武汉大学电子信息学院计算机网络
课程设计报告

基于 TCP 的局域网多人聊天室

小　组　成　员：夏可为　2015301200168
潘鑫海　2015301200201
高迎紫　2015301200290
刘斯齐　2015301200294
张瑞祥　2015301200087
指　导　教　师　：杨珺老师

二〇一八年六月

摘　要

随着计算机网络技术的发展,各种各样基于网络的应用也随之诞生,比如基于互联网的信息发布,通信,数据共享等等。局域网的发展也同样迅速。很多政府机构,企业,学校,都是先以一个统一的局域网联结在一起,再分别接入 INTERNET 因此基于局域网的即时通信工具,就这样应运而生了。所以本课程设计提出了一个基于 TCP 局域网多人聊天室设计,并在 Visual Studio 所提供的 win32 平台上实现了将客户端和服务端两个应用程序在局域网内相互通信的功能,并用多线程实现不同的并行任务，且进行了人性化的界面设计,使用起来更加简单方便；功能也十分合理,又易于扩展以及个性化定制。本系统采用典型的 CS(客户端/服务器)的框架模式,在该方案中采用 Socket（套接字)原理实现网络通信,最终实现了通过服务器中转的文字聊天、文件传输功能。本报告中介绍了该系统实现的基础原理、实现方法与结果展示。

关键词：TCP； Socket； 多线程； 通信

第一章 系统实现的基础原理

1. 传输控制协议（Transmission Control Protocol，TCP）

TCP 是 TCP/IP 体系中一个复杂的协议，其最主要的特点为:

（1） TCP 是面向连接的运输层协议，应用程序在使用 TCP 协议之前，必须建立 TCP 连接。

（2） 每条 TCP 连接只能连接两个端点，每一条 TCP 连接只能是点对点。

（3） TCP 提供可靠交付的服务，通过 TCP 连接传送的数据无差错、不丢失、不重复，并且按序到达。

（4） TCP 提供全双工通信。TCP 允许通信双方的应用进程在任何时候都能收发数据。TCP 连接的两端都设有发送缓存和接收缓存，用来临时存放双向通信的数据。在发送时，应用程序在把数据传送给 TCP 缓存后，就可以处理其它数据，而 TCP 在合适的时候把数据发送出去。在接收时，TCP 把收到的数据放入缓存，上层的应进程在合适的时候读取缓存中的数据。

（5） 面向字节流。TCP 中的“流”指的是流入到进程或从进程流出的字节序列。“面向字节流”的含义是：虽然该应用程序和 TCP 的交互是一次一个数据块，但 TCP 把应用该程序交下来的数据仅仅看成是一串无结构的字节流。TCP 并不知道所传送的字节流的意义。TCP 不保证接收方应用程序所收到的数据块和发送方应用程序所发送的数据块具有对应大小的关系，但是接收方应用程序收到的字节流和发送方应用程序发出的字节流完全一样。

2. TCP 的连接

每条 TCP 连接有两个端点，TCP 连接的端点为套接字（Socket）或插口。根据 RFC793 的定义：端口号拼接到 IP 地址即构成了套接字。因此，套接字的表示方法是在点分十进制的 IP 地址后面写上端口号，中间用冒号或逗号隔开。每条 TCP 连接唯一地被通信两端的两个端点（即两个套接字）所确定，即：

TCP 连接 ::= { socket1，socket2 } = { (IP1:port1)，(IP2：port2) }

3. socket 机制

socket 是应用进程为了获得网络通信服务而与操作系统进行交互时使用的一种机制。在网络编程时，常把 socket 作为应用进程和运输层协议之间的接口，在 socket 以上的进程是受应用程序控制的，而在 socket 以下的运输层协议软件则是由计算机操作系统控制。因此，只要应用程序使用 TCP/IP 协议进行通信，就必须通过 socket 与操作系统交互并请求其服务。

