

# 武汉理工大学毕业设计（论文）

## 基于二维码和移动终端的 SM2 密码数据处理研究

学院（系）： 信息工程学院

专业班级： 通信工程专业

通信 zy1701 班

学生姓名： 黎为楷

指导教师： 龙毅宏

## 学位论文原创性声明

本人郑重声明：所呈交的论文是本人在导师的指导下独立进行研究所取得的研究成果。除了文中特别加以标注引用的内容外，本论文不包括任何其他个人或集体已经发表或撰写的成果作品。本人完全意识到本声明的法律后果由本人承担。

作者签名：

年 月 日

## 学位论文版权使用授权书

本学位论文作者完全了解学校有关保障、使用学位论文的规定，同意学校保留并向有关学位论文管理部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权省级优秀学士论文评选机构将本学位论文的全部或部分内容编入有关数据进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

本学位论文属于 1、保密口，在 年解密后适用本授权书

2、不保密口。

（请在以上相应方框内打“√”）

（宋体小四号）

作者签名： 年 月 日

导师签名： 年 月 日

## 摘要

在使用浏览器调用密码模块对数据进行处理时，常用的方法是使用浏览器插件和 USB key 等密码硬件。但使用浏览器插件会面临技术复杂、兼容性差和不能跨平台等问题，而 USB key 则存在成本高的经济问题。本文提出了一种不需要浏览器插件和 USB key 等硬件设备的方案，并基于二维码技术和移动终端平台，设计了一个 Web 应用客户端网页和一个 Android 移动终端应用软件。经过测试，本设计能够实现对数据的密码处理功能。

本次设计使用的密码算法是 SM2 国密体制，相比传统的 RSA 密码体制有着安全性更高、密钥长度更短、抗攻击能力更强等优点，能实现密码处理的功能有加密、解密、数字签名和签名验证。

本文的主要研究内容如下：

- （1）分析 Web 应用客户端如何与用户交互来获取需求和数据，以及如何将这些信息以二维码的形式表示出来。
- （2）分析 Android 移动终端如何使用数据获取模块获取数据，如何使用本地的密码模块对数据进行密码处理，如何使用数据发送模块发送数据。
- （3）分析 Web 密码服务器如何接收 Android 移动终端发送的密码处理结果。

**关键词：** SM2 国密体制；数据密码处理；移动终端；二维码技术

## Abstract

When using browser to call password module to process data, the common method is to use browser plug-in and USB key and other password hardware. However, using browser plug-ins will face the problems of complex technology, poor compatibility and cross platform, while USB key has high cost economic problems. This thesis presents a scheme that does not need browser plug-in and USB key hardware equipment. Based on the two-dimensional code technology and mobile terminal platform, a web application client page and an Android mobile terminal application software are designed. After testing, the design can realize the password processing function of data.

The cipher algorithm used in this design is SM2 national secret system. Compared with the traditional RSA cipher system, it has the advantages of higher security, shorter key length, stronger anti attack ability, etc. it can realize the functions of cipher processing, including encryption, decryption, digital signature and signature verification.

The main contents of this thesis are as follows:

(1) This thesis analyzes how the web application client interacts with users to obtain requirements and data, and how to express these information in the form of two-dimensional code.

(2) This thesis analyzes how the Android mobile terminal uses the data acquisition module to obtain data, how to use the local password module to process the data, and how to use the data sending module to send data.

(3) This thesis analyzes how the web password server receives the password processing results sent by Android mobile terminal.

**Keywords:** SM2 state secret system; Data cipher processing; Mobile terminal; Two-dimensional code technology

# 目 录

第 1 章 绪 论.....	1
1.1 题目研究背景和意义.....	1
1.2 国内外研究现状.....	2
1.3 课题研究内容和章节安排.....	4
第 2 章 系统需求分析与整体方案设计.....	5
2.1 系统需求分析.....	5
2.2 系统技术方案.....	6
2.3 本章小结.....	7
第 3 章 系统的具体实现.....	8
3.1 Android 移动终端系统的实现.....	8
3.1.1 整体模块分析.....	9
3.1.2 数据获取模块的实现.....	10
3.1.3 数据处理模块的实现.....	11
3.1.4 数据发送模块的实现.....	14
3.1.5 密钥存储功能的实现.....	15
3.2 Web 应用客户端系统的实现.....	16
3.2.1 Java Web 开发介绍.....	16
3.2.2 前端用户交互界面的实现.....	17
3.2.3 登录功能的实现.....	19
3.2.4 利用 cookie 显示用户名在页面上.....	20
3.2.5 使用 Ajax 在前后端数据交互.....	21
3.2.6 接收 Android 端数据的实现.....	22
3.3 Android 与 Web 的数据交互的实现.....	23
3.4 本章小结.....	24
第 4 章 系统测试与分析.....	25
4.1 Android 移动终端的测试.....	25
4.2 Web 应用客户端的测试.....	27
4.3 进行数据发送的测试.....	30
4.4 本章小结.....	31
第 5 章 总结与展望.....	32
5.1 总结.....	32
5.2 展望.....	32
致 谢.....	33
参考文献.....	34

# 第 1 章 绪论

## 1.1 题目研究背景和意义

信息时代以前，人们记录信息的方式多为纸和笔，这种方式比较简单，但是通信起来颇为不便。随着信息时代的到来，计算机的出现给信息的保存和传输带来了翻天覆地的变化。以往通信所用的信件，现在则可动动手指便能以 Email 的方式代替。但是信息化时代也是一把双刃剑，产业化、数字化的飞速发展带来的效益虽然大大提升了，但通信安全也逐渐成为了一种隐患。计算机网络的一个特点是开放性，这种行为有利有弊，好处不言自明，弊端则是，有些不法分子会利用这个特点实施违法行为，从而严重危害计算机网络的安全。为了避免这些问题的发生，有效地实现计算机网络通信的应用，信息安全学应运而生<sup>[1]</sup>。

信息安全里的一个重要部分是密码学。从古至今，密码学一直是一门热门的学科。从封建时期的古典密码，到战争年代的摩斯电码，密码学发展到今天，已经信息化和商业化了。现代密码学是信息安全技术的理论基础，其原理是发送信息者将发送的数据经过一定的方法加密，将明文转变为密文，并将密文发送给接收方。接收方要做的事情是加密的逆过程，即将密文转变为明文<sup>[2]</sup>。

现代密码算法从体制可以分为对称密码体制和非对称密码体制。密码的处理过程大致如下，发送方若想发送一条加密信息，首先他需要使用一种加密算法，然后配合特定的密钥，将信息变成密文；对于接收方来说，若要从密文里得到原始信息，他需要使用相应的解密算法，然后配合特定的密钥，将信息变成明文。分组密码体制是对称密码体制里的一种。典型的分组密码体制有 DES、3DES、IDEA、AES 等<sup>[3]</sup>。公钥密码体制与对称密码体制的区别在于密钥的分配上。对称密码体制的密钥涉及一对密钥，发送方和接收方使用相同的密钥；而非对称密码体制的密钥一对存在，通常加密密钥给发送人，解密密钥给接收人。的公钥密码算法有 RSA、椭圆曲线密码 ECC、Rabin、ElGamal 和数论研究单位算法等<sup>[4]</sup>。

关于加密的设备，目前的发展也是五花八门。用不同的设备进行密码处理就会涉及到密钥的存储和使用问题。目前有关计算机通信中对密钥的存储主要有两种手段：第一，将用户密钥存放于用户的个人计算机中，当需要对数据密码处理时，调用计算机相应的密码处理模块，并使用本地密钥处理数据（如加密、解密、签名和签名验证）。这种方案会有安全隐患，当用户使用公共计算机时无法使用自己的本地密码模块。若通过移动存储设备，比如 U 盘，复制密钥然后使用时，有可能被他人获取到该密钥。第二，使用专门的密码硬件装置，比如 U 盾，插入计算机里对数据进行密码处理。该方法最大的特点是安全，适用于用户在网吧或其他公共场所需要加密的状况。但该方法也有相应的问题，除了购买和使

用 U 盾时会产生额外的费用外，有些公共场所出于安全的考虑，不提供 USB 接口，无法输入自己的密码装置。另外，如果密码的应用程序是浏览器，则通过浏览器调用本地的密码模块是非常困难的，需要使用额外的第三方插件，而这也面临着兼容性差、操作繁琐等实际问题<sup>[5]</sup>。而移动互联网的出现将很好地解决这个问题。

当今世界离不开互联网。截至 2019 年全球互联网用户已达 38 亿，其中中国以 21% 的占比位居世界第一<sup>[6]</sup>。移动互联网是互联网的后继，其鲜明的一个特点就是“随时随地随身”与他人通信。移动互联网的一个重要的应用领域是移动终端。与个人电脑相似，移动终端相当于“行动的”个人电脑，且经过多年硬件的升级进步，其在数据处理能力上已经能赶上个人电脑的水平了。同时移动终端还具备着比个人电脑更适配的触控屏、摄像头拍照和二维码扫描功能，因此随着时间的发展，它的高普及率、便携性、移动性、私密性，比个人电脑更受人们的喜爱。在移动终端里设计密码模块的好处，不仅解决了上文提到的电脑和专用加密装置带来的问题，同时也更符合时代发展的需要。

## 1.2 国内外研究现状

基于信息安全的相关计算机网络应用技术主要有以下几个方面：数据加密技术、防火墙技术和访问权限控制技术<sup>[7]</sup>。数据加密技术指的是密码学的应用，其重点是对数据的保护。防火墙技术是一种应用安全技术。企业内部网络和 Internet 之间几乎所有的边界都有防火墙，因为它可以抵御外部网络攻击和入侵。访问控制技术是指在访问资源时需要获取授权，使系统能够合法运行和处理请求。它是指用户的身份及其所属的定义组限制用户访问某些信息项或使用某些控制功能的技术，如 uninac 网络访问控制系统。

世界上有许多机构和个人都在探索数据保护技术的研究。1991 年，Weiser<sup>[8]</sup>提出了“泛在计算(Ubiquitous Computing)”的概念，即人们可以在任意时间、任意地点通过合适的终端与网络进行连接从而获取信息和服务，开启了对移动互联网研究的先河。2015 年，国内学者朱为朋<sup>[9]</sup>提出了移动终端及其数据加密方法，通过在移动终端所运行系统里建立安全核心领域，设计一个安全密码模块，在安全领域里对用户的敏感数据进行密码处理，并保存下来。移动终端使用的操作系统有 Android 和 ios，Android 操作系统的特点是开放性和包容性，因此引来了很多的开发者投入精力。现在已有基于 Android 的透明加密技术，和基于 Android 的移动智能终端数据安全防护技术，说明利用移动终端来进行密码处理的技术已逐渐成熟<sup>[10]</sup>。

二维码是相对于普通的一维条形码的相对说法，它能比普通条形码存储更多的数据量，因此得到了广泛的使用。其原理是用某种特定的几何图形按一定规律在平面上分布黑白相间的、记录数据符号信息的图形，类似于计算机的二进制数据“0”“1”比特流，使用若干个与二进制相对应的几何形体来表示文字数值信息，通过图象输入设备或光电扫描设备自

