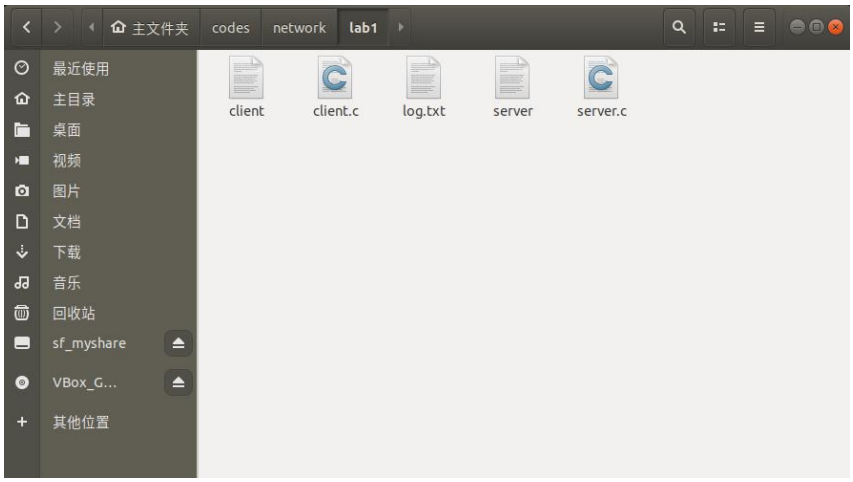
在命令行打开当前文件夹，输入如下代码，即可运行：

```
idaguo@idaguo-VirtualBox:~/codes/network/lab1$ ./server
```

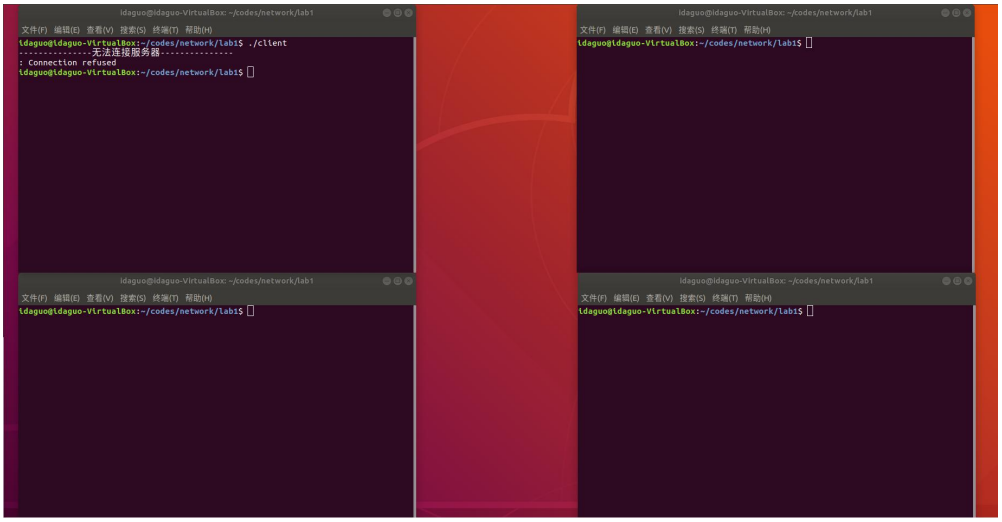
```
idaguo@idaguo-VirtualBox:~/codes/network/lab1$ ./client
```