当应用进程（客户或服务器）需要使用网络进行通信时，必须首先发出 socket 系统调用，请求操作系统为其创建一个“socket”。这个调用实际效果是请求操作系统把网络通信需要的一些系统资源（存储器空间、CPU 时间、网络带宽等）分配给该应用进程。操作系统为这些资源的总合用一个叫做套接字描述符的号码来表示，然后把这个套接字描述符返回给应用进程。此后，应用进程所进行的网络操作（建立连接、收发数据、调整网络通信参数等）都必须使用这个套接字描述符。因此，几乎所有的网络系统调用都把这个套接字描述符作为 socket 的许多参数中的第一个参数。在处理系统调用的时候，通过套接字描述符，操作系统就可以识别出应该使用哪些资源来完成应用进程所请求的服务。通信完毕后，应用进程通过一个用于关闭 socket 的 close 函数通知操作系统回收与该套接字描述符相关的所有资源。下图为使用 socket 机制的系统调用流程图。

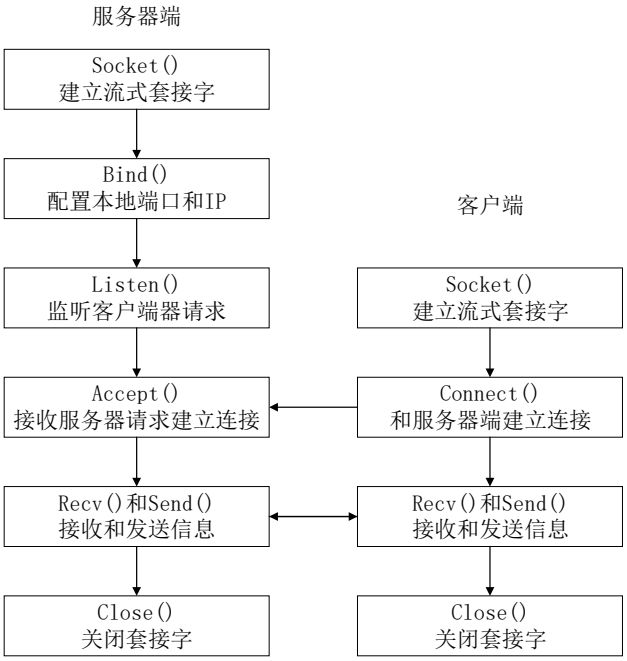

图 1.1 利用 socket 机制的系统调用流程图

3. 停止等待协议

TCP 是面向流的提供可靠的全双工通信的协议。为保证发送端和接收端之间能够进行可靠通信，最简单的方式就是使用停止等待协议。停止等待协议用于通信系统中，两个相连的设备相互发送信息时使用，以确保信息不因丢包或包乱序而丢失，是最简单的自动重传请求方法。当发送端 A 发送一个数据包后，A 处于等待状态，接收端 B 接收到了一个数据包后会发送一个确认“消息”，当 A 端接收到这个消息后，才会继续发送下一组的数据包。

为了更深入的理解停止等待协议，我们的程序模仿停止等待协议也进行了一定的流量控制（虽然计算机底层运行 TCP 协议本身已经保证了可靠通信）。客户端每发送一条消息给服务器之后会阻塞等待，直到收到服务器返回的确认消息才会发送下一条消息或者继续监听用户输入。这样一方面实现了简易版的停止等待协议，另一方面也保证了客户端的数据发送速率不会过快导致服务器端来不及接收而崩溃。

第二章 系统的实现方法

1. 系统功能

该系统是一个基于 TCP 协议的多线程聊天室。该系统只有一个服务器端但可以有多个客户端与其通信，能实现最基本的文字交流与文档传输。系统主要由客户端与服务器端组成，服务器端主要是负责监听并转发客户端发送来的消息，利用 TCP 线程并发服务器模型实现对客户端的监听接收。客户端主要用来向服务器端发送数据。

系统的主要实现的功能如下:

