



南开大学  
Nankai University

南 开 大 学

网 络 空 间 安 全 学 院

计算机网络课程报告

---

socket 编程：多线程实现群聊

---

学号：2013018

姓名：许健

年级：2020 级

专业：信息安全

2022 年 10 月 22 日

## 目录

<b>一、 实验要求</b>	<b>1</b>
<b>二、 协议设计</b>	<b>1</b>
(一) 协议的语法	1
(二) 协议的语义	1
(三) 协议的时序	1
<b>三、 各模块功能</b>	<b>2</b>
(一) 客户端	2
1. 接收消息线程	3
2. 发送消息模块	3
(二) 多线程流式服务端	4
1. 创建接受并转发消息的线程	4
<b>四、 程序界面展示及运行说明</b>	<b>5</b>
<b>五、 实验过程中遇到的问题及分析</b>	<b>9</b>
(一) 实验过程	9
(二) 未来改进	9

## 一、 实验要求

1. 使用流式 Socket，设计一个两人聊天协议，要求聊天信息带有时间标签。请完整地说明交互消息的类型、语法、语义、时序等具体的消息处理方式。
2. 对聊天程序进行设计。给出模块划分说明，模块的功能和模块的流程图
3. 在 Windows 系统下，利用 C/C++ 对设计的程序进行实现。程序界面可以采用命令行方式，但需要给出使用方法。编写程序时，只能使用基本的 Socket 函数，不允许使用对 socket 封装后的类或架构。
4. 对实验的程序进行测试。

## 二、 协议设计

### （一） 协议的语法

通信方使用的均为聊天报文，属于文本协议。

报文包含 name 字段、time 字段、content 字段。通讯方发送的每条消息都会携带时间和姓名信息。

C++ 使用 Buffer 结构体实现聊天报文。

---

```
1 struct Buffer {  
2     char name[16];  
3     char time[32];  
4     char content[1024];  
5 };
```

