长沙理工大学

CHANGSHA UNIVERSITY OF SCIENCE & TECHNOLOGY

毕业设计（论文）

题目： 网络保密通信系统的设计与实现

学生姓名： 肖娜

学　　号： 201258080202

班　　级： 网络 1202 班

专　　业： 网络工程

指导教师： 熊兵

2016 年 6 月

网络保密通信系统的设计与实现

学生姓名：肖娜

学　　号：201258080202

班　　级：085811202

所在院(系)：计算机与通信工程学院

指导教师：熊兵

完成日期：2016 年 6 月

网络保密通信系统的设计与实现

摘要

当今时代，各种通信软件层出不穷。作为用户的我们，并不知晓其中的奥妙。试想一下：如果某款软件出现了漏洞，那我们的个人信息岂不岌岌可危了？然而，我们却能放心地使用这些软件，是因为软件制造商为我们作出了相应的承诺。为此，在虚拟的网络环境中，如何开发出一款能够确保用户通信安全的软件，是各位软件制造商所共同面临的问题。而结合各种密码算法来实现保密通信，是解决这一问题的关键所在。

本课题基于 Visual Studio 2013 平台，采用数据加密技术，开发一个网络保密通信系统。系统采用 C/S 模式，服务器和客户端之间的通信采用密码算法进行加密，从而实现保密通信。系统的核心算法包括：RSA、DES 以及 MD5。其中，RSA 用于密钥协商，DES 用于会话加密，MD5 用于对数据库中的数据进行保护。此外，本系统也具备一个通信系统应有的功能，比如：登录、聊天等功能。

关键词： Visual Studio 2013；RSA；DES；MD5

Design and implementation of network secure communication system

ABSTRACT

In this day and age, a variety of communication softwares are cropping up. As a user, we don't know the secret of them. Just think about it: if there is a software vulnerability, our personal information would be in danger! However, we can use these software safely, because the software manufacturers has made the solemn promise. For that, it's a common problem for the software manufacturers to develop a software to ensure the users' communication security in the virtual network environment.Using cryptographic algorithm in secure communication system is the key to solve this problem.

This project will develop a secure communication system based on Visual Studio 2013 platform with data encryption technology.This system uses C/S mode.the communication between the server and the client is encrypted,so that it can realize secure communication.The core algorithms of this system are DES、RSA and MD5.Among them,RSA is used for key agreement,DES is used for session encryption,and MD5 is used to protect the data in the database.In addition,the system also have the basic functions of a communication system,such as:login,chat and so on.

Keywords: Visual Studio 2013；RSA；DES；MD5

目录

1.绪 论

网络为生活提供了巨大的便利，购物、转账等等都可以在网上进行，我们渐渐习惯了用网络来解决各个方面的问题。一方面，我们在享受着网络所带来的便捷服务，另一方面，我们却在承担着网络所隐含的一些风险[1]。木马、病毒从来都没有远离我们的视线，打开电脑，总免不了一番安全扫描和病毒查杀。在危机四伏的网络环境中，如何保障用户的通信安全？这是本文将要重点讨论的内容。

1.1.背景

自 1969 年 ARPANET 诞生以来，计算机网络经历了诸多的变化和发展。到如今，它已经笼罩了各个行业领域，无论是学习、工作还是购物等，我们似乎都离不开网络。由于网络能够提供高速、快捷的服务，我们对它已经产生了深深的依赖。现在，几乎没有什么可以取代网络在我们的社会与生活中发挥同样的作用。尽管我们享受着网络所提供的优质服务，但是同时我们也承担着一些网络安全方面的风险。在涉及资金安全的问题上，我们不得不对网络安全一再引起高度重视。尽管当前各类通信软件在用户安全方面做得很完善，但是历史上也曾出现过一些信息泄露的大事件，这不得不让我们警惕。一个典型的例子就是：2011 年 CSDN 就因数据库被黑客所攻击而导致 600 万用户的账号信息泄露。这样的事件并不少见，而且给社会和人们带来了很多负面的影响和不利的因素。因此，如何全方位地保障用户的通信安全，是每一款保密通信系统所需要面对的问题。为了解决这一问题，首先我们需要了解对网络安全造成威胁的攻击有哪些。

网络安全面临的威胁多种多样，除了计算机病毒、蠕虫、木马[2]等耳熟能详的攻击外，还有一些攻击也在悄悄地接近我们。比如说：中间人攻击、流量分析、恶意程序等。对于大多数攻击而言，它们往往试图获取一些有用的信息。怎样做才能阻止它们获取我们的信息呢？最好的方式就是采用数据加密技术对信息进行高强度的加密。这样，攻击者们就只能获取一堆毫无意义的密文，而无法达到攻击的目的。也正因为这个原因，数据加密技术[19]是保密通信系统核心的技术之一。

1.2.发展现状

随着时代的进步，保密通信系统的发展日趋完善。各种类型的保密通信系统层出不穷，其中最权威的要数支付宝了。据统计，支付宝的实名用户已超过 3 亿，每一天，都有超过 4500 万笔的手机支付量。为什么大家都如此信赖支付宝？这离不开支付宝强有力的安全保障。通过巧妙地结合 RSA 等一系列加密算法，为用户提供值得信赖的安全环境。除此之外，支付宝还在研究生物识别技术。声纹、掌纹、人脸等个人特征，在将来都有可能成为身份鉴别的密码。

尽管当前资金安全已经在算法上得到了很好的保障，但是在人为因素上却依然存在一些漏洞。由于网银和第三方支付往往采用短信验证码以及数字密码的方式，并没有很好地将“支付”和“鉴权”相分离，如果手机丢失或者中毒，很可能导致支付风险。针对这一现象，腾讯提出了相应的解决方案：将“支付”和“鉴权”分离，采用双因子认证的方式来实现支付功能。据报道，腾讯在不久前曾推出一款名为“Qkey”的智能手环，通过和手机配合使用来实现支付功能。即使手机丢失，只要手环还在，用户就无需担心资金安全。由此可见，一个好的保密通信系统，不仅需要安全的密码算法，还需要从程序设计思路上考虑到各种外界因素的干扰。

除了在日常生活中发挥作用外，保密通信系统在军事上也发挥着不可或缺的作用。由于军事关系国家安全，因此每一项军用业务都需要采用最先进的保密技术和最前沿的科技。在几十年以前，美国就已经建立了完善的军用安全通信网，同时拥有电报、数据、话音、图像等保密机系列。虽然军事上的一些最新技术往往作为机密而不被公开，但是过了这么多年，军用保密通信系统也一定发生了翻天覆地的变化。相比于生活中各类通信软件所用的保密技术，军用保密技术一定先进得多。由此可见：保密通信系统依然有很大的进步空间。

网络保密通信系统的核心在于密码算法。通过密码算法[20]将明文转变为不可识别的密文，使被动攻击无法实施，从而保障用户的通信安全。密码算法分为好几类，其中对称密码算法适用于对大规模的数据进行加密。在对称密码算法中，DES 是一种相当经典

的算法。自 1977 年被确立为标准以来，它一直在保密通信系统中发挥着不可小觑的作用。由于它很难被破解，而且加密速度快，因此被运用于各种类型的保密通信系统中。然而，随着科技的进步，密码算法普遍面临着被破解的风险。尽管 DES 算法未被破解，但是 56 位的密钥长度实在太短[12]，穷举攻击依然可以成为攻击的手段。为了消除密钥长度所带来的安全隐患，我们有必要推出一种密钥长度更长，安全强度更高的算法。于是高级加密标准 AES 算法顺势诞生了。AES 的密钥长度可以为 128 比特、192 比特或者 256 比特。相比 DES 的 56 位密钥，其安全性能自然大幅度提高。所以，AES 算法近年来在保密通信系统中大受欢迎，大有取代 DES 算法的趋势。

除了对称密码算法之外，在保密通信系统中，非对称密码算法的用处也很大。非对称密码算法加密速度比较慢，只适合对少量的关键数据进行加密，它能实现的功能主要有：身份鉴别、数字签名、密钥协商等。在保密通信系统中最常用到的非对称密码算法有：ECC、RSA、DSA 这三种。其中，大家最熟悉的就是 RSA 算法了，它基于大数分解的难题[14]，几乎能够抵抗所有的密码攻击，只要密钥的位数足够长，要想破解它比登天还难。按理来说，RSA 是一种理想的算法。然而，在计算能力如此强大的今天，RSA 算法也只能通过增加其密钥长度来保障通信安全不受威胁。从最初的 256 位到如今的 1024 位，RSA 的安全性虽然得到了保障，但是其运行效率却明显下降了不少，给机器造成的损耗也相当大。与 RSA 相比，ECC 则无需担心因密钥长度过长所带来的一系列问题。据了解，密钥长度为 234 位的 ECC 算法其安全性能[15]就远高于密钥长度为 2048 位的 RSA 算法。由此可见：RSA 算法终会渐渐淡出保密通信系统的视线，而 ECC 算法则将取代它继续为保密通信[9]系统保驾护航。

对于保密通信系统而言，哈希算法同样是一味不可缺少的调料。通过哈希算法对重要的数据进行散列，然后保存在数据库中，这样就能大大增强数据库的安全度。不仅如此，哈希算法还能对重要的文件进行签名，防止其被篡改。在这一领域，比较著名的算法有 Ron Rivest 开发的 MD5 以及 NIST 提出的 SHA 系列算法。然而，MD5 算法和 SHA-1 算法由于碰撞缺陷，皆已被破译。这导致许多大公司不得不淘汰这两种算法，而提前推出了新的替换算法。由此可见，在单向散列函数这一领域有很大的发展空间值得人们去探索和发现。

1.3.本文的主要内容

该论文共分为五个部分，从开发背景到设计实现，逐步描述软件实现过程的各个阶段。

第一部分首先从当前的网络环境出发，讲述了网络保密通信系统存在的必要性。然后通过几款著名的通信软件，进一步说明保密通信技术的重要性。最后，详细介绍当前常用的密码算法。