1.用户登录:设立一个服务器,实现多个用户实时通信。用户在登陆时候输入明确的用户名称,以便各用户之间能够相互识别。

2.文本聊天:同一个聊天室中聊天的内容所有的人都能够看到，同时，在聊天界面中用“您”来区分自己和其他用户。

3.文件传输:多线程,收发可随时进行,相互之间没有相互影响,文件传输速度快。

2. 系统流程

整个系统有两个主要部分，分别为客户端（Client）和服务器端（Server），服务器端首先设置端口(Port),开启主线程监听是否有来自客户端的连接请求，根据判断的结果从而确定下一步任务。当有请求连接时，则开启新的用户线程；当用户退出聊天室时，终止并销毁此线程。

客户端通过填写服务器端的 IP 地址和端口号，并且设置用户名登录，连接服务器。成功连接后，可以发送聊天文字或者聊天文件，由服务器进行监听并转发给聊天室内的其他客户端。

2.1 服务器

服务器端需要设置一个固定的端口，还需要提供服务器的 IP 地址，然后等待客户端的连接请求，并监听所有客户端发送的数据。在客户端连接成功后，再

进行下一步处理。

服务器端流程图如下图所示：

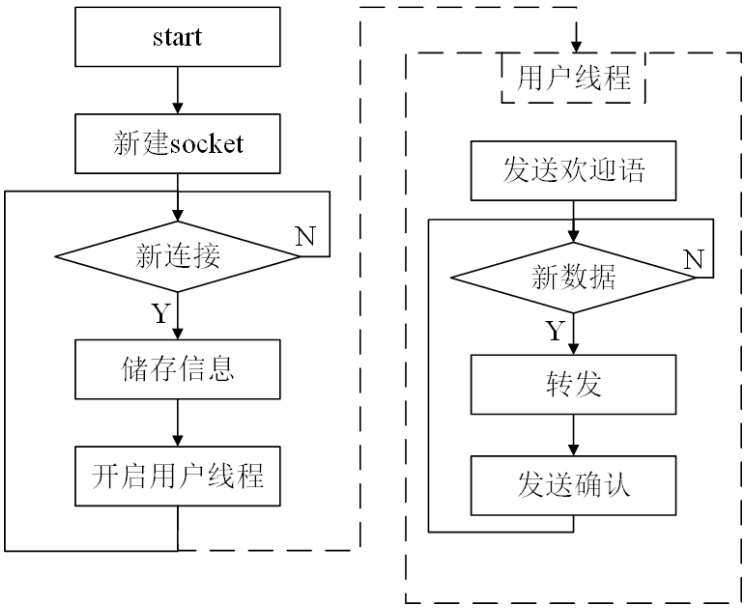

图 2.1 服务器工作流程

服务器端程序首先进行初始化，然后持续监听一个固定的端口，当收到一个新的客户端连接请求之后，开启一个新的子线程与客户端建立 socket 连接（主线程继续监听连接），进行数据交换；当有用户退出聊天室的时候关闭对应的连接并销毁 socket。

具体实现方法为：服务器创建一个共享内存空间函数以及服务器端的简单界面，利用套接字模型机制实现服务器的简易模型，利用 socket()创建流式套接字，并可以返回对应的套接字编号；利用 bind()实现套接字与本地地址相连，listen()通知 TCP 服务器准备好监听客户端的连接，accept()阻塞等待客户端的连接，建立连接之后 accept 返回新的客户端的套接字，然后运用多线程技术为该用户开启一个独立的子线程，并利用 recv()/send()函数收发数据。

2.2 客户端

客户端设置对应服务器端的 IP 和 Port，根据 Port 的值与服务器端建立连接，连接建立后，则可以将数据发送至服务器端，若判断数据有效，则可以进行正常通信。客户端流程图如下图所示：