六、实验结果：

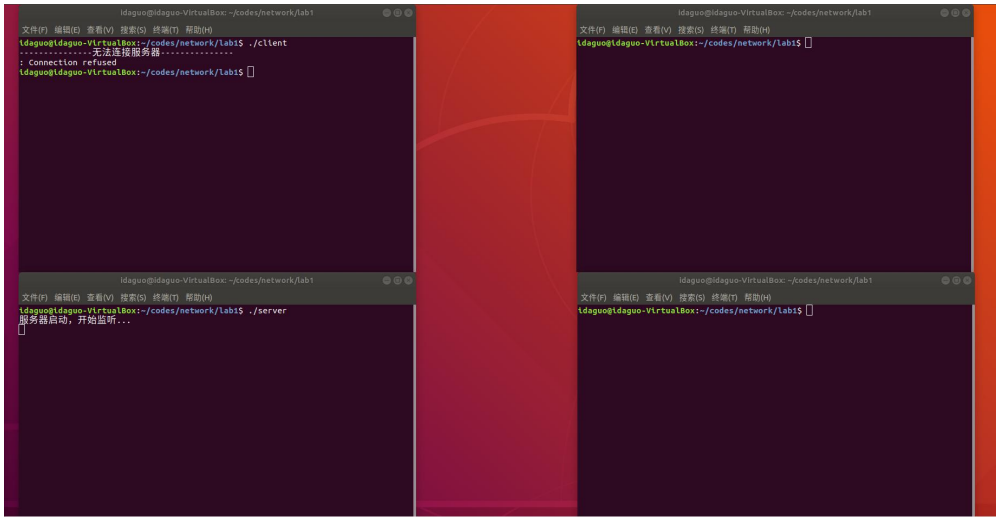
因可以设置聊天室人数上限，故为了便于演示，设置上限为 2，即最多只能有两人同时在聊天室，演示如下：



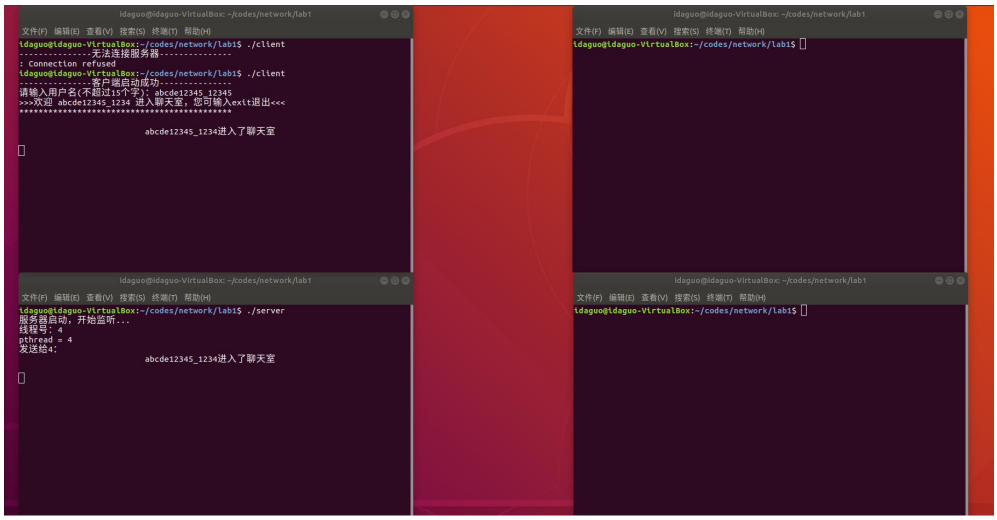
以上是文件列表，分别是客户端可执行文件、客户端源代码、监听日志、服务器可执行文件、服务器源代码。通过在命令行输入./xxx（xxx 为可执行文件名）来运行。



