

# 北京理工大学

## 本科生毕业设计(论文)

基于国产密码算法的云计算网络信息传输认证系统设计与实现

Design And Implementation Of Information Transmission And Authentication System In Cloud Computing Network Based On Domestic Cryptography Algorithm

学    院:	计算机学院
专    业:	软件工程
学生姓名:	侯添久
学    号:	1120161912
指导教师:	闫怀志

2020 年 1 月 3 日

# 北京理工大学

## 本科生毕业设计（论文）任务书

题目类别：毕业论文

题目性质：理论研究

### 基于国产密码算法的云计算网络信息传输认证系统设计与实现

**Design And Implementation Of Information Transmission And Authentication System In Cloud Computing Network Based On Domestic Cryptography Algorithm**

学 院：计算机学院

专 业：软件工程

学生姓名：侯添久

学 号：1120161912

指导教师：闫怀志

### 题目内容:

本课题主要进行基于国产密码算法的网络信息传输认证系统原型设计与实现工作。国产密码算法是中国自主设计并实现的成熟加密算法系列，未来将在我国信息系统的安全设计和实现中获得广泛应用。本课题拟根据具体信息系统中的网络信息传输认证需要，对已有的国产密码算法的具体实现进行分析和研究，选择适用的对称加密、非对称加密以及消息摘要算法，实现云计算信息系统中的网络信息传输认证，并进行实际系统的应用效果分析与验证。

### 任务要求:

1、了解云计算网络信息传输认证和国产密码算法相关应用领域背景知识，了解国内外行业标准、规范和技术发展趋势，了解相关行业的政策和法律法规；

2、在指导教师指导下阅读国内外文献和相关知识，对已有的云计算网络信息传输认证和国产密码算法进行安全性分析和研究，根据云计算网络信息传输认证原理理解云计算安全保护的实现原理和方法。结合对现有云计算网络信息传输认证的研究，实现国产密码算法在其中的具体应用，最后在实际生产环境下进行分析与验证；

3、需要解决的问题有：A、云计算安全保护需求；B、云计算网络信息传输认证的实现原理与方法；C、实现对云计算网络信息传输认证的国产密码算法适配分析；D、实现一个简要的云计算信息系统中的网络信息传输认证的原型；

4、开发环境：操作系统：Windows 7 及以上和 Linux 发行版；开发语言：C、Java、Python 等，可根据实际系统选择；

5、完成毕业设计论文并提交相关文档；

6、毕业设计进度安排：

a. 查阅相关资料，完成基础知识的积累。（第 1 周-第 2 周）

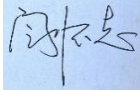
b. 研究分析云计算网络信息传输认证的基本流程。（第 3 周-第 4 周）

c. 根据已有的云计算网络信息传输认证解决方案，实现基于国产密码算法的云计算网络信息传输认证系统设计与实现工作。（第 5 周-第 10 周）


d. 对云计算网络信息传输认证进行安全性分析，并验证其实际效果。（第 11 周-第 12 周）

e. 完成毕业论文，提交论文及相关文档。（第 13 周-第 15 周）

f. 完成本科生毕业论文答辩。（第 16 周）

指导教师签字:  \_\_\_\_\_

2020 年 1 月 3 日

教学单位负责人签字:  \_\_\_\_\_

2020 年 1 月 4 日

责任教授签字:  \_\_\_\_\_

2020 年 1 月 5 日

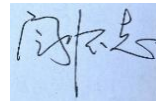
## 毕业设计（论文）评语表（一）

指导教师对毕业设计（论文）的评语：

该同学从文献综述入手，对云计算网络传输问题，研究并实现了对云计算系统的云计算网络传输设计，设计对云计算的设计与实现也进行了详细论述。

该同学工作认真，有较强的独立工作能力，按照计划进行，较好地完成了毕业设计各指标点要求，毕业设计合格。论文写作规范，结构清晰。论文中还对云计算设计进行了详细论述，建议继续深入研究。

指导教师

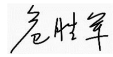
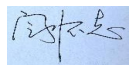


（签字）

2020 年 6 月 11 日



## 毕业设计（论文）评语表（二）

答辩委员会（小组）成员			
姓 名	职称	主要分工	签 字
危胜军	副教授	主任	
戴银涛	副教授	委员	
田东海	讲师	委员	
闫怀志	副教授	委员	
孙建伟	讲师	委员	
林中	研究员	委员	

答辩中提出的主要问题及回答的简要情况:

问题一: 你的系统是如何设计的?

回答: 分为客户端与服务端, 客户端使用相应国产密码算法加密, 服务端对密文处理, 将密文结果返回客户端, 客户端解密拿到数据, 采用 socket 网络传输。

问题二: 解释下什么是同态性?

回答: 密码算法同态是指使用密文进行计算处理所得到的结果与直接使用明文一样。典型的如 RSA, 满足乘法同态。

问题三: SM2, SM3, SM4 是国产密码算法吗?