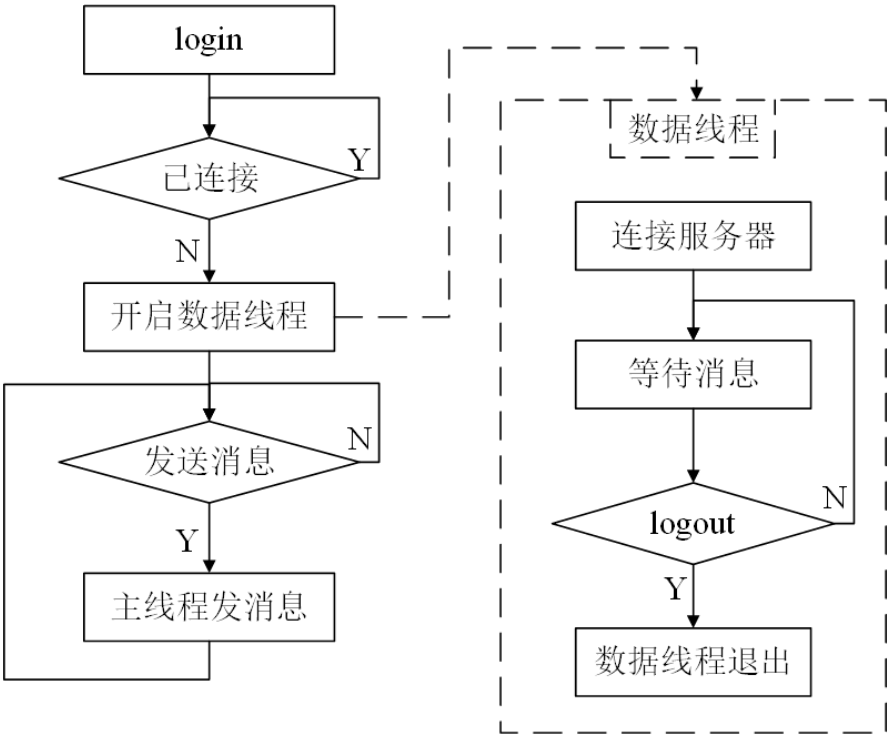

图 2.2 用户端工作流程

客户端程序首先初始化客户程序，然后连接到某个服务器上，建立 socket 连接，通过 socket 连接向服务器发送请求信息，通信结束后中断与服务器端的连接。

具体实现方法为：首先定义运行时候需要的参数，同样利用 socket()建立流式套接字，返回套接字号；操作系统内核会自动分配动态端口号，然后调用 connect()将套接字与远程服务器连接，通过 recv()和 send()实现与服务器端的数据接收和发送操作，通信结束后 close()关闭套接字，关闭对话。

在本项目中，我们使用了多线程的方式对系统进行优化，将收发数据分别放在两个线程，实现了真正的全双工通信，同时采用多线程的方式还可以避免服务器端的 accept()和客户端的 recv()函数执行时主程序被卡死。