---

### （二） 协议的语义

当 content 字段输入“depart”时，表示客户端退出群聊，服务器会将该消息转发给其余客户端。

输入其他内容，服务器会正常转发给每一个客户端。

### （三） 协议的时序

实验采用多线程实现即时通讯系统，并支持群聊。

客户端连接进入服务器时需要输入自己的姓名，并自动发送一条报文表示进入聊天室，服务器会将该报文转发给所有客户端。

客户端可以在任意时刻发出信息，服务器接受并转发给每个客户端，信息携带姓名和发送时间信息。

当客户端输入“depart”时，表示退出群聊，结束与服务器的连接。

### 三、 各模块功能

客户端和服务端的整体流程如图1所示

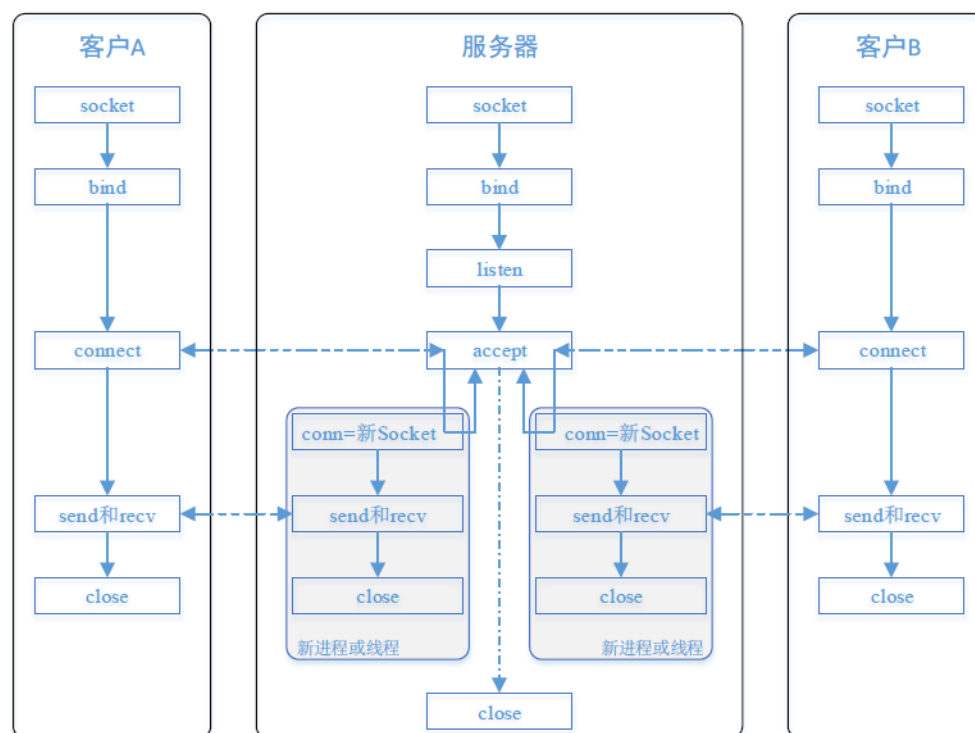


图 1: 整体流程图

不同的是，客户端的 send 和 recv 分属不同的线程，互不干扰。

服务端对于每一个客户端 socket 会创建一个 Thread\_Recv 线程用于接受消息并转发给其余客户端，服务器只转发消息而不会参与群聊。

#### (一) 客户端

客户端需要干的事情包括：

1. 创建 socket
2. 输入待连接服务器的 IP 和 Port，与服务器建立连接
3. 创建接收消息的线程
4. 在主线程创建发送消息模块
5. 结束聊天时关闭 socket

创建 socket、建立连接和关闭 socket 只需要使用 socket 函数、connect 函数、closesocket 函数即可实现，出现错误可以使用 GetLastError 函数获取错误信息，再次不做赘述。主要介绍接收消息的线程和发送消息的模块。

## 1. 接收消息线程

在这里, 创建线程使用的是 `CreateThread` 函数而不是 C++11 标准加入的 `Thread` 函数。释放线程使用 `CloseHandle` 函数。

在 `while(1)` 循环中使用 `recv` 接收消息, 区分消息的类型进行不同的处理措施。

接收消息线程

```

1 HANDLE hThread_recv = CreateThread(NULL, NULL, &handler_Recv, LPVOID(
    ClientSocket), 0, NULL);
2 CloseHandle(hThread_recv);
3
4
5 DWORD WINAPI handler_Recv(LPVOID lparam) {
6     SOCKET recvSocket = (SOCKET) (LPVOID) lparam;
7     Buffer recvBuf;
8     while (1) {
9         memset((char*)&recvBuf, 0, sizeof(recvBuf));
10        recv(recvSocket, (char*)&recvBuf, sizeof(recvBuf), 0);
11        if (strcmp(recvBuf.content, "depart") == 0) {
12            cout << recvBuf.time << " " << recvBuf.name << "离开了聊天室" <<
                endl;
13        }
14        else {
15            cout << recvBuf.time << " " << recvBuf.name << ": " << recvBuf.
                content << endl;
16        }
17    }
18    closesocket(recvSocket);
19    return 0;
20 }

```

## 2. 发送消息模块

在 `while(1)` 循环中使用 `send` 函数给服务器发送消息, 每条消息都会附加上姓名和时间信息, 时间信息是通过编写 `getTime()` 函数封装 `ctime` 库获得的。

根据消息的内容是聊天还是退出聊天, 如果是 `depart` 则退出群聊并释放 `socket`。

发送消息模块

```

1 Buffer sendBuf;
2 while (1) {
3     memset((char*)&sendBuf, 0, sizeof(sendBuf));
4     strcpy(sendBuf.name, name);
5     cin.getline(sendBuf.content, sizeof(sendBuf.content));
6     string time = getTime();
7     strcpy(sendBuf.time, time.c_str());
8     send(ClientSocket, (char*)&sendBuf, sizeof(sendBuf), 0);
9     if (strcmp(sendBuf.content, "depart") == 0) {
10        cout << "You have disconnect with the server" << endl;

```

```
11         break;
12     }
13 }
14 closesocket (ClientSocket);
15 return 0;
16 }
```

## (二) 多线程流式服务端

服务端需要干的事情包括：

1. 创建 socket
2. 输入网卡的 IP 和 Port，对 socket 进行地址和端口的绑定
3. 监听客户端的连接
4. 接受客户端的请求
5. 为每一个客户端 socket 创建一个接收并转发信息的线程

绑定地址端口和监听客户端请求只需要 bind 函数、listen 函数即可，使用 accept 函数接受客户端的请求，会返回客户端的 socket，我们需要为每一个 socket 创建一个接收并转发消息的线程。

