
# 《网络安全》
# 报告

组　　员：孔浩 胡洋 张毛毛　班号：193152
学　　号：20151001606　组长：许晓晖
院（系）：计算机学院　专业：网络工程
指导教师：姚 宏　职称：副教授

2018 年 6 月

独立工作成果声明

本人声明所呈交的《网络安全》报告，是我个人在导师指导下进行的程序编制工作及取得的成果。

尽我所知，除文中已经标明的引用内容，和已经标明的他人工作外，本报告未包含任何抄袭自他人的工作成果。对本报告的工作做出贡献的个人，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

报告作者签名：

日期：　　年　　月

# 目录

# 第一章　系统及开发环境概述

## 1.1.1 系统

系统为基于 KERBEROS 的多人聊天室系统。

## 1.1.2 环境

（1）环境系统

Window 7 系统版本及以上

Java jdk 1.8.0 版本及以上

（2）运行平台

Eclipse Java EE IDE for Web Developers.

Version: Oxygen.2 Release (4.7.2)

Build id: 20171218-0600

（3）开发语言：

JAVA

（4）硬件配置

个人笔记本电脑 4 台

（5）源代码管理工具

GitHub for Windows 1.0

（6）数据库

MYSQL 数据库。

## 1.1.3 任务分工

许晓晖（组长）：主要负责对整体系统的设计。其中

1、Kerberos 的空数据包发送流程（数据包内容未解析）。

2、初期设计报告的主体内容。

3、客户端应用服务模块。

4、服务器应用服务模块。

5、X.509 模块身份认证。

6、语音播放服务。

7、后期上机整体代码修改。

8、RSA 加密模块。

孔浩：

1、服务器 V 的 kerberos 模块。

2、DES 加密模块。

3、android 端的开发模块。

4、android 端语音识别。

胡洋：

1、Kerberos 的 AS 认证细节。

2、AS ui 界面。

3、状态机维护。

张毛毛：

1、TGS 的认证细节处理。

2、TGS ui 界面。

3、状态机维护。

# 第二章　UI 设计

## § 2.1 引言

### 2.1.1 设计概要

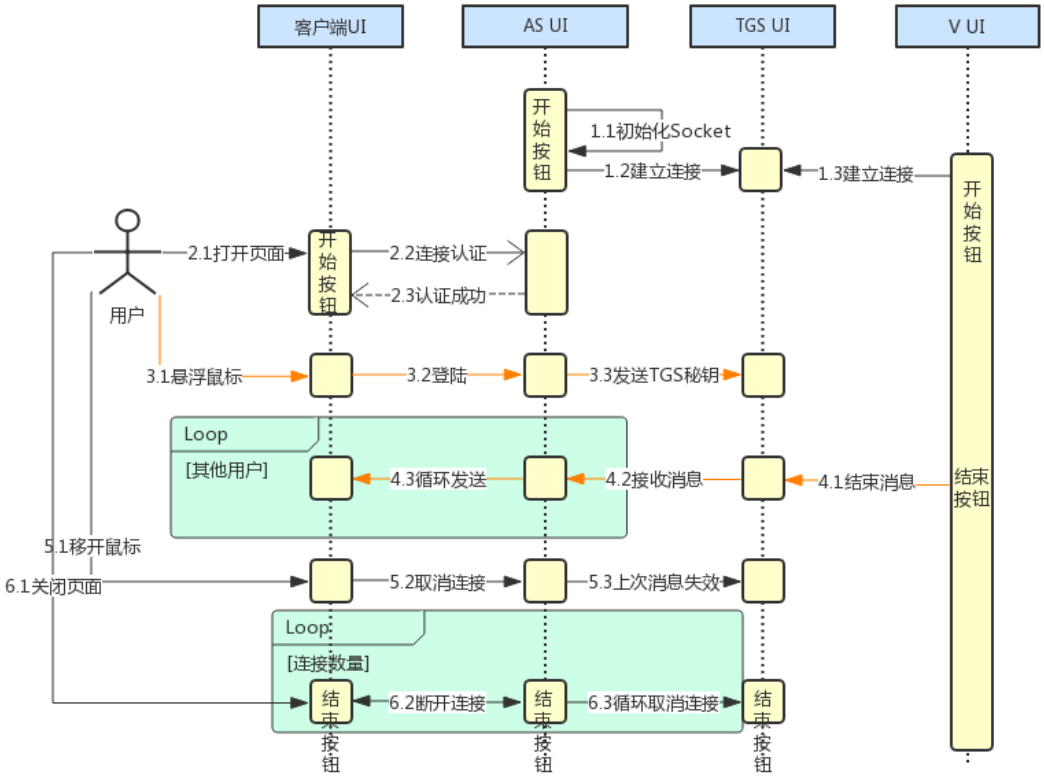


图 二-1 UI 设计概要

根据应用服务所需内容，UI 共有客户端注册 UI、客户端登陆 UI、客户端应用服务 UI、服务器 UI、AS 的 UI、TGS 的 UI 共 6 部分组成。

UI 流程一般为：

第一， 开启 AS 的 UI，启动 AS 服务器。

第二， 开启 TGS 的 UI，启动 TGS 服务器。

第三， 开启 V 的 UI，启动应用服务器。

第四， 开启客户端 UI，准备登陆或者注册。

每个部分根据需求将内容呈现在 UI 中，其具体细节如下所述。

### 2.1.2 登陆界面

用户需要被验证，所以设立登陆界面。需要做如下事情：

1、 输入用户名

2、 输入密码

3、 输入验证码

4、 设计界面如下


图 二-2 注册界面

**按钮解释——登陆**

用户点击登录按钮后，根据 KERBEROS 处理信息流程，依次发出请求信息。

第一步，按钮点击后，客户端发出 3 号包，发往 AS 请求访问 TGS，如果通过，则 AS 发回 4 号包。客户端发出 5 号包，请求访问服务器 V，TGS 如果通过，则 TGS 发回 6 号包。客户端发出 7 号包，请求服务，V 如果通过，在 V 发回 8 号包，保内标记信息，如果请求成功，则返回信息为 YES，代表登陆成功。否则返回信息 NO，代表登陆失败。登陆成功后弹出聊天室界面。

**按钮解释——注册**

用户点击注册按钮后，进入注册界面，内容如注册界面部分所述。

**代码——详解**

设置 frame 的 setlayout 为（null），这样页面布局可以自己随意布置。

根据注册功能需要，需要添加文本框 3 个，分别用 jpanel 显示为用户名、密码、验证码。输入区域设置 JTextField。按钮 2 个，分别为登陆和注册，类型 JButton。

其中验证码初始化形式如图所示。

```
//验证码文本框
verCodeTxt = new JTextField();
verCodeTxt.setSize(100, 20);
verCodeTxt.setLocation(170, 180);
```

图 二-3 按钮代码

按钮具体事件的处理，采用actionPerformed(ActionEvent e)，获得点击按钮上的信息，

JButton bt = (JButton)e.getSource();

获取按钮上显示的文本

String str = bt.getText();

再使用判断语句判断文本是什么即可执行相应的语句。

其中验证码使用Graphics方式显示，使用for循环random预先设定的几个字符，分别为大小字母和数字。生成方式如图所示。

```
//生成随机验证码
char[] tmp = ("0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJ
sb = new StringBuilder();
for(int i = 0;i<4;i++)
{
    int pos = r.nextInt(tmp.length);
    char c = tmp[pos];
    sb.append(c + " ");
}
```

图 二-4 验证码

## 2.1.3 注册界面

用户需要被验证，所以设立登陆界面。需要做如下事情：

1、 输入用户名

2、 输入密码

3、 确认密码

设计界面如下：

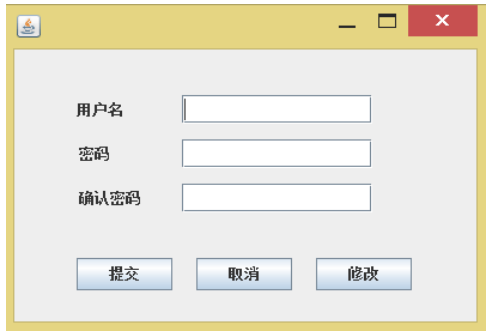


图 二-5 登陆界面

**按钮解释——提交**

首先需要对 AS 身份确认，利用模仿 X.509 协议。

设计的 X.509 内容，主要包括此协议的版本号、序列号、第三方签名信息，对应主体的 RSA 公钥（n 和 e）。

客户端如果点击提交按钮。则会 socket 发送 000 字符串，告诉 AS 请求发回其 X.509 协议内容，进行身份确认。AS 将 X.509 信息转换为字符串封装为 33 号数据包发往 Client，Client 接收到后，首先按照 33 号包格式将信息提取，验证 AS 的身份正确性。如果正确，则保存 AS 的公钥私钥，准备发送本 Client 端身份信息 32 号包，包格式和 33 号一样，代表客户端的 X.509 信息。如果客户端身份认证确认争取，则保存其公钥私钥。

确认身份后，客户端提交密码

AS 期待接收 1 号包。C 期待返回 2 号包。（包的格式已在下面章节定义好）。

当用户输入用户名、密码以及重复密码后，点击注册按钮。将用户名和密码从 UI 框中提取为 String 类型，使用 RSA 加密算法，用 AS 公钥加密用户名和密码，包头标记为 1 号（包头不加密），发送给 AS。

AS 获取 1 号包后，按照 1 号数据包数据格式，使用自己私钥获取明文，然后将用户名在数据库中进行比对，如果已经存在此用户，返回 2 号数据包，包内信息为 NO1，表示已存在用户注册失败，提示用户重新注册。

如果 AS 存入数据库或者其他问题，返回保内信息 NO2，表示注册失败，提示重新注册。

如果数据库中存入成功，则返回 YES。

所有的返回 2 号数据包均用客户端的公钥加密，包头标记 2 号不加密。

客户端接收 2 号包，使用自己私钥解密，比对信息，查看是否注册成功，否则提示用户重新点击登录按钮聊天室界面。

**按钮解释——修改**

首先需要对 AS 身份确认，利用模仿 X.509 协议。

设计的 X.509 内容，主要包括此协议的版本号、序列号、第三方签名信息，对应主体的 RSA 公钥（n 和 e）。

客户端如果点击修改按钮。则会 socket 发送 000 字符串，告诉 AS 请求发回其 X.509 协议内容，进行身份确认。AS 将 X.509 信息转换为字符串封装为 33 号数据包发往 Client，Client 接收到后，首先按照 33 号包格式将信息提取，验证 AS 的身份正确性。如果正确，则保存 AS 的公钥私钥，准备发送本 Client 端身份信息 32 号包，包格式和 33 号一样，代表客户端的 X.509 信息。如果客户端身份认证确认争取，则保存其公钥私钥。

确认身份后，客户端提交更改的密码。

AS 期待接收 1 号包。C 期待返回 2 号包。（包的格式已在下面章节定义好）。

当用户输入用户名、密码以及重复密码后，点击注册按钮。将用户名和密码从 UI 框中提取为 String 类型，使用 RSA 加密算法，用 AS 公钥加密用户名和密码，包头标记为 1 号（包头不加密），发送给 AS，末尾加注字符串”xiugai“信息，表明是修改而不是注册。

AS 获取 1 号包后，按照 1 号数据包数据格式，使用自己私钥获取明文，然后将用户名在数据库中进行比对。首先判断数据长度，如果是修改密码，则字符串长度会大于

45，所以是修改流程。

如果用户修改成功，则返回 XIU。表明密码修改成功，提示用户可以进行其他操作。


图 二-6 修改成功提示

如果 AS 存入数据库或者其他问题，返回包内信息 XIUB，表示注册失败，提示重新修改。

如果 AS 查询数据库后发现没有指定的用户，则返回包内信息 BUCZ，表明不存在此用户，提示用户应该注册。


图 二-7 不存在用户提示

所有的返回 2 号数据包均用客户端的公钥加密，包头标记 2 号不加密。

客户端接收 2 号包，使用自己私钥解密，比对信息，查看是否注册成功，否则提示用户重新点击登录按钮聊天室界面。

**代码——详解**

设置 frame 的 setlayout 为（null），这样页面布局可以自己随意布置。

根据注册功能需要，需要添加文本框 3 个，分别用 jpanel 显示为用户名、密码、确认密码。输入区域设置 JTextField。按钮 3 个，分别为提交，取消，修改，类型 JButton。

其中 JButton 初始化形式如图所示。

```
button_1 = new JButton("取消");
button_1.setBounds(152, 174, 80, 27);
frame.getContentPane().add(button_1);
button_1.addActionListener(this);
```

图 二-8 按钮代码

按钮具体事件的处理，采用actionPerformed(ActionEvent e)，获得点击按钮上的信息，

JButton bt = (JButton)e.getSource();

获取按钮上显示的文本

String str = bt.getText();

再使用判断语句判断文本是什么即可执行相应的语句。

### 2.1.4 聊天室

提供多人实时聊天功能。需要做如下事情：

1、 用户输入内容区域。

2、 聊天信息显示。

3、 好友列表。

4、 私聊输入区域。

5、 传递文件。

6、 历史消息。

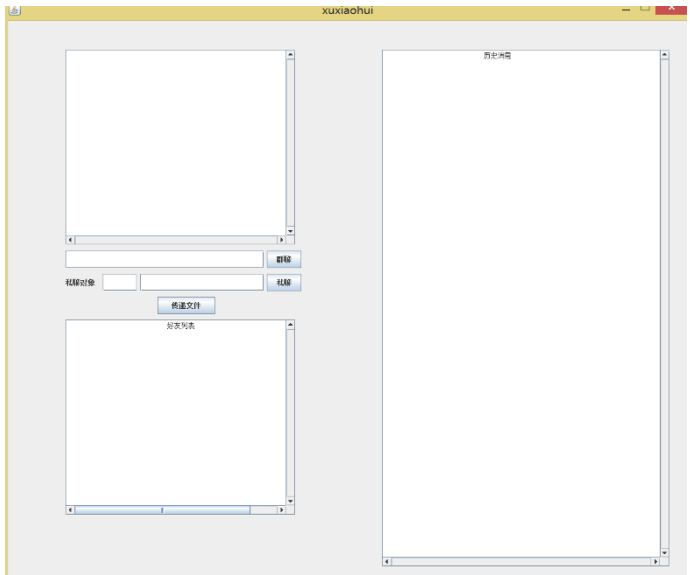


图 二-9 聊天室界面

**按钮解释——群聊**

用户弹出本聊天室界面后，在群聊输入框内输入要发送的信息，点击发送，则会将信息发送至聊天室。发送 9 号数据包至服务器，服务器广播至所有客户端。

其他客户端收到数据包 DES 解密，则聊天框内容显示刚才的聊天信息。

每个客户端重复以上操作。

**按钮解释——私聊**

用户弹出本聊天室界面后，在群聊输入框内输入要发送的信息，点击发送，则会将信息发送至聊天室。发送 9 号数据包至服务器，服务器解密 9 号数据包。解密完成后，根据客户端提交的私聊对象名称，服务器查找到对应用户，只发送给对应用户消息内容。