第三章 系统的实现效果

1. 服务器端

服务器端的开启后等待连接界面如下：

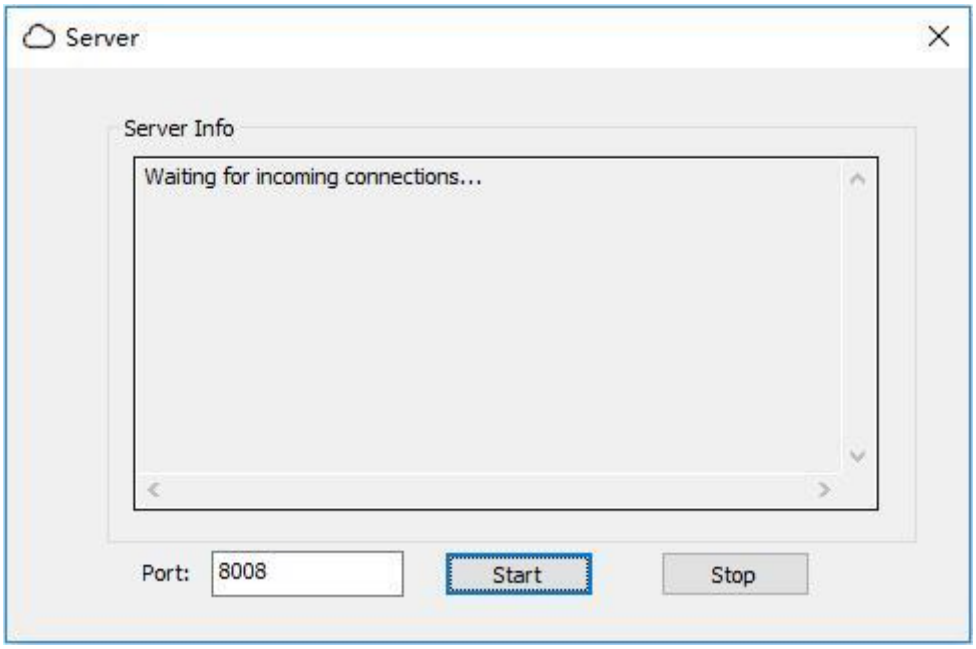

图 3.1 服务器监听界面

后台的服务器端显示：

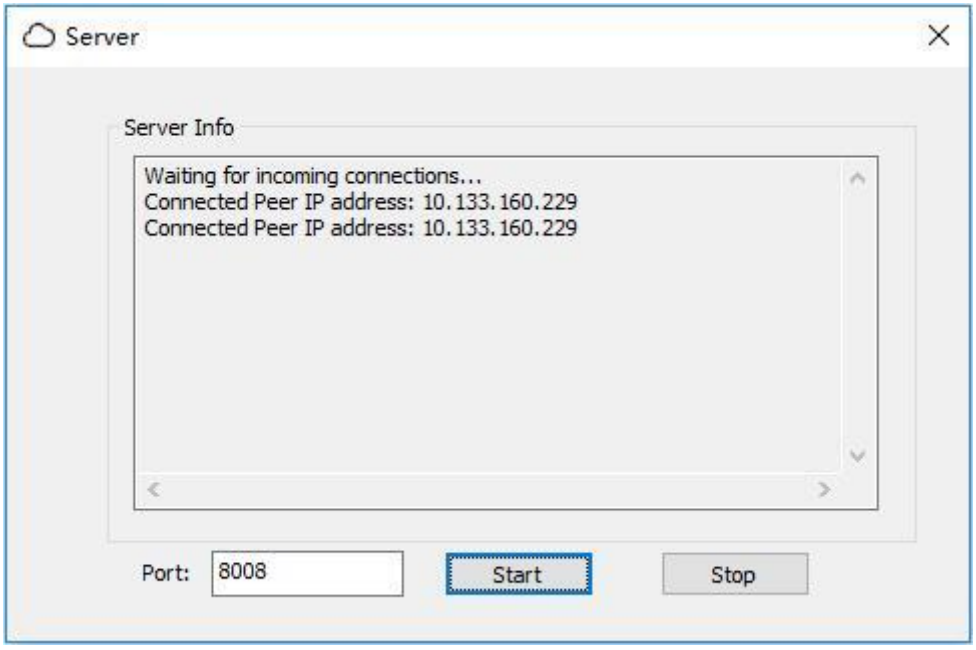

图 3.2 服务器后台数据

2．客户端

客户端登陆界面如下：