第二部分首先从开发环境入手，介绍系统使用的编程软件；然后对系统所使用的技术进行了一番详细的解读。

第三部分首先从系统的功能、运行情况等方面进行了需求分析；然后，按照功能将系统划分为若干模块，并组织起来，呈现出整体的架构；最后，对各个模块进行详细的设计与分析。

第四部分是展示阶段，通过引用项目的部分关键代码和提供程序的运行截图，展示项目的最终成果。

第五部分是对本系统的总结以及对开发过程中的学习心得的描述。

2. 系统理论技术基础

本文接下来对系统的基本状况以及所运用的关键技术进行一番详细的描述，从而回答“为什么该系统具有保密通信功能？”的问题。

2.1.系统概述

本系统是一款基于 Visual Studio 2013 平台，通过密码算法对网络流进行加密的通信系统。系统能够实现登录、聊天、注册、文件传输等功能。系统使用的核心算法有：RSA 算法、DES 算法以及 MD5 算法。其中，RSA 算法用于实现密钥协商的功能，DES 算法用于实现文本及文件加密的功能，MD5 算法用于对用户的账号及密码进行 Hash，将定长的散列值存储在数据库中，从而保障数据库中的数据安全。

2.2.开发环境

本系统是基于 Visual Studio 2013 开发环境，采用 MySQL-5.7.10 数据库开发的一款软件。Visual Studio 2013 为开发人员创建了便捷的开发环境，从用户体验的角度对以前的版本做出了大量的改进，方便程序员对代码进行调试和跟踪错误，从而帮助程序员快速开发新的应用程序。同时，Visual Studio 2013 还提供了许多新的功能，比如：轻量代码注释、支持 Windows 8.1 APP 开发、敏捷项目管理等功能。此外，在编程方面，Visual Studio 2013 具有丰富的提示功能，能够在你编码过程中出现语法错误时或者存在内存泄露问题时进行提示。通过这一系列的机制，Visual Studio 2013 能够帮助程序员最大程度地减轻负担。

Visual Studio 2013 的最大优势在于能够提供多种语言编写的环境以及轻松调用数据库的功能。由于本项目的算法部分采用 C 语言[6]编写，其它部分由 C#语言编写而成，同时涉及对数据库的访问。因此，Visual Studio 2013 无疑是最好的选择！

2.3.相关技术

2.3.1 DES

DES 算法基于扩散和混乱的理论[2]，来实现对数据的加密和解密。整个过程是这样的：首先，明文按照 64 位的固定长度进行分组；然后，将这个分组经过一个初始置换，再执行 16 轮运算；最后经过一个末置换，便得到了最终的密文，并且密文的长度也是 64 位。

表 1 初始置换表

58	**50**	**42**	**34**	**26**	**18**	**10**	**2**
60	**52**	**44**	**36**	**28**	**20**	**12**	**4**
62	**54**	**46**	**38**	**30**	**22**	**14**	**6**
64	**56**	**48**	**40**	**32**	**24**	**16**	**8**
57	**49**	**41**	**33**	**25**	**17**	**9**	**1**
59	**51**	**43**	**35**	**27**	**19**	**11**	**3**
61	**53**	**45**	**37**	**29**	**21**	**13**	**5**
63	**55**	**47**	**39**	**31**	**23**	**15**	**7**

初始置换是将 64 位数据按照表 1 的形式进行位置的转换，从而达到打乱顺序的目的。这样，经过一轮初始置换后，明文已经转换成了无法直观理解的乱码。

表 2 末置换表

40	**8**	**48**	**16**	**56**	**24**	**64**	**32**
39	**7**	**47**	**15**	**55**	**23**	**63**	**31**
38	**6**	**46**	**14**	**54**	**22**	**62**	**30**
37	**5**	**45**	**13**	**53**	**21**	**61**	**29**
36	**4**	**44**	**12**	**52**	**20**	**60**	**28**
35	**3**	**43**	**11**	**51**	**19**	**59**	**27**
34	**2**	**42**	**10**	**50**	**18**	**58**	**26**
33	**1**	**41**	**9**	**49**	**17**	**57**	**25**

既然存在初始置换，那么势必会有末置换。末置换采用相同的原理，对 16 轮运算之后生成的数据进行顺序的打乱，最后得到我们所需要的密文。

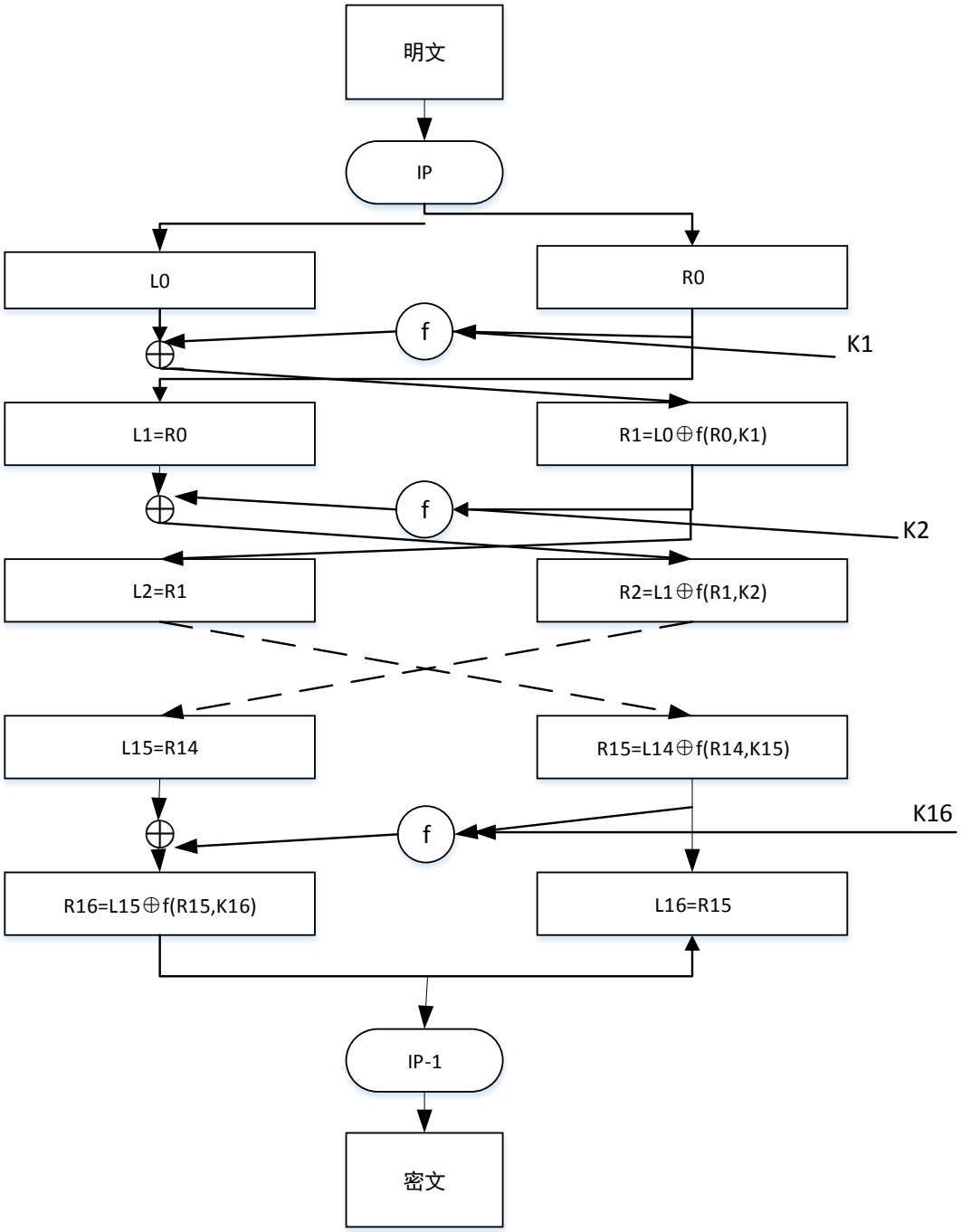

图 2.1 DES 全过程

对前后的置换过程有了一定的了解之后，我们难免不会产生这样的好奇：在这两个置换之间的中间过程到底是怎样的呢？现在，就让我们了解一下这最关键的中间过程。DES 算法的中间过程是 16 轮几乎相同的运算。如图 2.1 所示，前 15 轮运算完全相同，

第 16 轮运算略有不同，而唯一的区别在于：第 16 轮运算的左右两部分并未进行交换。虽然过程相同，但是细心的我们仍旧可以从图中看出，每一轮运算所用到的密钥并不相同。原来，生成密钥也要经历一个较为复杂的过程，其中包括循环移位以及压缩置换[13]等操作。