**按钮解释——传递文件**

用户首先在私聊对象输入区域内输入对象名称，再点击传递文件，则会弹出如下选择对话框。

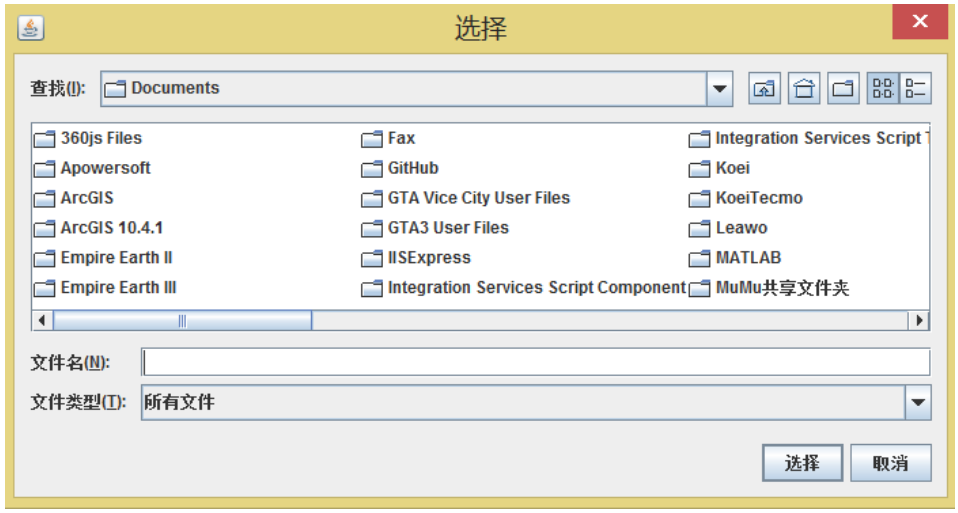


图 二-10 选择文件

选中后，点击选择，则会将文件发送给对应的用户，发送成功提示如下对话框。


图 二-11 发送成功

对应客户端接收到后，弹出如下对话框。


图 二-12 成功接受文件

**历史消息区域**

此区域用来展示接收到消息的密文，原文。直观地呈现加解密信息。呈现格式如图所示。


图 二-13 历史消息

如果是发送的信息，则先展示完全加密的密文。第二行代表原数据包解密后完整的呈现，第三个明文代表信息明文，最后一行代表签名，防止消息抵赖。如果签名不正确，会有对话框提示。

**好友列表**


图 二-14 好友列表

好友列表用来展示当前在线的好友。

**代码——详解**

设置 frame 的 setlayout 为（null），这样页面布局可以自己随意布置。

根据注册功能需要，需要添加文本框 3 个，分别用 jpanel 显示为群聊输入区域、私聊输入区域、私聊对象名输入区域。输入区域设置 JTextField。按钮 3 个，分别为群聊，私聊，传送文件，类型 JButton。需要文本区域 3 个，分别为显示聊天信息，显示好友列表，显示历史信息，类型为 JTextArea。其中 Jtextarea 里的滑动框需要使用 scrollpane。

滑动框设置方式如图所示。

```
//分别设置水平和垂直滚动条总是出现
scrollPane.setHorizontalScrollBarPolicy(
JScrollPane.HORIZONTAL_SCROLLBAR_ALWAYS);
scrollPane.setVerticalScrollBarPolicy(
JScrollPane.VERTICAL_SCROLLBAR_ALWAYS);
```

图 二-15 滑动框

对于JTextArea大小的设置，因为scrollpane创建时使用Jtextarea，所以要对scrollpane进行设置大小，相当于对Jtextarea大小进行设置。

按钮具体事件的处理，采用actionPerformed(ActionEvent e)，获得点击按钮上的信息，JButton bt = (JButton)e.getSource();

获取按钮上显示的文本

```
String str = bt.getText();
```

再使用判断语句判断文本是什么即可执行相应的语句。

# 第三章　多线程 socket 运转

## § 3.1 引言

分为 3 个服务器，分别为 AS、TGS、V。

分为 2 个客户端，分别为注册客户端 Register 和 Client。

| 名称 | 端口 |
| --- | --- |
| AS | 5432 |
| TGS | 5433 |
| V | 5434 |

### 3.1.1 运转代码结构

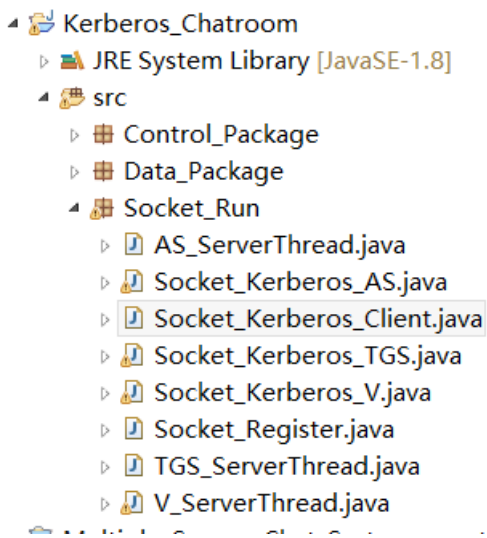


图 三-1 代码结构

各个客户端和各种服务器如图所示。命名方式统一按照模块设计中规定。

它们的建立 socket 方式如下，分别对应服务器和客户端。

### 3.1.2 代码详解——各个客户端

```
Socket s = new Socket("127.0.0.1", 5432);
OutputStream out = s.getOutputStream();
DataOutputStream dout = new DataOutputStream(out);
dout.writeUTF(str1);
InputStream in = s.getInputStream();
DataInputStream din = new DataInputStream(in);
st = din.readUTF();
```

【功能】：socket 连接服务器，传递加密密文。

【方式】：创建 Socket 对象，传入地址与端口。

```
Socket s = new Socket("127.0.0.1", 5432);
```

定义输出流 OutputStream，数据流 DataOutputStream，使用数据流方法 writeUTF 将字符串发送至服务器。

接收方式定义输入流 OutputStream，数据流 DataOutputStream，使用数据流方法 readUTF 将字符串接收服务器信息。

### 3.1.3 代码详解——各个服务器（多线程）

```
Socket cs = s.accept();
new ServerThread(cs).start();
System.out.println("接收了 第"+i+"个请求");
InputStream in = sock.getInputStream();
DataInputStream din = new DataInputStream(in);
String name = din.readUTF();
Decipher deph=new Decipher();
System.out.println(name+" 收到!!!!!!!!!!!!!");
name=deph.decipher(name);
OutputStream out = sock.getOutputStream();
DataOutputStream dos = new DataOutputStream(out);
```

【功能】：响应客户端请求，调用解密代码。

【方式】：建立 socket 之后，使用监听方法 Socket cs = s.accept();监听客户端请求。

定义输出流 OutputStream，数据流 DataOutputStream，使用数据流方法 writeUTF 将字符串发送至客户端。

接收方式定义输入流 OutputStream，数据流 DataOutputStream，使用数据流方法 readUTF 将字符串接收客户端信息。

### 3.1.4 运转结果截图

```
Socket_Kerberos_AS [Java Application] C:\Program Files\Java\jdk1.8.0_121
start AS service:
接收了 第1个请求
包1 收到!!!!!!!!!!!!!
```

图 三-2 注册结果

```
<terminated> Socket_Kerberos_Client [Java Application] C:\Program Files\Java\jdk1.8.0_121\jre\bin\javaw
From AS Server:
From TGS Server
From V Server
交互成功
```

图 三-3 client 结果

```
Socket_Kerberos_AS [Java Application] C:\Program Files\Java\jdk1.8.0_121\jre\bin\javaw.exe (2018年5月
接收了 第1个请求
包1 收到!!!!!!!!!!!!!
接收了 第2个请求
包3 收到!!!!!!!!!!!!!
接收了 第3个请求
包3 收到!!!!!!!!!!!!!
```

图 三-4 AS 结果

```
Socket_Kerberos_TGS [Java Application] C:\Program Files\Java\jdk1.8.0_121\jre\bi
start TGS service:
接收了 第1个请求
包5 收到!!!!!!!!!!!!!
接收了 第2个请求
包5 收到!!!!!!!!!!!!!
```

图 三-5 TGS 结果

```
Socket_Kerberos_V [Java Application] C:\Program Files\Java\jdk1.8.0_121\jre\bin\
start V service:
接收了 第1个请求
包7 收到!!!!!!!!!!!!!
接收了 第2个请求
包7 收到!!!!!!!!!!!!!
```

图 三-6 V 结果

# 第四章　服务器设计

## § 4.1 引言

### 4.1.1 设计概要

服务器代表 Kerberos 流程最后一步，在本小组设计中，Kerberos 流程最后一步所用控制包为 7 号和 8 号。其余服务器所应用是数据包，详细内容如下。

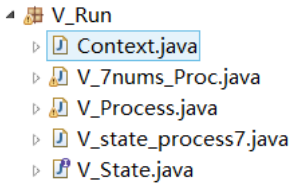


图 四-1 V 代码结构展示

其中，V_Process.java 代表服务器处理主要流程。V_State 代表状态机。V_state_process7 代表 7 号数据包的状态处理。Context 代表状态机的执行类。

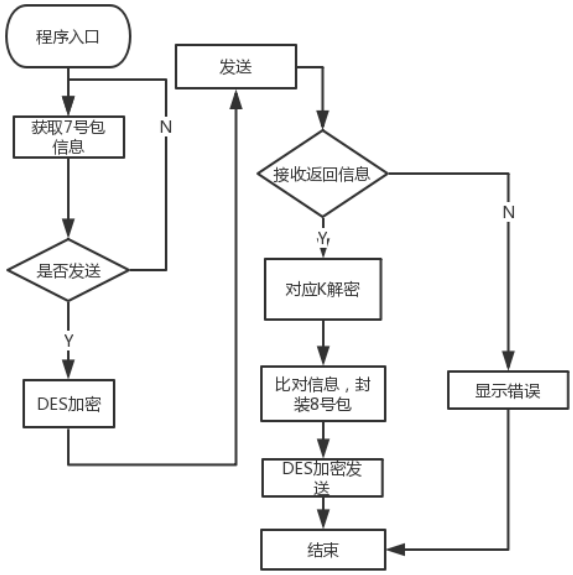


图 四-7 V 控制包信息

### 4.1.2 实现功能

1、 正确接收 C 请求的控制包和数据包。
2、 正确解开 C 请求的控制包和数据包。
3、 比较是否 C 控制信息是否正确。
4、 处理 C 的数据包，正确交互聊天代码。
5、根据约定格式封装和加密数据包。
6、Socket 发出至客户端。

### 4.1.3 模块输入

长度为 107 的 Stirng 控制包。
长度为的 200 的 String 数据包。

### 4.1.4 模块输出

长度为的 9 的 String 控制包。
长度为的 200 的 String 数据包。

### 4.1.5 状态机

数据包 9 触发应用服务。
数据包 10 触发发送消息服务（群聊和私聊）。
数据包 11 表示登陆流程。
数据包 12 表示用户退出。
数据包 19 触发文件操作。

### 4.1.6 接收控制包

**注意：**必须转为 Sting 类型。
1、属于控制包范围，格式标记为 7。
2、要求接受到的字符串 DES 解密。
3、7-69 位 char 经转换，格式参照 Ticket（v）格式，代表 Ticket（v），否则结束本模块。
4、69-107 位 char 经转换，格式参照 Authenticator(c)，否则结束本模块。
5、必须共 107 位，否则结束本模块。

| 名称 | 信息 |
| --- | --- |
| 7 号 | C-V |
| 代号 | 1-6 位 |
| Ticketv | 7-69 位 |
| Authenticator | 70-107 位 |

### 4.1.7 发送控制包

**注意：**必须转为 Sting 类型。

1、属于控制包范围，格式标记为 8。

2、前 6 位 char 经转换为 int 类型 8，代表控制包种类，否则结束本模块。

3、后 7-9 位，char 内容为 YES 或者 NO ，不够的补空格（末尾补空格）。

| 名称 | 信息 |
| --- | --- |
| 8 号 | V-C |
| 代号 | 1-6 位 |
| 信息 | 7-9 位 |

### 4.1.8 程序流程

1、文字描述：

始终监听客户端请求。

始终监听客户端提交的信息。

使用 K（v）DES 解密密文。如果解开，则继续。否则，结束本模块。

提取前 6 位，比对是否是 C 发来的请求服务的控制报文，否则，结束本模块。

提取 7-69 位，比对 Ticketv 内容。提取 K（c，v），IDc 和 ADc，IDtgs，TS4，和 LIFETIME4。分别存入对应字段。

提取 72-109 位，提取 Authenticatorc 内容，内容前 3 位为 IDc，4-18 位为 ADc，最后 20 位为 TS3。

比对 C 的 IDc 和 ADc 是否相同，相同则继续，否则结束模块。

比对 TS3 和 LIFETIME2，算出时间差，计算剩余时间，如果大于 0，则继续，否则

在剩余时间内开放聊天室服务。转发数据包。

根据数据包包头 6 位比对数据所对应的信息，执行相应的函数操作。比如 9 代表多人聊天室，数据内容转发至多人聊天室。

### 4.1.9 代码解析

**1、多线程 socket**

通过 s= new ServerSocket(5439);建立服务器 5439 端口，通过死循环 while（ture），Socket cs = s.accept()。不断监听是否有客户端请求连接。如果有，thread.start()，开启多线程服务。

服务器对应的线程类，其构造函数接口有参数 socket，代表对应客户端的 socket，RSA 代表存储的此客户端 X.509 关键信息（公钥、数字签名），JTextArea tx 代表服务器的接受到信息的 UI 界面及时更新，JTextArea tx1 代表服务器 UI 界面的在线用户名称更新。Lock lock，代表服务器的线程锁，防止多用户同时操作导致程序运行流程出错。

**2、7 号包信息**

服务器的 run()函数，要先检查客户端的 7 号包内容。

InputStream in = socket.getInputStream();

DataInputStream din = new DataInputStream(in);

通过得到 socket 的输入流，使用 DataInputStream 创造对象，读取客户端提交的信息。

按照数据包格式约定，对 Ticketv 进行解密，对 authentic 进行解密，并分别校验其正确性。

对于 ticektv，首先利用本 v 的秘钥进行解密。如果解不开，代表客户端错误。则停止本流程。解开后，对于解开的原文，根据 ticketv 的信息内容格式进行提取，放到 ticketv 类中对应的内容中。比如 Ticket_v.substring(0, 8)，利用 string 的 substring 函数提取在对应位的客户端和服务器的秘钥 kcv。

解开 ticketv 后，获得了 kcv，则可以比对 autc 的正确性。获取 ADc，IDc，TS 的信息。判断信息生存时间是否到期。

SimpleDateFormat("yyyy/MM/dd HH:mm:ss");用来将时间格式转换为 2016-08-10 20:40，分别获取客户端的时间和服务器的时间内容，将时间转换为 int 类型进行相减，如果结果大于 0 ，则正确，如果小于 0，则返回信息 error，强制客户端退出。