动识读以实现信息自动处理<sup>[11]</sup>。二维码的出现无疑为信息传递增添了极大的便利。

密码学是一门成熟的学科，其正式的发展历史从上个世纪七十年代开始算起，已有了 50 个年头。该学科在研究初期，其初衷主要用于军事和外交领域。比如在第二次世界大战期间，德国军方启用“恩尼格玛”密码机，密码学在战争中起了非常重要的作用<sup>[12]</sup>。但密码技术离我们并不遥远，小到使用手机扫码移动支付，大到使用专门的远程控制设备打开仓库大门，我们无时不在使用密码学。因此在信息化数字化快速转型后，一些其他的领域，比如金融和通信，也有了信息加密的需求，这让普通民众和一些企业开始从事于密码和信息安全相关的研究与开发。数据加密技术使用密钥进行密码处理，它将明文信息映射成其他类型的数据，也就是变成没有规律、不容易翻译的密文。密文信息被收到之后，可以使用解密密钥对将数据重新映射回去，即变成能阅读的明文。保证信息安全送达就是计算机网络数据安全的核心技术<sup>[13]</sup>。传统的数据加密算法有以下四种：置换表算法、改进的置换表算法、循环移位和 XOR 操作算法和循环冗余校验算法。为了更好的对市场进行管控，著名的“美国数据加密标准（DES）”里提及了一些新的密码算法，包括 RSA、DES、SHA 等。近期又出现了加密强度更高的 AES、ECC 等更为先进的密码算法<sup>[14]</sup>。

计算机网络安全中大量充斥着数据加密技术。网络数据库加密、软件加密、电子商务和虚拟专用网里都会有它的身影。网络数据库管理系统的安全等级通常有 C1 和 C2 之分，使用的操作系统一般是 windows nt 或 UNIX，因此容易遭受攻击，比如在计算机存储模块和数据传输通道模块。这样会给数据安全带来隐患，会让个人电脑易被窃取或篡夺有用的数据和各种密码<sup>[15]</sup>。软件加密的应用场景大多是杀毒软件或杀毒软件，因为在检查数据是否感染病毒的过程中，需要保密<sup>[16]</sup>。SSL 安全协议、数字证书、电子签名和其他加密技术通常用于确保交易的成功<sup>[17]</sup>。VPN 数据加密主要体现在路由器硬件加密上。当发送方从路由器发送数据时，加密文本通过加密模块直接在互联网上传输。

主流密码算法的设计原则是抵抗密码攻击。然而，对于所有实际应用，性能和实现成本也是重要的考虑因素。如果密码算法过于繁重而无法使用，即使它具有很高的安全性，也不会应用，因为这些密码的使用场景通常是电子商务、银行和在线交易。密码模块通常集成在嵌入式系统中，因此需要考虑硬件的计算能力。对于 DES、3DES、AES 和 blowfish 等主流加密算法，aamer Nadeem 等人在 2005 年的实验中总结了<sup>[18]</sup>。当他们使用统一的编程语言，使用自己的标准密码规范，并在两个不同的硬件平台上进行测试时，blowfish 的加密速度最快。虽然与 blowfish 集成的嵌入式系统性能最强，但其安全性并不是最高的。因此，在实际应用中，我们应该平衡算法的性能和安全性，采取折衷的方法。

RSA 密码算法的理论基础是数学上的一个难题——大数分解问题。近年来，由于大数分解问题逐渐攻破，计算机数据处理能力不断增强以及计算机网络和移动互联网的飞速演



进，为了实现安全加密，就需要对 RSA 密码提出更高的要求，其中最常用的手段就是增加密钥长度。但是增加密钥长度会带来新的问题：更复杂的解密设备和更大的时间复杂度，这无疑会严重阻碍 RSA 密码的应用场景。为了寻找一种新的密码算法，1985 年 N.Koblitz 和 Miller 提出<sup>[19]</sup>一种基于椭圆曲线 ECC 的新加密算法，其理论基础也是数学上的一个难题——有限域中的椭圆曲线上点群中的离散对数问题 ECDLP。相比于大数分解问题，ECDLP 的算法复杂度大大提升，呈指数级。因此，使用 ECC 算法的密码将比使用 RSA 算法的密码具备更强的抗攻击性、更少的 CPU 占用率、更低的网络消耗和更快的加密速度<sup>[20]</sup>。

SM2 是一种国家密码标准，其主要使用的密码算法就是椭圆曲线 ECC 算法。2010 年，SM2 第一次亮相。2012 年，SM2 成为中国商用密码标准。2013 年，孙荣燕，蔡昌曙，周洲<sup>[21]</sup>等人将 SM2 数字签名算法与 ECDSA 算法进行了对比与分析研究，对 ECDSA-SM2 算法的数字模型进行了正确性证明。2016 年，SM2 成为中国国家密码标准，正式标志其在公钥密码领域成为了最高级别的算法。同年，陈泽凯<sup>[22]</sup>实现了一个基于身份的 SM2 椭圆曲线数字签名方案 SM2-IBS，具有效率高、签名长度短、不依赖 PKI 等特点。

### 1.3 课题研究内容和章节安排

本论文的章节安排结构如下：

第一章，介绍题目的研究背景和意义，即当前浏览器调用密码模块功能会遇到的一些问题，然后分析国内外研究现状，最后制定本次设计的研究内容和本片论文的章节安排。

第二章，对题目的要求进行全面的需求分析，制定整个系统的设计方案与使用的技术手段。即分别使用移动终端来对数据进行密码处理，使用 Web 网页端与用户进行交互。

第三章，详细地论述系统各个组成部分的具体实现，即在移动终端，我们需要完成数据获取、数据密码处理和数据发送；在 Web 网页端，我们需要实现用户交互页面和登录功能；在 Web 服务器端，我们要实现登录验证，接收数据和与浏览器交互等功能。

第四章，对已实现的系统进行功能的测试，需要分别测试移动终端每个模块的功能能否实现，测试 Web 网页端能否正确生成二维码，测试 Web 服务器能否得到密码处理的结果。分析测试结果，若这些功能均能实现，我们便可认为本次设计成功。

第五章，总结与展望。在这里我们回顾整个项目的创新点，并指出设计过程里遇到的问题和困难，展望可能提出的解决方案。

## 第 2 章 系统需求分析与整体方案设计

### 2.1 系统需求分析

针对浏览器调用密码功能需要浏览器插件所带来的问题，包括技术复杂、兼容性差、不能跨平台，以及使用 USB key 等密码硬件所带来的成本高等问题，以 SM2 密码算法为例，研究开发一个针对浏览器的不需要浏览器插件、不采用 USB key 密码硬件的密码数据处理技术，研究开发的技术具有如下工作方式和功能：

（1）用户的 SM2 私钥与公钥均存在于用户移动终端中，公钥与私钥是基于 ECC 密码算法生成的一对密钥。移动终端实现各种密码运算，能够实现包括基于 SM2 的加密、解密，签名、签名验证这四项功能；

（2）当浏览器需要对数据进行密码处理时，浏览器通过一个用户交互界面获取用户的需求，然后得到待处理的数据，最后将需要处理的数据以二维码的方式展现；

（3）使用移动终端扫码二维码获得待处理数据，然后由移动终端中的密码模块对数据进行密码处理，包括 SM2 密码运算处理，然后手机将处理后的结果返回到 Web 系统，进行进一步的处理；

（4）移动终端是手机设备，使用的操作系统是 Android 或 ios 操作系统。

## 2.2 系统技术方案

本设计结合以上目标，设计了一种使用移动终端作为密钥存储和进行密码处理的装置，对数据进行加密、解密、数字签名和签名验证，技术方案及措施如下：

（1）如图 2.1 所示。系统的整体框架由 Web 应用客户端、Android 移动终端和 Web 密码服务器三部分组成。Web 应用客户端和 Web 密码服务器采用 Spring Boot 平台开发，开发环境是 IntelliJ Idea。Android 移动终端使用的操作系统是目前市面上应用最为广泛的 Android 操作系统，原因是其的开放性和包容性，开发环境是 Android Studio。Web 应用客户端、Android 移动终端和 Web 密码服务器之间的交互采用的是 Apache Http 协议。Web 密码服务器使用 Spring Boot 框架中自带的 Tomcat。Web 应用客户端与 Web 密码服务器之间前后端的交互采用异步的 JavaScript 和 XML (Asynchronous JavaScript and XML, AJAX) 技术。Android 移动终端的数据存储使用的技术是 SharedPreferences，这是一种简便的数据持久化技术，能够方便地存储用户密钥。

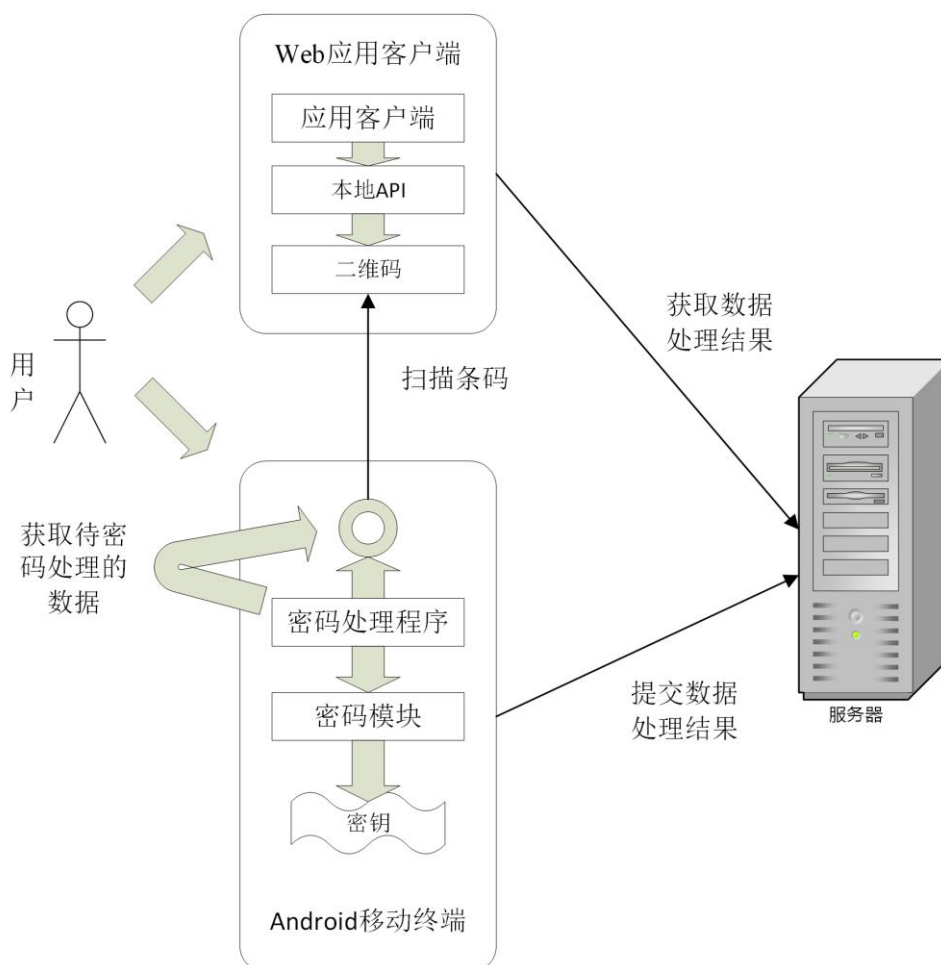


图 2.1 系统整体设计方案

（2）Web 应用客户端提供可供用户交互的浏览器页面，同时结合二维码技术，将用户输入的用户名、用户输入的原始数据、用户选择的密码数据处理方式和 Http 协议中通过 session 交互的 sessionId 组合在一起存放于二维码中，并将该二维码显示在客户端浏览器页面上。