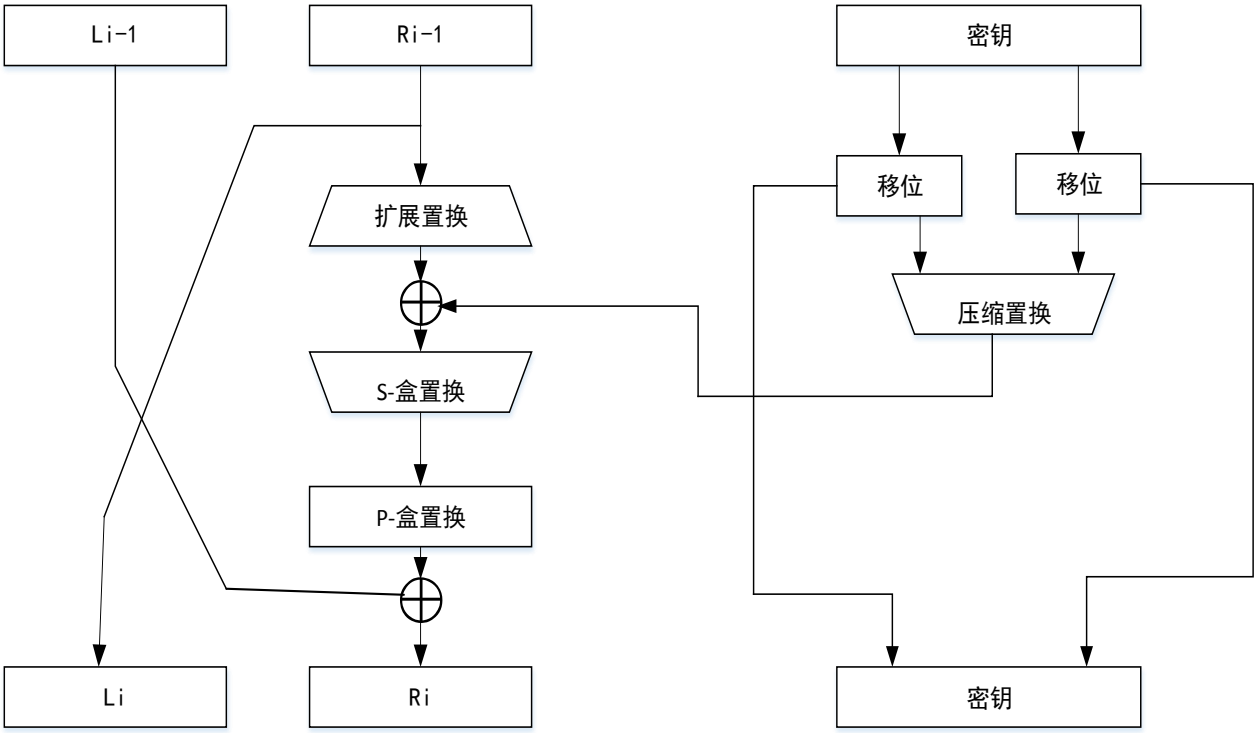

图 2.2 一轮 DES

尽管存在 16 轮运算，但是每一轮的运算过程几乎一致，因此，我们只需要对其中的一轮运算进行深入了解即可。如图 2.2 所示，DES 算法的一轮运算需要经过如下几个步骤：

1.将 64 位数据均分为两个部分：左半部分 L-Part 和右半部分 R-Part。

2.将 56 位密钥进行压缩置换得到 48 位的 K。

3.R-Part 通过扩展置换运算，得到 48 位的 R。

4.将 R 与 K 进行异或，得到 48 位新数据 D。

5.D 通过 S 盒代替获得 32 位新数据 M。

6.M 进行 P 盒置换获得 32 位新数据 A。

7.将 A 与 L-Part 进行异或得到 Data.

8.将 R-Part 作为新的左半部分，将 Data 作为新的右半部分。

16 轮运算过后，将结果通过末置换处理，密文就可以呈现在眼前了。通过这样的处理，信息就被加密成其他人无法识别的乱码。而通信的双方却可以相互解密，因为 DES 算法是对称密码算法，我们只需要将加密过程中每一轮运算的密钥顺序倒过来，就得到了解密密钥链。经过同样的算法过程，我们可以获得明文。

经过一番了解，我们对系统的设计有了一定的思路：采用 DES 算法来作为网络保密通信系统的核心算法。在密钥只有发送方和接收方知道的基础上，其他人无法窃取并解密信息。这样，就从一定程度上保障了通信的安全。但是单凭 DES 算法，能否做到保密通信呢？首先，我们冷静的分析一下：该算法的安全性由密钥的保密程度所决定。当只有进行会话的两端知道密钥时，才是最安全的。这时，问题出现了：通信双方如何传输 DES 密钥而不被其他人发现呢？如果是在现实生活中，可能大家一起吃个饭，找个没人的地方，商量个暗号，就可以作为密钥。但是在网络世界中，肉眼根本无法辨别周围环境是否安全，如果简单的把密钥按照明文的形式进行传输，如果被怀有恶意的人所截获，则整个会话会曝光在他的眼前。因此，仅靠 DES 算法来保证系统的安全是不够的。这时，借助非对称密码算法[16]正好能够弥补这一缺陷。

2.3.2 RSA

由上可知，非对称密码算法的主要功能在于密钥协商。那么，它是怎么实现这一过程的呢？首先，我们得了解一下非对称密码算法的工作原理。正如其名字所显示的，非对称密码算法提供了两把钥匙：私钥和公钥。公钥用于向公众公开[14]，而私钥则用于个人保存。其中，两把密钥可以互相解密。因此，我们可以得出密钥协商的过程如下：

1.A、B 的公钥分别向对方公开。

2.A 生成会话密钥 K。

3.会话密钥 K 经 B 的公钥加密后，被发送给 B。

4.B 收到密文。

5.密文经 B 的私钥解密后得到 K。

在这个过程中，由于其他人没有用于解密的 B 的私钥，因此，他们无法窃取会话密钥。于是，整个会话对于其他人而言毫无意义。在本系统中，服务器充当 A 的角色，客户端充当 B 的角色，并采用 RSA 算法完成这一关键的步骤。虽然我们前面谈到了 RSA 算法，但是对于它的具体过程并不了解。在这里，我们将对它进行深层次的接触。

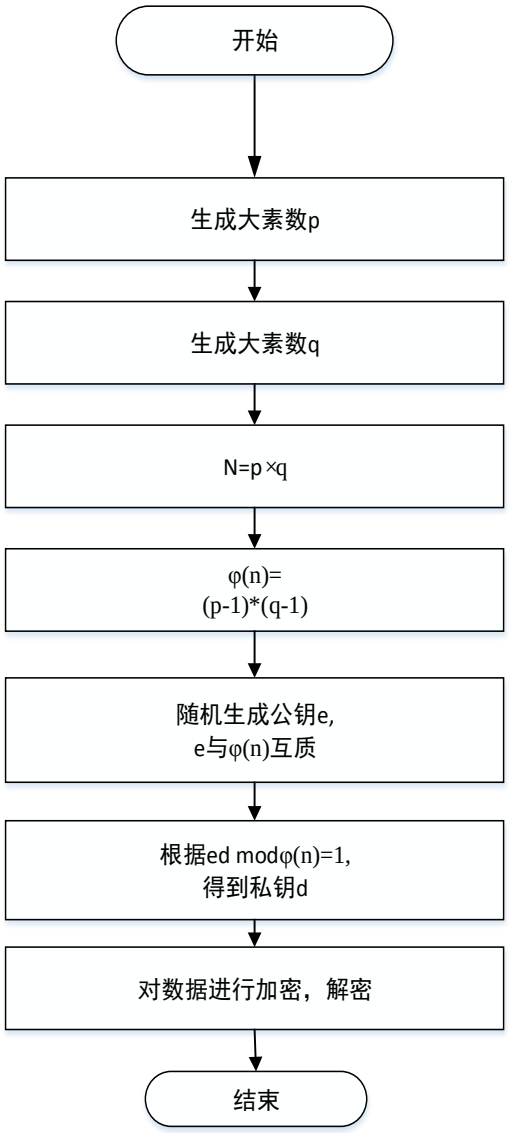

图 2.3 RSA 算法

如图 2.3 所示，RSA 算法的运算过程有如下几个步骤：

1.随机生成大素数：p、q。

2.计算乘积：n=p*q。

3.计算欧拉函数 φ(n) = (p-1)(q-1)。

4.随机选择一个整数 e，其中 e 满足：1< e < φ(n)，且 e 与 φ(n) 互质。

5.根据 ed ≡ 1 (mod φ(n)) ，计算 e 对于 φ(n)的模反元素 d。

6.得到私钥（d,n)和公钥（e,n)。

7.实现：C=E_d（M）以及 M=D_e（C）。

2.3.3 MD5

用户名和密码在数据库中用明文存放是极其不安全的，历史上就曾经出现过因数据库被盗而导致用户信息泄露的大事件。为了消除这一潜在的危险，我们需要寻找一种方式，让用户的关键信息能够保存下来而且不被轻易解读。这时候，哈希算法就派上用场了。

不论是什么样的数据，哈希算法总能找到唯一且定长的密文与之对应。将密文保存在数据库中，我们就无需担心数据库被盗所带来的一系列的危险。为了更好地保障用户的通信安全，本系统将采用 MD5 算法对用户的关键信息进行处理。MD5 算法是一种经典的哈希算法[17]，曾被广泛运用于各种保密通信系统中。然而随着科技的进步，彩虹表的存在，加之王小云教授已经成功破解了 MD5 算法，单纯利用 MD5 算法来对数据进行散列的想法已不再可靠。那么，难道 MD5 算法就要退出密码学的舞台吗？答案是否定的。我们可以通过加盐的方式对 MD5 算法进行加工。由于 MD5 算法能够将相似的数据散列成完全不同的密文，因此，我们只要在原始数据中添加一点料[18]，这样，攻击者就无法根据散列值推出原始数据。这种散列方法就是加盐的 MD5。在本系统中，具体的过程是这样的：当用户注册时，随机生成 Salt，并将 Salt 添加在账户及密码的指定位置，以获得新的字符串。然后新的字符串经过 MD5 算法处理后，就得到了可以存储在数据库中的密文。最后，将 Salt 写入特殊的文件中。当用户下次登录时，通过访问固定的文件夹来获取 Salt，再通过加盐的 MD5 算法对账户及密码取散列值，并将该值与数据库中的存储项进行对照，如果发现完全符合条件的项，则登录成功；否则登录失败。

2.3.4 基于 TCP 的应用编程技术

本系统基于 TCP 应用编程的技术，实现用户之间的交流与通信。具体的过程如下：

1.首先，服务器建立 TcpListener 并绑定端口。

2.然后，客户端通过建立 TcpClient 对象[5]，请求得到服务器的连接。

3.服务器同意与客户端在指定端口进行连接，于是两者便可以互相传递信息。

在本系统中，基于 TCP 的应用编程技术贯穿整个项目，无论是注册模块、登录模块还是通信模块，都需要通过建立连接，彼此交换指令和数据来实现密钥协商以及传送文本等功能，由此可见该技术的重要性。

2.3.5 数据库访问技术

本系统采用 MySQL 数据库，来存储用户数据。在登录过程中，通过对数据库[4]进行查询来验证用户身份的真伪。在修改密码的过程中，用户通过提交验证信息，待数据库检验通过后，密码方可重新设置，否则，密码设置失败。