**3、8 号控制包信息**

如果 7 号包信息正确，则可以往客户端发送 8 号包控制包。首先获得服务器的当前时间，使用 kcv 进行加密。发送客户端 8 号控制包。

# §4.2 子模块——聊天室设计

## 4.2.1 流程图

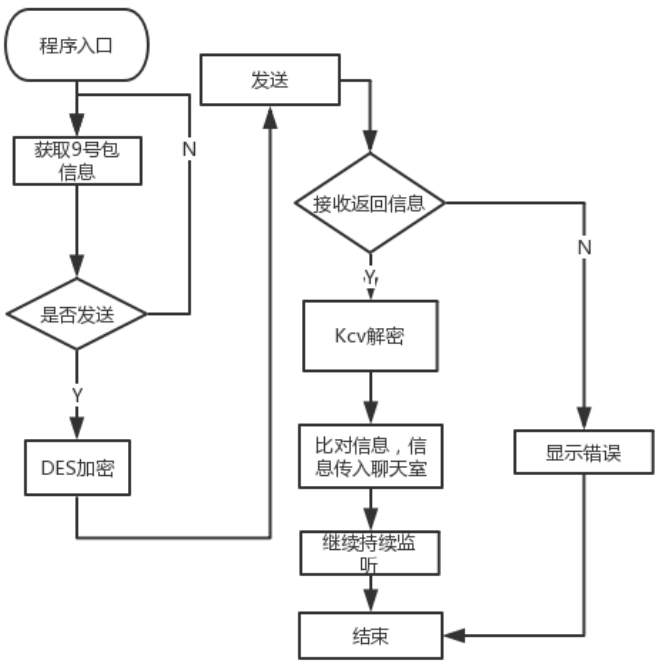


图 四-2 服务器处理数据包

## 4.2.2 设计概述

1、使用公有类 OnlineUsers 保存连接到服务器的客户端信息，包括 socket、rsa 公钥信息、用户名、客户端秘钥。

2、使用工厂模式设计用户连接过程。

```
private static final String DBDRIVER = "com.mysql.jdbc.Driver";
//private static final String DBURL = "jdbc:mysql://localhost:3306/shop?useU
private static final String DBURL = "jdbc:mysql://localhost:3306/chatroom";
private static final String DBUSER = "root";
private static final String DBPASSWORD = "";
private Connection conn;
```

图 四-3 数据库连接

数据库连接，一般常用如图所示常用语句。第一句代表驱动数据库服务，第二句代

表使用数据库的表 chatroom，第三句话是用户名，password 因为我的 mysql 没有密码，所以为空，最后是数据库连接 connection。使用方法 forName(DBDRIVER);开始数据库服务，使用 DriverManager.getConnection(DBURL,DBUSER,DBPASSWORD);链接数据库。

数据库创造用户和查询用户信息使用工厂模式。

接口有如下图中所示方法。

```
public boolean doCreate(User user) throws Exception;
public User findByUserName(String name) throws Exception;
public User findByUserNameandPasswd(String name,String password)throws Exception;
public String updateuserpasswd(String name,String password) throws SQLException, Exception;
```

图 四-4 接口方法

有一个生产类，负责生产用户。类中继承接口的方法，对方法进行重写。

对于 doCreate 方法。负责在数据库中创造用户。首先使用语句 insert into user (username,passwd,grade,birth,sex,email,phone,registertime) values (?,?,?,?,?,?,?,?)";，调用 prepareStatement()方法一次对语句中的？进行写入，依次为用户名、密码、年级、性别、生日、邮箱等信息（设计初期定的，内容尽量多不要少）。再调用 executeUpdate()方法执行，如果执行成功，则数据库中会放入用户的这些信息。

对于 findByUserName()方法。负责在数据库中查找通过用户名查找用户，返回的是 User 类。使用语句 select username,passwd,grade,birth,sex,email,phone,registertime from user where username=?，查找用户的信息，repareStatement(sql);处理语句，方法 executeQuery()执行语句，使用 rs.getString()获得 User 类中对应的信息存入 User 类中，返回这个类。

对于 findByUserNameandPasswd()方法。负责在数据库中查找通过用户名查找用户，返回的是 User 类。使用"selectusername,passwd,grade,birth,sex,email,phone,registertime from user where username=? and passwd=?"，查找用户的信息，repareStatement(sql);处理语句，方法 executeQuery()执行语句，使用 rs.getString()获得 User 类中对应的信息存入 User 类中，返回这个类。

对于 updateuserpasswd()方法。负责在数据库中查找通过更新用户信息，返回的是 String，代表修改成功或者失败。首先要调用使用方法 findByUserName()，判断是否 User 为空，"update user set passwd=? where username=?"，查找用户的信息，repareStatement(sql); ，setString 方法处理语句， executeQuery()执行语句，如果执行成功，则返回中文完成，如果查找不到用户，则返回找不到用户。

1、 服务器转发信息和文件

无论是客户端发送信息还是传送文件，服务器都始终扮演着中间者的角色。

发送信息时，客户端首先将信息发送给服务器，服务器接收到后，根据其要发送的用户转发给相应的用户。

发送文件时，客户端将文件发送给服务器，服务器先将文件接收保存完毕，再将文件发送给对应的用户，返回客户端发送成功。

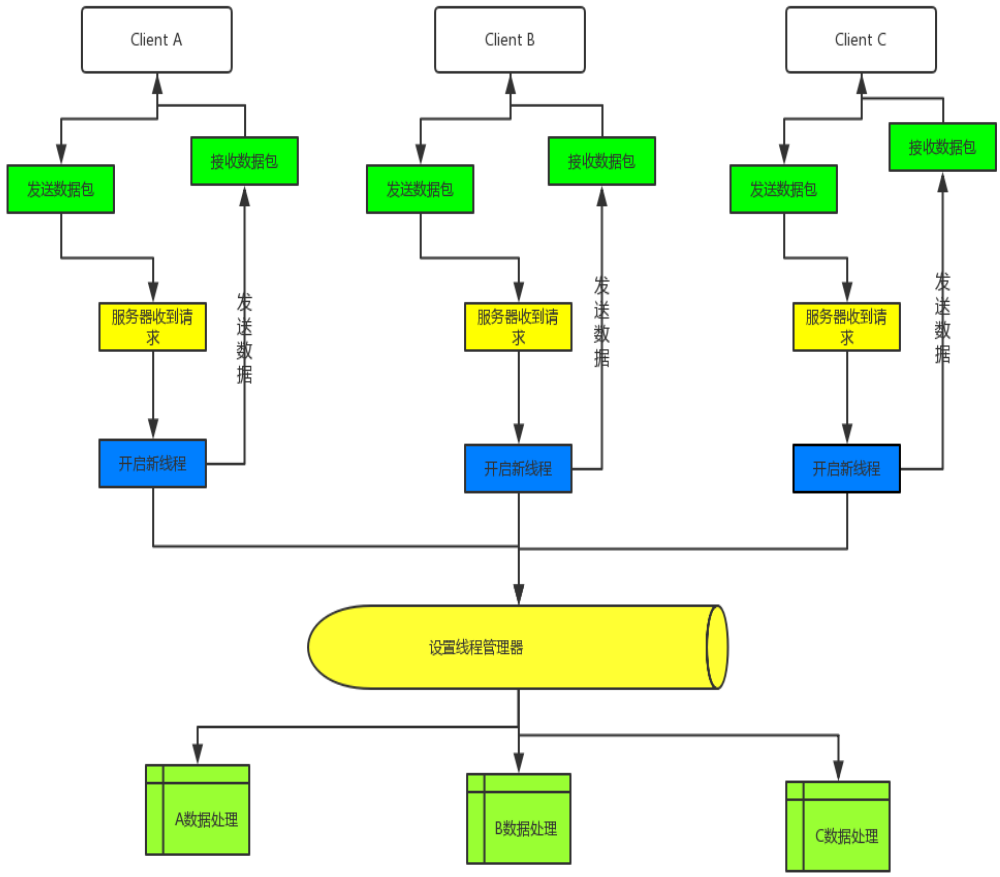


图 四-5 服务器对应多个 client 流程

认证成功之后，当服务器收到各客户端的数据包之后，将数据包进行解析，分解出数据项之后进行解密，将解密之后的会话内容广播到所有在线 client 端，服务器端主要流程如下：

（1） 监听客户端发来的数据报文
（2） 如果收到则解析数据包对数据部分解密
（3） 解密完之后将数据部分加密封装成包
（4） 广播给所有在线客户端

### 4.2.3 实现功能

1、显示用户是否在线。
2、显示用户历史消息。
3、用户之间在聊天室相互聊天。
4、私聊功能。

5、传递文件。
6、语音播放信息。

### 4.2.4 模块输入

数据长度限制为 200 位。

### 4.2.5 模块输出

数据长度限制为 200 位。

### 4.2.6 发送数据包格式

**注意：**必须转为 Sting 类型。
1、属于数据包范围，格式标记为 9-31。
2、前 6 位 char 经转换为 int 类型 9-31，代表数据包种类。
3、后 7-200 位，char 内容为聊天内容数据，不够的补空格（末尾补空格）。

### 4.2.7 程序流程与要求

1、 文字描述
客户端设计为多线程模式。可以在一台机子上运行多个用户。
客户端一直处于监听状态，如果收到服务器发来的数据包，则进行拆分，然后利用自己的 DES 进行解密，将解密后的数据显示在客户端界面上。
客户端监听用户输入，如果有数据发送，则加密封装成数据包发送给服务器。
服务器设计为多线程模式，可以接入多个客户端。
服务器开启后，时刻监听客户端交互信息。
服务器如果收到客户端的消息，进行解密
将客户端消息封装成数据包转发给所有已连接客户端信息。
2、 流程图

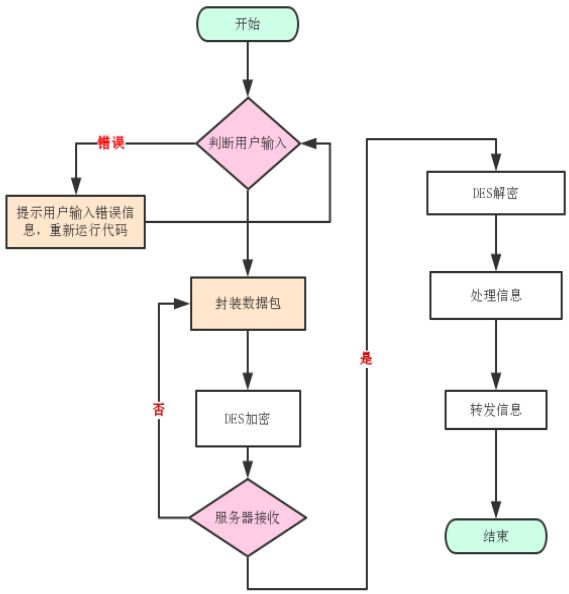


图 四-6 聊天室

## 4.2.8 代码详解

**1、【功能】：用户信息服务器中的保存**

【方式】：在开启应用服务器前，首先需要保存当前连接的 socket 信息，创建 hashmap 对象，分别保存 socket 和 String 类型，socket 代表客户单的 socket，String 代表用户名，这样在服务器以后收到信息知道根据用户名信息应该发往哪个 socket。除此之外，还要服务器为每个客户端的分配秘钥 kcv，用于将发送的信息 DES 加密。保存每个 socket 对应的公钥信息，用户数字签名，防止消息抵赖。

```
OnlineUsers.ssm = new HashMap<String,Socket>();
OnlineUsers.sskey=new HashMap<String,Socket>();
OnlineUsers.onlineUsers  = new ArrayList<String>();
OnlineUsers.clientList = new ArrayList<Socket>();
OnlineUsers.name_passwd=new HashMap<String,String>();
```

图 四-7 服务器的客户端信息

其中，OnlineUsers 的内容就是上图所示的几个哈希表和 list 表。

```
public class OnlineUsers {
    static public ArrayList<Socket> clientList;
    static public  ArrayList<String> onlineUsers;
    static public  Map<String,Socket> ssm;
    static public  Map<String,Socket> sskey;
    static public  Map<String,String> name_passwd;
}
```

图 四-8 OnlineUsers 类

**2、【功能】：登陆处理**

【方式】：登陆数据包是 11 号数据包。调用 ClientDataGramAnalyzer 的 getNname 查找用户名，getpassword 查找密码。如果首先判断用户是否登陆，查看调用 onlineUsers.contains，查看用户名是否存在，如果存在，则返回 13 号数据包，代表用户已经登陆。

如果用户没有登陆，调用方法 findByUserNameandPasswd 查找 User，如果密码错误，则返回 12 号数据包，代表登陆密码错误。

如果完全正确，则用户正确登陆。获取当前在线的所有用户，使用方法 onlineUsers.toArray(new String[OnlineUsers.onlineUsers.size()])。将用户名称放入数组里，使用 for 循环方法，提醒当前在线的所有用户此用户登陆，即 18 号数据包。for(int i = 1; i< accepters.length;i++)。mapOnlineUsers.ssm.get(accepters[i]).getOutputStream()方法获得用户名对应的 socket，将要发送的信息封装成 18 号数据包格式，首位是 type 标识，后面依次为发送方用户名称。

发送完毕后，将 OnlienUsers 的信息增加此用户。

```
OnlineUsers.clientList.add(this.socket);
OnlineUsers.onlineUsers.add(this.user.getUserName());//********
OnlineUsers.ssm.put(this.user.getUserName(),this.socket);
OnlineUsers.sskey.put(keypass, this.socket);
OnlineUsers.name_passwd.put(this.user.getUserName(),keypass);
```

图 四-9 用户增加

调用 hashmap 的 add 方法即可。

**3、【功能】：发送消息**

【方式】：发送消息使用 10 号数据包。

接收到信息后，利用包头需要首先判断是几号包（状态机激发），进入 10 号包处理流程。在代码里的 10 号数据包代表 0 信息处理过程。首先将信息调用 des 解密，解密秘钥使用对应客户端的秘钥。秘钥解开后，因为服务器 UI 的需要，需要首先将 Jtextarea 的显示内容更新，使用 Jtextarea 的 append 方法。调用 ClientDataGramAnalyzer 类的 getAccepters 提取信息的内容。

ClientDataGramAnalyzer 类，有 gettype 方法，此方法表示获取数据包的类型。GetName 获取数据包中用户的名字。Getpassword 获得数据包中的密码等等。数据信息格式统一为以”&&&“进行信息区分。对于收到的信息，直接调用 String 的 split 方法以&&&转换为 String 数组，每个则数组的 0 处放置 type，代表包的类型。1 处存放用户名或者用

户组，2 处存放信息。

使用 getAccepters 提取信息的内容后，即可提取到信息要转发的目标用户。如果是群发，则数组里有所有用户的名称。如果是私聊，则只有一个用户。使用 for 循环，循环限制在用户的数量，即数组的大小。for(int i = 1; i< accepters.length;i++)。使用这个哈希 map 的 OnlineUsers.ssm.get(accepters[i]).getOutputStream()方法获得用户名对应的 socket，将要发送的信息封装成 10 号数据包格式，首位是 type 标识，后面依次为发送方用户名称、发送的信息内容。再调用 DES 加密接口将信息除了标识位不加密其他部分加密，利用找到的客户端的 socket 发送给对应客户端。

**4、【功能】：用户退出**

【方式】：用户退出代表用户离线。需要将用户从服务器的用户记录中删除。使用 hashmap 的 remove 方法依次将用户的 socket、用户名、Rsa 等信息全部删除。在调用 for(int i = 1; i< accepters.length;i++)。使用这个 OnlineUsers.ssm.get(accepters[i]).getOutputStream()方法获得用户名对应的 socket，将要发送的信息封装成 16 号数据包格式，首位是 type 标识，后面依次为离线用户名称。告诉其他客户端他已经离线，并且更新其用户列表。