（3）Web 密码服务器用于处理用户在 Web 应用客户端中的数据，判断用户名的存在和接收 Android 移动终端提交的数据处理结果。

（4）Android 移动终端的密码应用程序可以分为三个独立的部分，它们分别用于数据获取、数据密码处理和数据发送。其中数据获取模块结合二维码扫描技术，通过扫码获取 Web 应用客户端的密码处理数据和功能请求。数据处理模块用于实现密码处理，包括加密、解密、数字签名和签名验证。密码算法使用 SM2 国密体制。数据发送模块用于发送数据到 Web 密码服务器，包括数据处理的结果和会话标识。

## 2.3 本章小结

本章首先明确了系统的需求分析，即使用移动终端来进行密码处理，使用二维码来传递数据的具体细节；然后做出了实现本系统所需要的技术方案和总体设计思路。

## 第3章 系统的具体实现

本系统可以概括为两个平台、三个模块，分别是 Android 和 Web 两个平台，Android 移动终端、Web 应用客户端和 Web 密码服务器三个模块。Web 应用客户端会先以 HTML 语言里表单的形式获取用户的需求，然后以二维码的形式呈现在浏览器上；Web 密码服务器用来验证用户的登录，获取浏览器与服务器之间会话连接的会话标识以及接收 Android 移动终端发送来的数据；Android 移动终端通过扫码获取信息，通过算法解析信息，然后进行相应的密码处理，再将处理后的结果发送给 Web 密码服务器。Android 移动终端是实现密码功能的数据处理中心，是整个系统最重要的组成部分。

### 3.1 Android 移动终端系统的实现

Android 移动终端是整个系统里最重要的组成部分，如图 3.1 所示，由三个模块组成：数据获取模块、数据处理模块和数据发送模块。

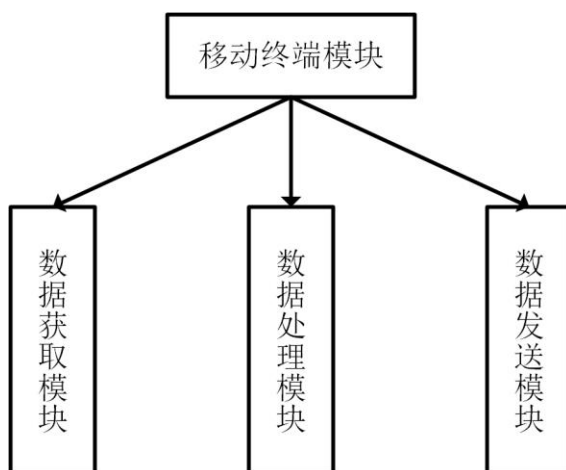


图 3.1 Android 移动终端子系统

数据获取模块：通过扫描二维码实现信息的获取，先将信息解析成数据信息和操作信息，然后发送给数据处理模块进行相应的处理。

数据处理模块：数据获取模块传送过来的数据信息和操作信息有：用户名 `username`，会话标识 `sessionId`，数据处理方式 `dataHandle` 和待处理数据 `data`，当用户需要进行签名验证操作时还会有数字签名 `signature`。操作信息 `dataHandle` 有四种取值，分别是 `Encrypt`、`Decrypt`、`Sign` 和 `Signature`。数据处理模块会根据不同的取值进行不同的处理方式，处理完成后会得到处理结果，并把结果递交给数据发送模块。

数据发送模块：数据发送模块的任务是将数据处理后的结果和会话标识 `sessionId` 一起发送给 Web 密码服务器。

### 3.1.1 整体模块分析

Android 移动终端的开发环境是 Android Studio+Android SDK，使用 PC 通过数据线连接 Android 手机的方式进行调试。

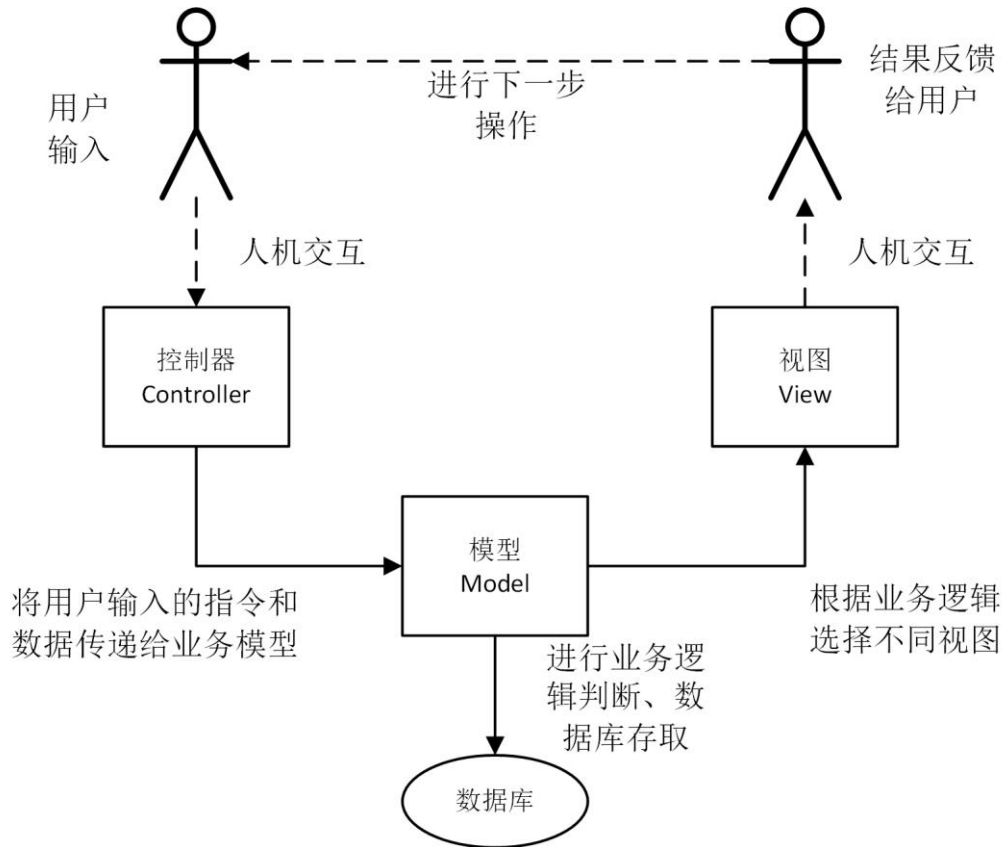


图 3.2 整体模块框架图

一个功能完整的项目开发离不开好的框架，如图 3.2 所示，本项目采用的框架是 MVC。MVC 是 Model、View 和 Controller 的缩写，其中文释义分别为业务模型、用户界面和控制器<sup>[23]</sup>。它将逻辑、数据和界面的代码分开来组织，将业务逻辑聚集到一个部件里面，这样在改进和个性化定制界面及用户交互的同时，不需要重写业务逻辑。

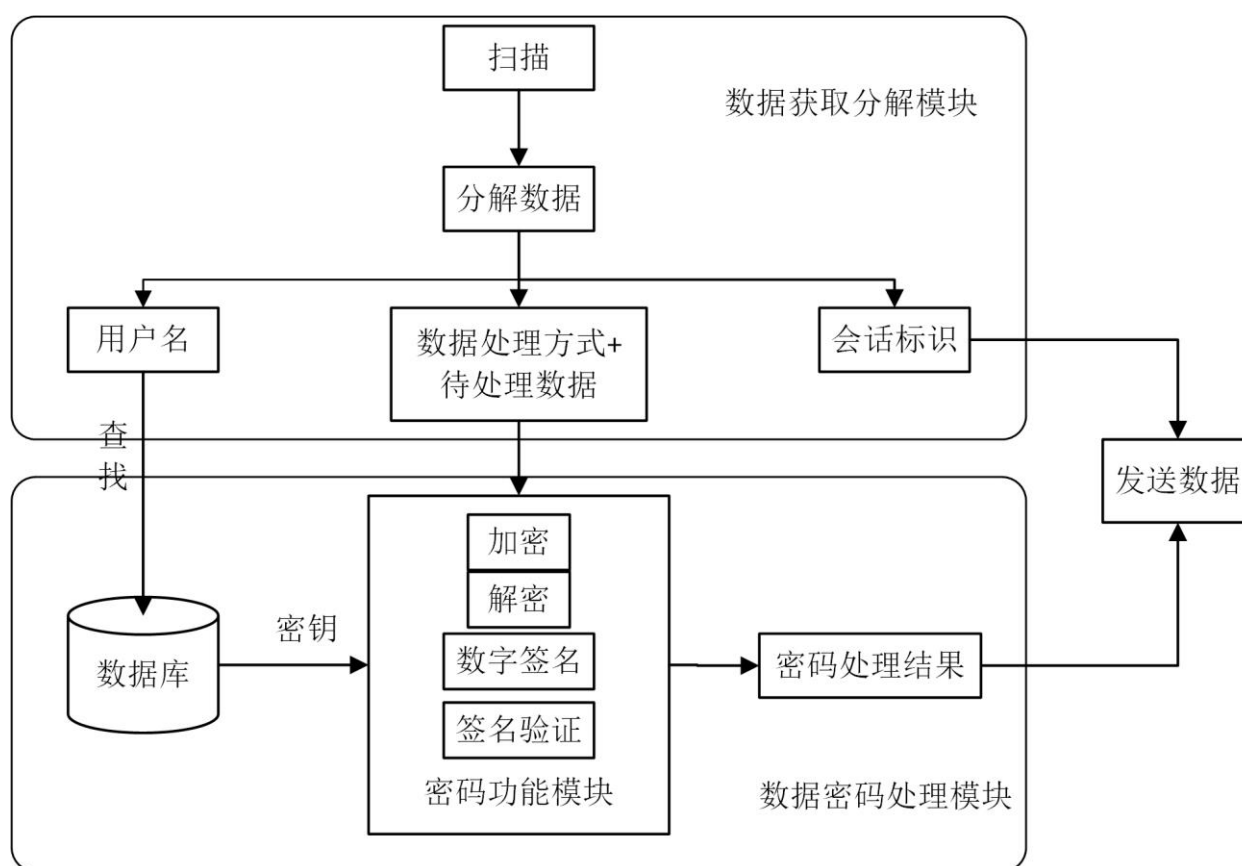


图 3.3 模块功能实现框图

整体模块功能实现的框图如图 3.3 所示。数据分解模块首先需要通过二维码扫描技术获取到存储在二维码中的信息。该信息将多个数据以字符串的形式组合在了一起，因此获取到信息后需要分解，以得到用户名、待密码处理数据、数据处理方式和会话标识。其中，用户名用来在数据库中查找对应的密钥，数据处理方式用来指定密码功能模块以何种方式进行密码处理（加密、解密、数字签名或签名验证），同时使用密钥对待密码处理数据进行密码处理，以得到密码处理结果。最后，需要将密码处理结果和会话标识一起发送给 Web 密码服务器。