### 1. 创建接受并转发消息的线程

限定最大接受客户端的数量，使用一个全局 socket 数组用于分配。

对于每一个接受的客户端 socket 创建一个线程，用于接受消息，并根据消息的类型进行不同情况的转发，在 while 循环中还要确认 socket 是否合法。

#### 接收并转发消息线程

```
1 while (1) {
2     if (index < MaxSize) {
3         ClientSocket[index] = accept(ServerSocket, (SOCKADDR*)&
4             ClientAddr, &ClientAddrlen);
5         HANDLE hThread_recv = CreateThread(NULL, NULL, &handler_Recv,
6             LPVOID(ClientSocket[index]), 0, NULL);
7         index++;
8         CloseHandle(hThread_recv);
9     }
10    else {
11        cout << "当前人数已满" << endl;
12    }
13 }
14
15 DWORD WINAPI handler_Recv(LPVOID lparam) {
16     SOCKET recvSocket = (SOCKET)(LPVOID)lparam;
17     Buffer recvBuf;
18     while (1) {
19         if (recvSocket == INVALID_SOCKET) {
```

```

18         closesocket(recvSocket);
19         break;
20     }
21     memset((char*)&recvBuf, 0, sizeof(recvBuf));
22     recv(recvSocket, (char*)&recvBuf, sizeof(recvBuf), 0);
23     if (strcmp(recvBuf.content, "depart") == 0) {
24         cout << recvBuf.time << " " << recvBuf.name << "离开
25         了聊天室" << endl;
26         for (int i = 0; i < index; i++) {
27             if (ClientSocket[i] != INVALID_SOCKET &&
28                 ClientSocket[i] != recvSocket)
29                 send(ClientSocket[i], (char*)&recvBuf,
30                     sizeof(recvBuf), 0);
31         }
32         closesocket(recvSocket);
33         break;
34     }
35     else {
36         cout << recvBuf.time << " " << recvBuf.name << ": "
37         << recvBuf.content << endl;
38         for (int i = 0; i < index; i++) {
39             if (ClientSocket[i] != INVALID_SOCKET)
40                 send(ClientSocket[i], (char*)&recvBuf,
41                     sizeof(recvBuf), 0);
42         }
43     }
44 }
45 return 0;
46 }

```

## 四、 程序界面展示及运行说明

演示一下群聊的整体流程，我们选用一个服务端和两个客户端，主要过程包括：

1. 服务器绑定地址端口，开启监听

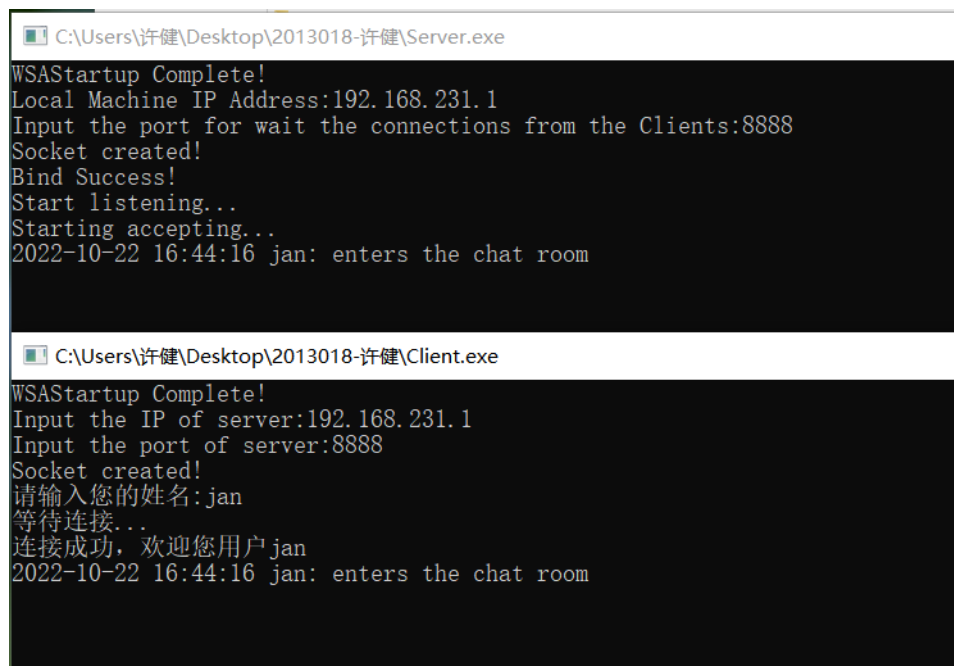
```

C:\Users\许健\Desktop\2013018-许健\Server.exe
WSASStartup Complete!
Local Machine IP Address:192.168.231.1
Input the port for wait the connections from the Clients:8888
Socket created!
Bind Success!
Start listening...
Starting accepting...