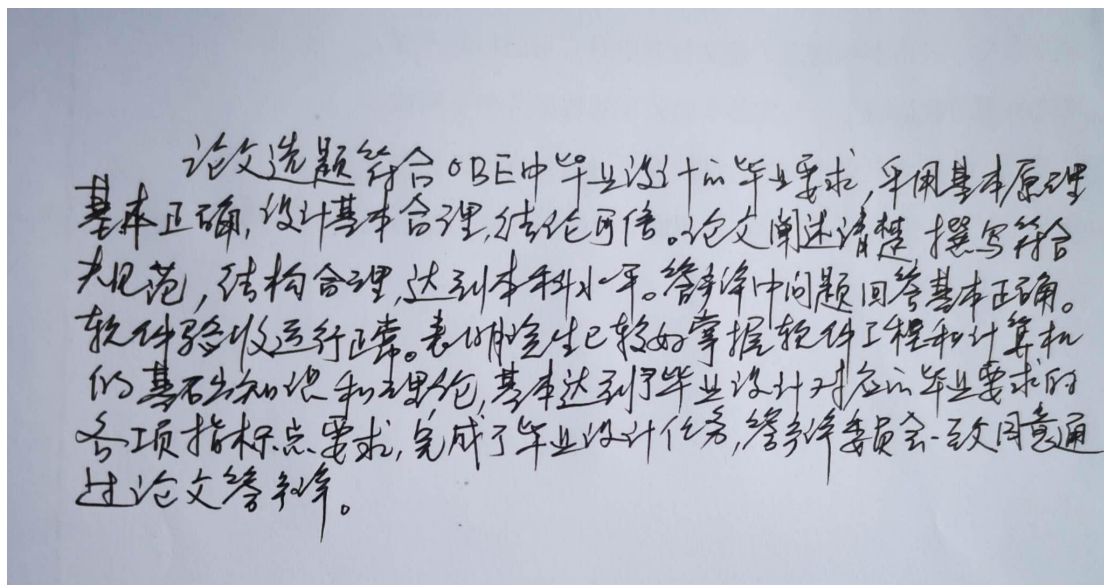
回答: 根据国密标准, SM2、SM3、SM4 都是我国自主研发设计的, 属于国产密码算法。

答辩委员会 (小组) 代表 戴和兴 (签字)

2020 年 6 月 18 日



答辩委员会（小组）的评语：



答辩委员会（小组）代表 危胜军 （签字）

2020 年 6 月 18 日

答辩委员会（小组）给定的成绩：

中

答辩委员会（小组）主任 危胜军 （签字）

2020 年 6 月 18 日

毕业设计（论文）开始日期 2020 年 1 月 3 日

截止日期 2020 年 6 月 20 日

毕业设计（论文）答辩日期 2020 年 6 月 18 日

## 原创性声明

本人郑重声明：所呈交的毕业设计（论文），是本人在指导老师的指导下独立进行研究所取得的成果。除文中已经注明引用的内容外，本文不包含任何其他个人或集体已经发表或撰写过的研究成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。

特此申明。

本人签名：侯添久

日期：2020年6月10日

## 关于使用授权的声明

本人完全了解北京理工大学有关保管、使用毕业设计（论文）的规定，其中包括：①学校有权保管、并向有关部门送交本毕业设计（论文）的原件与复印件；②学校可以采用影印、缩印或其它复制手段复制并保存本毕业设计（论文）；③学校可允许本毕业设计（论文）被查阅或借阅；④学校可以学术交流为目的，复制赠送和交换本毕业设计（论文）；⑤学校可以公布本毕业设计（论文）的全部或部分内容。

本人签名：侯添久

日期：2020年6月10日

指导老师签名：

闫志

日期：2020年6月10日

# 基于国产密码算法的云计算网络信息传输认证系统设计与实现

## 摘 要

随着云计算的发展，越来越多的应用都在使用云计算，然而，数据的隐私安全也变得格外重要，作为云计算服务的使用者，他们不希望自己的数据服务提供者所得到，只是借助云计算强大的计算和存储能力来帮助完成一些任务。如果服务使用者发送密文数据到相应的云上，云服务对密文进行运算，得到的密文结果再发送给服务使用者，服务使用者解密拿到的结果和直接使用明文进行相同的操作得到的结果是相同的，若密码算法具备这种性质，则称为同态加密算法，这也是目前解决云计算数据隐私安全最强有力的手段。

本课题主要对云计算环境下的网络信息传输认证系统的设计与实现，信息传输使用的国产密码算法需要具有同态性。主要研究的国产密码算法是非对称加密算法 SM2，对称密码算法 SM4，数字摘要算法 SM3 三种密码算法，先通过实验确定其是否具有同态性，再通过时间衡量其加解密的效率，最后选择合适的算法以及模式设计传输认证系统。

研究主要通过 socket 通信，通信双方分为发送者和接收者模拟用户和云服务器，发送者为用户端，接收者为服务端，对数据的加密操作在用户端进行，用户端再将加密后的数据通过网络发送给服务端，服务端对客户端传递来的数据进行某种计算，并且将得到的密文结果返回给客户端，客户端解密后拿到结果，再判断与使用明文进行相应计算得到的结果是否相同。通过此种方式模拟用户数据传输到云服务器，云服务进行只进行相应的计算操作，客户端拿到密文后解密得到明文的结果。