### 3.1.2 数据获取模块的实现

数据获取是通过扫描二维码的方式实现的。这里使用的技术是开源的代码 Zxing 库。使用 Zxing 库的步骤如下：首先我们自定义一个 CaptureAct 类用来继承 Zxing 库包里的二维码扫描类 CaptureActivity。在 MainActivity 里，定义一个 scanCode 方法，在该方法里新建一个 IntentIntegrator 对象，使用 Integrator 对象的 setCaptureActivity() 关联 CaptureAct.class，并对其设置一些参数。这里的 IntentIntegrator 对象即用于跳转到 Zxing 的扫码界面。

```
public void scanCode(){
```

```

IntentIntegrator integrator = new IntentIntegrator(this);
integrator.setCaptureActivity(CaptureAct.class);
integrator.setOrientationLocked(false);
integrator.setDesiredBarcodeFormats(IntentIntegrator.ALL_CODE_TYPES);
integrator.setPrompt("Scanning Code");
integrator.initiateScan();
    }

```

其次，获取扫描的结果，使用 IntentIntegrator 对象的 parseActivityResult 方法即能得到扫描结果 result。

```

IntentResult result = IntentIntegrator.parseActivityResult(requestCode, resultCode, data);

```

得到 result 后需要进行条件判断，如果扫描成功，则弹出一个对话框，对话框里显示扫描获取的数据。对话框里还设置了两个按钮，分别是“Scan Again”和“finish”。点击“ScanAgain”按钮会退回上一页面重新扫描，点击“finish”按钮则会跳转到数据处理页面 handleActivity。

这里从一个页面跳转到另一个页面的技术叫 Intent。使用 Intent 不仅可以实现页面的跳转，还能传递数据。首先新建一个 Intent 对象，传入的参数为源 Activity 和目标 Activity；然后使用 intent.putExtra()方法用来传递数据；最后使用 startActivity()方法启动 Intent。

```

Intent intent = new Intent(MainActivity.this, handleActivity.class);
intent.putExtra("scan_data",result.getContents());
startActivity(intent);

```

综上所述，数据获取模块的流程图如图 3.4 所示。

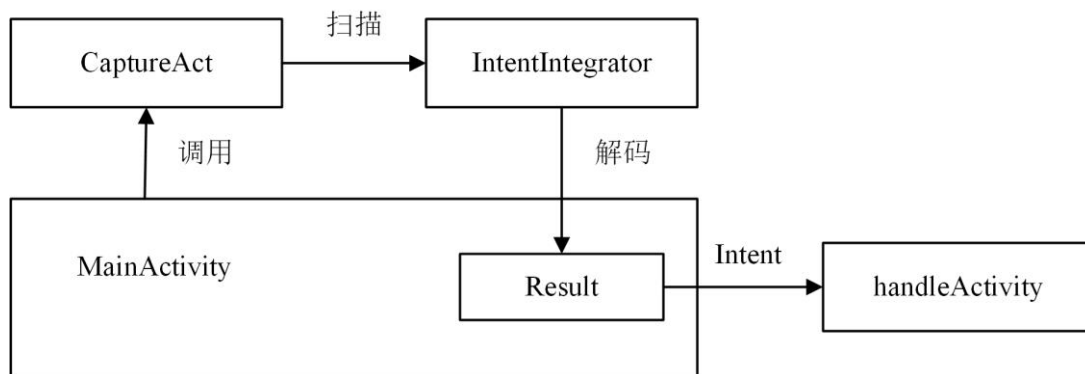


图 3.4 数据获取模块的流程图

### 3.1.3 数据处理模块的实现

首先通过 getIntent().getStringExtra()方法获取到扫描结果 scanData。



```
String scanData =getIntent().getStringExtra("scan_data");
```

数据获取模块获取到的 scanData 是一个组合的字符串，在数据处理模块处理之前需要先解析。scanData 的字符串格式如图 3.5 所示，可以发现，其中包含 username, ID, data 和 dataHandle，如果是验证数字签名操作，则还包含 signature，如图 3.6 所示。

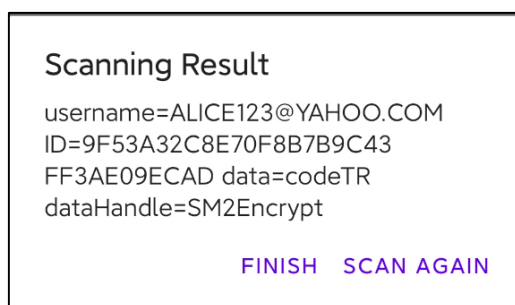


图 3.5 result 字符串格式 1

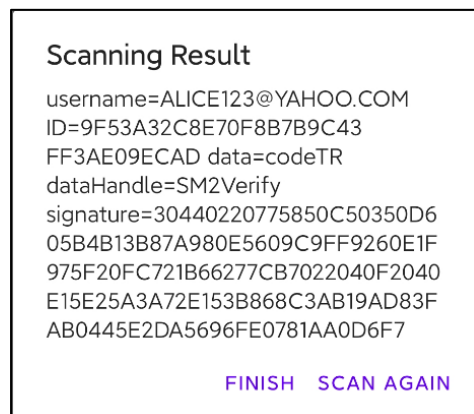


图 3.6 result 字符串格式 2

然后使用字符串的通用方法 split 方法，将 scanData 解析成几个字符串，并存放在数组中。

```
String[] array = scanData.split(" ");
```

同理，我们可以解析出 username, ID, data, dataHandle 以及 signture 的值，以便后面调用。

```
String[] usernameArray = array[0].split("=");
String[] IDArray = array[1].split("=");
String[] dataArray = array[2].split("=");
String[] handleArray = array[3].split("=");
if(array.length==5){
    signatureArray = array[4].split("=");
}
```

获取到 dataHandle 的值后，可以使用 switch 语句判断对数据进行何种处理。case “SM2Encrypt”表示对数据进行加密处理；case “SM2Decrypt”表示对数据进行解密处理；case "SM2Sign"表示对数据进行签名处理；case "SM2Verify"表示对数据进行签名验证处理；default: break 表示不做任何处理退出 switch 语句。

```
switch( handleArray[1] ){
    case “SM2Encrypt”:
    case “SM2Decrypt”:
```

```

    case "SM2Sign":
    case "SM2Verify":
    default: break;
}

```

下面分别介绍如何实现四种密码处理方式。

对数据进行密码处理会使用一个名称为 `org.bouncycastle` 的开源代码，该代码有一个工具类 `SM2Utils`，用于存放对数据进行密码处理的各种方法：`encrypt()`、`decrypt()`、`sign()`和`verifySign()`。先建立一个 `SM2Utils` 对象 `sm02`。

```
SM2Utils sm02 = new SM2Utils();
```

（1）数据加密方法 `encrypt()` 的实现。`encrypt()` 方法传入两个字符串格式的参数 `originaldata` 和 `publickey`。`originaldata` 为待密码处理的数据，`publickey` 为用户公钥，返回值是密文。传入的 `publickey` 不能直接用于加密，需要先进行 16 进制数到字节格式的类型转换 `Util.hexToByte(publickey)`，然后再进行 Base64 格式的编码 `Base64.encode()`。使用 `sm02.encrypt()` 方法对数据加密，将结果存放在 `encryptdata` 字节数组中。返回值是 16 进制的密文，所以需要使用 `Util.byteToHex()` 方法进行字节格式到 16 进制数的转换。

```

public String encrypt(String originaldata,String publickey) throws IOException {
    String publickeyS = new String(Base64.encode(Util.hexToByte(publickey)));
    byte[] encryptdata =
    sm02.encrypt(Base64.decode(publickeyS.getBytes()),originaldata.getBytes());
    return Util.byteToHex(encryptdata);
}

```

（2）数据解密方法 `decrypt()` 的实现。`decrypt()` 方法传入两个字符串格式的参数 `encryptdata` 和 `privatekey`。`encryptdata` 为密文，`privatekey` 为用户私钥，返回值是明文。传入的 `privatekey` 不能直接用于解密，同样需要先进行 16 进制数到字节格式的类型转换，然后再进行 Base64 格式的编码。使用 `sm02.decrypt()` 方法对数据解密，将结果存放在 `decryptdata` 字节数组中。返回值是字符串形式的明文，所以需要使用 `Util.byteToString()` 方法进行字节格式到字符串的转换。

```

public String decrypt(String encryptdata,String privatekey) throws IOException {
    String privatekeyS = new String(Base64.encode(Util.hexToByte(privatekey)));
    byte[] encryptData = Util.hexStringToBytes(encryptdata);
    byte[] decryptdata = sm02.decrypt(Base64.decode(privatekeyS.getBytes()),encryptData);
    return Util.byteToString(decryptdata);
}

```

```
}
```

（3）数据签名方法 `sign()`的实现。`sign()`方法传入三个字符串格式的参数 `username`、`originaldata` 和 `privatekey`。`username` 为用户名，`originaldata` 为待签名数据，`privatekey` 为用户私钥，返回值是数字签名。使用 `sm02.sign()`方法对数据加密，将结果存放在 `signature` 字节数组中。返回值是字符串形式的数字签名，所以需要使用 `Util.getHexString()`方法进行字节格式到 16 进制数字字符串的转换。

```
public String sign(String username,String originaldata,String privatekey) throws
IOException {
    String privatekeyS = new String(Base64.encode(Util.hexToByte(privatekey)));
    byte[] signature =
    sm02.sign(username.getBytes(),Base64.decode(privatekeyS),originaldata.getBytes());
    return Util.getHexString(signature);
}
```

（4）数据签名验证方法 `verify()`的实现。`verify()`方法传入四个字符串格式的参数 `username`、`originaldata`，`publickey` 和 `signature`。`username` 为用户名，`originaldata` 为原始数据，`publickey` 为用户公钥，`signature` 为数字签名，返回值是布尔值。使用 `sm02.verifySign()`方法对数据加密，将结果存放在布尔类型的变量 `isValid` 中，并返回 `isValid`。

```
public boolean verify(String username,String originaldata,String publickey,String signature)
throws IOException {
    String publickeyS = new String(Base64.encode(Util.hexToByte(publickey)));
    boolean isValid =
    sm02.verifySign(username.getBytes(),Base64.decode(publickeyS.getBytes()),
    originaldata.getBytes(),Util.hexStringToBytes(signature));
    return isValid;
}
```