```

图 2: 过程 1

## 2. 客户端 1 连接服务器

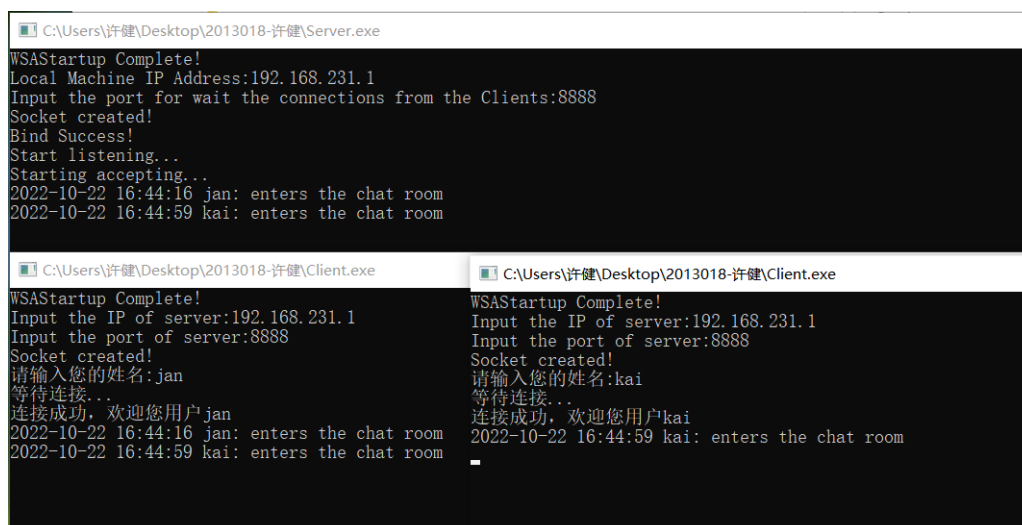


```
C:\Users\许健\Desktop\2013018-许健\Server.exe
WSAStartup Complete!
Local Machine IP Address:192.168.231.1
Input the port for wait the connections from the Clients:8888
Socket created!
Bind Success!
Start listening...
Starting accepting...
2022-10-22 16:44:16 jan: enters the chat room

C:\Users\许健\Desktop\2013018-许健\Client.exe
WSAStartup Complete!
Input the IP of server:192.168.231.1
Input the port of server:8888
Socket created!
请输入您的姓名:jan
等待连接...
连接成功, 欢迎您用户 jan
2022-10-22 16:44:16 jan: enters the chat room
```