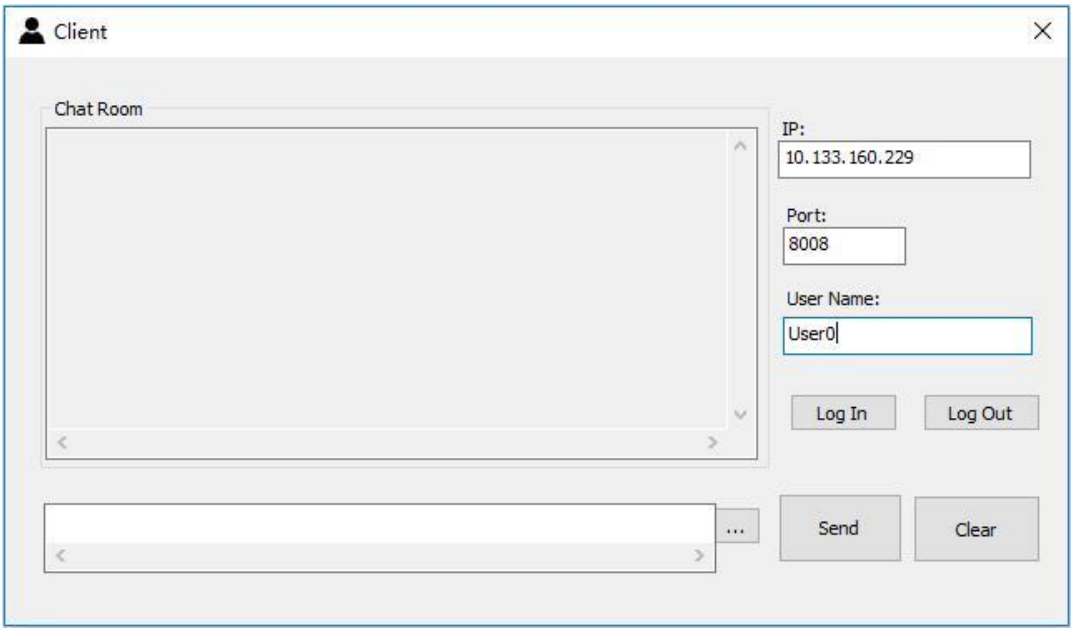

图 3.3 客户端登入界面

可以看到右上方选择的请求的服务器的端口号与 IP 地址与服务器对应，则服务器端可以监听到该客户的连接请求，并建立连接，实现两者的通信。当多个客户端连接到同一服务器，则可以实现以服务器为中转的客户端与客户端之间的通信。下图为多客户端连接服务器实现聊天功能的实际效果图：