3. 系统分析与设计

在项目的初始阶段，需求分析是一个很重要的步骤，如果需求分析考虑得很全面，将会为软件设计者的工作省去许多不必要的烦恼，但是如果分析不够全面，就会在后期造成很多困扰，严重的时候会导致软件设计者在某些地方需要进行返工。因此，一个好的项目的实现需要有一个全面具体的需求分析方案。

3.1.需求分析

作为一个通信工具，本系统需要实现的功能有：用户注册、登录、通信、修改密码等功能。由于系统具备保密特性，所以加密技术应该贯穿于这些功能的实现过程中。其中，每一个功能的实现应包含一系列的子过程。例如，登录、注册、修改密码的实现过程相似，都需要包含下述子过程：

1.密钥协商。

2.文本加密、解密。

3.指令传输。

4.数据库访问。

5.文件访问。

而通信功能的实现过程则包括以下子过程：

1.文本加密、解密。

2.文件加密、解密。

3.文件传输。

4.文本传输。

5.文件访问。

由上我们可以看出：密钥协商、文本的加解密以及数据库的访问几乎贯穿于整个项目。其中，密钥协商的关键在于 RSA，文本和文件的加解密过程依赖于 DES，而数据库保护则需要通过加盐的 MD5 算法来实现。因此，在项目编写的初始阶段，完成 RSA、DES 以及 MD5 这三大算法是最为重要的任务之一。

在算法编写完成之后，我们就可以进一步实现注册、登录等各项功能。在这当中，无论是指令传输还是文本、文件的传输，都离不开网络通信。这时，我们需要运用 TCP 应用编程技术在服务器和客户端之间搭建一条桥梁，传递各类信息。

在编写项目的同时，我们需要从系统的可扩展性、可维护性以及美观程度等方面进行考虑，最好能够达到以下要求：

1.代码精简程度高。

2.密码算法的运行速度快[10]。

3.空间占用率低。

4.方便扩展。

5.方便维护。

6.界面设计美观。

3.2.系统架构

在本系统中，可以依据功能来对模块进行划分，主要模块包括：注册模块、修改密码模块、登录模块以及通信模块。由于算法是整个项目的核心，所以我们可以单独划分一个模块——算法模块。因此，整个系统的功能模块图如图 3.1 所示：

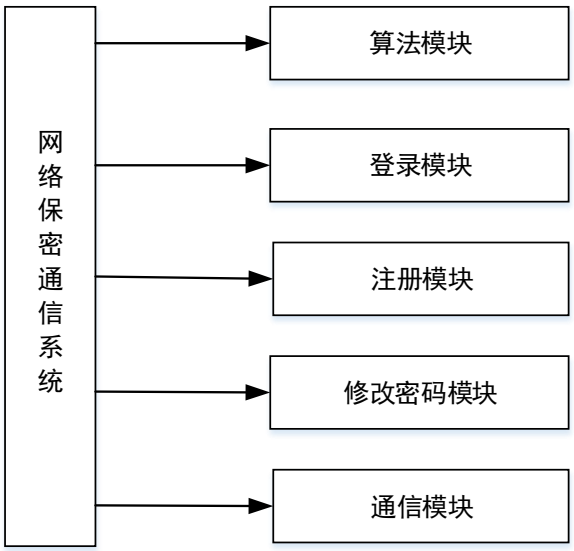

图 3.1 系统模块图

各个模块之间并不是独立的，如图 3.2 所示，我们可以看出：算法模块对其它四个模块均有重要的影响。其中，注册模块、修改密码模块、登录模块都需要使用到算法模块中的三大算法：RSA、DES 和 MD5；而通信模块由于用户的身份已经被验证，并且在登录模块中就已产生了会话密钥，所以只需要进行文本、文件、指令传输之类的信息交互过程，需要引用的算法就是 DES 算法。

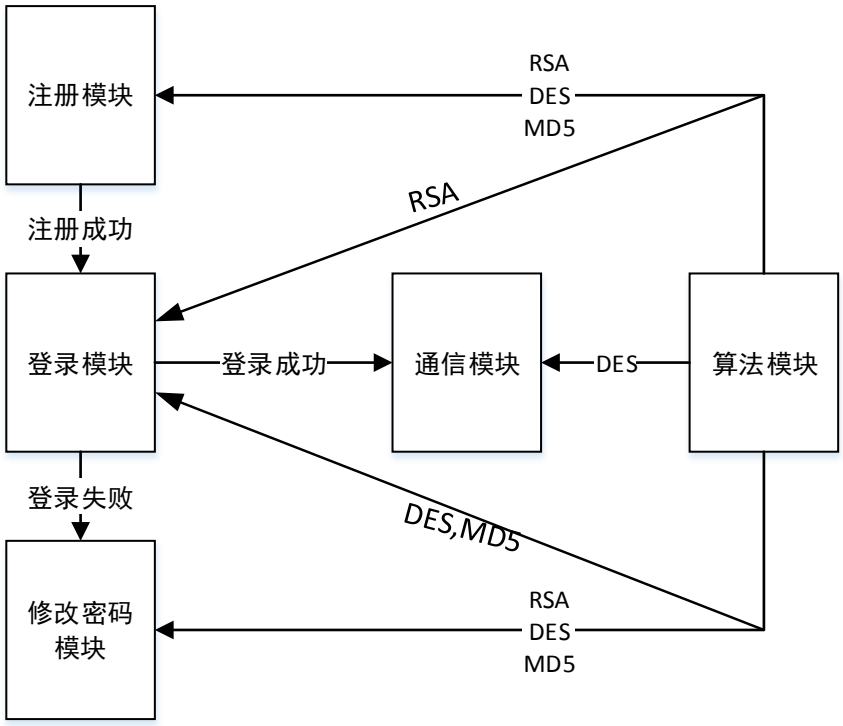

图 3.2 模块关系图

3.3.模块设计

3. 3. 1. 算法模块

经过上面的分析，我们可以知道：算法模块是系统的核心，而其他模块则通过调用算法模块中的相关函数来实现数据加密和解密的功能。因此，在项目的初期阶段，首要任务是完成算法模块，并为其他模块留出相应的函数接口。如图 3.3 所示，算法模块包括三个主要算法:RSA、DES 以及 MD5。

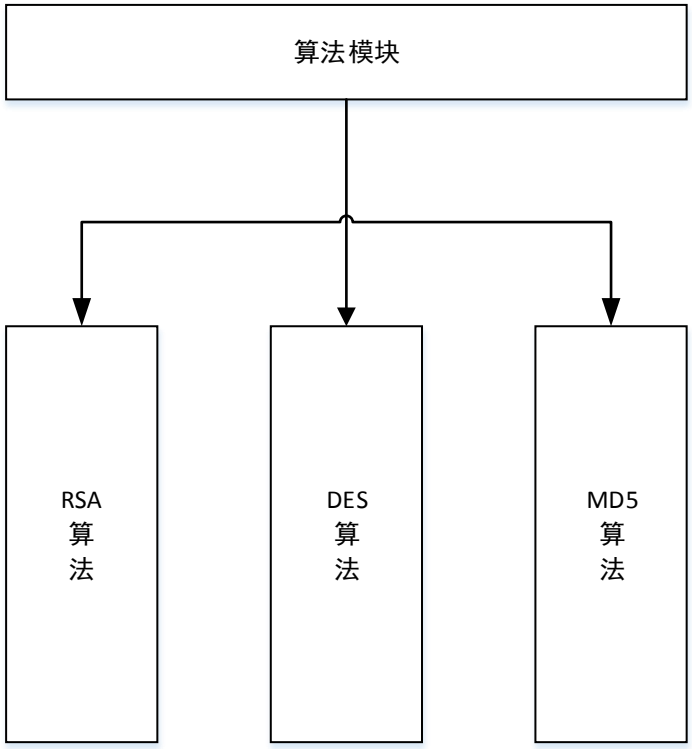

图 3.3 算法模块

因为 RSA 算法用于密钥协商过程，因此 RSA 算法应该为其他模块提供以下函数接口：

1.自动生成公钥/私钥对。

2.文本加密。

3.文本解密。

DES 算法负责文本、文件的加解密，因此 DES 算法应该提供以下函数接口：

1.文本加密。

2.文本解密。

MD5 算法用于对数据进行哈希处理，在数据库访问时需要用到这些函数：

1.加盐函数。

2.哈希函数。

在代码编写的过程中，我们需要为系统提供以上的函数接口。这样，在算法模块编写成功后，我们可以将其制作成为动态链接库，然后在其他模块中调用这些函数，实现密钥协商、网络流加密等功能。

3.3.2. 注册模块

在用户注册这一环节当中，我们可以联想一下平时进行注册时的场景。首先，我们填入的注册信息不能为空，不然无法进行提交。然后信息提交成功后，也会出现两种情况：一种是用户注册成功，这当然是我们所希望的；另一种情况是该用户名已被注册，我们需要重新进行注册。因此，在注册过程中，我们也需要考虑到这些情况。如图 3.4 所示，我们制定的注册计划是：

1.首先，用户在注册界面中输入账户、密码以及验证身份的信息。

2.点击提交按钮后，客户端会获取界面的信息，并进行一番检测。如果存在信息为空，则提交无效，并提示：用户需完善信息后才能进行提交。

3.成功通过检测之后，客户端发送“reg”指令给服务器。

4.服务器见到“reg”指令后，和客户端进入协商会话密钥的阶段。

5.注册信息经客户端加密后，传递给服务器。

6.服务器解密获得注册信息。

7.验证用户名是否存在。由于数据库中存放的是经哈希所得的密文。所以，首先要对注册信息中的用户名进行哈希取值，再进行数据库查询。如果经过查询证实该用户名已被注册，则向客户端发送“regFail”指令；否则，进行下一步。