### 3.1.4 数据发送模块的实现

数据发送模块是 Android 移动终端三大模块中的最后一个，其主要技术手段是 Android `HttpURLConnection`。`HttpURLConnection` 是一种多用途、轻量级的 HTTP 客户端，使用它来进行 HTTP 操作可以适用于大多数的应用程序。`HttpURLConnection` 的使用步骤如下：

（1）创建一个 URL 地址 `path`，使用这个地址连接到 Web 密码服务器；

```
String path = "http://ip4:8080/ServletForGetMethod";
```

（2）创建一个 `HashMap`，使用 `params.put()`方法存放希望通过 `HttpURLConnection` 发

送的键值对；

```
Map<String, String> params = new HashMap<String, String>();
params.put(key, value);
```

(3) 将 path 解析, 然后再调用 URL 的 openConnection() 方法来获取 HttpURLConnection 对象实例；

```
StringBuilder sb = new StringBuilder(path);
HttpURLConnection conn =
    (HttpURLConnection) new URL(sb.toString()).openConnection();
```

(4) 使用 setRequestMethod() 方法设置 HTTP 请求的方式为 GET；

```
conn.setRequestMethod("GET");
```

(5) 使用 setConnectTimeout() 方法设置连接超时的毫秒数；

```
conn.setConnectTimeout(5000);
```

(6) 将需要发送的数据拼装成一个字符串后发送出去。比如要发送加密后的密文：

```
String transData = "处理结果:" + encryptData + " " + "SessionId:" + IDArray[1];
result = UserInformationService.save("TransmittedData", transData);
```

其中 transData 为待发送数据，UserInformationService.save() 方法是自定义的数据发送方法。

以上是 Android 移动终端 HttpURLConnection 发送数据的代码，Web 密码服务器端在接收数据时也会有相应的代码，在之后的章节会详细介绍。

### 3.1.5 密钥存储功能的实现

本项目最初的设计思想就是将用户密钥存放在用户的手机上，因此在 Android 移动终端上需要实现密钥的本地存储功能。Android 端存储数据的技术主要有三种：一是 SharedPreferences，以键值对的形式存放数据，其特点是只能存放少量数据，比如数据量低的配置信息的存储；二是 SQLite，是一个轻量级的关系型数据库，集成于 Android 系统，能存放一些比较复杂的数据结构，特别适合对象存储；三是 File Save，适合存放比较大的文件（例如日志，图片缓存，apk 包等）或者某些特殊的配置文件。因为本设计只需要存储用户名和密钥对，因此选择更方便和轻量的 SharedPreferences 技术。

SharedPreferences 的使用步骤如下：

(1) 创建 SharedPreferences 对象：

```
private SharedPreferences
mSharedPreferences=getSharedPreferences(SharedPreferencesName,
Context.MODE_PRIVATE);
```

上面的第一个参数是配置文件名称，第二个参数是存储模式，有下面几种：

MODE\_PRIVATE，该配置文件只能被自己的应用程序访问。

MODE\_WORLD\_READABLE，该配置文件除了自己访问外还可以被其它应该程序读取。

MODE\_WORLD\_WRITEABLE，该配置文件除了自己访问外还可以被其它应该程序读取和写入。

MODE\_APPEND，检查文件是否存在，存在就往文件追加内容，否则就创建新文件。

（2）将密钥存储进 SharedPreferences 中。比如这里自定义了一个 savePrivateKey()方法来存储私钥。首先获得 SharedPreferences.Editor 对象 editor，然后在 editor 中使用 editor.putString()方法添加数据，最后使用 editor.commit()进行同步。

```
public void savePrivateKey(String username, String Privatekey){
    SharedPreferences.Editor editor = mSharedPreferences.edit();
    String key = username+"-privateKey";
    editor.putString(key, Privatekey);
    editor.commit();
}
```

（3）从 SharedPreferences 里获取密钥。比如这里自定义了一个 QueryPrivateKey()方法来获取私钥。将 username 修改成“username-privateKey”的格式作为获取私钥的键名，然后使用 sharedPreferences 的 getString()方法得到私钥。

```
public String QueryPrivateKey(String username){
    String key = username+"-privateKey";
    return mSharedPreferences.getString(key, null);
}
```

## 3.2 Web 应用客户端系统的实现

### 3.2.1 Java Web 开发介绍

通常我们使用 Java Web 页面开发的知识来对网页应用系统进行开发。在英语中，Web 的意思是一个网站的页面，这也就是说它是互联网域名上的一个数据源，或者说一个可以供外部世界去访问的地址。在互联网上，对于需要外部世界来访问的 Web 资源可以分为如下几类：静态的 Web 资源（比如 HTML 页面），这里的“静态”指人们访问浏览器页面时呈现的数据始终是相同的；动态的 Web 资源（比如 JSP/Servlet, ASP, PHP 等），这里说的“动态”则表示，人们访问浏览器页面时得到的数据是会收到程序控制的，在不同的时间

点访问时看到的内容也会不尽相同。

一个 Web 应用就是一个程序，这个程序可供浏览器来访问。举例来说，有许多 Web 资源，像 HTML 文件，它们可以用来对外部世界提供服务。与此同时，这些复杂的 Web 资源应该被置于一个目录里，以便组成一个 Web 应用。通常来说，由静态 Web 资源和动态 Web 资源共同组成一个 Web 应用系统，这些资源包括：HTML 文件、CSS 文件、JS 文件、JSP 文件、Java 代码、支持 Java 包、配置文件等等。如果你想要更好得访问外部世界，在完成 Web 应用系统开发之后，你需要递交一份目录，这个目录与 Web 应用系统关联起来，并交由 Web 服务器进行管理。这种映射关系同被我们称为虚拟目录映射过程。

Web 服务器也是一个应用程序，只不过它设置在一种与互联网连接的特殊的计算机上。它的特点是，当浏览器发出请求（request）时，能够做出响应（response），比如提供文档。服务器处理请求并发送文件给浏览器的前提是浏览器要连接到服务器并发送请求命令，这时从服务器送回的文件里会讲明浏览器该做的事，比如如何展示信息在页面上。本次设计使用的是 Tomcat 服务器，它是一个小型的免费服务器，由 Apache 和其他公司联合开发，能够实现 Java EE 的标准，并且因为其稳定性、开源性和先进性收到众多 Java 开发者的青睐。

现代 Java Web 的开发有很多框架可以选择，使用框架，我们可以更简单高效地实现代码的编写。本次设计使用的框架是由 Pivotal 公司开发的 Spring Boot。Spring Boot 使用起来非常简单，只需要三步：

（1）在 pom.xml 中添加相关依赖。比如支持 web 的模块：

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
</dependency>
```

（2）编写 controller.java 文件

Controller 文件用来注册 html 静态文件，通过 @RequestMapping(URL) 将 URL 和 html 文件映射起来。@RequestMapping(URL) 通常标注在某个方法上，当浏览器访问该 URL 时，会执行该方法。

（3）运行项目，启动服务器，在浏览器地址栏里输入 [http://localhost:8080/\\*\\*](http://localhost:8080/**) 即可访问 html 文件了。

### 3.2.2 前端用户交互界面的实现

能在浏览器上显示出来的内容都属于 Java Web 的前端，本次设计需要设计一个供用户交互的前端界面。该界面可以实现的功能是：欢迎用户，将其用户名显示在页面上；用户

可以选择数据密码处理的方式，且能输入待处理的密码数据；显示浏览器与服务器会话的会话标识；生成二维码，将有用的信息和数据存放在二维码里。如图 3.7 所示，为用户交互页面的效果图。

### 数据密码处理

欢迎您: ALICE123@YAHOO.COM      SessionId: CB5E4D8A23C95519B344EEF858310C6C

请选择何种数据处理: ☐加密 ☐解密 ☐数字签名 ☐签名验证

需处理的数据:

数字签名:

default

重置

生成二维码

生成的二维码如下:

图 3.7 数据密码处理的用户页面

html 文件里通常使用表单来收集不同类型的用户输入。表单可以定义一些控件，比如常用的文本框（textarea）、按钮（button）、单选框（radio-buttons）等等，使用这些控件，我们能轻松地与用户交互。

html 语言里还会经常使用到的语言是 JavaScript。在 html 文件里使用 JavaScript 能够控制网页的行为，比如：对事件的反应、改变 html 的内容、验证输入等。使用 JavaScript 语言，我们能轻松获取到表单里输入的信息，并进行其他操作。比如我们需要获取需处理的数据的文本框中的内容时，先通过 document.getElementById()方法获取目标 id 的标签，并存放在 originaldataobj 这个变量里。

```
var originaldataobj = document.getElementById("data");
```

当我们获取到表单里的数据后，需要将其结合成一个字符串并以二维码的形式生成出来。生成二维码时会用到开源代码 Qrcode。添加 Qrcode 的依赖非常简单，只需要通过

jQuery 技术关联一下源代码地址的 URL 即可。

```
<script type="text/javascript" src="//cdn.staticfile.org/jquery/2.1.1/jquery.min.js">
</script>
<script src="//cdn.bootcss.com/jquery/1.11.3/jquery.min.js"></script>
<script type="text/javascript" src=
"//static.runoob.com/assets/qrcode/qrcode.min.js"></script>
```

使用以下步骤可以生成二维码：

（1）document.getElementById()方法获取 div 的 id，并新建一个 QRcode 对象 qrcode。其中 div 是一个标签，用来显示二维码。

```
var qrcode = new QRCode(document.getElementById("qrcode"), {
    width : 300,
    height : 300
});
```

（2）将表单数据组合成一个字符串 dataString

```
var dataString = "username="+username+" ID="+SessionId+" data="
+originaldataobj.value+" dataHandle="+datahandle+" signature="
+signobj.value;
```

（3）使用 qrcode 的 makecode()方法生成二维码，传入的参数 dataString 就是二维码里要显示的内容

```
qrcode.makeCode(dataString);
```

### 3.2.3 登录功能的实现

在进入数据密码处理的用户页面之前，通常需要一个登录页面，以实现用户的验证操作。登录页面是一个 html 表单文件，登录验证则是在 web 密码服务器中实现的。登录页面的效果图如图 3.8 所示。

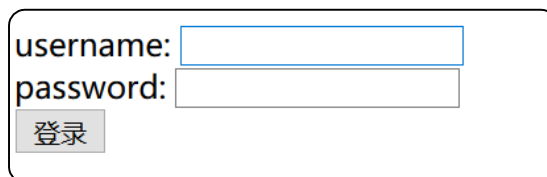


图 3.8 登录页面效果图

登录验证的方法是在 controller.java 里实现的。表单还有一个重要的功能就是提交参数到指定地址，这样便能实现将客户端与服务器的关联。首先获取用户提交参数。



```
String username = request.getParameter("username");
```

```
String password = request.getParameter("password");
```

然后验证用户名和密码，如果验证通过则跳转到 `success.html`。`success.html` 页面即是上文介绍的数据密码处理的用户页面。如果验证失败则跳转到 `fail.html`。