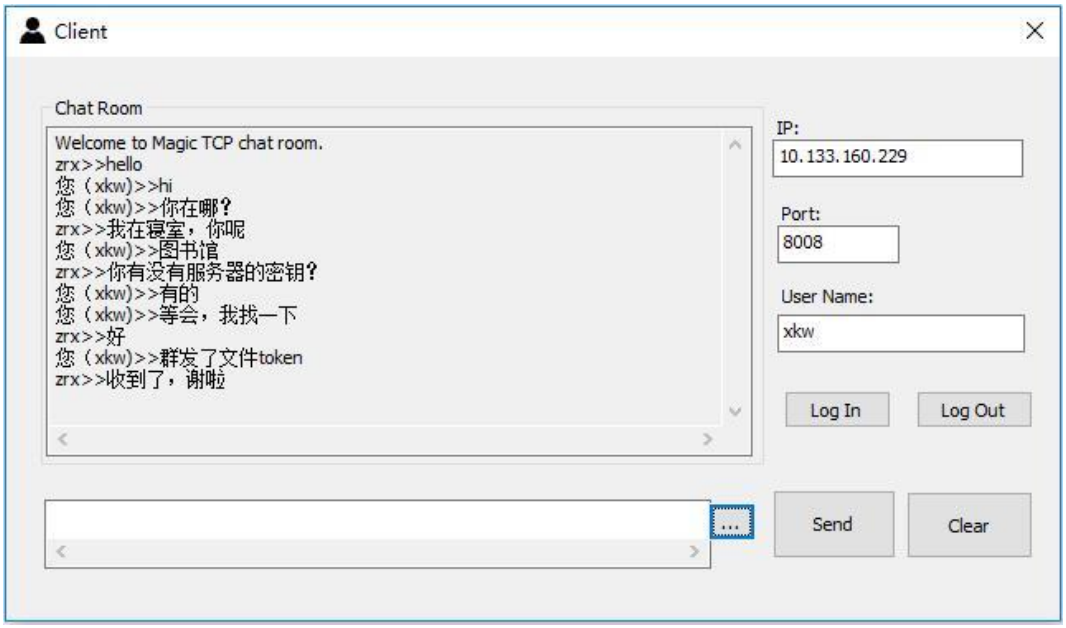

图 3.4 系统客户端聊天窗口显示 1

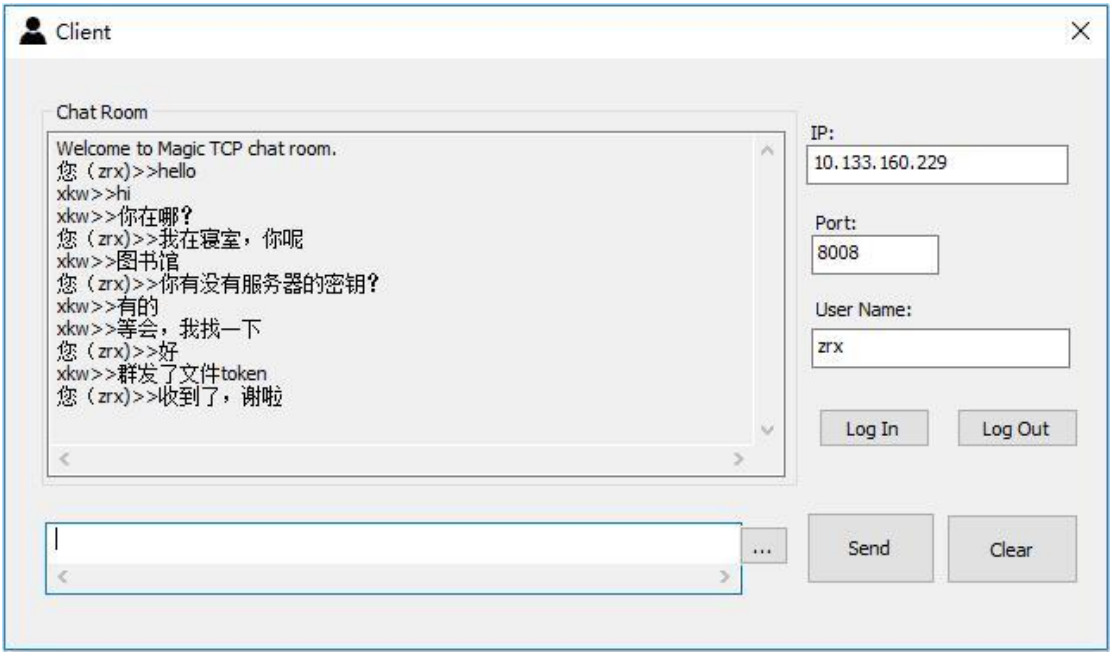

图 3.5 系统客户端聊天窗口显示 2

第四章 总结与拓展

在本次课程设计中，我们设计的基于 TCP 局域网多人聊天室系统将计算机网络课程的知识应用到实际中，让我们更加深入地理解了 TCP 协议。

但是本系统只能在同一个局域网下运行，对此，我们尝试了将服务器端程序部署到云端，从而可以允许外网访问。这样就可以在不处于同一局域网的情况下依旧实现多人聊天。由于云端服务器的 IP 地址是固定的，端口也由代码默认指定，所以客户端登录所需要的 IP 地址和端口也是固定的，无需再由用户手动输入与服务器对应的端口与 IP 地址。对此，我们重新设计了客户端的登录界面，如下图所示：