**5、【功能】：文件发送**

【方式】：文件发送需要服务器先接受被发送文件的名称和要发送对象的用户名称信息。接受客户端提交的 19 号数据包，数据包后面以&&&带着用户的发送文件名称。然后使用 socket 的 in.readLine()方法将文件接收。接收文件使用方法 BufferedWriter，客户端发送文件完毕后，需要发送一个结束信息，否则服务器会陷入这里不断接收信息发生错误。代码中发送字符串”comple“表示文件发送完毕。接收到的字符要先 DES 解密后调用 BufferedWriter 的 write 方法写入文件里。接收完成后，向发送文件的客户端发送 10 号数据包，代表发送成功。

服务器接收完毕文件后，要将文件发送给指定的用户客户端。通过 getAccepters 方法从哈希表中获得用户名对应的 socket。需要先向对应的客户端，服务器先发送被发送文件的名称和要发送文件的用户名称信息。接受客户端提交的 19 号数据包，数据包后面以&&&带着用户的发送文件名称。发送文件使用 socket 的 println 方法，发送之前先 DES 加密。发送完毕后，想对应客户端发送字符串”comple“表示文件发送完毕。

**6、【功能】：用户注册**

【方式】：登陆数据包是 14 号数据包。调用 ClientDataGramAnalyzer 的 getNname 查找用户名，getpassword 查找密码。如果首先判断用户是否已经注册，查看调用 onlineUsers.contains，查看用户名是否存在，如果存在，则返回 3 号数据包，代表用户已经登陆。

### 4.2.9 结果截图

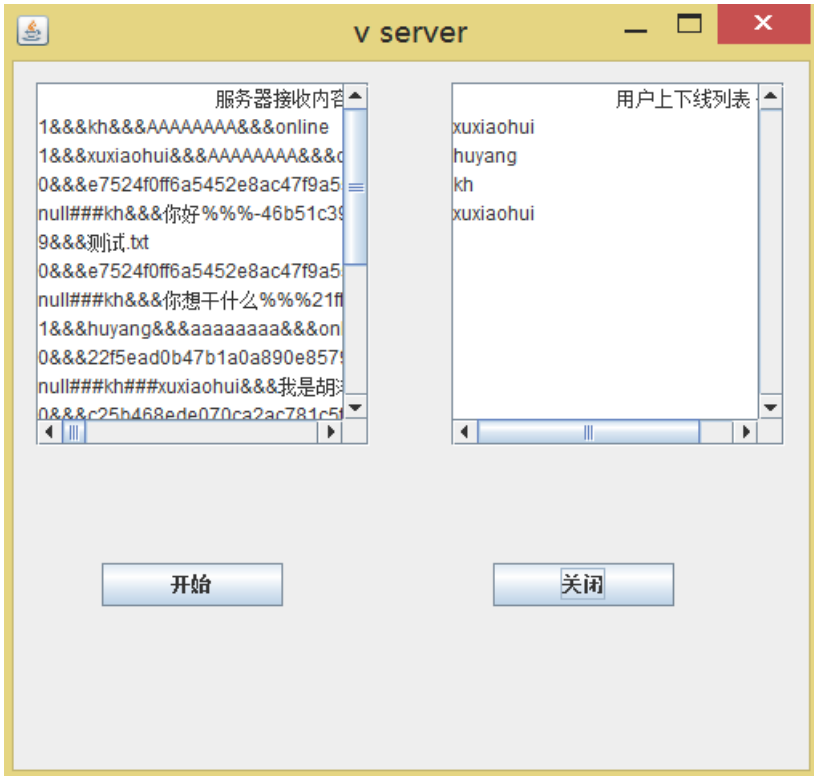


图 四-10 服务器截图

左侧服务器接受内容区域框，分别为用户上线信息展示，用户消息接收到的密文展示和消息解析后的展示。以及接收到文件的加密显示。

右侧是用户的上下线展示。

# 第五章　客户端

CLient 要实现功能为：请求访问 as、tgs、v，实现聊天功能。

## 5.1.1 设计概要

代码结构如图所示。

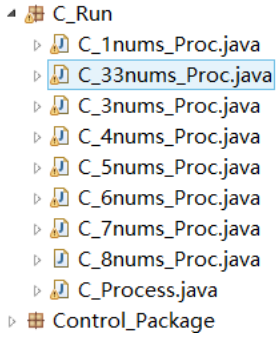


图 五-1 代码结构

Kerberos 的发送 3、4、5、6、7 号包和接收 8 号包流程。

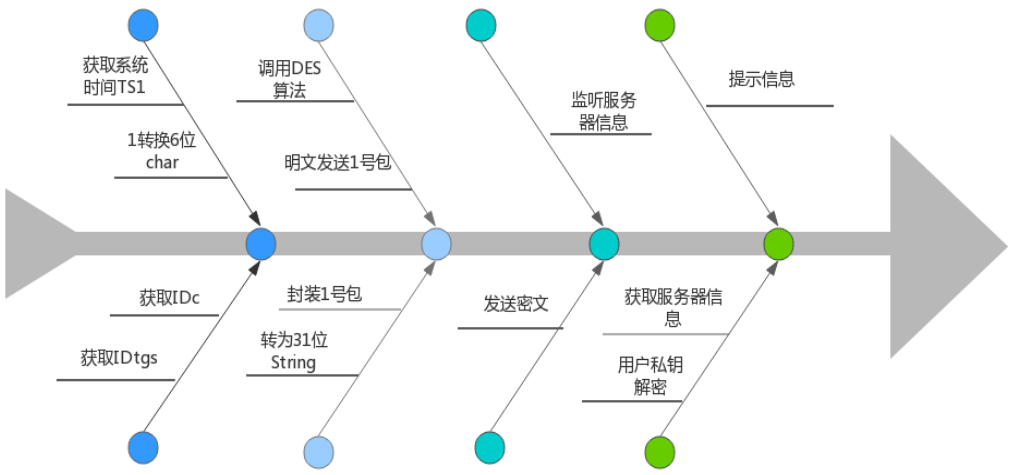


图 五-2 C 和 AS 交互流程

**3 号.C -> AS　IDc|IDtgs|TS1**

客户端向 AS 请求获取 Tickettgs，发送的报文中包含客户端的 ID，TGS 的 ID，当前系统时间 TS1。

客户端操作流程如下：

（1） 获取 IDc

（2） 获取 IDtgs

（3） 获取 TS1

（4） 拼接成数据包

（5） 发送给 AS

**4 号. AS-> C EKc[Kc,tgs|IDtgs|TS2|lifetime2|Tickettgs]**

当 AS 收到客户端的报文后，AS 会随机生成一个 C 和 tgs 之间的对称加密的秘钥 Kc,tgs，这个秘钥的作用是 C 与 TGS 之间的会话秘钥，在下一步的访问 TGS 的时候使用。IDtgs 表示访问的 TGS 的 ID，TS2 表示当前系统时间，lifetime2 表示 Tickettgs 的生命周期，进行消息重放的时候使用，太短则用户需要重复输入口 令，太长则为攻击者提供了大量机会。Tickettgs 是客户端访问 TGS 是的票据，里面包含了 Kc,tgs|IDc|ADc|IDtgs|TS2||lifetime2 然后采用 EKtgs 加密封装，EKtgs 是只有 AS 与 TGS 之间才知道的秘钥，以预防 Tickettgs 被篡改。

TGS 端在收到票据以后可以利用 EKtgs 解密获取 Kc,tgs 与 Client 会话。

AS 操作流程如下：

（1） 解析 Client 发来的包

（2） 随机生成 Kc,tgs

（3） 获取 IDtgs

（4） 获取 TS2

（5） 获取 lifetime2

（6） 生成 Tickettgs

（7） 拼装各字段

（8） 发送给 Client

生成 Tickettgs 的流程如下：

（1） 获取 Kc,tgs

（2） 获取 IDc

（3） 获取 ADc

（4） 获取 IDtgs

（5） 获取 lifetime2

（6） 获取 EKtgs（AS 与 TGS 之间的秘钥）

（7） 拼装各字段

（8） 采用 EKtgs 加密

（9） 生成 Tickettgs EKtgs[ Kc,tgs|IDc|ADc|IDtgs|TS2||lifetime2]

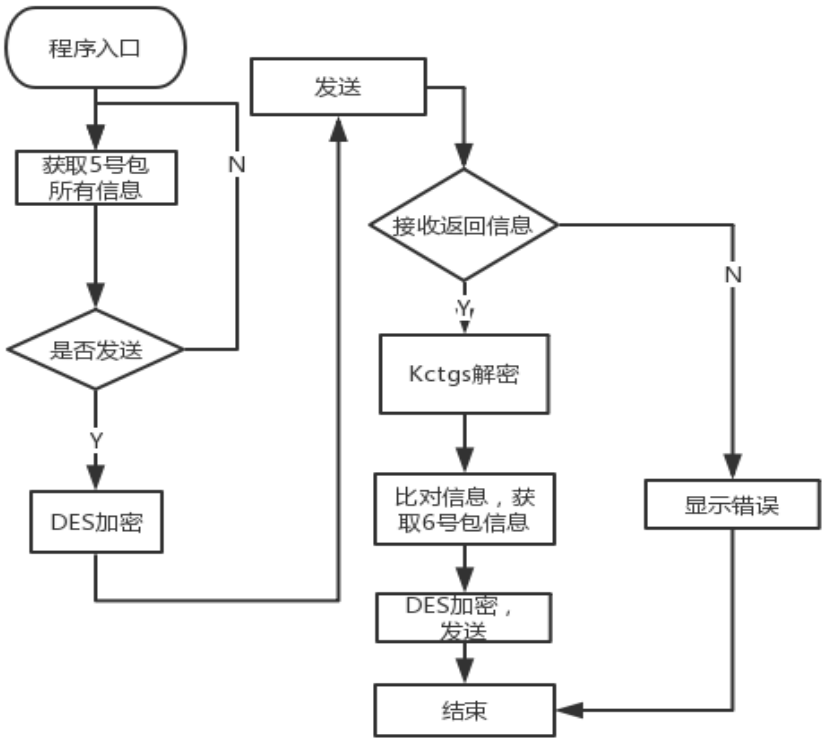


图 五-3 C 和 TGS 流程

**5 号. C->TGS　IDv|Tickettgs|Authenticator**

Tickettgs EKtgs[ Kc,tgs|IDc|ADc|IDtgs|TS2||lifetime2]

Authenticator：Kc,tgs [ IDc|ADc|TS3]

客户端收到 AS 发来的包后对包进行解析，首先输入口令 EKc 进行解密，解密的结果是可以获取 Kc,tgs（AS 生成的 client 与 TGS 会话的秘钥）、IDc、ADc 和 Tickettgs 等，然后就可以利用 Kc,tgs 对 IDc|ADc|TS3 加密生成 Authenticator，然后 IDc|Tickettgs|Authenticator 发送给 TGS。这样我们就明白了 AS 随机生成 Kc,tgs,以后是如何把 Kc,tgs 告诉 Client 和 TGS 的。

Client 操作流程如下：

（1） 输入 EKc 解析 AS 发来的包，获取 Kc,tgs 以及 Tickettgs

（2） 获取 IDv

（3） 获取 ADc

（4） 获取 TS3

（5） 生成 Authenticator

（6） 拼接 IDv|Tickettgs|Authenticator

（7） 发送给 TGS

**6 号.TGS->C EKc,tgs[Kc,v|IDv|TS4|Ticketv ]**

TGS 收到 Client 发来的包后，首先分割出来 IDv，Tickettgs，Authenticator。然后通过自己存的 Ektgs（AS 与 TGS 之间的会话秘钥）解开 Tickettgs，获取 Kc,tgs,IDc,ADc,TS2。然后利用 Kc,tgs 解开 Authenticator，获取 IDc 和 ADc 和 TS3 与 Tickettgs 中的数据进行校验，如果不符合要求，则认证失败。否则，TGS 随机生成一个 KCv（Client 与 Server 之间的秘钥），然后生成一个 Ticketv，采用 EKc,tgs 加密发送给客户端。

TGS 操作流程如下：

（1） 解析 Client 发来的包，获取 IDv、Tickettgs、Authenticator

（2） 用 EKtgs 解析 Tickettgs 获取 Kc,tgs,IDc,ADc 等

（3） 用 Kc,tgs 解析 Authenticator 获取 IDc,ADc 等

（4） 校验数据信息

（5） 随机生成一个 Kc,v

（6） 获取 IDv

（7） 获取 TS4

（8） 生成 Ticketv

（9） 拼接各字段，采用 EKc,tgs 加密

（10） 发送给 Client

生成 Ticketv 的流程如下：

（1） 获取 Kc,v

（2） 获取 IDc

（3） 获取 ADc

（4） 获取 IDv

（5） 获取 TS4

（6） 获取 LifeTime4

（7） 采用 EKv（TGS 与 V 之间的秘钥）加密

（8） 生成 Ticketv EKv[ Kc,v|IDc|ADc|IDv|TS4|LifeTime4 ]

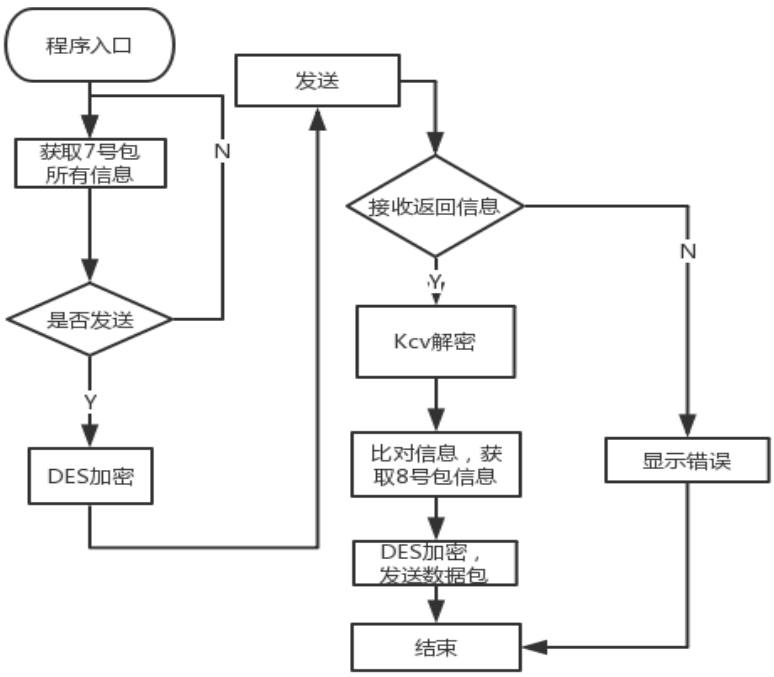


图 五-5 c 和 V 流程

**7 号. C->V　Ticketv|Authenticator**

Ticketv　EKv[ Kc,v|IDc|ADc|IDv|TS4|LifeTime4 ]

Authenticator EKc,v[IDc|ADc|TS5]