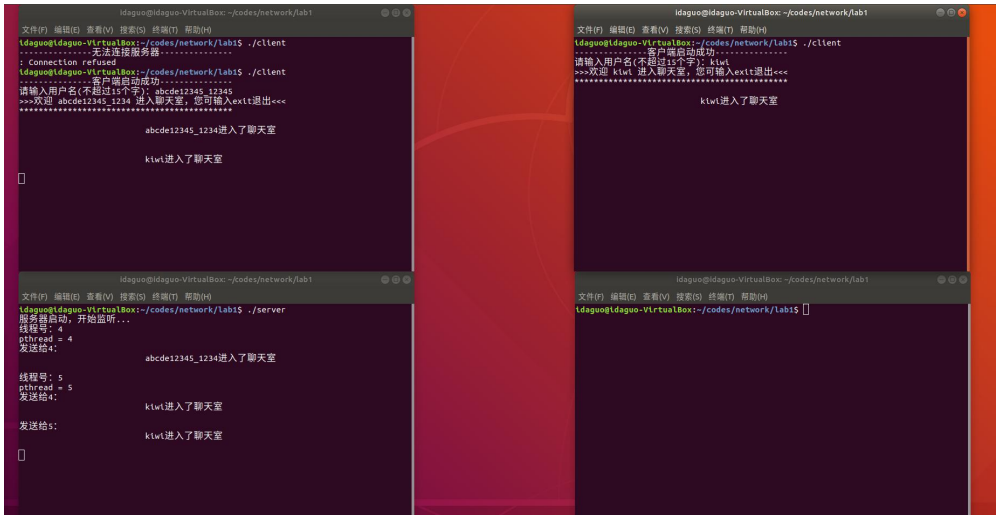
（1）在未开启服务器时开启客户端，显示无法连接



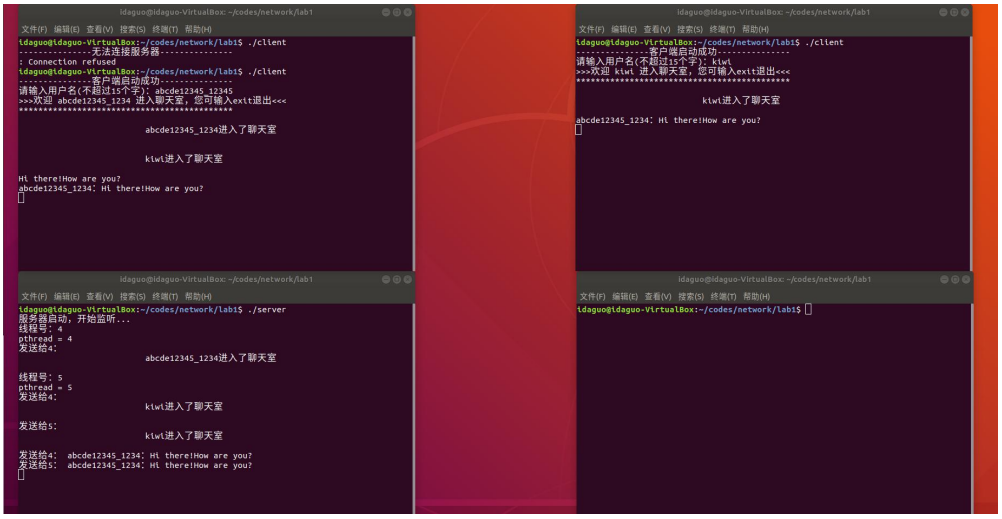
（2）开启服务器，服务器持续监听中



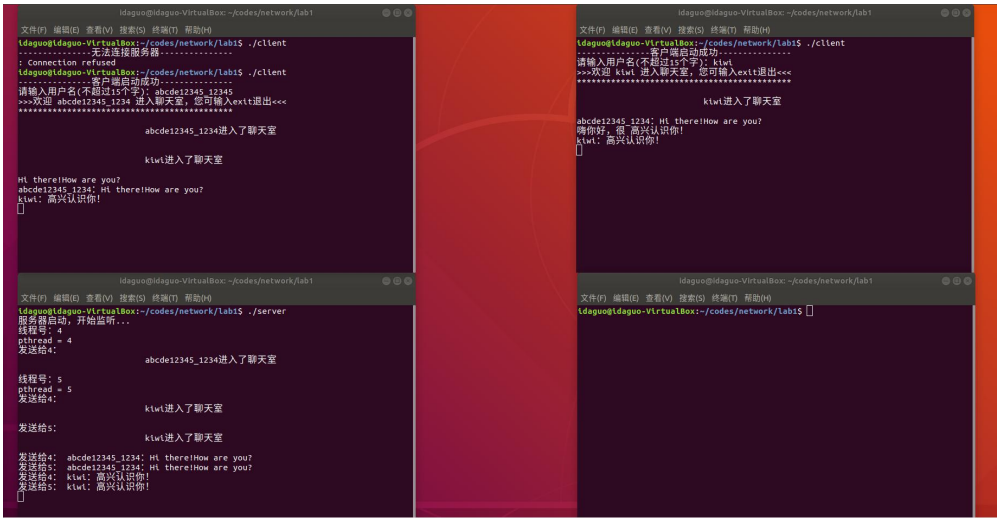
（3）开启客户端 1，输入用户名 abcde12345_12345，只截取前 15 位，进入聊天室