```
if(username.equals("ALICE123@YAHOO.COM")&&password.equals("123456")){
    return "success";
}
return "fail";
```

### 3.2.4 利用 cookie 显示用户名在页面上

cookie 是服务器通知客户端保存键值对的一种技术，客户端有了 cookie 后，每次请求都发送给服务器。因此使用 cookie 我们可以实现将表单里输入的用户名显示在数据密码处理的页面上。

使用 cookie 传递数据的步骤如下：

（1）将用户名以键值对的形式放入 cookie 中，`cname` 是键，`cvalue` 是值。

```
function setCookie(cname, cvalue) {
    document.cookie = cname + "=" + cvalue + ";";
}
```

（2）通过自定义的 `getCookie()` 获取用户名，其中传入的参数 `cname` 是键名，返回值是键值。

```
function getCookie(cname) {
    var name = cname + "=";
    var decodedCookie = decodeURIComponent(document.cookie);
    var ca = decodedCookie.split(';');
    for(var i = 0; i < ca.length; i++) {
        var c = ca[i];
        while (c.charAt(0) == ' ') {
            c = c.substring(1);
        }
        if (c.indexOf(name) == 0) {
            return c.substring(name.length, c.length);
        }
    }
}
```

```
return "";
```

```
}
```

（3）将用户名显示在 id 为 “id01” 的标签上。

```
document.getElementById("id01").innerHTML = username;
```

### 3.2.5 使用 Ajax 在前后端数据交互

表单生成二维码时需要将会话标识也加入进去，但是会话标识只能在服务器里获得，有一种从前端获取后端服务器里数据的技术就是 AJAX。如图 3.9 所示，为 AJAX 的工作原理框图。

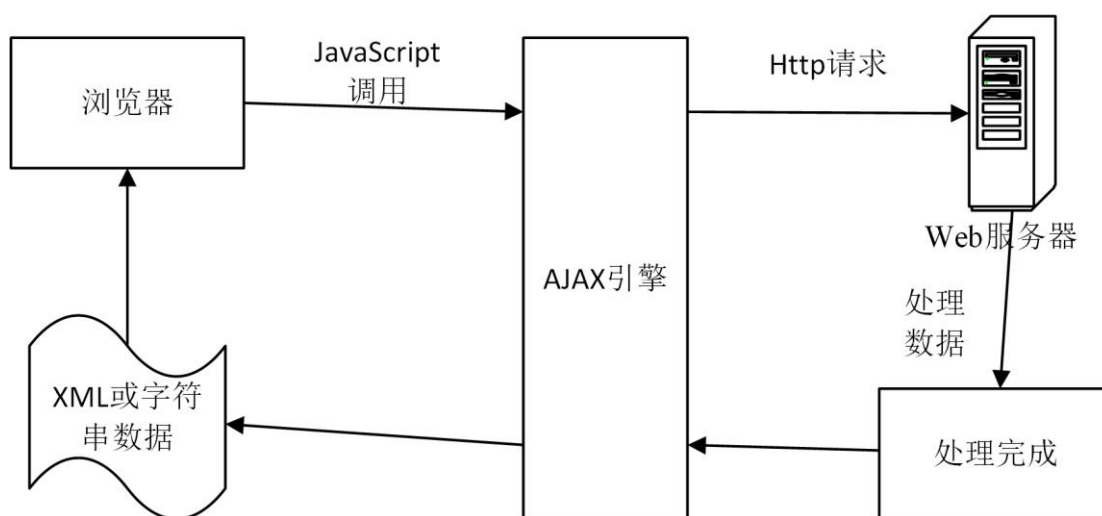


图 3.9 AJAX 工作原理图

异步的 Javascript 和 XML(Asynchronous JavaScript and XML, AJAX)是一种与服务器交互的技术，它能实现将后端的数据返回前端并进行展示，且不需要更新整个页面。

前端的 AJAX 代码是以脚本的形式写在<script>标签里的，其写法有固定的格式，当设置好请求地址 URL 后，就可以向服务器发送请求了。服务器返回的数据存放在 data 里，然后显示在标签上。一下为 AJAX 获取 sessionId 的代码。

```
<script>
$(document).ready(function () {
    $.ajax({
        url:"ajaxServlet",
        type:"get",
        data:"",
        dataType:"text",
        error:function () {
```

```

        },
        success:function (data) {
            $("#id02").text(data);
        }
    })
});
</script>

```

相应 AJAX 的服务器端的代码是写在 Servlet 里的。使用 Servlet 可以收集来自网页表单的用户输入，呈现来自数据库或者其他源的记录，还可以动态创建网页。本设计定义了一个 AjaxServlet.java 文件，专门用来响应来自前端的 AJAX 请求。

在 Spring Boot 里使用 @WebServlet(name = "AjaxServlet",urlPatterns = "/ajaxServlet")来设定 servlet 的请求路径。在 servlet 的 doGet()方法里使用如下的代码获取到会话标识，其中 setContentType()方法为了解决响应的中文乱码问题。

```

response.setContentType("text/html;charset=UTF-8");
HttpSession session = request.getSession(false);
String sessionId = (String) session.getAttribute("sessionId");
然后使用如下的方法将 sessionId 发送回前端浏览器。
PrintWriter writer = response.getWriter();
writer.write(sessionId);

```

### 3.2.6 接收 Android 端数据的实现

上一节介绍了 Android 移动终端用 HttpURLConnection 发送数据到 web 密码服务器的实现。这里介绍使用 web 密码服务器接收数据的实现，定义一个 ServletForGetMethod.java 文件来处理请求。在 servlet 的 doGet()方法里使用如下的代码获取发送过来的数据。

```

使用 request.getParameter()方法获取传送过来的数据。
String TransmittedData= request.getParameter("TransmittedData");
使用 split()方法解析字符串，并将其存放在数组里。
String[] array = TransmittedData.split(" ");
String[] resultArray = array[0].split(":");
String[] idArray = array[1].split(":");
从数组里获取处理结果和会话标识，并打印在控制台上。
String handleResult = resultArray[1];
String sessionId = idArray[1];

```