图 3: 过程 2

## 3. 客户端 2 连接服务器



```
C:\Users\许健\Desktop\2013018-许健\Server.exe
WSAStartup Complete!
Local Machine IP Address:192.168.231.1
Input the port for wait the connections from the Clients:8888
Socket created!
Bind Success!
Start listening...
Starting accepting...
2022-10-22 16:44:16 jan: enters the chat room
2022-10-22 16:44:59 kai: enters the chat room

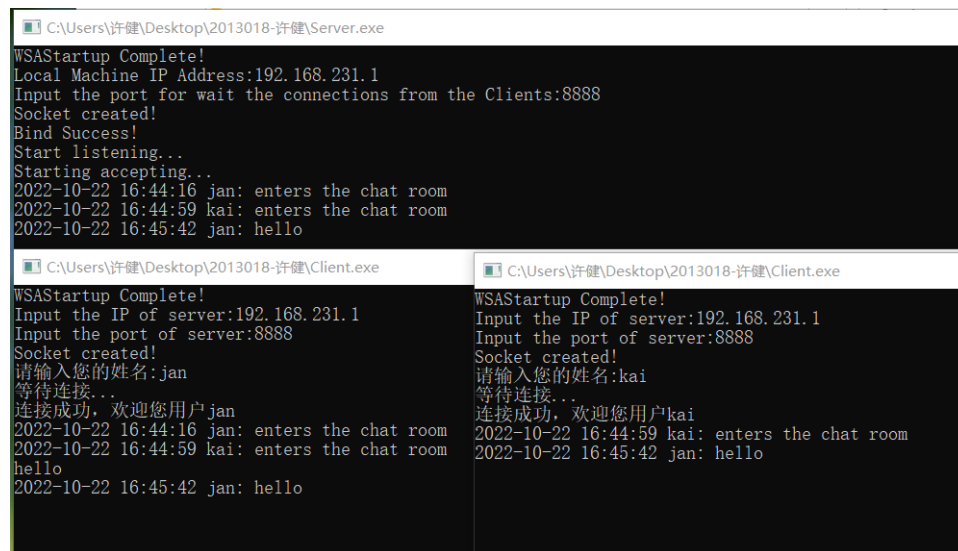
C:\Users\许健\Desktop\2013018-许健\Client.exe
WSAStartup Complete!
Input the IP of server:192.168.231.1
Input the port of server:8888
Socket created!
请输入您的姓名:jan
等待连接...
连接成功, 欢迎您用户 jan
2022-10-22 16:44:16 jan: enters the chat room
2022-10-22 16:44:59 kai: enters the chat room

C:\Users\许健\Desktop\2013018-许健\Client.exe
WSAStartup Complete!
Input the IP of server:192.168.231.1
Input the port of server:8888
Socket created!
请输入您的姓名:kai
等待连接...
连接成功, 欢迎您用户kai
2022-10-22 16:44:59 kai: enters the chat room
```

图 4: 过程 3



## 4. 客户端 1 发送消息



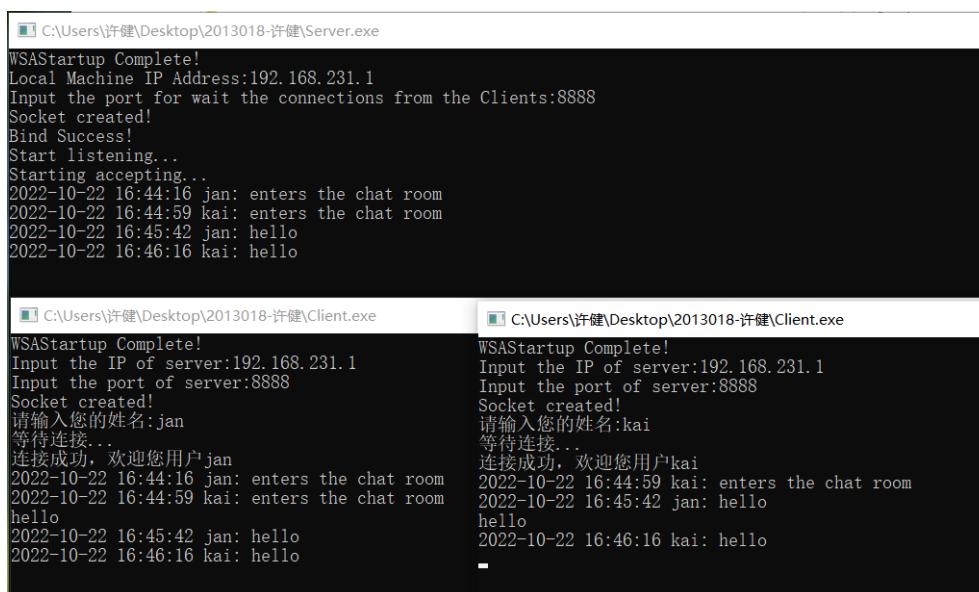
```
C:\Users\许健\Desktop\2013018-许健\Server.exe
WSAStartup Complete!
Local Machine IP Address:192.168.231.1
Input the port for wait the connections from the Clients:8888
Socket created!
Bind Success!
Start listening...
Starting accepting...
2022-10-22 16:44:16 jan: enters the chat room
2022-10-22 16:44:59 kai: enters the chat room
2022-10-22 16:45:42 jan: hello

C:\Users\许健\Desktop\2013018-许健\Client.exe
WSAStartup Complete!
Input the IP of server:192.168.231.1
Input the port of server:8888
Socket created!
请输入您的姓名:jan
等待连接...
连接成功, 欢迎您用户 jan
2022-10-22 16:44:16 jan: enters the chat room
2022-10-22 16:44:59 kai: enters the chat room
hello
2022-10-22 16:45:42 jan: hello

C:\Users\许健\Desktop\2013018-许健\Client.exe
WSAStartup Complete!
Input the IP of server:192.168.231.1
Input the port of server:8888
Socket created!
请输入您的姓名:kai
等待连接...
连接成功, 欢迎您用户kai
2022-10-22 16:44:59 kai: enters the chat room
2022-10-22 16:45:42 jan: hello
```