8.如果证实该用户名并不存在于数据库中。余下的用户信息会通过哈希处理过程，比如：密码以及身份验证信息。最后将所得密文写入数据库中。

9.存储成功后，将“regSuccess”指令传递给客户端。

10.客户端依据指令做出相应的显示。

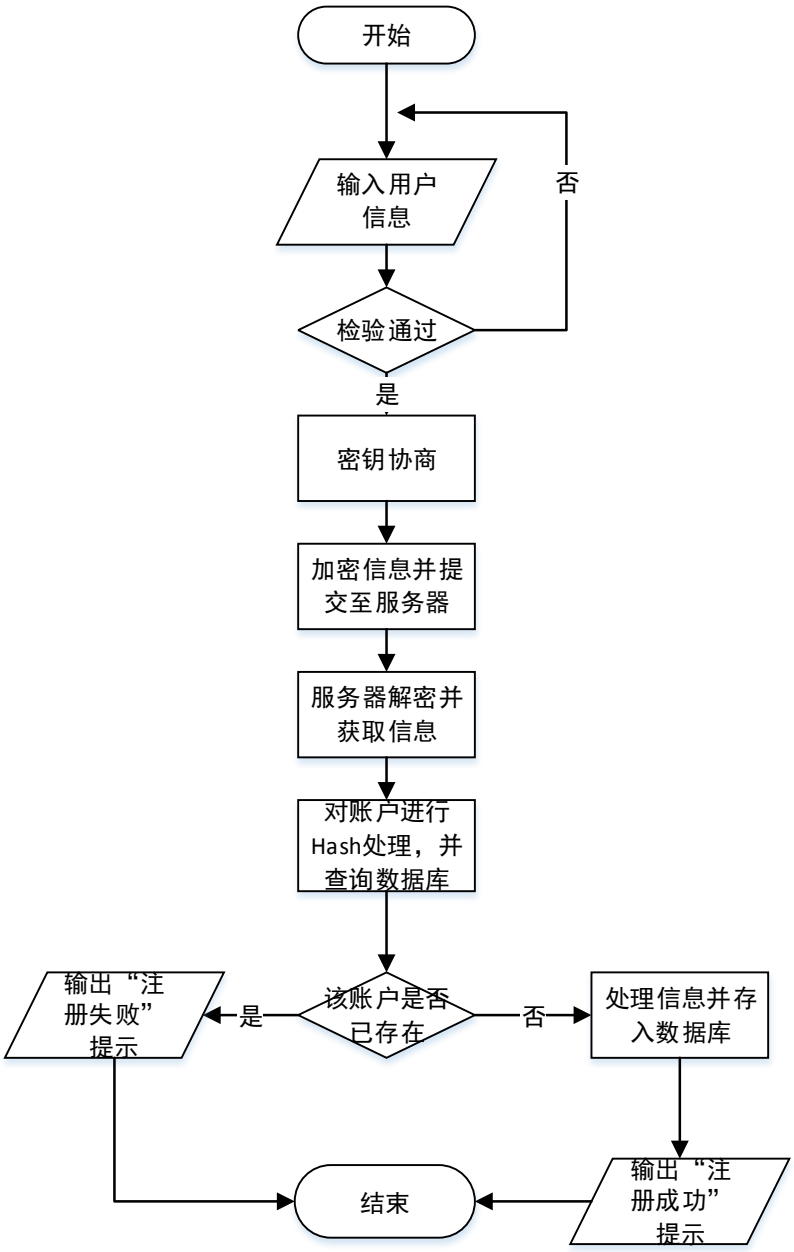

图 3.4 注册流程

3.3.3. 登录模块

登录模块用于辨别用户的真伪。只有当用户输入的账号及密码在数据库中存在对应项时，用户才能进入主界面。其中，登录模块的工作流程如下图所示。

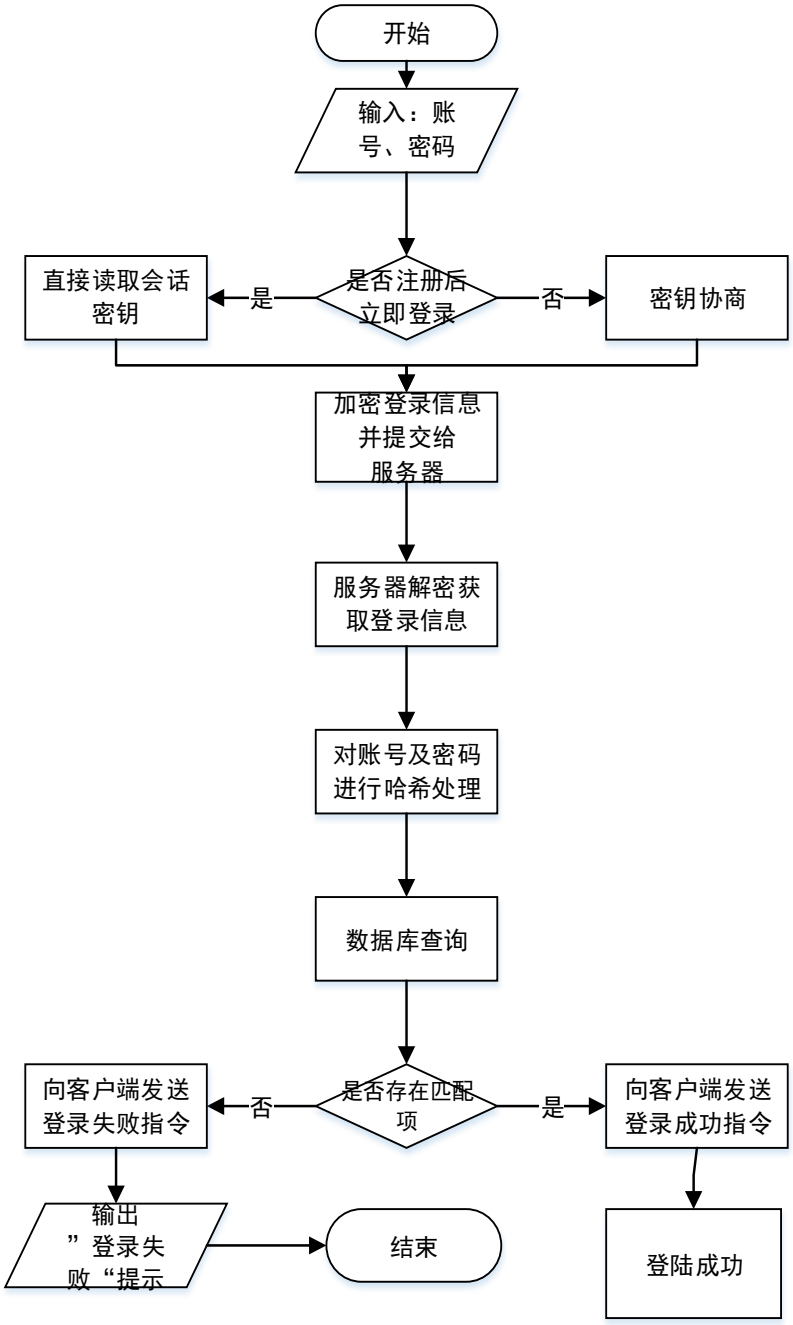

图 3.5 登录流程图

如图 3.5 所示，登录过程与注册过程有着相似之处，都需要通过访问数据库来判断是否执行某项操作。其中，具体过程如下所示：

1.首先在登录界面输入账号及密码。

2.提交之后，程序会进行一个判断：这次登录是否为新用户注册后立即登录。如果是，直接读取在注册过程中产生的会话密钥，继续与服务器进行通信。否则，进入协商会话密钥的阶段。

3.登录信息经会话密钥加密后，传递至服务器。

4.服务器接收并解密得到登录信息，同时分别对账户及密码取哈希值。将所得哈希值进行数据库查询，如果存在匹配项，向客户端回馈一个登录成功指令“loginSuccess”；否则，向客户端回馈一个登录失败指令“loginFail”。

5. 客户端接收指令并执行相应的操作，如果是“loginSuccess”，则登录成功并进入主界面；否则，弹出“登录失败”的提示框。

3.3.4. 修改密码模块

本模块为用户提供修改密码的途径。修改密码模块和注册模块一样：首先是协商会话密钥的过程，然后再进行信息的传递。它的工作原理是：先告知服务器用户要修改密码，然后将新密码连同验证信息传至服务器，服务器对信息进行哈希处理之后，再对数据库进行查询。如果经数据库验证通过，则更新密码；否则，不能进行修改。本模块通过验证信息来鉴别用户，能够保障用户的账户安全，防止他人对密码进行篡改。

3.3.5. 通信模块

当用户登录成功，即可进入通信模块。在本模块中，我们需要实现的功能包括：文本加密传输和文件加密传输。为了让本系统更加符合用户的需求，通信对象不应该只局限于一个用户，而应该扩展到多个用户。参考现在最流行的通信系统，我们可以将通信模块划分为私聊子模块和群聊子模块。

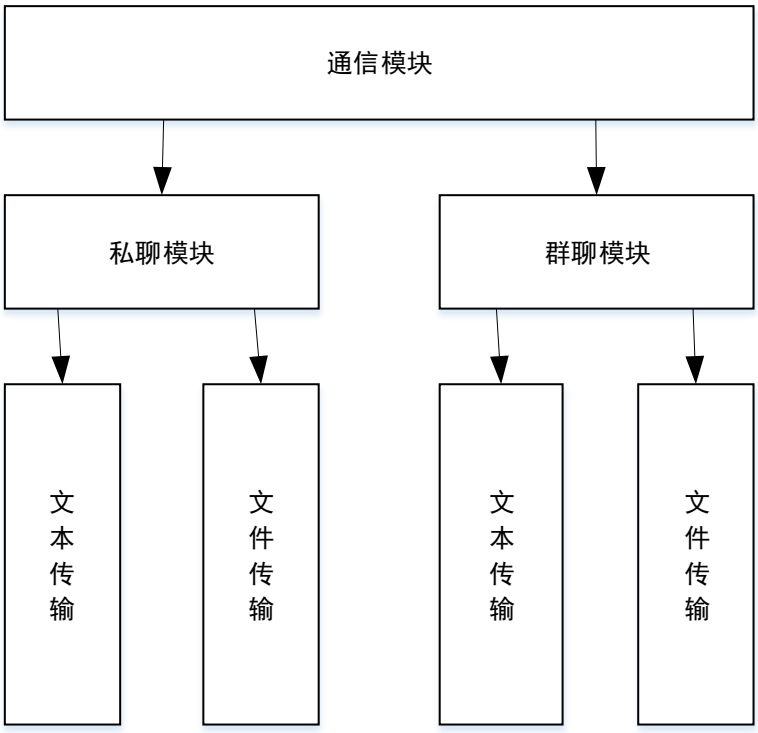

图 3.6 通信模块功能框架图

如图 3.6 所示，不论是选择私聊还是选择群聊，其核心功能都是文本以及文件的传输。由于通信模块具有保密传输的性质，因此，原始信息需经过 DES 算法加密后生成密文，然后密文发送给服务器，由服务解密并依据其中包含的指令类型进行对应的操作。