客户端收到 TGS 发来的包后，先用 Ec,tgs 解密，获取到 Ticketv，Kc,v 等字段。

然后利用 Kc,v 加密 IDc、ADc、TS5 等字段生成 Authenticator，然后与 Ticketv 一起发送给服务器 V。

Client 操作流程如下：

（1） 采用 EKc,tgs 解密 TGS 发来的包，获取 Ticketv 和 Kc,v

（2） 获取 IDc

（3） 获取 ADc

（4） 获取 TS5

（5） 用 Kc,v 生成 Authenticator

（6） Authenticator 与 Tickev 拼接发送给 V

**8 号. V->C　EKc,v[ TS5+1]**

服务器收到 Client 端的包之后，首先分割出 Ticketv，然后利用自己保持的 Kv 对 Ticketv 进行解密，获取 Kc,v、IDc、ADc、IDv 等字段，然后分割出 Authenticator，利用 Kc,v 进行解密，解密之后获取 IDc、ADc 等字段进行校验。如果校验成功将 TS5 的值加 1 然后用 Kc,v 加密之后传给客户端，到此认证成功。

服务器操作流程如下：

（1）解析 clien 发来的包，获取 Ticketv 和 Authenticator

（2）用 Kv 解密 Ticketv，获取 Kc,v 等字段

（3）用 Kc,v 解密 Authenticator，获取 IDc、ADc，TS5

（4）进行校验。

（5）Kc,v 加密[TS5+1]发送给客户端，表示认证成功。

## 5.1.2 Client 之间的关系

当用户采用私聊的方式时，多个 Client 之间是点对点的关系，即 client 端不通过服务器进行数据传递，两个 Client 之间进行点对点之间的数据传输。两个 Client 之间进行证书的交换之后，确定采用 RSA 加密算法获取会话秘钥，然后根据会话秘钥采用 DES 对称加密进行数据传输。

当用户采用群聊的方式，多个 Client 之间没有关系，统一由服务器来进行数据包的中转与控制，每个 Client 与服务器构成 C/S 架构，服务器将用户发来的包进行广播，用户接受服务器发来的数据包并显示在本地聊天室中。

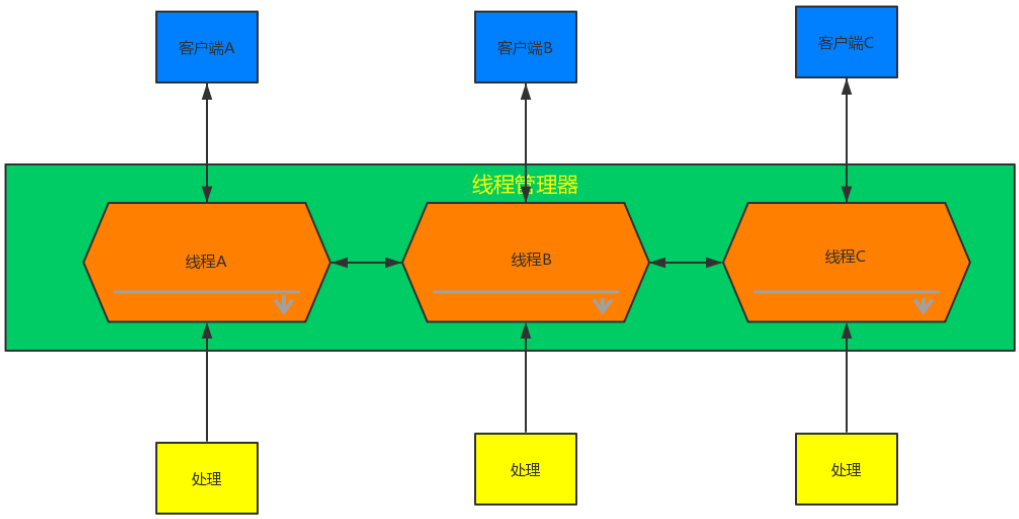


图 五-6 客户端在服务器中关系

## 5.1.3 实现功能

1、 正确发送 C 请求 AS 的身份校验注册控制包。

2、 正确解开 AS 的控制包。

3、 正确解开 v 的控制包。

4、正确发送 c 发往 v 的数据包。

5、正确解开 v 的数据包。

6、比较是否 C 信息是否正确。

7、根据约定格式封装和加密数据包。

8、Socket 发出至客户端。

### 5.1.4 模块输入

长度为 9 的 String 的 8 号控制包。

长度为 200 的 String 的 9 号数据包。

长度为 9 的 String 的 2 号控制包。

### 5.1.5 模块输出

长度为 45 的 String 的 1 号控制包。

长度为 107 的 String 的 7 号控制包。

长度为 200 的 String 的 9 号数据包。

### 5.1.6 状态机

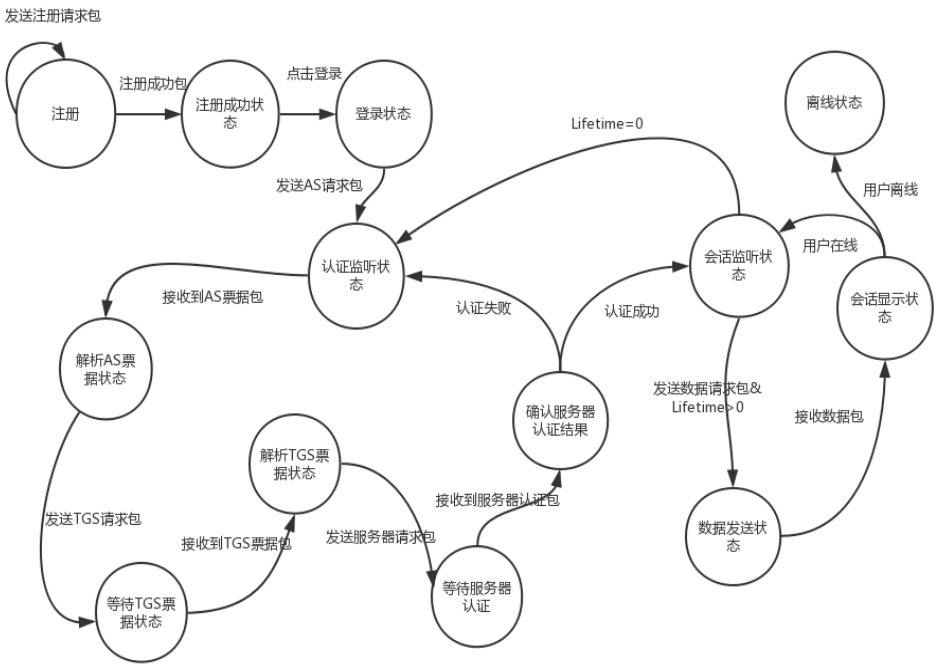


图 五-7 状态机

33 号包触发处理 AS 身份认证流程。

32 触发 Client 发送 X.509 信息至 AS。

1 触发注册，2 触发接收注册信息。3、4、5、6、7、8 依次为 AS、TGS、V 认证，每

个对应两个。

### 5.1.7 发送注册信息

**注意：**必须转为 Sting 类型。

因为 AS 使用 while(readutf)不断听取客户端请求，刚开始要告诉 AS，0 号信息是请求 X.509 公钥交换与身份认证。

1、 客户端发送 0 号包，内容为”000000”，请求服务器发回 33 号控制包。

客户端接收 33 号包，因为使用 RSA 加密解密，而使用的 RSA 算法是对一个 char 分别加密，所以需要以一个标记来区分转为字符串的加密后的信息。这里以”&“来区分。

客户端发送 32 号包，使用 AS 端的公钥信息加密。使用的 RSA 算法时对一个 char 分别加密，所以需要以一个标记来区分转为字符串的加密后的信息。这里以”&“来区分。

| 名称 | 信息 |
| --- | --- |
| 32 号 | 控制使用 |
| 代号 | 1-6 位 |
| 信息 | 7-60 位，X.509 信息 |

| 名称 | 信息 |
| --- | --- |
| 33 号 | 控制使用 |
| 代号 | 1-6 位 |
| 信息 | 7-60 位，X.509 信息 |

### 5.1.8 发送 1 号控制包

**注意：**必须转为 Sting 类型。

1、属于控制包范围，格式标记为 1。

2、要求接受到的字符串 RSA 解密。

3、前 6 位 char 经转换为 int 类型 1，代表控制包种类，否则结束本模块。

4、7-26 位 char 经转换为 String 类型，长度必须小于等于 20，代表用户名，否则结束本模块。

5、27-29 位 char 经转换为 int 类型，且大于等于 0 小于 8，标记客户端代号，否则结束本模块。

6、30-45 位 char 经转换为 String 类型，代表用户秘钥，必须小于 16 位，且字符串内无空格，否则结束本模块。

7、必须共 45 位，否则结束本模块。

| 名称 | 信息 |
| --- | --- |
| 1 号 | 注册使用 |
| 代号 | 1-6 位 |
| 用户名 | 7-26 位 |

| IDc | 27-29 位 |
| --- | --- |
| 密码 | 30-45 位 |

### 5.1.9 接收 2 号控制包

**注意：**必须转为 Sting 类型。

1、属于控制包范围，格式标记为 2。

2、要求接收到的字符串 RSA 解密。

3、前 6 位 char 经转换为 int 类型 2，代表控制包种类，查看是否是其他控制包，否则结束本模块。

4、7-9 位 char 经转换为 string 类型，且为 YES 和 NO，标记是否注册成功，否则结束本模块。

5、必须共 9 位，否则结束本模块。

| 名称 | 信息 |
| --- | --- |
| 2 号 | AS 返回注册信息 |
| 代号 | 1-6 位 |
| 信息 | 6-9 位 |

### 5.1.10 发送 3 号控制包

**注意：**必须转为 Sting 类型。

1、属于控制包范围，格式标记为 3。

2、要求接受到的字符串 DES 解密。

3、前 6 位 char 经转换为 int 类型 3，代表控制包种类，否则结束本模块。

4、7-9 位 char 经转换为 int 类型，且大于等于 0 小于 8，标记客户端代号，否则结束本模块。

5、10-11 位 char 经转换为 int 类型，且大于等于 0 小于 4，标记访问 tgs 标号，否则结束本模块。

6、11-31 位 char 经转换为 String 类型，代表系统时间，格式必须为年月日时分秒，且字符串内无空格，样例：2018 年 4 月 28 日 8:35:42，否则结束本模块。

7、必须共 31 位，否则结束本模块。

| 名称 | 信息 |
| --- | --- |
| 3 号 | C-AS |
| 代号 | 1-6 位 |
| IDv | 7-9 位 |
| IDtgs | 10-11 位 |
| TS | 12-31 位 |

### 5.1.11 接收 4 号控制包

**注意：**必须转为 Sting 类型。

1、属于控制包范围，格式标记为 4。

2、要求接收到的字符串 DES 解密。

3、前 6 位 char 经转换为 int 类型 4，代表控制包种类，查看是否是其他控制包，否则结束本模块。

4、7-9 位 char 经转换为 int 类型，且大于 0 小于等于 8，标记是哪个客户端，如果不是本机，否则结束本模块。

5、10-25 位 char 经转换为 String 类型，且必须 16 位，代表 K（c，tgs）秘钥。

6、26-27 位 char 经转换为 int 类型，且大于 0 小于等于 4，否则结束本模块。

7、28-47 位 char 经转换为 Stirng 类型，代表系统时间。

8、48-53 位 char 经转换为 String 类型，代表存活时间。

9、54-112 位 char 经转换，为 Ticket（tgs），参考其格式转换。

10、必须 112 位，否则结束本模块。

| 名称 | 信息 |
| --- | --- |
| 4 号 | AS-C |
| 代号 | 1-6 位 |
| Kctgs | 7-22 位 |
| IDtgs | 23-24 位 |
| TS4 | 25-44 位 |
| Lifetime | 45-50 位 |
| Tickettgs | 51-112 位 |

### 5.1.12 发送 5 号控制包

**注意：**必须转为 Sting 类型。

1、属于控制包范围，格式标记为 5。

2、要求接收到的字符串 DES 解密。

3、前 6 位 char 经转换为 int 类型 5，代表控制包种类，否则结束本模块。

4、7-9 位 char 经转换为 int 类型，且大于等于 0 小于 8，标记服务器 v 代号，否则结束本模块。

5、10-71 位 char 经转换，格式参照 Ticket（tgs）格式，代表 Ticket（tgs），否则结束本模块。

6、72-109 位 char 经转换为 int 类型，格式参照 Authenticator(c)，否则结束本模块。

7、必须共 109 位，否则结束本模块。

| 名称 | 信息 |
| --- | --- |
| 5 号 | C-TGS |

| 代号 | 1-6 位 |
| --- | --- |
| IDv | 7-9 位 |
| Tickettgs | 10-71 位 |
| Authenticator | 72-109 位 |

## 5.1.13 接收 6 号控制包

**注意：**必须转为 Sting 类型。

1、属于控制包范围，格式标记为 6。

2、要求接受到的字符串 DES 解密。

3、前 6 位 char 经转换为 int 类型 6，代表控制包种类，查看是否是其他控制包，否则结束本模块。

4、7-22 位 char 经转换为 String 类型，长度必须小于等于 16，代表 k（cv）客户端与服务器 v 的秘钥，否则结束本模块。

5、23-25 位 char 经转换为 int 类型，且大于等于 0 小于 8，标记服务器 v 代号，否则结束本模块。

6、26-45 位 char 经转换为 String 类型，代表系统时间，格式必须为年月日时分秒，且字符串内无空格，样例：2018 年 4 月 28 日 8:35:42，否则结束本模块。

7、46-108 位经转换，参考 Ticket（v）。

8、必须共 108 位，否则结束本模块。

| 名称 | 信息 |
| --- | --- |
| 6 号 | TGS-C |
| 代号 | 1-6 位 |
| Kcv | 7-22 位 |
| IDv | 23-25 位 |
| TS4 | 26-45 位 |
| Ticketv | 46-108 位 |