（4）开启客户端 2，输入用户名 kiwi，进入聊天室

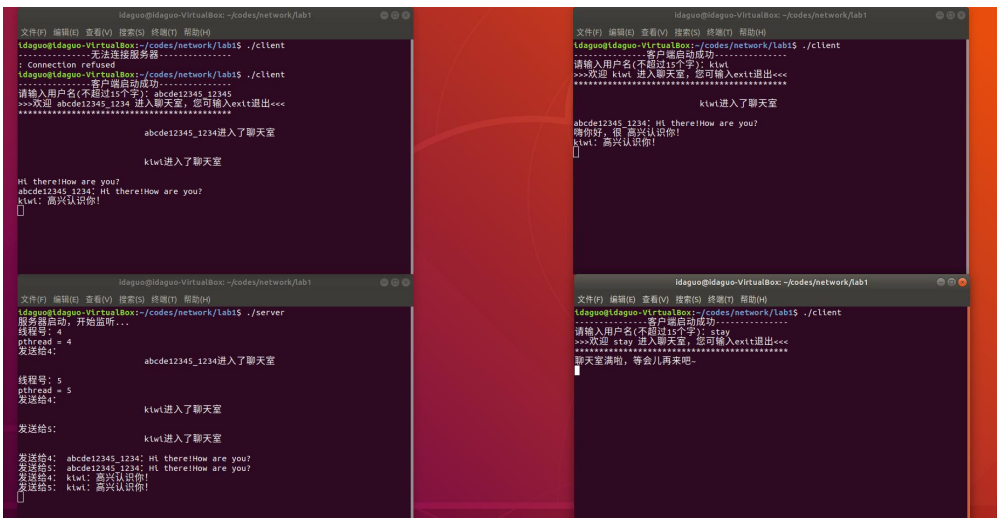


（5）用户 1 发送了英文信息，在用户 1 页面、用户 2 页面和服务器监听页面均有显示

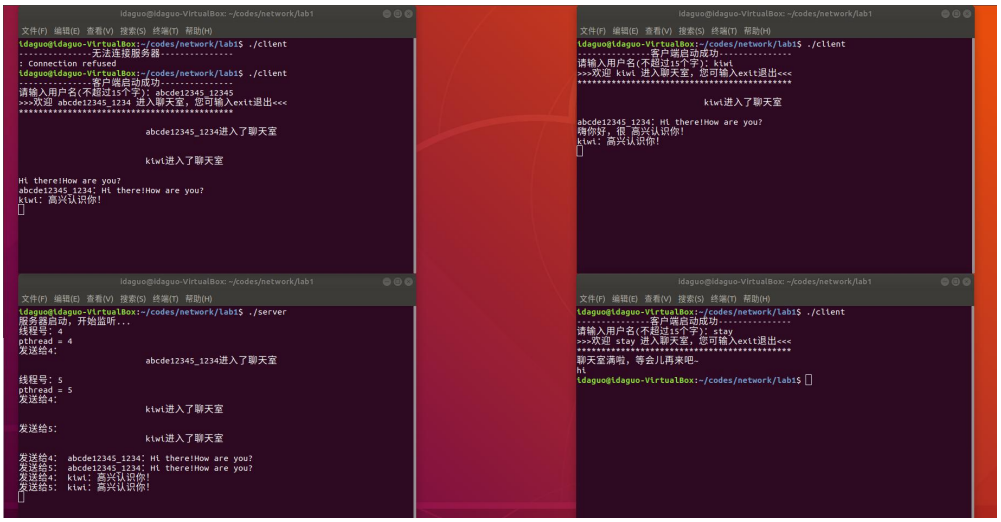


（6）用户 2 发送了中文信息，在用户 1 页面、用户 2 页面和服务器监听页面均有显示

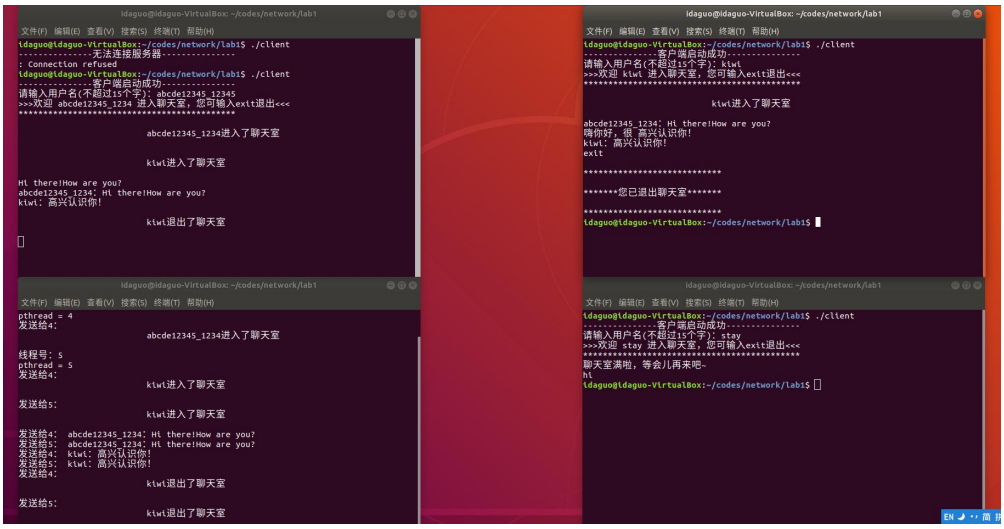
【此处输入中文之后按删除键，只能删除后一部分，前一部分只留在输入行，没有真正输入】