4. 系统实现与效果

经过一段时间的努力，本系统终于实现了其基本的功能，虽然还有一些不完善的地方，但是它已经能够很好地体现保密通信这一要求。接下来，我们将从系统实现过程以及运行效果两个方面分别对其进行描述。

4.1 系统实现

本系统采用 C 和 C#两种语言进行编写。其中，算法模块由 C 语言编写而成，其他模块由 C#语言编写而成。通过将 C 程序[7]制成动态链接库并导入项目中，然后通过 C#语言对 dll 中的函数进行调用，从而实现对网络流的加解密以及密钥协商等功能。

4.1.1. 总体实现

本系统具备一个通信系统所应有的基本功能，比如说：聊天、发送文件、修改密码等功能。如图 4.1 所示：进入软件的界面后，我们不论选择注册、登录还是修改密码，第一步都是进行密钥协商，因为只有商定好了会话密钥，双方才能进行信息的保密传输。如果我们进入了注册界面，注册信息通过服务器端的审核之后，就可以进入登录界面；如果我们选择的是修改密码，就会进入修改密码界面，当修改密码成功后，我们也可以进入登录界面；当登录信息通过审核后，客户端会进入通信主界面，在这里我们可以点击群聊按钮进入群聊界面，也可以双击好友列表中的某一项进入私聊界面。在主界面中，用户可以实现聊天、发送文件、接收文件等功能。

对系统的基本工作流程有了大概的认识之后，接下来我们将从程序设计以及代码实现等方面对系统进行详细的描述。

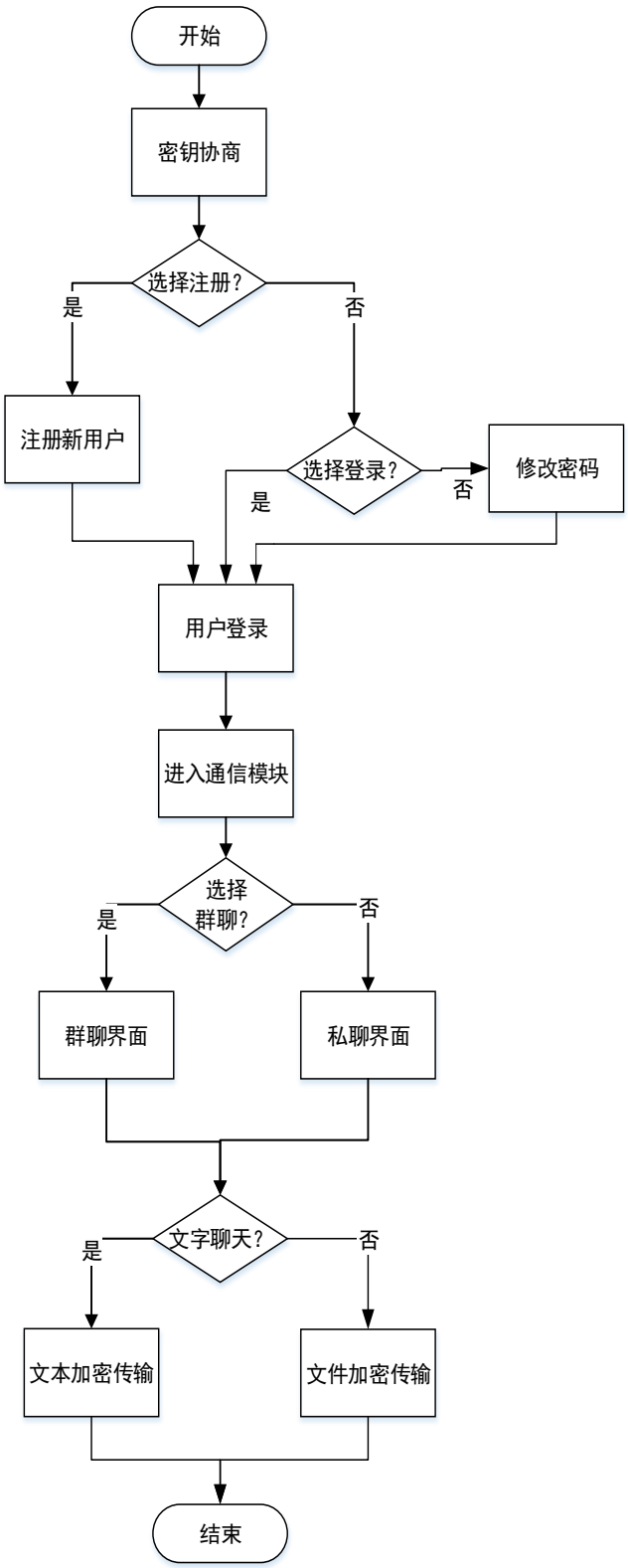
开始
密钥协商
选择注册?
是
否
注册新用户
选择登录?
修改密码
是
否
用户登录
进入通信模块
选择
群聊?
是
否
群聊界面
私聊界面
文字聊天?
是
否
文本加密传输
文件加密传输
结束

图 4.1 系统工作流程图

4.1.2. 密码算法实现

密码算法的实现步骤有以下几个：

1.首先，用 C 语言编写 RSA 算法、DES 算法以及 MD5 算法。

2.新建一个 Visual C++[11]控制台项目，在应用程序类型中选择 DLL.

3.将 Cpp 文件中的代码导入到新建的项目文件中。

4.然后将_declspec(dllexport)放置在项目需要使用的函数前面。

5.点击运行即可生成 dll。

6.将 dll 放置到网络保密通信系统的 Debug 文件夹下。

7.在 C#项目中用 dllImport 即可完成对 dll 的引用。比如需要引用 DES 中的加密函数，则可以使用这几句程序语句来实现：

[DllImport("Des.dll", EntryPoint = "des", CallingConvention = CallingConvention.Cdecl)]

private extern static int des(byte[] data, byte[] key, byte[] ciphertext, int len, int sign);

4.1.3. 注册功能实现

进入注册界面后，客户端会新建一个 TcpClient,并连接服务器的特定端口。待服务器同意连接之后，客户端程序会启动注册线程，双方通过协商生成共同的会话密钥。然后客户端将注册信息加密发送给服务器，服务器解密得到信息后与数据库进行匹配，如果不存在匹配项，新用户就能注册成功。最后，服务器向服务器发送一个加密的指令，客户端解密该指令，就能知道自己是否注册成功，并弹出相应的提示信息框。如果注册成功，则可以马上跳转到登录界面。其中，密钥协商的流程如下所示:

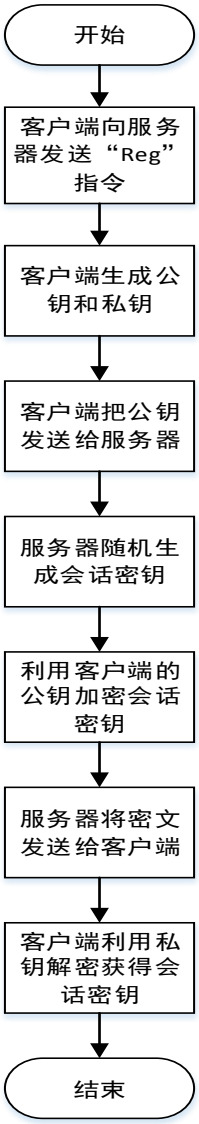

图 4.2 密钥协商

关于密钥协商，调用 RSA 算法产生公钥/私钥对的代码只有一句：

```
rsa(user.es, user.ds, user.ns);
```

服务器生成会话密钥并进行加密的代码如下所示：

```
Random r = new Random(unchecked((int)DateTime.Now.Ticks));//初始化随机数
```

```
int temp;//临时变量

for (int i = 0; i < 8; i++)//生成 8 个字节的会话密钥

{

    temp = r.Next(0, 255);//随机数在 0-255 之间

    user.key[i] = (byte)temp;    //将随机数转变成字节

}

byte[] keyCipher = new byte[600];//keyCipher:存放会话密钥加密后的结果

int len = textEncry(user.key, user.key.Length, user.ec, user.nc, keyCipher);//加密会话密钥
```

4.1.4. 修改密码功能实现

是不是随随便便就可以对密码进行修改呢？当然不是，在修改密码前，我们需要对用户的身份进行验证。其中，验证信息就是用户注册时所录入的信息。在本系统中，一共设置了两个验证问题。如果你的答案与注册时所提交的答案不相符，这就说明你很有可能是冒充其他用户，密码不能修改。如果你的验证信息完全符合，就允许你修改密码。在这一模块中，服务器和数据库所起的作用尤为重要。下面呈现的是其中的部分代码。

```
sql = "update user set user_pwd='" + user_pwd + "'where user_name='" + user_name + "'";//修改密码的 sql 语句

if(conOpen()==true)

{
```

```
        MySqlCommand msc = new MySqlCommand(sql, con);

        msc.ExecuteNonQuery();//更新数据库

    }

    con.Close();
```

4.1.5. 登录功能实现

用户如果想体验本系统的功能，首先得通过登录。因此，登录模块是本系统的一扇门，将真正的用户迎接进来，将不速之客拒之门外。登录的两个要素是账户及密码。客户端向服务器登录信息，服务器再对登录信息进行检测，如果检测通过，则发送“LoginSuccess”指令，否则发送“LoginFail”指令。整个过程如图 4.3 所示：