## 5.1.14 发送 7 号控制包

**注意：**必须转为 Sting 类型。

1、属于控制包范围，格式标记为 7。

2、要求接受到的字符串 DES 解密。

3、7-69 位 char 经转换，格式参照 Ticket（v）格式，代表 Ticket（v），否则结束本模块。

4、69-107 位 char 经转换，格式参照 Authenticator(c)，否则结束本模块。

5、必须共 107 位，否则结束本模块。

| 名称 | 信息 |
| --- | --- |
| 7 号 | C-V |

| 代号 | 1-6 位 |
| --- | --- |
| Ticketv | 7-69 位 |
| Authenticator | 70-107 位 |

### 5.1.15 接收 8 号控制包

**注意：**必须转为 Sting 类型。

1、属于控制包范围，格式标记为 8。

2、前 6 位 char 经转换为 int 类型 8，代表控制包种类，查看是否是其他控制包，否则结束本模块。

3、后 7-9 位，char 内容为 YES 或者 NO ，不够的补空格（末尾补空格）。

| 名称 | 信息 |
| --- | --- |
| 8 号 | V-C |
| 代号 | 1-6 位 |
| 信息 | 7-9 位 |

### 5.1.16 发送 9 号数据包

**注意：**必须转为 Sting 类型。

1、属于数据包范围，格式标记为 9-31。

2、前 6 位 char 经转换为 int 类型 9-31，代表数据包种类。

3、后 7-200 位，char 内容为聊天内容数据，不够的补空格（末尾补空格）。

| 名称 | 信息 |
| --- | --- |
| 9 号 | 聊天 |
| 代号 | 1-6 位 |
| 信息 | 7-200 位 |

### 5.1.17 程序流程

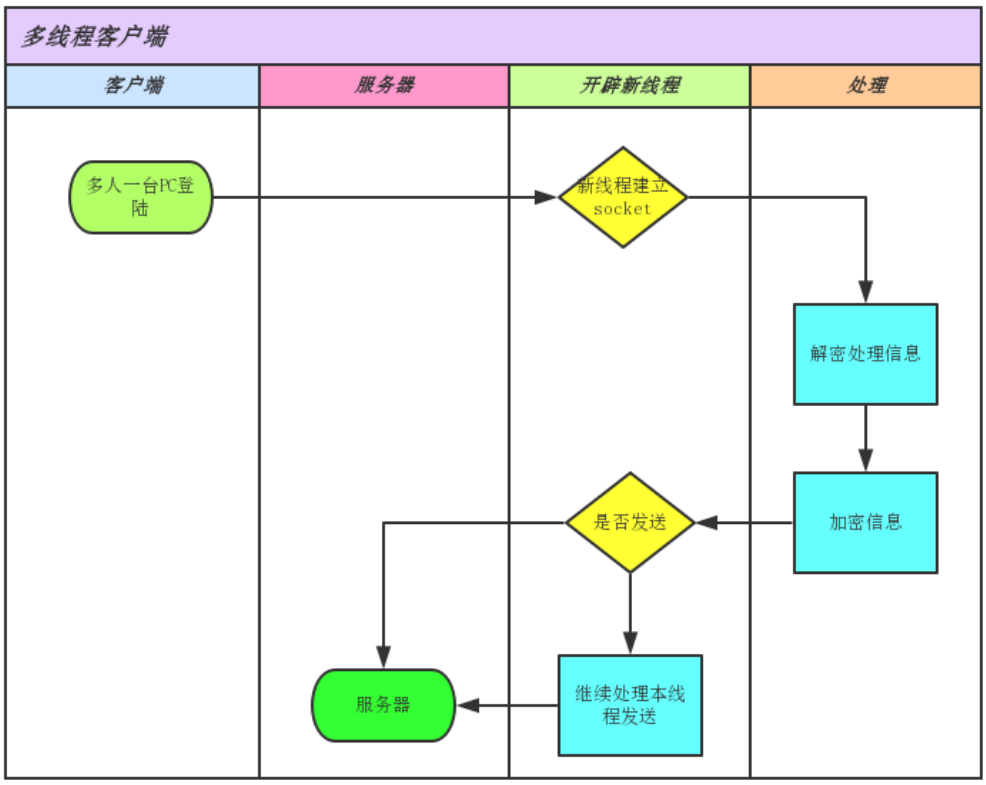


图 五-8 同 PC 多人登陆流程

## 5.1.18 代码解析

**1、注册修改流程**

需要先从注册 UI 界面中获取用户输入的用户名、密码信息，使用 Rsa 加密算法，将获得的 AS 的公钥进行加密。加密后，带上包头 000001 代表这是 1 号控制包进行注册流程。如果是修改用户信息，只需要将 1 号控制包最后加入一个字符串”xiugai“，改变 1 号数据包长度，AS 进行判断即可。

发送后，期待接收 2 号控制包。如果收到的信息是 YES，则表示注册成功。如果收到的信息是”HIND“，则表示已经存在此用户。如果收到的是”NO“表示注册失败。如果收到 XIU，则表示修改成功。如果收到 XIUB，则表示修改失败。如果收到 BUCZ，则表示修改过程中不存在这个用户信息。

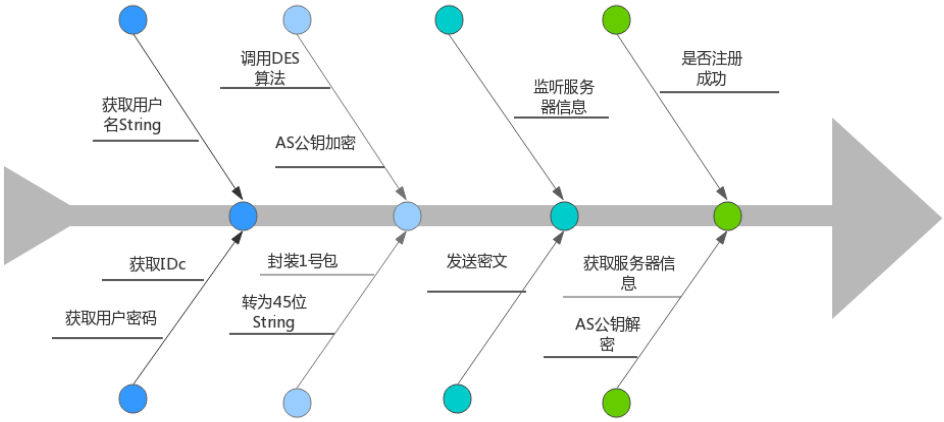


图 五-9 注册流程

2、Kerberos 流程如流程概述所述，只要根据数据包规定位数，按照 String 的 substring 方法进行提取信息，对信息转化。发送出去的和接收到的数据包都按照规定格式来就是正确的。

# §5.2 子模块——聊天室设计

## 5.2.1 实现功能

1、 群聊发送
2、 私聊发送
3、 文件发送
4、 好友列表
5、 历史消息显示
6、 登录前的身份验证

## 5.2.2 设计概述

客户端的每个模块分别是独立的，只要状态机写好，判定正确每种包该执行哪种操作，这种操作对应的结果是正确的即可。

主要集中在对服务器发来的消息进行分析和判断。

有自己定义的类 ServerDataGramAnalyzer 对服务器消息进行判断。

ServerDataGramAnalyzer 有方法 gettype、getname、getmessage 等。按照包的格式，其每个关键信息都以&&&作为分隔符，所以对发来的信息解密后需要使用 String 的 spli 方法分隔成其他的 String 数组。分隔完毕后，String 数组的 0 处放置包的类型，1 处放置发

送方用户名称，2 处放置 message 信息。最后是发送方的数字签名，用来放置消息抵赖。

```
public static String getState(String datagram){
    String []stringarr = datagram.split(delimiter);
    return stringarr[2];
}
```

图 五-10 客户端消息处理

分析出包的类型后，根据状态机的设计，执行每种包对应的操作即可。每种模块是分散的，并没有像服务器那样有在线用户之间的关系。

**. C->V EKc,v[ data]**

认证成功之后，客户端与服务器进行数据通信，因为数据量比较大采用 DES 加密算法，采用的秘钥就是 C 与 V 通过认证流程获取的 EKc,v,为了防止秘钥被破解，C 与 V 之间的秘钥当生存时间到了之后就需要进行重新生成，进行 AS 重放，重放时间不宜过短也不宜过长，过短会导致重放次数太多，过长会导致用户秘钥被破解的时间增加。

客户端主要流程如下：

（1） 监听服务器数据报文

（2） 如果接收到服务器数据包，则进行解密将数据显示在本地

（3） 如果有消息发送则输入聊天数据进行加密

（4） 加密完之后封装成数据包

（5） 发送给 V

## 5.2.3 数据包统一信息格式

| 报文类型（type）6 位 | 源用户名称 | 目的用户名称 | 消息内容 | 数字签名，防止消息抵赖 |
|---|---|---|---|---|
| 各个关键信息，因要数据统一处理，但每种包位数不同，信息之间以&&&区分，用户名称之间以###进行区分。 | | | | |

## 5.2.4 消息包 10

| 名称 | 信息 |
|---|---|
| 10 号 | 信息 |
| 代号 | 1-6 位 |
| 信息 | 7-200 位，长度不定，但一般小于 200 位。包括待接收方的用户名字，上线状态，消息内容，消息签名，以标识符&&&对信息进行区分。根据用户名进行群聊私聊。 |

### 5.2.5 登陆包 11 号

| 名称 | 信息 |
| --- | --- |
| 11 号 | 登陆信息 |
| 代号 | 1-6 位 |
| 信息 | 7-200 位，包括用户名、密码，上线状态，以标识符&&&对信息进行区分 |

### 5.2.6 退出包 12 号

| 名称 | 信息 |
| --- | --- |
| 11 号 | 登陆信息 |
| 代号 | 1-6 位 |
| 信息 | 7-200 位，包括用户名，状态，以标识符&&&对信息进行区分 |

### 5.2.7 文件包 19 号

| 名称 | 信息 |
| --- | --- |
| 19 号 | 登陆信息 |
| 代号 | 1-6 位 |
| 信息 | 7-200 位，包括用户名、文件信息，以标识符&&&对信息进行区分 |

### 5.2.8 代码详解

**1、【功能】：发送前准备**

【方式】：在客户端登陆服务器成功后，需要先获得服务器的分发的消息秘钥，使用 14 号数据包获取器新发放的 DES 秘钥，分发过程使用 RSA 的公钥加密。

**2、【功能】：发送消息**

【方式】：客户端的任务主要集中在更新客户端 UI 和发送消息。更新客户端 UI 采用 JTextarea 的 append 方法，将 UI 的组件传入发送消息处理状态机中使用这个函数即可对 UI 进行更新。

采用 socket 的 readline 函数对 socket 内容进行读取。读取后，对于读取到的字符串使用自己写的类 ServerDataGramAnalyzer 的 gettype 方法对其信息包的类型判断。

如果收到 10 号数据包，则说明收到了其他客户端发送来的信息。需要先对信息进行解密，使用小组设计的 DES 解密接口，解密完成后，需要先判断消息的签名是否正确。使用 String 的 split 方法提取最后的数字签名。认证过程由 RSA 部分说明，如果认证成功，可以防止消息抵赖。再分别调用 ServerDataGramAnalyzer 的 getname、getmessage 等方法，提取发送方信息和发送的消息。提取完成后，再用每个组件的 append 分别更新到

UI 上即可。

**3、【功能】：语音播放**

【方式】：收到的消息提取到 message 后，对客户端的 UI 界面 append 更新完毕后，调用语音播放函数对其进行语音播放。语音播放功能需要配置 windows 的执行环境。

将 jacob.jar 工程包复制到工程目录，在 eclipse 右键该文件→Build Path→Add to 添加到工作环境 jacob-1.17-M2-x64.dll 添加到 JDK 的 bin 目录和 Windows 的 system32 目录。其调用代码如下：

```
调节音量 0-100
sap.setProperty("Volume", new Variant(100));
语音朗读速度 -10 到 +10
sap.setProperty("Rate", new Variant(-2));
获取执行对象
Dispatch sapo = sap.getObject();
执行朗读
while (msg != null) {
Dispatch.call(sapo, "Speak", new Variant(msg));
content = br.readLine();
}
关闭执行对象
sapo.safeRelease();
```

其驱动语音播放主要通过语句 ActiveXComponent sap = new ActiveXComponent("Sapi.SpVoice");获得对语音播放功能的支持。

**4、【功能】：更新好友列表**

【方式】：更新好友列表使用数据包 11 号。主要对获得的信息使用 ServerDataGramAnalyzer 的 getname 方法获得其发送消息方的姓名。收到后，使用客户端的 UI 界面 append 更新。

**5、【功能】：发送文件**

【方式】：在 UI 框中选定文件后，使用文件的 getName 获得文件名。需要先将文件名和要接收方用户名称发送至服务器，告诉服务器基本的文件信息。将信息封装成 19 号数据包格式，仍然以&&&作为区分分隔符。

发送文件使用 socket 的 println 方法，在 while 循环中通过 readline 函数读取文件中的信息，发送之前先 DES 加密。发送完毕后，想对应客户端发送字符串”comple“表示文件发送完毕。

```
while((line=bufr.readLine())!=null)
{

    line=client_DES_encry(line,0,line.length(),thread.get_server_key());
    System.out.println(line);
    out.println(line);//发送文件内容
}
```

图 五-11 发送文件

**6、【功能】：接收文件**

【方式】：接收文件同样先接受到文件名，再通过 new BufferedWriter(new FileWriter("F:\\Desktop\\测试\\"+filename));其中 F:\\Desktop\\测试\\"+filename)是文件存放的路径，通过 Filewriter 创造这个对象。BufferWriter 对象获得后，通过 while 接收服务器发来的文件信息，先 DES 解密，在通过其 write 方法写入文件。

当收到服务器发来的 complete 消息后，代表文件发送完毕，结束接收文件即可。

接收完毕文件后，会弹出对话框提示接收到文件。

JOptionPane pane = new JOptionPane("接收到文件： "+filename);

JDialog dialog = pane.createDialog("恭喜");

dialog.show();

JOptionPane 用来创建对话框的内容，createDialog 创建对话框，show 函数将对话框进行显示。

**7、【功能】：用户上线提醒**

【方式】：接收到 18 号数据包，利用 ServerDataGramAnalyzer 的 getname 方法获得其发送消息方的姓名。更新客户端 UI 采用 JTextarea 的 append 方法，将 UI 的组件传入发送消息处理状态机中使用这个函数即可对 UI 进行更新。

**8、【功能】：用户退出，重新显示好友列表**

【方式】：接收到 16 号数据包，利用 ServerDataGramAnalyzer 的 getname 方法获得其发送消息方的姓名。更新客户端 UI 采用 JTextarea 的 append 方法，将 UI 的组件传入发送消息处理状态机中使用这个函数即可对 UI 进行更新。

将客户端的好友信息哈希表内容进行更新。使用哈希 map 的 remove 方法，找到用户名称，OnlineUsers.onlineUsers.remove(ServerDataGramAnalyzer.getName(stocdatagram));将用户名字移除。

```
for (String name : OnlineUsers.onlineUsers){
    System.out.println("******" + name);
    textArea2.append(name + "\n");
}
```

图 五-12 更新用户

对更新后的 UI，需要先清空显示，利用 append（null）即可清空。

通过 for 循环，利用组件的 append 方法重新用户列表。

**9、【功能】：登陆失败**

【方式】：接收到 12 号数据包。接收到后，显示对话框即可。

JOptionPane.showMessageDialog(null,"登录失败", "信息错误",JOptionPane.ERROR_MESSAGE)。

**10、【功能】：重复登陆**

【方式】：接收到 13 号数据包。接收到，显示对话框即可。

JOptionPane.showMessageDialog(null,"已经在线上", "信息错误",JOptionPane.ERROR_MESSAGE)