```
System.out.println("处理结果是: "+handleResult);
System.out.println("sessionId 是: "+sessionId);
```

### 3.3 Android 与 Web 的数据交互的实现

数据的交互是用会话（HttpSession）来实现的。浏览器的一个页面里可以有多个请求（request），通过 request.setAttribute("key",value)方法传入键值对参数，可以将数据通过 request 发送出去。在同一 request 下，通过 request.getParameter("key")可以得到 value。如果不是同一 request，则无法获得 value。想要在不同的 request 下通过同一个 key 获得 value，只要在一个页面内，则可以通过 HttpSession 实现。在服务器中，系统会为每一个会话维护一个 session。不同的会话，对应不同的 session。系统是如何识别各个 session 对象的呢？分为如下几步：

#### （1）写入 session 列表

当用户发送第一次请求的时候，在 servlet 里先获取 session，然后往 session 里放属性。底层服务器看到你用了 session，会马上生成一个 32 位长度的随机字符串，然后再创建一个 session 对象。然后以这个 32 位长度的随机字符串作为 key，以新创建的 session 对象作为 value 放在 session 列表中，session 列表是一个 map。如表 3.1 所示。

表 3.1 服务器的 session 列表

key	value
D329C89C88DEC2189888FDF365961A87	HttpSession 的引用 1
94C9C89C88EDC21898188FDF36591F82	HttpSession 的引用 2
.....	.....
E2A9C89C88DEC2189838FDF36596189C	HttpSession 的引用 n

#### （2）服务器生成并发送 cookie

服务器同时还做了一个工作，即把这 32 位长度的随机字符串包装成了一个 cookie，cookie 的 key 是 sessionId，value 就是 32 位长度的随机字符串。以在发送回客户端的响应中，报头里会添加 cookie，这样便实现了 cookie 的发送。

#### （3）客户端接收并发送 cookie

浏览器接收到 cookie 后，会将 cookie 保存到客户端浏览器的缓存中。当客户端再次发送请求时，会首先将浏览器缓存的 cookie 里的 sessionId 放在请求的头部信息中，并发送给服务器。

（4）从 session 列表中查找

服务器接收到 cookie 后，会拿到 sessionId 的值，然后到 session 列表中查找，并找到对应的 session，然后就能从 session 里读数据了。

### 3.4 本章小结

本章介绍了基于二维码和移动终端的 SM2 密码数据处理的系统的具体实现，可以分为三个子系统部分，分别是 Android 移动终端子系统的实现、Web 应用客户端子系统的实现以及 Android 与 Web 的数据交互子系统的实现。

## 第 4 章 系统测试与分析

### 4.1 Android 移动终端的测试

使用数据线连接手机和个人 PC，在 Android Studio 里进行应用调试，会显示个人手机的调试信息，如图 4.1 所示。

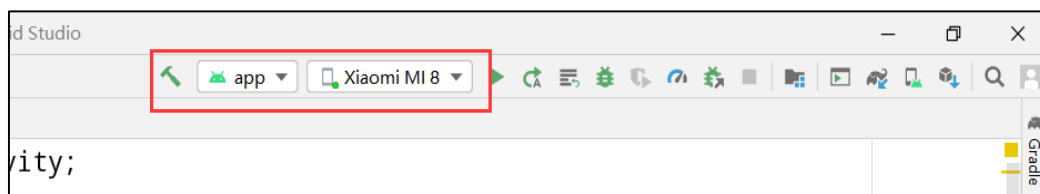


图 4.1 Android Studio 调试界面

在手机里进入提前安装好的 Android 移动终端的应用程序 AndroidMobileClient，会显示扫码页面，如图 4.2 所示。点击“scan now”按钮，会进入扫码窗口，如图 4.3 所示。

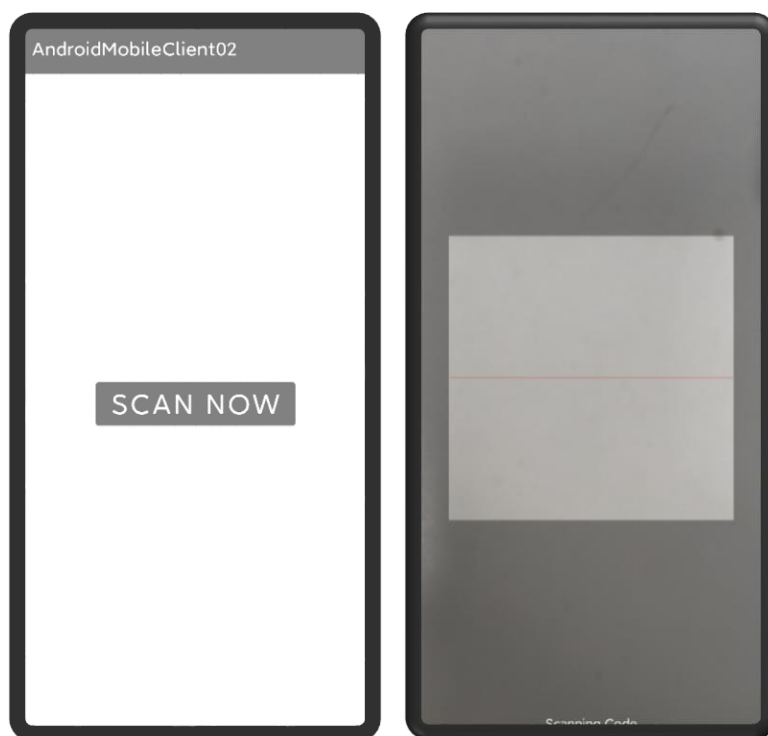
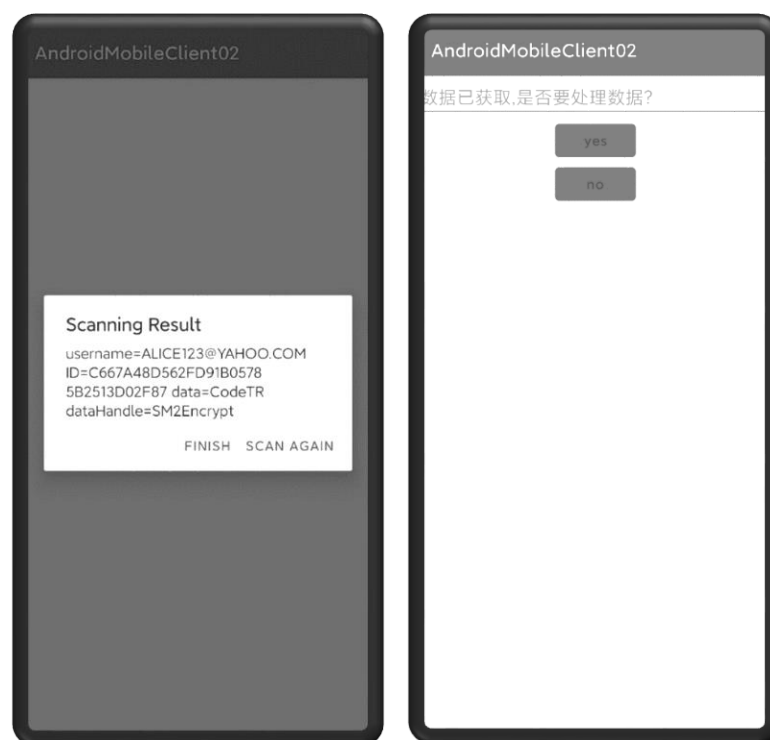


图 4.2 Android 端应用程序扫码页面 图 4.3 Android 端应用程序扫码窗口



4.4 Android 移动终端获取到的扫描结果 图 4.5 Android 移动终端的数据处理页面

我们使用 Web 应用客户端生成的二维码，对二维码扫描，可以得到扫描结果，如图 4.4 所示。可以看到，扫描结果是一个对话框，里有 **username**，**ID**，**data**，**dataHandle** 这四种数据，还有“Finish”和“Scan Again”两个按钮。我们点击“Scan Again”按钮后，会重新返回到图 4.3 所示的二维码扫码窗口；我们点击“Finish”按钮后，会进入数据处理页面，如图 4.5 所示。

数据处理页面有三个组件，分别是一个文本框，和两个按钮。文本框用来显示数据处理的结果，其默认值已经给用户提示，提示其是否要处理数据；当用户不想处理数据时，需要点击“no”按钮，此时页面会返回到最初的图 4.2 所示的扫码页面；当用户想要处理数据时，需要点击“yes”按钮，此时页面会执行两项操作：首先，文本框内会显示用户密码数据的处理结果；然后，页面会弹出一个提示，提示是否已成功将处理结果发送给了 Web 密码服务器。如果发送成功，会提示“Succeeded to save”，如果发送失败，会提示“Failed to save”。如图 4.6 所示，展示的是数据处理结果以及发送成功的页面。



图 4.6 Android 端应用程序数据处理页面的结果

至此，Android 应用客户端的所有功能均已进行了实验测试。经验证，数据的密码处理算法确实为 SM2 国密体制，且能成功实现加密、解密、数字签名和签名验证四种密码功能。系统有关 Android 移动终端的需求得以成功实现。

## 4.2 Web 应用客户端的测试

进入 IntelliJ Idea，打开按钮启动程序调试，如图 4.1.7 所示。

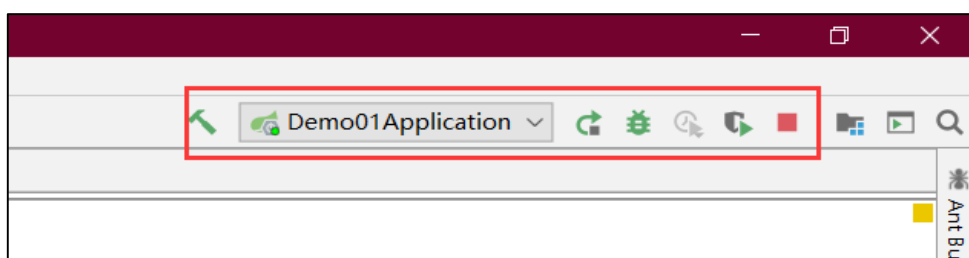


图 4.7 IntelliJ Idea 启动项目调试

打开火狐浏览器，在浏览器的地址栏里输入 <http://localhost:8080/login>，则可以进入到 Web 应用客户端的登录页面，如图 4.8 所示。登录页面是一个表单，有两个文本框和一个按钮，分别用于输入用户名和密码，点击登录。



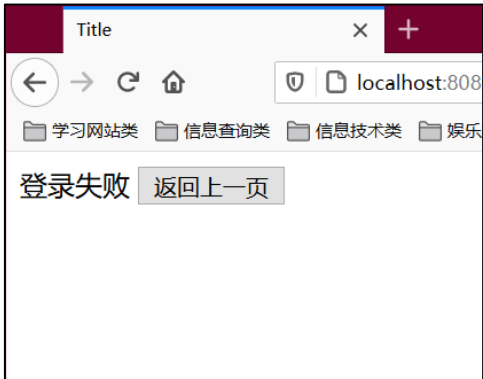
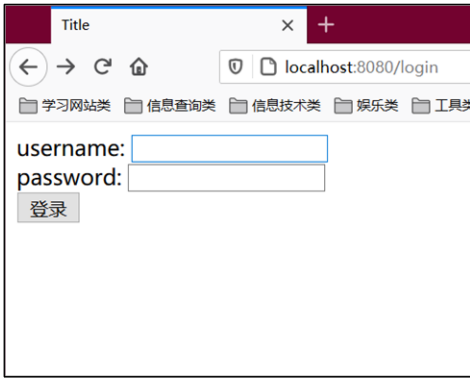


图 4.8 Web 应用客户端的登录页面      图 4.9 登录失败页面

当我们输入用户名和密码后，点击登录按钮，会有两种结果：如果登录失败，会跳转到失败页面，如图 4.9 所示。此时页面会提示用户登录失败，用户可以点击“返回上一页”按钮重新返回到图 4.8 所示的登录页面。如果登录成功，则会跳转到成功页面，如图 4.10 所示。

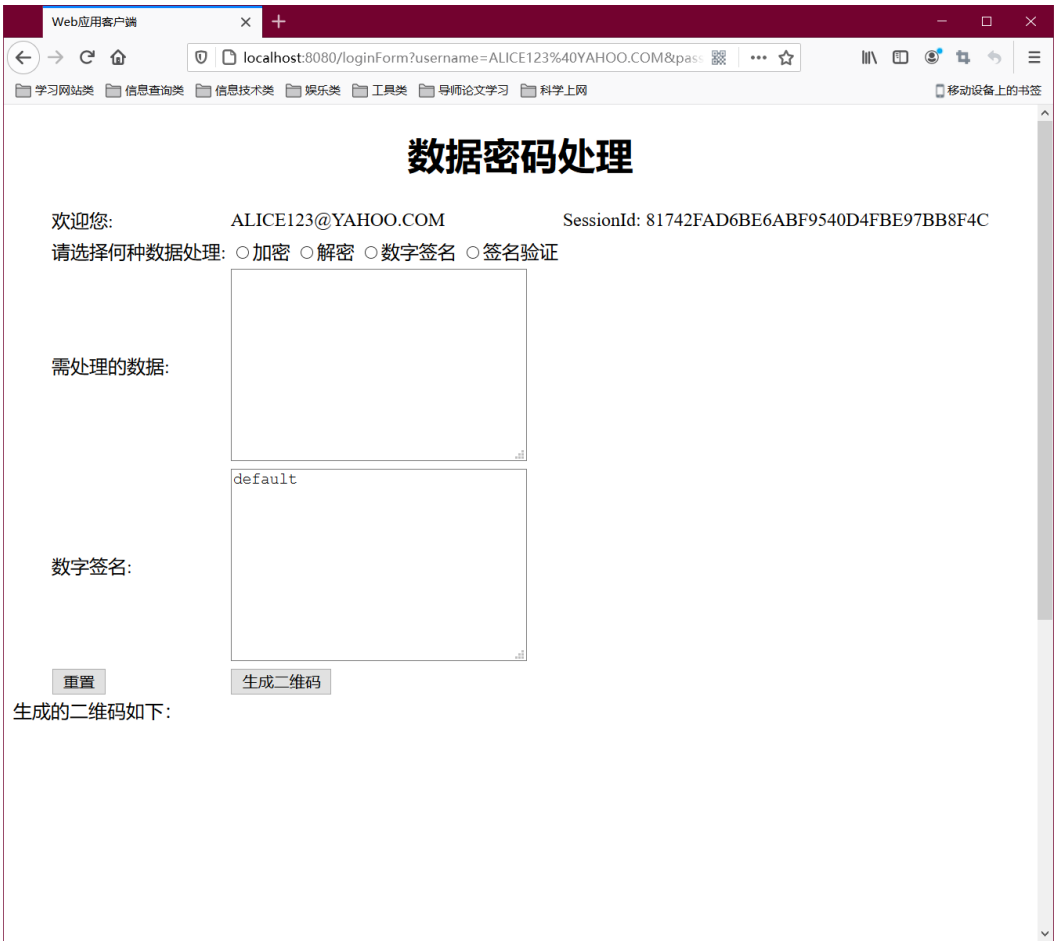


图 4.10 登录成功页面

登录成功页面是一个表单，标题是“数据密码处理”。表单里有如下的一些组件：第一行是文本框，用来显示登录用户的用户名和浏览器与服务器之间会话的会话标识(SessionId)；

第二行是单选框，用户可以选择进行何种密码处理，比如加密、解密、数字签名或签名验证；

第三行是文本输入框，用户在这里输入需处理的数据；

第四行也是文本输入框，当用户选择签名验证这种处理方式时，需要在这里输入待验证的数字签名；

第五行是两个按钮，名称是“重置”和“生成二维码”，它们分别代表的功能是刷新当前页面和根据用户输入的内容生成二维码。当用户输入了相关的信息后，可以生成二维码，如图 4.11 所示。

Web应用客户端

localhost:8080/loginForm?username=ALICE123%40YAHOO.COM

### 数据密码处理

欢迎您: ALICE123@YAHOO.COM SessionId: 81742FAD6BE6ABF9540D4FBE97BB8F4C

请选择何种数据处理: ☐ 加密 ☐ 解密 ☐ 数字签名 ☒ 签名验证

codeTR

需处理的数据:

数字签名:

30440220775850C50350D605B4B13B87  
A980E5609C9FF9260E1F975F20FC721B  
66277CB7022040F2040E15E25A3A72E1  
53B868C3AB19AD83FAB0445E2DA5696F  
E0781AA0D6F7

重置 生成二维码

生成的二维码如下:

图 4.11 登录成功页面成功生成的二维码

此时二维码里会包含如下信息：username，ID，data 和 dataHanlde，当用户选择的密码处理方式为签名验证时，还会包含 signature。

至此，有关 Web 应用客户端前端的功能均已进行了全部测试，系统需求分析有关 Web 应用客户端的需求得以成功实现。

### 4.3 进行数据发送的测试

下面我们分别对四种密码处理方式进行测试。

（1）数据加密。在 Web 应用客户端的数据密码处理页面点击加密选项，在需处理的数据文本框里输入英文字符，如“codeTR”，然后点击生成二维码按钮，页面会生成一个二维码。在 Android 移动终端里扫描该二维码，获取到相关数据后进行密码处理，处理成功后发送到 Web 密码服务器的控制台上。发送的结果如图 4.12 所示。