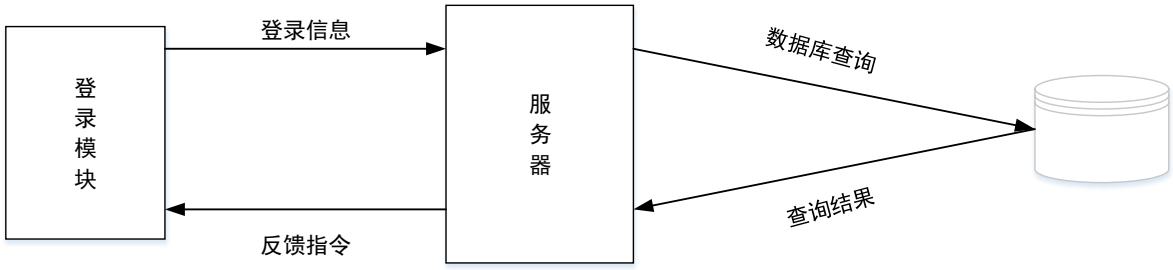

图 4.3 登录过程

在本模块中，最关键的地方是服务器对信息的处理：采用加盐的 MD5 算法将重要信息转换成定长的密文，再通过数据库对密文进行查询。

```
MD5(dataPwd, dataPwd.Length, pwdByte, dataName, dataName.Length);

MD5(dataAns1, dataAns1.Length, nameByte, dataName, dataName.Length);

MD5(dataAns2, dataAns2.Length, ans2Byte, dataName, dataName.Length);
```

4.1.6. 通信功能实现

客户端接收到“LoginSuccess”指令后，就进入了通信主界面。当一个用户登录成功，服务器会向它发送在线的用户列表，用户接收之后以好友列表的形式在主界面上显示。当用户双击朋友列表中的某一项，可以进入用于聊天的窗口。向其中发送消息或者文件，相应的好友就能接收到。

在通信过程中，存在一个线程负责处理服务器发来的消息。如果消息头部为“user”，客户端会从消息的剩余部分获取好友列表；如果消息头部为“add”，代表有新的用户登陆成功，客户端会将该用户添加到好友列表中；如果消息头部为“talkAll”,客户端会开辟一个新的群聊窗口；如果消息头部为“message”,客户端会获取聊天信息并将其添加到特定的私聊窗口中；如果消息头部为“msg”，客户端会将聊天信息添加到群聊窗体中；如果消息类型为“newFile”，代表有人向该用户发送了文件，客户端会弹出一个是否接收文件的选择框。其中，客户端接收消息的关键代码如下所示:

```
int len = reader.ReadInt32();

int length = (len / 8 + 1) * 8;

byte[] cipher = new byte[length];

cipher = reader.ReadBytes(length);

int l = length + 8;

byte[] temp = new byte[l];

des(cipher, key, temp, cipher.Length, 1);
```

在聊天过程中，文件的发送与接收具有重要的意义。如图 4.4 所示，上传文件的过程可以分为这几个步骤：首先，客户端打开用户指定的文件，获取文件长度，按照每次

发送 1024 个字节的方式计算出发送数据的总次数；然后，发出上传文件的指令，并告知服务器数据传递的总次数。紧接着，从文件中读取指定长度的字节，经加密后传递给服务器。服务器进行解密，将得到的原数据保存系统预先定义的路径下。重复这个过程，直到数据接收完毕，然后用 close()函数关闭文件。至此，文件上传成功。下载文件的过程与之相似，客户端给服务器传达下载文件的指令，服务器读取用户需要的文件，将数据陆续传至客户端。客户端选择一个路径保存数据。最后，文件得以下载成功。

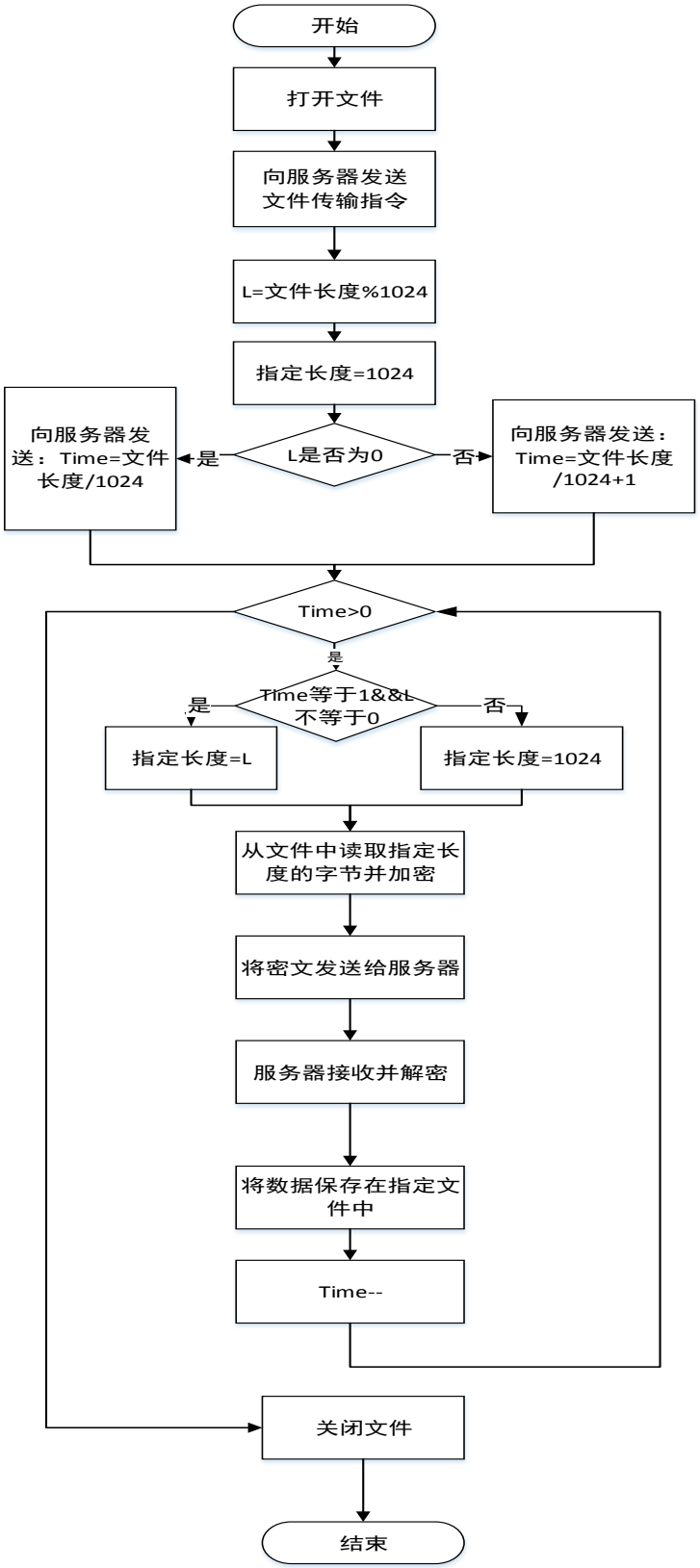
开始
打开文件
向服务器发送
文件传输指令
L=文件长度%1024
指定长度=1024
向服务器发送：Time=文件长度/1024
是
L是否为0
否
向服务器发送：Time=文件长度/1024+1
Time>0
是
Time等于1&&L不等于0
是
否
指定长度=L
指定长度=1024
从文件中读取指定长度的字节并加密
将密文发送给服务器
服务器接收并解密
将数据保存在指定文件中
Time--
关闭文件
结束

图 4.4 文件上传过程

如图 4.5 所示，服务器相当于文件中转站，当某个用户向另一用户发送文件时，首先将文件上传到服务器中，然后由另一个客户端决定是否接收该文件，如果接收该文件，则立刻从服务器中下载该文件；如果选择不接收，该客户端的通信主界面会暂存该文件的名字以及获取路径。如果某一个时刻，该客户端想获取该文件，只需双击待接收文件列表中对应的项，即可从服务器中下载。

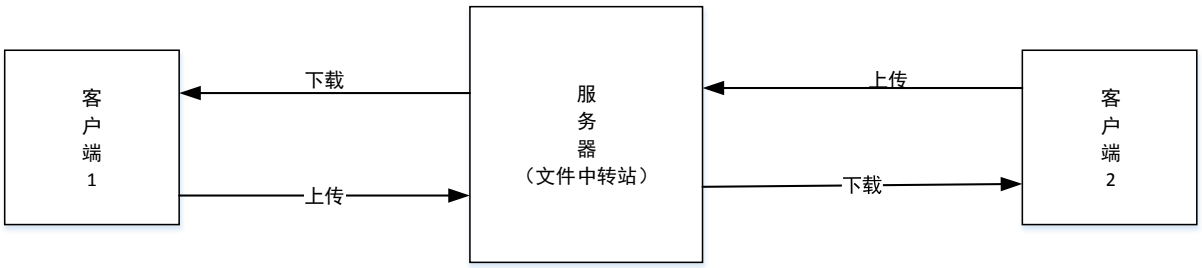

图 4.5 文件传输

其中，文件传输的部分代码如下所示：

```
for (long i = 0; i < time; i++)//在计算好的次数 time 内
{
    data = br.ReadBytes(1024);//br 在文件中一次读取 1024 个字节
    des(data, key, cipher, 1024, 0);//加密读取的字节
    writer.Write(cipher);//发送密文
    writer.Flush();
    remainLength -= 1024;//文件的剩余长度减少 1024
}
```