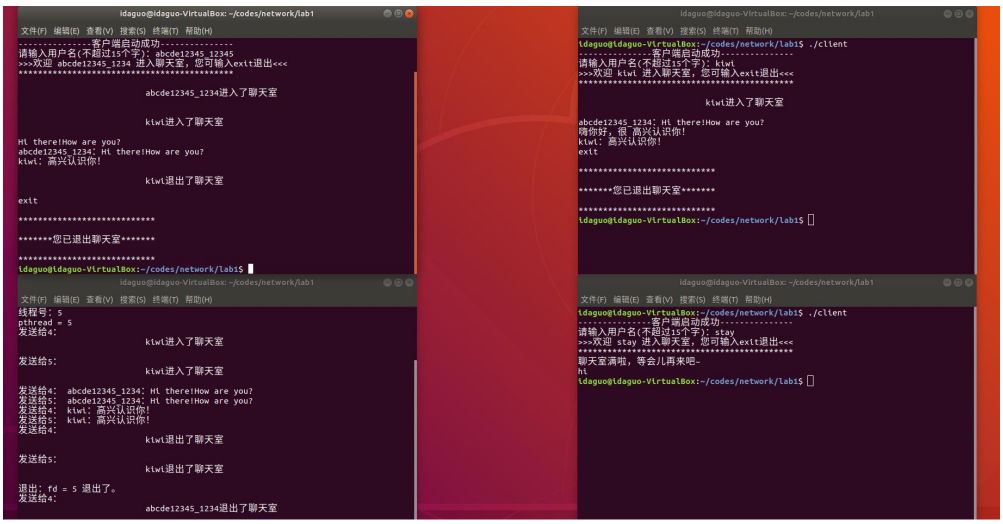
（7）开启客户端 3，输入用户名 stay，聊天室已满员，故无法进入聊天室



（8）客户端 3 不论输入什么都会直接退出，因为聊天室已满



（9）用户 2 退出聊天室，在用户 1 页面、用户 2 页面和服务器监听页面均有显示



（10）用户 1 退出聊天室，在用户 1 页面和服务器监听页面均有显示

（11）用 gedit 打开监听日志 log.txt，如上图所示

七、问题与不足：

1. 用户名输入，想限定在 15 个字以内，超过的全部截断，于是考虑了以下方法：

（1）scanf(“%14s”,name)：超过的部分无法截断，留在后续的输出；

（2）前 15 个字用 getchar，之后传入空循环：无法修改之前的昵称；

（3）先构造一个较长的缓冲区，输入之后再截取前 15 位送入 name：若用户数过多，则空间消耗过大。

经过权衡，出于准确性和空间考虑，决定使用方法（2）。

2. 聊天输入，采用 scanf("%s",inputBuf)的方式，发现输入英文时只读取了第一个单词，scanf 函数无法读入空格，故设置限定 100 字输入缓冲区，采用逐个 getchar()的方法来输入。经过检验，发现此处输入数字、英文字母、英文符号均可全部删除掉，但输入中文则删除不干净（见实验结果第 6 步），暂时没有想到如何解决。

3. 使用 vscode 的 code runner 插件直接编译运行，报错“对 pthread_create 未定义的引用”：程序中使用了线程的操作，但 pthread 库不是 Linux 系统默认的库，连接时需要链接-lpthread，故在命令行使用 gcc -o xxx xxx.c -lpthread 命令生成可执行文件，再使用./xxx 命令运行。