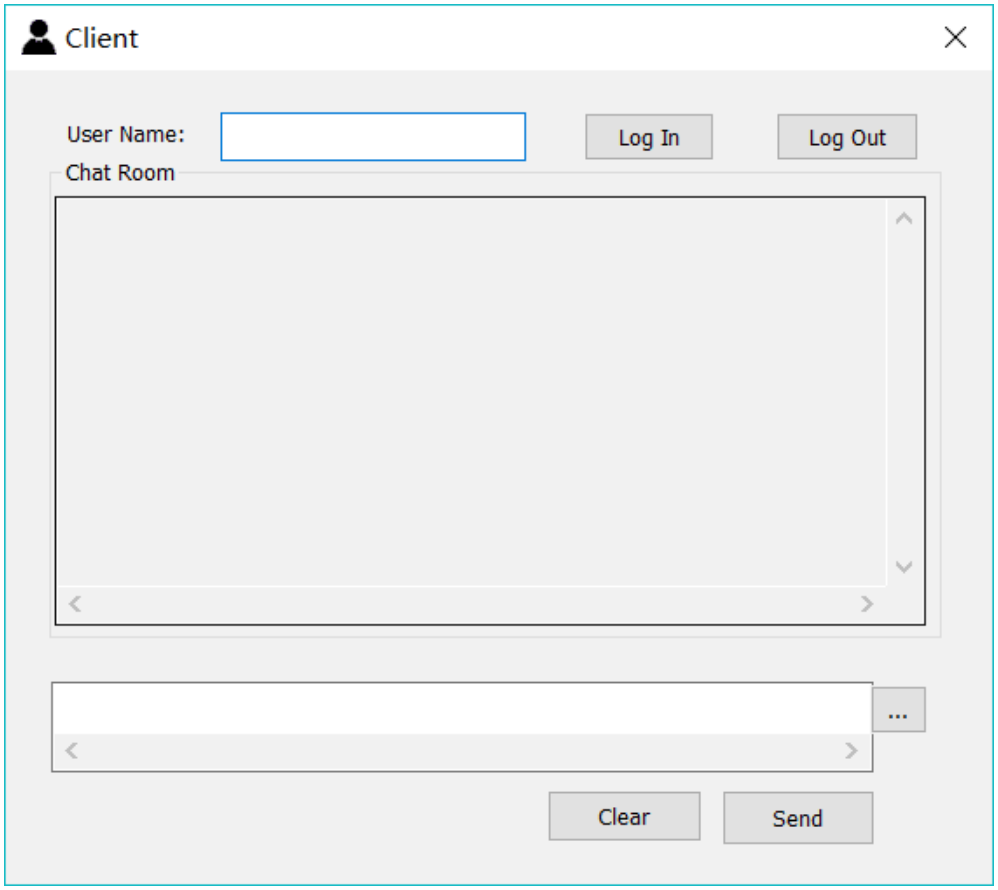

图 4.1 客户端登录界面

经测试，改进版的多人聊天系统可以实现以云端服务器为中转的在不同局域网下用户间的文字与文件传输。

参考文献

[1] 谢希仁，编著． 计算机网络． 第四版． 北京：电子工业出版社

[2]高莹.基于 TCP 的网络聊天室的设计研究[J].无线互联科技,2011(08):17.

[3]陈忠菊.一种基于 TCP 的网络聊天软件实现[J].电脑编程技巧与维护,2015(03):26-27.

致谢

在本课程将要结束之际，我们向本课程老师杨珺老师致以诚挚的感谢。老师扎实的专业知识，严谨的教学态度及精益求精的工作作风是我们小组成员的榜样。老师在课堂上不仅传授知识点，而且强调了这门课做实验的重要性，并表示了没有开设实验课的遗憾。虽然没有开设实验课，但是该课程设计为我们提供了一次实践与展示的机会。

通过该课程设计的工作，我们学会了如何在实际问题中将学过的知识进行重构，怎样分工，并以一个团队的形式共同完成一个课程项目等，这些对于我们今后的学习与科研相关工作都是不可多得的宝贵财富。在此，我们再一次真诚地表示感谢！