图 4.12 Web 密码服务器接收到的加密后数据

观察到，控制台上显示着四行信息，前两行为服务器登录成功后输出的字符串，分别是用户名 username=ALICE123@YAHOO.COM 和会话标识 sessionId；后两行为服务器接收到的数据，分别为处理结果和会话标识。经验证，该结果与 Android 端密码处理的结果一致，说明数据发送成功。

（2）数据解密。在 Web 应用客户端的数据密码处理页面点击解密选项，在需处理的数据文本框里输入数据加密生成的密文，然后点击生成二维码按钮，页面会生成一个二维码。在 Android 移动终端里扫描该二维码，获取到相关数据后进行密码处理，处理成功后发送到 Web 密码服务器。发送的结果如图 4.13 所示。



图 4.13 Web 密码服务器接收到的解密后数据

同样的，通过后两行信息可以观察到解密的结果是“codeTR”，说明数据发送成功，解

密功能成功实现。

（3）数字签名。在 Web 应用客户端的数据密码处理页面点击数字签名选项，在需处理的数据文本框里输入“codeTR”，然后点击生成二维码按钮，页面会生成一个二维码。在 Android 移动终端里扫描该二维码，获取到相关数据后进行密码处理，处理成功后发送到 Web 密码服务器。发送的结果如图 4.14 所示。



图 4.14 Web 密码服务器接收到的数字签名

（4）签名验证。在 Web 应用客户端的数据密码处理页面点击签名验证选项，在需处理的数据文本框里输入“codeTR”，在数字签名文本框里输入待验证的数字签名，然后点击生成二维码按钮，页面会生成一个二维码。在 Android 移动终端里扫描该二维码，获取到相关数据后进行密码处理，处理成功后发送到 Web 密码服务器。发送的结果如图 4.15 所示。

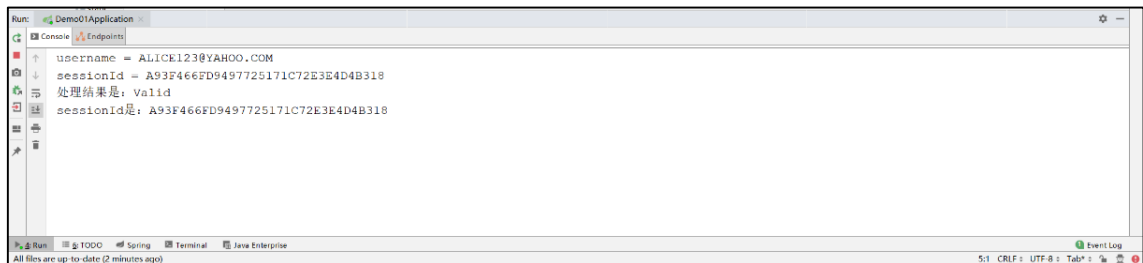


图 4.15 Web 密码服务器接收到的签名验证

可以观察到验证的结果是“Valid”，说明数字签名和签名验证功能成功实现。

## 4.4 本章小结

本章对设计好的系统进行了功能的测试，并分析了测试结果。通过分析结果，我们得出结论，该系统能很好的满足预期。

## 第5章 总结与展望

### 5.1 总结

本文设计了一种在浏览器上对数据进行密码操作的新方法，即使用二维码技术和 Http 网络技术将数据转移到移动终端，通过调用存放在用户移动终端里的用户密钥，对数据进行密码操作。这样能够实现不使用浏览器插件和 USB key 等硬件对数据进行密码操作的需求。

第一章主要研究了题目的研究背景和意义，国内外的研究现状。通过分析移动设备在当今和未来的应用现状和使用前景，对实现本题目的必要性和可能性做了一个研究。第二章实现了本系统的需求分析，并提出了系统的整体设计方案。第三章完成了系统的具体实现。整个系统可以分为两个平台，Web 应用客户端和 Android 移动终端。Android 移动终端是本设计的重点，其主要功能有三个：获取数据、处理数据和发送数据。Web 应用客户端的设计分为前端的用户交互界面和后端的密码服务器。前端的用户交互页面能够实现登录页面，用户密码处理页面和生成二维码的功能；后端的密码服务器则负责访问的控制、登录验证、接收 Android 移动终端发送的数据等功能。第四章完成了系统的测试与分析，通过分析测试结果，得出题目研究的最终结论。

本设计的创新点在于，使用的密码处理算法为 SM2 国密体制，相比目前已有的 RSA 密码算法，能够实现更高的安全性。

### 5.2 展望

本次设计实现了基于二维码和移动终端的 SM2 密码数据的处理研究，但其中仍有些地方还不够完善：

首先，本次设计不论是 Android 移动终端还是 Web 应用客户端，只对逻辑功能进行了实现，页面并不够美观，后续应使用更美观的布局文件进行渲染；其次，使用二维码进行数据的传递时，数据量不能太大，且只能发送英文字符，后续应采用一种存储量更大、兼容性更好的技术；再者，移动终端存储的密钥为事先生成的密钥对，没有实现随机生成，今后应使用能够随机生成密钥的密码库；最后，Web 密码服务器在接收到移动终端发送来的数据后仅简单地显示在了控制台上，没有做成成型的系统，今后可以做一个数据显示页面将数据显示出来。

## 致 谢

时光如梭，四年的本科生时光就要结束了。这四年不仅让我在知识技能上得到了成长，也教会了我如何做人，因此我想在这里表达感谢。

首先我要感谢父母，没有他们的辛勤工作，我或许现在不会坐在教室里念书，谢谢他们，一直在背后默默鼓励我；也谢谢他们，在我面临人生的抉择时为我指点迷津。

其次我要感谢我的导师，龙毅宏教授。在毕业设计的期间让我学到了很多，给予了我很多的指导，每次不懂的问题都会细心的讲解，直到问题解决。龙老师每周都会开一次组会，在会上所有组成员都要汇报自己的研究进展，通过这种方式，我们养成了规律学习的习惯，保证了毕业设计能顺利的完成。因此我再一次谢谢龙老师在这几个月里对我的帮助。

最后要感谢和我一批做毕设的同学，每次做设计遇到问题同学们都会跟我讨论，一起解决问题，我能够完成毕业设计，也离不开他们的指导和帮助。

天下没有不散的宴席，到了离别的时间，有很多的不舍，有很多的祝愿，希望老师们身体健康，希望同学们前程似锦，希望未来的我更加的优秀！

## 参考文献

- [1] 陈慧.基于移动终端的密码数据处理与交换方法的设计与实现[D], 武汉: 武汉理工大学, 2016.05.
- [2] Christof Paar,Jan Pelz 著.马小婷译.深入浅出密码学[M].北京: 清华大学出版社,2012.
- [3] Preeti Sharma, M.Tech Student. Elliptic Curves and their Applications in Cryptography[J]. International Journal of Engineering Research & Technology (IJERT). April-2017, Vol.6 Issue 04, pp.963-969.
- [4] 国家密码管理局. GM/T 0003-2012. SM2 椭圆曲线公钥密码算法[S]. 2012.
- [5] 龙毅宏. 一种 Web 应用中基于移动终端的密码数据处理与交互方法. 中国, 201510196956.8[P]. 2015-4-23.
- [6] Mary Meeker's. Internet Trends Report[DB], 2019.
- [7] 王刚.基于信息安全的计算机网络应用分析[J].电脑编程技巧与维护,2021(05):157-158.
- [8] 罗军舟,吴文甲,杨 明.移动互联网: 终端、网络与服务[J]. 计算机学报, Nov.2011, Vol.34 No.11.
- [9] 朱为朋.移动终端及其数据加密方法. 中国专利, CN201410740502.8[P].2015-3-25.
- [10]王艳敏,李永忠,吕少伟.Android 平台下文件透明加密技术的研究与实现[J].计算机技术与发展,2014,24(09):137-140.
- [11]陈炯. QRcode 编解码技术的研究与实现[D]. 西安电子科技大学, 2012.
- [12]郑东,赵庆兰,张应辉.密码学综述[J].西安邮电大学学报,2013,18(06):1-10.
- [13]何锬.数据加密技术在计算机网络信息安全中的应用分析[J].通讯世界,2019,26(04):40-41.
- [14]秦志光.密码算法的现状和发展研究[J].计算机应用,2004(02):1-4.
- [15]葛晓健.基于 Android 的智能收银终端设计与应用[J]. 计算机与网络. 2019(11)
- [16]肖承望,张宇,黎惟梁.计算机网络通信安全中数据加密技术的应用研究[J].科技风,2019(36):71.
- [17]叶小艳.一种 AES 算法和 HASH 认证结合的文件加密方案[J]. 计算机技术与发展. 2019(03).
- [18]程雨芊,李智超.基于余数系统蒙哥马利模乘器的 RSA 密码算法[J]. 计算机仿真. 2021(01).
- [18]A. Nadeem, M. Y. Javed. A Performance Comparison of Data Encryption Algorithms[C]. 2005 International Conference on Information and Communication Technologies,2005,pp.84-89.
- [19]范恒英, 何大可, 卿铭.公钥密码新方向:椭圆曲线密码学[J]. 通信技术, 2002, 000(007):82-84.
- [20]Wang Q. The Application of Elliptic Curves Cryptography in Embedded Systems[C]. International Conference on Embedded Software and Systems. IEEE Computer Society, 2005: 527-530.
- [21]孙荣燕,蔡昌曙,周洲等.国密 SM2 数字签名算法与 ECDSA 算法对比分析研究[J]. 网络安全技术与应用, 2013(02):60-62.

[22]陈泽凯.一种基于身份的 SM2 椭圆曲线数字签名方案及其在离线文档认证中的应用[D]. 2016.

[23]Ning W,Liming L, Yanzhang W.Research on the Web Information System Development Platform Based on MVC Design Pattern[J]. IEEE, 2011, 3:203-206.