## 5.2.9 结果截图

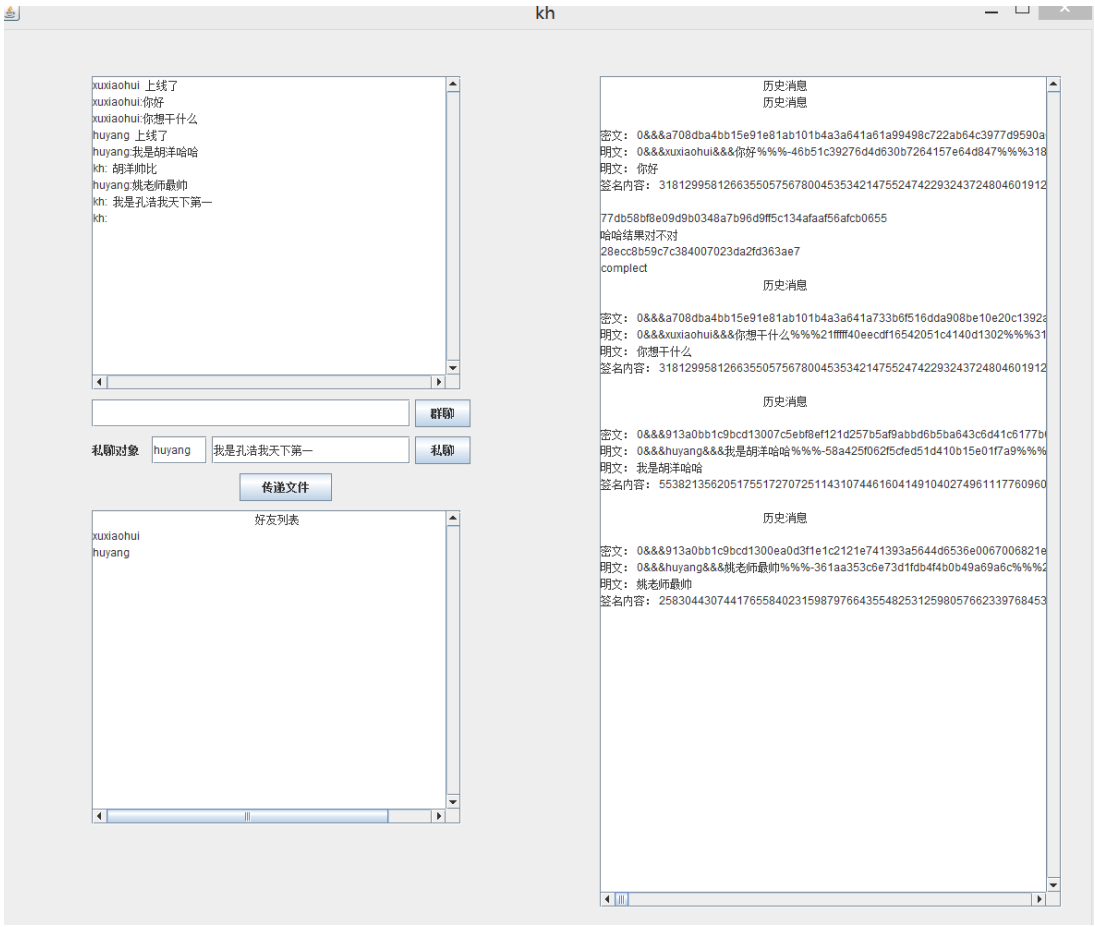


图 五-13 客户端结果展示

```
测试.txt 接收到文件名
开始接受文件
ec14d2d867150961d41f84902f0c07689fb31fb05aff1410
ec14d2d867150961d41f84902f0c07689fb31fb05aff1410 567LNY2C
f21928670ccfe6663ec121ed4396b7ff
f21928670ccfe6663ec121ed4396b7ff 567LNY2C
结束接受文件
开始发送文件
```

图 五-14 接收文件 log

```
Thu Jun 14 10:20:20 CST 2018 WARN: Establishing SSL connection without server's identity verification is not recommended. Accord
0&&&e7524f0ff6a5452e8ac47f9a554148e760601dc51d9641af84e5cc1b2fde6ca50046fa48a294f9a8e8b07de4bcc8a14921d40b02ef1f5a501a75cc8b0f6f
e7524f0ff6a5452e8ac47f9a554148e760601dc51d9641af84e5cc1b2fde6ca50046fa48a294f9a8e8b07de4bcc8a14921d40b02ef1f5a501a75cc8b0f6fada0
0&&&xuxiaohui&&&你好%%%-46b51c39276d4d630b7264157e64d847%%%3181299581266355057567800453534214755247422932437248046019128224926747
```

图 五-15 发送信息 log

```
0010007fef29de7192a96126d19c3da258bf5d97ff7614c891949b74c899efaa7a5e18
client socket open!!!!!!!!!!!!!!!!!!!!!
1&&&kh&&&AAAAAAAA&&&online ^^^^^^^^^^^^^^^^^^^^^^
3
```

图 五-16 上线包 log

```
11
1&&&null
6&&&kh
```

图 五-17 下线包

```
1&&&xuxiaohui&&&AAAAAAAA&&&online ^^^^^
3&&&null
Exception in thread "Thread-2" java.lan
```

图 五-18 登陆重复 log

```
33   !!!!!!!!!!!!!!!!!!!!
身份信息 100001&5&3260506453768671988809467257212227750921069808188741142087183111560049690825960635182041482335380733409973921439440

t fdjkdjfkdjfkadjfkdjkfjd
5 3260506453768671988809467257212227750921069808188741142087183111560049690825960635182041482335380733409973921439440
```

图 五-19 身份验证 log

```
2   !!!!!!!!!!!!!!!!!!!!
0000101252332576&4437053125&1350125107&5904900000
BUCZ  ************
```

图 五-20 不存在用户

```
2    !!!!!!!!!!!!!!!!!!!!
000010681472&389017&614125
XIU  ************
```

图 五-21 修改成功

# 第六章　RSA 加密解密

加密解密不需要加解密包头前 6 位。

加解密信息输出在控制台和客户端的 UI 历史显示区域。

## 6.1.1 RSA

RSA 在注册时使用，使用 AS 的公钥加密，AS 使用私钥解密，这是 1 号控制包。

C 公钥加密，C 使用私钥解密，这是 2 号控制包。

## 6.1.2 代码详解——RSA 加密

```
ArrayList<BigDecimal> arrayList = new ArrayList<BigDecimal>();
for(int i=0;i<num;i++) {
int mid_num=(int)str_orignal.charAt(i);
BigDecimal mid1=new BigDecimal(mid_num);
mid1=mid1.pow(e);
BigDecimal[] results = mid1.divideAndRemainder(BigDecimal.valueOf(n));
arrayList.add(results[1]);
```

【功能】：将明文使用 RSA 加密。

【方式】：因为可能产生的数据运算溢出，所以使用 java 内置类 BigDecimal，可以保证无限运算，防止出现错误。

这里要注意的是，BigDecimal 是一个类，不是一种类型。

1、字符转数字

因为加密是转换成数字，对于每一个字符，它都有自己对应的 ASCII 码，利用字符与码的对应关系，转换成相应的数字。Java 可以直接(int)char 强制转换为对应的数字。

2、如何保证数字转回字符

要想保证数字正确转回字符，需要将每个字符转换后的数字单独保存，这样就可以直接找到转换前对应的字符是什么。单独保存使用动态保存 Arraylist，类型 BigDecimal。

3、加密步骤

加密需要使用对方的公钥加密，调用函数 pow( e )，可以实现明文的 e 次方，然后调用 divideAndRemainder(BigDecimal.valueOf(n))，可以进行取余运算，这就是加密后的结果，使用函数 add 保存在 ArrayList 里。使用 for 循环不断重复此次步骤，直到字符串为空。返回 ArrayList 数组，里面每一项即为加密内容。

### 6.1.3 代码详解——RSA 解密

BigDecimal mid1=number; mid1=mid1.pow(d);

BigDecimal[] results = mid1.divideAndRemainder(BigDecimal.valueOf(n)); int result=results[1].intValue(); char c=(char)result; str=str+c;

【功能】：将明文使用 RSA 解密。

【方式】：因为可能产生的数据运算溢出，所以使用 java 内置类 BigDecimal，可以保证无限运算，防止出现错误。

这里要注意的是，BigDecimal 是一个类，不是一种类型。

1、数字转字符

因为解密密是转换成字符，对于每一个数字，它都有自己对应的 ASCII 码对应的字符，利用字符与码的对应关系，转换成相应的数字。加密时使用(int)char 直接强制转换为对应的数字。所以，只要解密步骤正确，直接(char)int，即可转为原来字符。

2、如何保证数字转回字符

要想保证数字正确转回字符，需要将每个字符转换后的数字单独保存，这样就可以直接找到转换前对应的字符是什么。单独保存使用动态保存 Arraylist，类型 BigDecimal。

3、解密步骤

解密需要使用自己的私钥解密，调用函数 pow( d )，可以实现密文的 d 次方，然后调用 divideAndRemainder(BigDecimal.valueOf(n))，可以进行取余运算，这就是解密后的结果，在转换为字符，使用 String 的+，将每次产生的 char 字符加到 Stirng 上，直到 ArrayList 为空，退出 for 循环。产生的 String 即为原来明文。

证书交换采用提前物理分发的方式使 Client 与 AS 分别获得对方的秘钥。根据 RSA 消息认证方案，将用户口令首先采用 md5 进行 hash 变换生成 H（M），然后将 H（M）用 client 的私钥进行签名得到 Sig[H(M)],然后与 M 进行拼接，然后采用 PKb 即 AS 的公钥进行加密，生成 EK[M||Sig[H(M)]]，然后发送给 AS，AS 收到数据以后，首先利用自己的私钥 SKb 进行解密得到 M||Sig[H(M)，然后将 M 与 Sig[H(M)]分离，Sig[H(M)]利用 client 的公钥 PKa 进行签名认证，解密得到 H(M)，然后将它之前分离的 M 采用 md5 进行 Hash，得到的结果与 H(M)进行对比，如果一致，则说明用户口令传输是完整的，AS 接收到的 M 对的。

### 6.1.4 代码详解——RSA 签名

```
ArrayList<BigDecimal> arrayList = new ArrayList<BigDecimal>();
(int i=0;i<num;i++) {
int mid_num=(int)str_orignal.charAt(i);
BigDecimal mid1=new BigDecimal(mid_num);
mid1=mid1.pow(d);
BigDecimal[] results = mid1.divideAndRemainder(BigDecimal.valueOf(n));
arrayList.add(results[1]);
```

【功能】：将明文使用 RSA 签名。

【方式】：只需要将加密的 e 换位 d 即可。

因为可能产生的数据运算溢出，所以使用 java 内置类 BigDecimal，可以保证无限运算，防止出现错误。

这里要注意的是，BigDecimal 是一个类，不是一种类型。

1、字符转数字

因为加密是转换成数字，对于每一个字符，它都有自己对应的 ASCII 码，利用字符与码的对应关系，转换成相应的数字。Java 可以直接(int)char 强制转换为对应的数字。

2、如何保证数字转回字符

要想保证数字正确转回字符，需要将每个字符转换后的数字单独保存，这样就可以直接找到转换前对应的字符是什么。单独保存使用动态保存 Arraylist，类型为 BigDecimal。

3、签名步骤

签名需要使用自己的私钥签名，首先算出 hash 结果，对其调用函数 pow( d )，可以实现明文的 d 次方，然后调用 divideAndRemainder(BigDecimal.valueOf(n))，可以进行取余

运算，这就是签名后的结果，使用函数 add 保存在 ArrayList 里。使用 for 循环不断重复此次步骤，直到字符串为空。返回 ArrayList 数组，里面每一项即为加密内容。

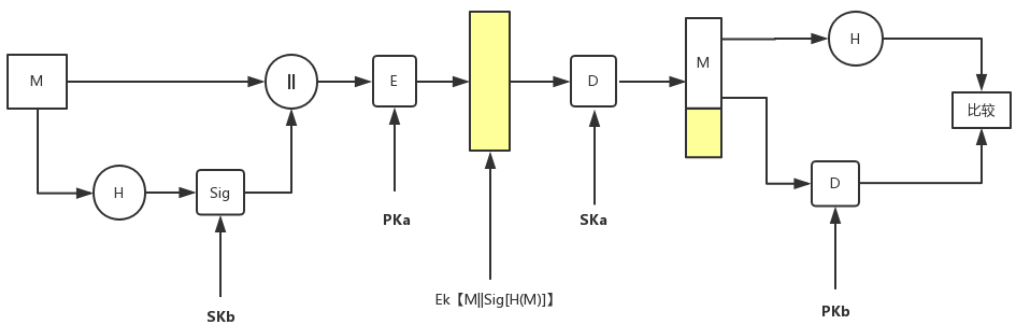


图 六-1 认证

## 6.1.5 代码详解——RSA 认证

BigDecimal mid1=number; mid1=mid1.pow(e);

BigDecimal[] results = mid1.divideAndRemainder(BigDecimal.valueOf(n)); int result=results[1].intValue(); char c=(char)result; str=str+c;

【功能】：将明文使用 RSA 认证。

【方式】：因为可能产生的数据运算溢出，所以使用 java 内置类 BigDecimal，可以保证无限运算，防止出现错误。

这里要注意的是，BigDecimal 是一个类，不是一种类型。

1、数字转字符

因为加密是转换成数字，对于每一个字符，它都有自己对应的 ASCII 码，利用字符与码的对应关系，转换成相应的数字。加密时使用(int)char 直接强制转换为对应的数字。所以，只要解密步骤正确，直接(char)int，即可转为原来字符。

2、如何保证数字转回字符

要想保证数字正确转回字符，需要将每个字符转换后的数字单独保存，这样就可以直接找到转换前对应的字符是什么。单独保存使用动态保存 Arraylist，类型为 BigDecimal。

3、认证步骤

认证需要使用对方公钥认证，将传过来的明文 hash 结果，对其调用函数 pow( e )，可以实现 e 次方，然后调用 divideAndRemainder(BigDecimal.valueOf(n))，可以进行取余运算，这就是认证后的结果，在转换为字符，使用 String 的+，将每次产生的 char 字符加到 Stirng 上，知道 ArrayList 为空，退出 for 循环。产生的 String 即为原来 hash 明文结果。判断认证出来的结果与传过来的明文的 hash 是否相同。