图 5: 过程 4

## 5. 客户端 2 发送消息



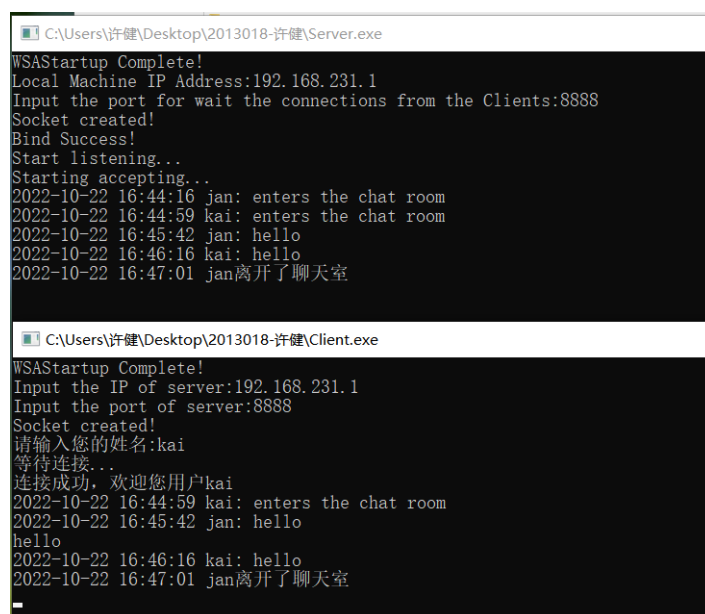
```
C:\Users\许健\Desktop\2013018-许健\Server.exe
WSAStartup Complete!
Local Machine IP Address:192.168.231.1
Input the port for wait the connections from the Clients:8888
Socket created!
Bind Success!
Start listening...
Starting accepting...
2022-10-22 16:44:16 jan: enters the chat room
2022-10-22 16:44:59 kai: enters the chat room
2022-10-22 16:45:42 jan: hello
2022-10-22 16:46:16 kai: hello

C:\Users\许健\Desktop\2013018-许健\Client.exe
WSAStartup Complete!
Input the IP of server:192.168.231.1
Input the port of server:8888
Socket created!
请输入您的姓名:jan
等待连接...
连接成功, 欢迎您用户 jan
2022-10-22 16:44:16 jan: enters the chat room
2022-10-22 16:44:59 kai: enters the chat room
hello
2022-10-22 16:45:42 jan: hello
2022-10-22 16:46:16 kai: hello

C:\Users\许健\Desktop\2013018-许健\Client.exe
WSAStartup Complete!
Input the IP of server:192.168.231.1
Input the port of server:8888
Socket created!
请输入您的姓名:kai
等待连接...
连接成功, 欢迎您用户kai
2022-10-22 16:44:59 kai: enters the chat room
2022-10-22 16:45:42 jan: hello
hello
2022-10-22 16:46:16 kai: hello
```