4.2.软件运行效果

4.2.1. 注册

注册时需要提交有关将来账号安全的重要信息，其中账号必须未被使用过。在注册时，用户需要牢记验证信息，因为验证信息可用于修改密码。其中，注册界面如图 4.6 所示。

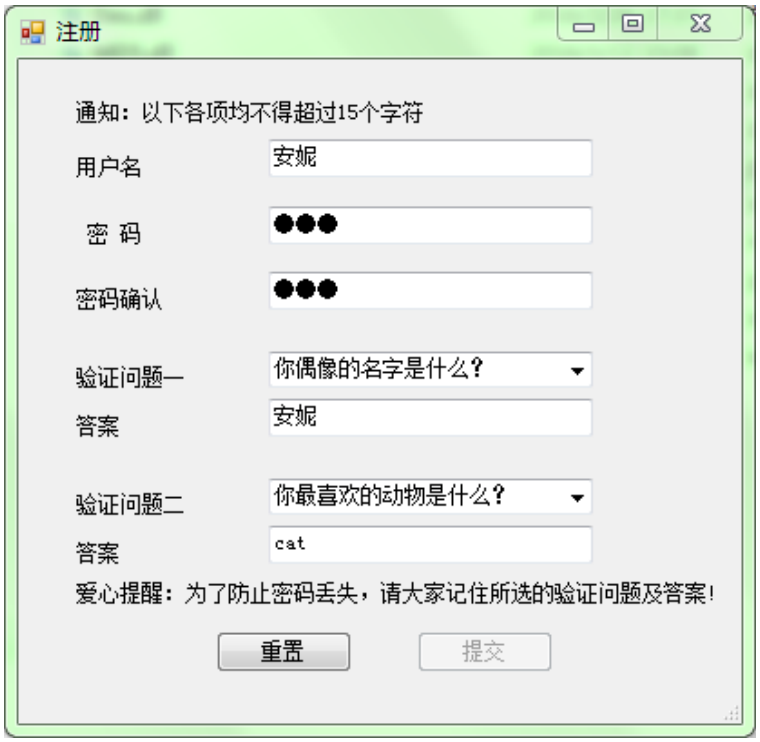

图 4.6 注册

4.2.2. 登录

当用户点击本系统的可执行文件，首先进入的是登录界面。

图 4.7 登录

4.2.3. 修改密码

点击忘记密码按钮，就进入了图 4.8 所示的界面。

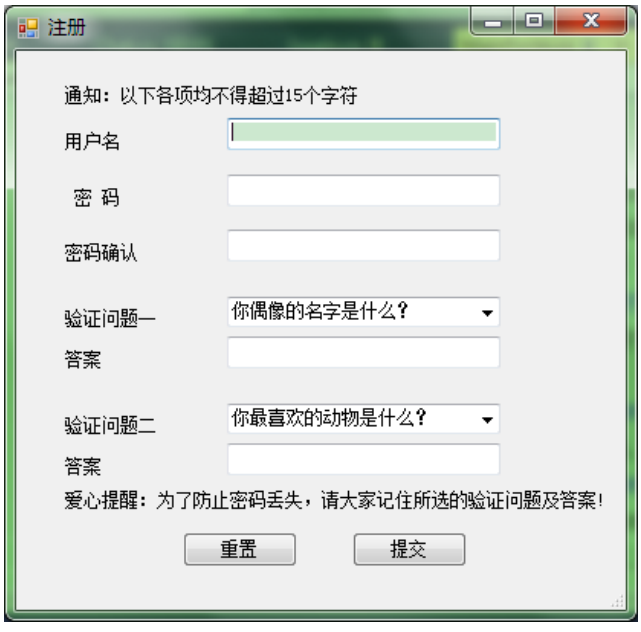

图 4.8 修改密码

4.2.4. 聊天

当用户点击好友列表中的某一项，就会进入私聊窗口，在这里可以发送信息以及文件。

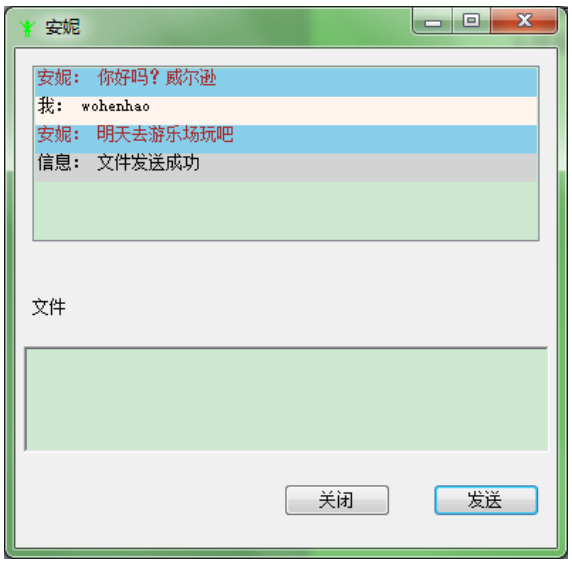

图 4.9 聊天

4.3.软件测试

在软件的制作过程中，测试是一项很重要的步骤，在测试的过程中，常常可以发现许多有趣的问题。通过不断地解决这些问题，软件用起来才舒服。同时，在测试的过程中，还能发现系统的不完善之处，经过细致的修改，就能使系统更加完善。

在测试的过程中，我遇到了几个有趣的问题：

问题 1:在对 MySQL 数据库进行操作时，出现找不到某一个键值的错误。

原因：数据库查询语句[8]中用到了本身带有”’字符的变量，导致 MySQL 数据库无法识别真正的变量。

解决方案：用’|’字符取代”’字符。

问题 2:在登录界面，每点一次登录按钮，都会导致一次登录过程。

原因：在按钮的 click 事件中，Enabled 属性没有被设置。

解决方案：将登录按钮的 Enabled 属性设置为 False；当用户登录失败时，再将登录按钮的 Enabled 属性设置为 True。

5. 总结与展望

5.1.总结

在软件的设计过程中，出现了许多意想不到的异常与错误，对于一个 C#新手而言，我面对的是不小的挑战。我依然记得：曾经有一个错误，一直卡在那里，我怎么改都解决不了。直到后来，我把代码重新写了一遍，才发现一些眉目：原来是我忽略了动态链接库中的一个参数。经过一番努力，我终于把那个问题解决了，然后紧接而来的是兴奋与快乐。现在想想：编程的过程中，这种快乐并不少见，然而这种快乐是建立在一番辛苦之上的。当然，辛苦并不是白费的，我不仅收获了快乐，也收获了知识。

5.2.展望

本系统虽然已经完成，但是其功能相对简单，在功能方面有待继续扩展。同时，本软件的核心在于算法，要想一直保持较高程度的通信安全，我们需要在算法上与时俱进，对算法不断改进。所以，在接下来的日子里，并不是一劳永逸的，而需要我们更加勤劳的钻研和学习。

参考文献

[1]Sebastian Roschke, Luan Ibraimi, Feng Cheng, Christoph Meinel.Secure Communication Using Identity Based Encryption[J].Lecture Notes in Computer Science,2010,6109:256-267

[2]谢希仁．计算机网络（第 5 版）[M]．电子工业出版社，2008

[3]Bruce Schneier 著.吴世忠 等译.应用密码学：协议、算法和 C 源程序(第 1 版)[M].北京：机械工业出版社，2001.1

[4]王珊，萨师煊.数据库系统概论(第 4 版)[M].北京：高等教育出版社，2006.5

[5]马骏,侯彦娥，等.C#网络应用编程(第 2 版)[M].北京：人民邮电出版社,2010.2

[6]谭浩强.C 程序设计(第四版)学习辅导[M].北京：清华大学出版社,2010.7

[7]陈倩诒，邓红卫.数据结构(C 语言版)[M].武汉：华中科技大学出版社,2013.1

[8]陶宏才，等.数据库原理及设计(第 3 版)[M].北京：清华大学出版社，2014

[9]DarrelHankerson，Alfred Menezes，Scott Vanstone 著.张焕国，等译.椭圆曲线密码学导论[M].北京：电子工业出版社,2005

[10]朱东生，赵建利，孙召伟，等.新编数据结构算法考研指导[M].北京：清华大学出版社，2010.7

[11]Brian W.Kernighan & Dennis M.Ritchie.The C Programming Language(Second Edition)[M].北京：机械工业出版社，2007

[12]张洁，朱丽娟.DES 加密算法分析与实现[J].软件导刊.2007(03)

[13]李少芳.DES 算法加密过程的探讨[J].计算机与现代化.2006(08)

[14]赵晓敏.RSA 算法及其安全性研究[J].科技资讯.2009(14)

[15]丁瑶，于志强.基于 ECC 算法的 PKI 系统设计与实现[J].计算机安全.2010(07)

[16]朱作付，徐超，葛红美.基于 DES 和 RSA 算法的数据加密传输系统设计[J].通信技术.2010(04)

[17]孙维国，李浩然.MD5 算法在数据安全中的应用及安全性分析[J].微计算机应用.2010(10)

[18]张绍兰，邢国波，杨义先.对 MD5 的改进及安全性分析[J].计算机应用.2009(04)

[19]黄志清.网络安全中的数据加密技术研究[J].微型电脑应用.2000(05)

[20]卿斯汉著.密码学与计算机网络安全[M].清华大学出版社，2001

致 谢

毕业设计即将结束，对于我而言，却是一段难忘的回忆。我记得：在这段日子里，我遇到了很多困难和挫折，也曾有过放弃的念头，但是我最终还是坚持了下来。也正因为这样，我看到了成功，感受到了成功之后的喜悦。我想说：一直以来所付出的心血并没有白费。虽然系统的功能比较简单，但是它却真实地达到了我所想要的效果。而我也不会轻易的丢弃这个作品，在以后的日子里，我会努力钻研，继续在原有的基础上扩展，让这个系统更加出色。

毕业设计的成功离不开同学的鼓励。在代码的编写过程中，程序上的一些错误很长时间都得不到解决，在这个时候，我真的感到很灰心。幸好，我的同学出来鼓励我，让我有继续下去的勇气。如果不是因为这些温暖的话，我想我现在都有可能陷在那些错误当中。

在这段时间里，老师给予了我们很大的帮助。老师的关心和指导让我们明白自己的方向和目标。我以前总是被界面的美观所蒙蔽，而忽视了后台的重要。所以在答辩的时候，总是找不到重点。是老师的一句话让我恍然大悟，我终于明白了：答辩的关键在于突出系统的重点。本系统强调的是保密通信，因此，我需要向大家展示的是实现保密通信的核心算法，而不是展示几张空洞无力的截图。我想：以后，我要学会抓住重点，尽可能用最具说服力的数据来让人眼前一亮。

附录

核心代码