判断此函数设置为 boolean，判断最后结果与被验证方发送回来的明文哈希结果进行判断比较，如果是，则返回 true，反之，false。

### 6.1.6 系统中 RSA 调用

因字数太多，不再赘述，这里仅说明主要用到的函数详细的设计内容。

具体的 RSA 在系统中的应用在服务器和客户端模块说明接口的使用。

# 第七章　时序图

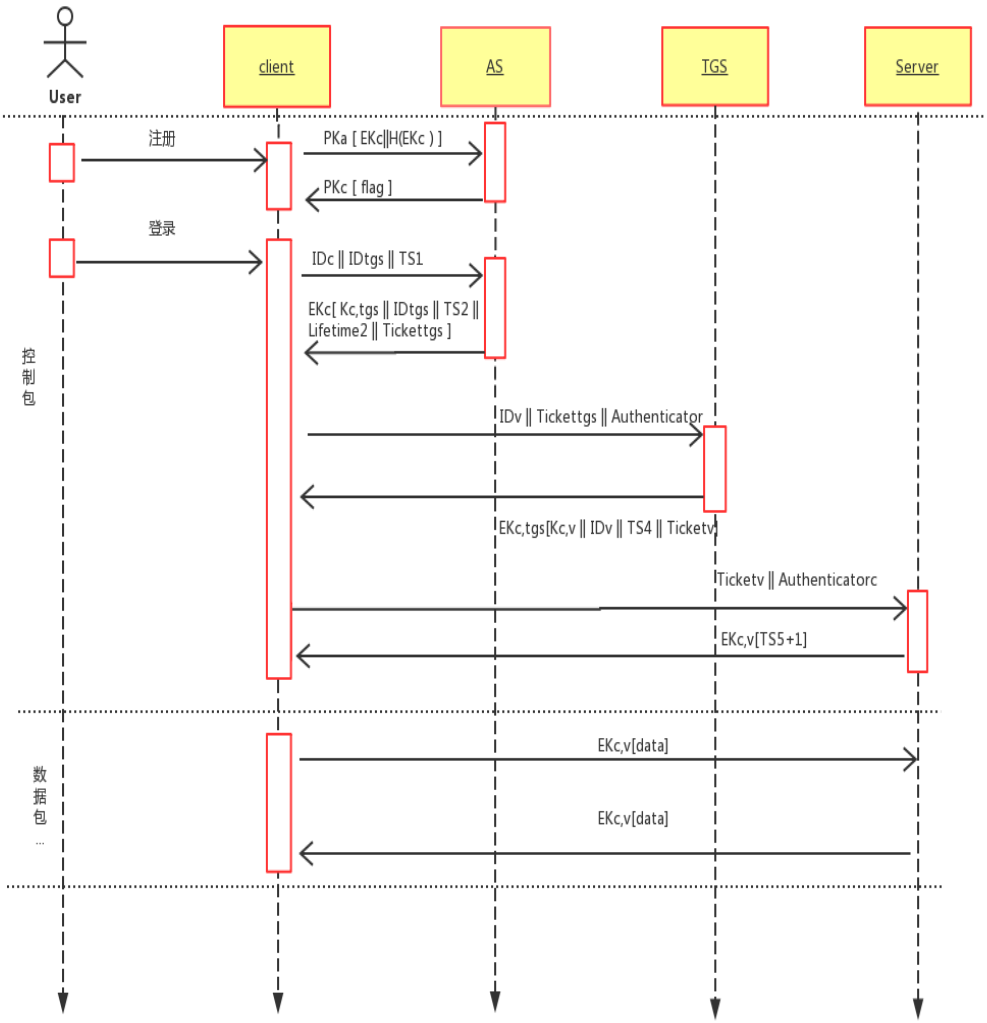


图 七-1 总时序图

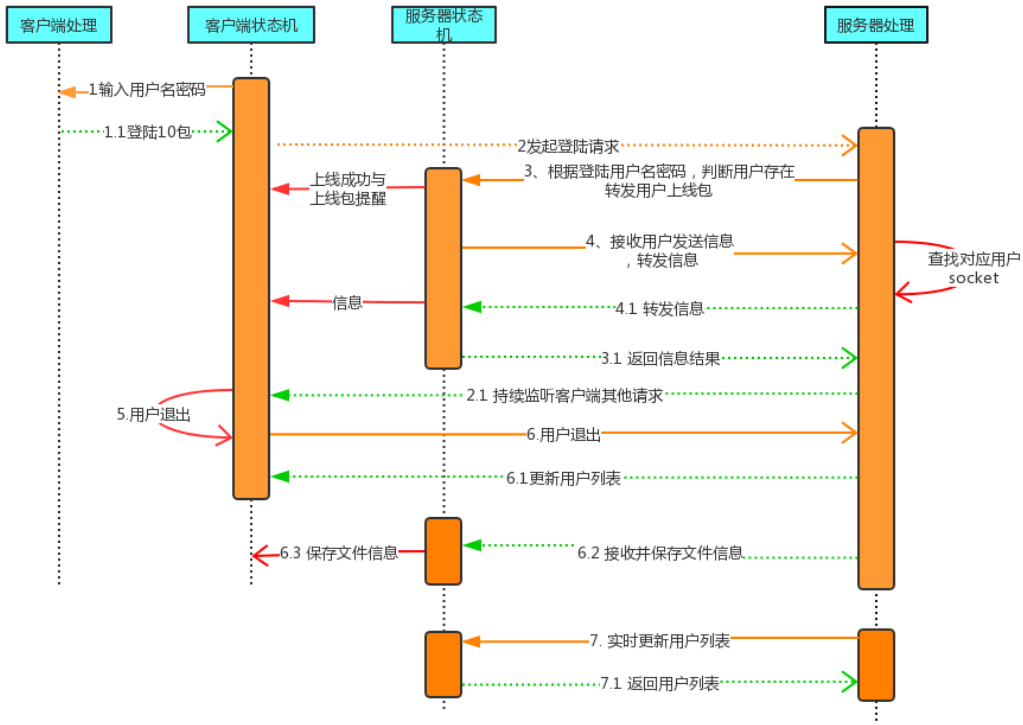
客户端处理
客户端状态机
服务器状态机
服务器处理
1.输入用户名密码
1.1登陆10包
2发起登陆请求
上线成功与
上线包提醒
3、根据登陆用户名密码，判断用户存在
转发用户上线包
4、接收用户发送信息
，转发信息
查找对应用户
socket
信息
4.1 转发信息
3.1 返回信息结果
2.1 持续监听客户端其他请求
5.用户退出
6.用户退出
6.1更新用户列表
6.3 保存文件信息
6.2 接收并保存文件信息
7. 实时更新用户列表
7.1 返回用户列表

图 七-2 聊天室时序图

# 第八章　开发进度

## 8.1.1 概述

**原预定代码进度：**规定每天每人在比较难的模块上 150 行，简单时 200 以上，得出如下工作安排。其中，测试阶段在代码开始一定程度后就需要不断测试与集成系统 ，防止后续重大错误。代码升级阶段花费 6 天，增加小组设计系统的功能。

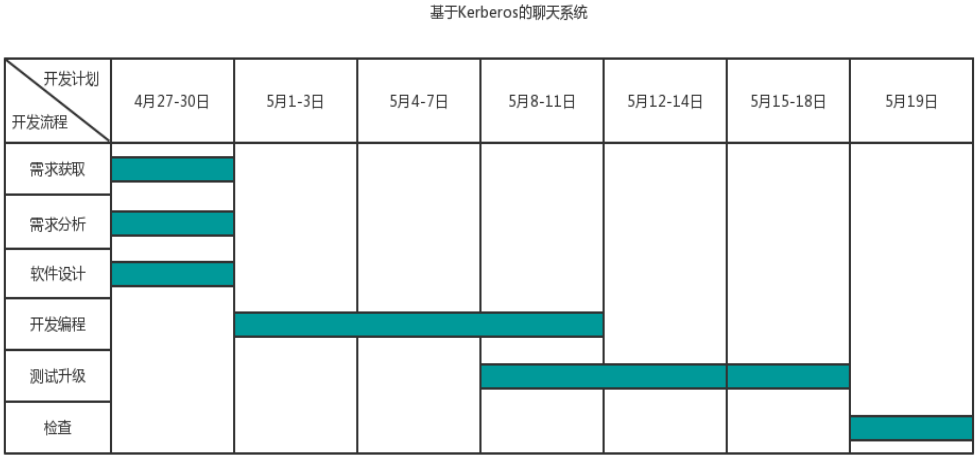


图 八-1 设计甘特图

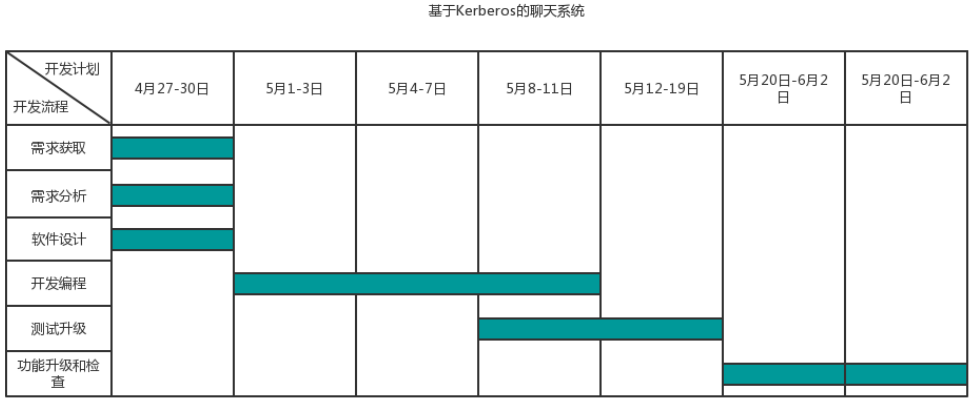


图 八-2 实际开发甘特图

**实际代码进度：**在我写完 Kerberos 流程后，小组确实是按照设计的流程进行开发。5 月 19 日我把群聊功能和拓展的私聊功能写完后，开始第一次上机。老师对小组的开发做出要求和指导，尤其对小组状态机的维护提出指导，我对小组成员写好的控制包流程进

行统一改进。

5 月 20-26 日，增加文件传送，消息包的处理、身份的认证等功能，老师检查后对小组文件加密和身份认证提出指导，我对代码继续进行改进，后期的小组的大部分改进部分主要由我完成。小组成员负责对 github 代码测试和自己完成的部分作出适当改进。其中因为根据一些小组现有代码，可以实现开发语音识别功能。小组孔浩根据写好的代码开发 android 端的实现。6 月 2 日，增加消息防抵赖，任何消息后面增加签名信息，完成了验收。

总体来说，对于基本功能的实现基本符合设计的工期，但后期的功能升级和完善重新花费时间到 6 月 2 日，我想到了实际工程至少是设计的 1.5 倍。

# 第九章　系统部署图

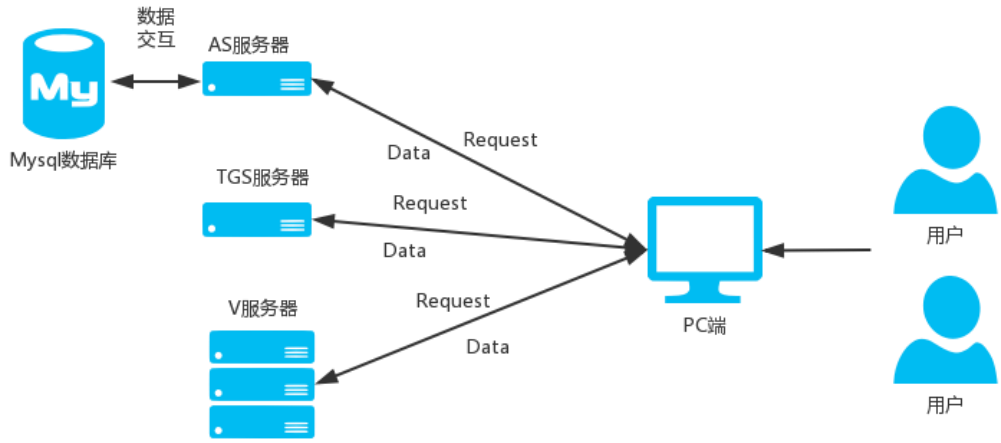


图 九-1 系统部署图

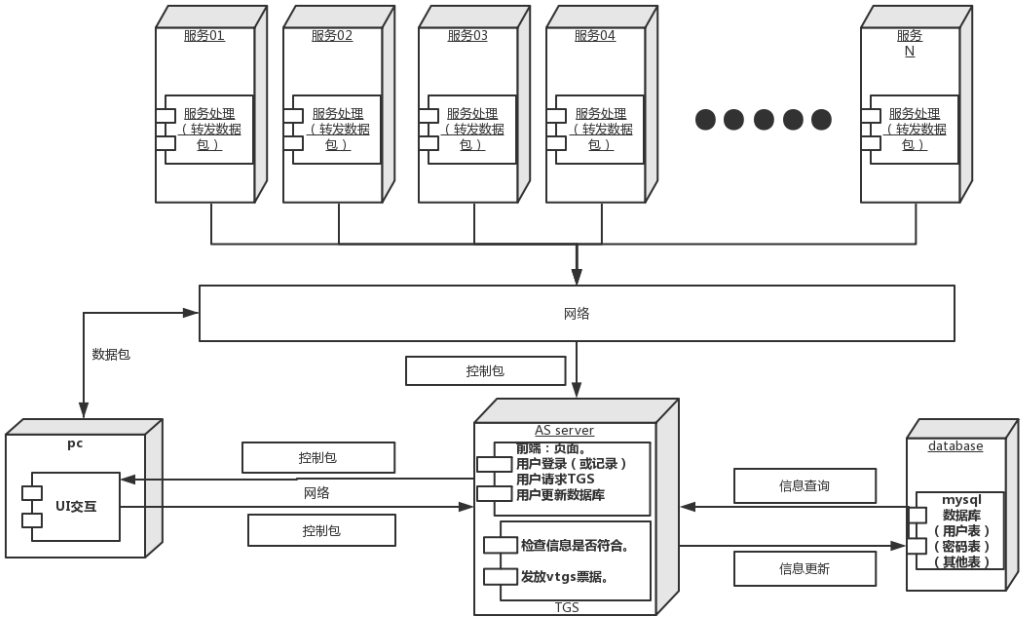


图 九-2 部署图

整体系统部署图按照文档设计来进行，其中在检查环节，组内胡洋代表 AS，张毛毛代表 TGS，许晓晖代表服务器，同时 4 台机子都当客户端。

# 第十章　总结感想

## 10.1.1 小组任务和分工总结

本次设计作为组长，完成了最初文档的设计主体内容，对 AS、TGS、C、V 的各种控制包和数据包做出规定，对其流程做出了文字描述，对其流程画出了流程图。对其部署实现画出部署图，对每一模块的 UI 提出要求。

完成了文档的设计后，为增快开发进度，我先对 kerberos 流程做出 socket 的空转代码实现，这样小组开发接口可以大部分直接定下，数据包的解析和信息比对只是模块里的内容，这样对小组进度有极大帮助。

根据预定分工，客户端对 AS、TGS、V 的开发，客户端对 AS 注册和身份认证流程，AS 身份认证和注册、应用服务器 V 的设计由我开发，服务器 V 的 Kerberos 的功能数据包解析由孔浩完成。AS 对 3 号数据包 4 号包的处理由胡洋完成。TGS 处理由张毛毛完成。

在 5 月 16 日第一次小组基本流程实现后分布式环境测试，因为最初空转流程的设计统一，控制包和数据包接口统一，测试一次通过。并且应用聊天室主要由我完成，在服务器和客户端之间分布式环境下也没有错误，主要还是代码设计逻辑和思路没有错误。

5 月 20 日，第一次上机之后，因为大部分功能已经实现，为增加功能，将电脑端实现的代码转向移动端的开发任务分给孔浩，因为我们之前有开发移动端的经验和语音识别代码接口，可以直接实现和使用。其他两位成员对自己的代码进行优化，我负责增加新的功能，后期老师提出的指导，我负责将整体代码改进，并且相继丰富文件传送、身份认证 UI 显示、消息防止抵赖、语音播放等功能。

6 月 2 日小组率先检查通过，完成了本次课程设计。

## 10.1.2 个人实现总结

本次开发，我明白了小组合作的重要性。组长的设计和进度规划对小组整体开发起到核心作用。小组组长也要相应多做一些工作，并且尽早提前将负责的工作做完，这样才能对小组其他成员起到激励和督促的作用。

Kerberos 的开发，让我熟悉掌握了 DES 和 RSA 在实际开发中的应用。不过，我认为，本次开发最主要的是熟悉了分布式系统开发文档设计、开发步骤等关键内容。在一个系统的开发过程中，最初的文档设计十分重要，代码的着手不能过早，否则反而对进度的顺利执行有负面影响。分布式系统的开发小组之间根据代码对设计模块进行对外接口对内输入接口的统一十分重要。

非常感谢这次课设，让积累 3 年的知识有了一个展示的平台，让我对代码的设计和设计模式的应用在较为多的代码量下有了实际的应用经验。