图 6: 过程 5

## 6. 客户端 1 离开群聊 (由于程序直接退出, 没有截图)

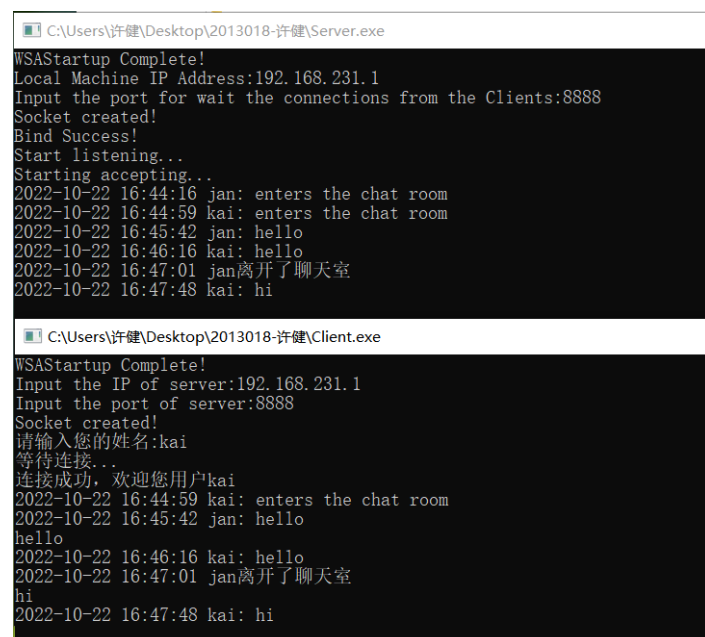


```
C:\Users\许健\Desktop\2013018-许健\Server.exe
WSAStartup Complete!
Local Machine IP Address:192.168.231.1
Input the port for wait the connections from the Clients:8888
Socket created!
Bind Success!
Start listening...
Starting accepting...
2022-10-22 16:44:16 jan: enters the chat room
2022-10-22 16:44:59 kai: enters the chat room
2022-10-22 16:45:42 jan: hello
2022-10-22 16:46:16 kai: hello
2022-10-22 16:47:01 jan离开了聊天室

C:\Users\许健\Desktop\2013018-许健\Client.exe
WSAStartup Complete!
Input the IP of server:192.168.231.1
Input the port of server:8888
Socket created!
请输入您的姓名:kai
等待连接...
连接成功, 欢迎您用户kai
2022-10-22 16:44:59 kai: enters the chat room
2022-10-22 16:45:42 jan: hello
hello
2022-10-22 16:46:16 kai: hello
2022-10-22 16:47:01 jan离开了聊天室
```

图 7: 过程 6

## 7. 客户端 2 继续聊天



```
C:\Users\许健\Desktop\2013018-许健\Server.exe
WSAStartup Complete!
Local Machine IP Address:192.168.231.1
Input the port for wait the connections from the Clients:8888
Socket created!
Bind Success!
Start listening...
Starting accepting...
2022-10-22 16:44:16 jan: enters the chat room
2022-10-22 16:44:59 kai: enters the chat room
2022-10-22 16:45:42 jan: hello
2022-10-22 16:46:16 kai: hello
2022-10-22 16:47:01 jan离开了聊天室
2022-10-22 16:47:48 kai: hi

C:\Users\许健\Desktop\2013018-许健\Client.exe
WSAStartup Complete!
Input the IP of server:192.168.231.1
Input the port of server:8888
Socket created!
请输入您的姓名:kai
等待连接...
连接成功, 欢迎您用户kai
2022-10-22 16:44:59 kai: enters the chat room
2022-10-22 16:45:42 jan: hello
hello
2022-10-22 16:46:16 kai: hello
2022-10-22 16:47:01 jan离开了聊天室
hi
2022-10-22 16:47:48 kai: hi
```

图 8: 过程 7

## 五、 实验过程中遇到的问题及分析

### （一） 实验过程

实验初期，主要是对 socket 编程和常用函数的熟悉，确保客户端和服务端可以建立连接、正确收发信息。

其次是对协议的设计，本次实验中设计的协议较为简单，只包括三个字段。一开始设计的聊天协议是不包含多线程的，也不支持群聊，协议规定了消息收发的次序，比如客户端先发信息然后轮到服务端，消息包括 exit、quit 等不同类型。缺点是消息字段大小固定且没有明确的控制字段，这是未来需要改进的地方。

实现简单的两人聊天协议后，尝试加入多线程改为即时通讯，时间成本主要包括熟悉创建线程函数、如何同步信息、调试多线程等。

最后尝试加入群聊功能，在服务端监听对于每一个客户端都新建一个线程，编写消息接收和消息转发，并释放关闭连接客户端的线程。至此一个简易的即时通讯群聊系统创建完毕。

### （二） 未来改进

包括以下方面：

1. 优化程序的性能，改进协议设计并尝试添加控制字段，支持更多种功能。
2. 尝试使用线程池回收线程，并进一步熟悉线程的工作原理。
3. 添加锁的机制，保证操作的原子性，防止出现错误。