研究最后得出在国产加密算法 SM4 的 ECB 模式下，使用分割字符串的方法拿到的结果解密之后与明文运算的结果相同。

**关键词：国产密码算法；同态加密技术；云计算；网络通信**

# **Design And Implementation Of Information Transmission And Authentication System In Cloud Computing Network Based On Domestic Cryptography Algorithm**

## **Abstract**

With the more and more scene use cloud computing, but the security of data has also become an very important problem. As users of cloud computing services, they do not want their data to be gotten by cloud computing providers, and they only want to use the powerful computing of cloud computing to help complete some problems. If the service users send the ciphertext data to the remote cloud, and the cloud service calculates result using the ciphertext and sends the ciphertext result to user. Users decrypt ciphertext result and get the plaintext result. It is the same as execute those operation on the plaintext directly. The cipher algorithm with this property is called homomorphic encryption algorithm. This is one powerful way to solve the problem of data security on the cloud.

In this paper, we mainly study how to design and implementation of transmission authentication system. Homomorphism is required for the domestic cryptographic algorithms used in information transmission. We mainly study some existing domestic cryptographic algorithms. They are SM2, SM3, and SM4. Firstly, we determine whether it has homomorphism by experiment, then measure the efficiency of encryption and decryption by time, and finally choose the appropriate algorithm and mode to design the transmission authentication system.

In this paper, I use socket to communication. Coding the socket of client and server independently, then encrypting the data in client, then client sends the encrypted data to server. Server calculates result, and returns the encrypted result to the client. Client decrypts the result, then judges if it is the same as using the corresponding operation in plaintext. In this way, It is simulated to transmit data to the cloud server, and the cloud service only performs the corresponding computing operation. The client gets the ciphertext result and decrypts it to get the plaintext result.

At last, it is concluded that under the ECB mode of the domestic encryption algorithm SM4, the result by using the method of splitting string is the same as result

using plaintext to operate.

**Key Words: Domestic Cryptography Algorithm; Homomorphic Encryption; Cloud Computing; Network Communications**

## 目 录

摘 要.....	I
Abstract.....	II
第一章 绪论.....	1
1.1 研究的背景与意义.....	1
1.2 研究的主要内容与方法.....	2
1.3 组织结构.....	2
1.4 本章小结.....	3
第二章 密码算法同态性及网络.....	4
2.1 密码算法简介.....	4
2.1.1 对称密码算法.....	4
2.1.2 非对称密码算法.....	5
2.1.3 数字摘要算法.....	5
2.1.4 应用.....	6
2.2 加密算法的同态性.....	8
2.2.1 完全同态加密技术.....	8
2.2.2 部分同态加密技术.....	8
2.3 加密模式介绍.....	9
2.3.1 ECB 模式.....	9
2.3.2 CBC 模式.....	10
2.4 网络通信介绍.....	10
2.4.1 传输层通信.....	11
2.4.2 数据传输过程.....	12
2.5 本章小结.....	13
第三章 国产密码算法与通信实现.....	14
3.1 国产密码算法实现.....	14
3.1.1 对称密码算法实现.....	14

3.1.2 非对称密码算法实现.....	15
3.1.3 数字摘要算法实现.....	16
3.2 网络通信实现.....	18
3.3 本章小结.....	19
第四章 国产密码算法同态性分析.....	21
4.1 方案设计.....	21
4.2 SM2 同态性分析.....	21
4.2.1 SM2 实验.....	21
4.2.2 SM2 结果分析.....	25
4.3 SM4 同态性分析.....	25
4.3.1 ECB 模式实验.....	26
4.3.2 CBC 模式实验.....	27
4.3.3 SM4 结果分析.....	28
4.4 密码算法同态性结果对比分析.....	29
4.5 本章小结.....	29
第五章 密码算法的性能对比.....	30
5.1 性能方案设计.....	30
5.2 SM2 加解密时间分析.....	30
5.3 SM3 加解密时间分析.....	33
5.4 SM4 加解密时间分析.....	34
5.4.1 ECB 模式加解密时间分析.....	34
5.4.2 CBC 模式加解密时间分析.....	35
5.5 密码算法性能分析.....	37
5.6 本章小结.....	37
第六章 传输认证系统设计与实现.....	38
6.1 实现方案.....	38
6.2 系统架构.....	38
6.3 实验分析.....	39
6.4 本章小结.....	40

## 北京理工大学本科生毕业设计（论文）

---

结 论.....	41
参考文献.....	42
致 谢.....	43



## 第一章 绪论

### 1.1 研究的背景与意义

近二十年来, 云计算技术的高速发展, 使得我们的生活发生了翻天覆地的变化, 并且也对许多传统行业影响是深远的。云计算的发展给我们带来方便的同时, 也随之带来了严峻的隐私数据安全问题。近年来, 云计算时常发生严重的安全问题, 其中用户隐私数据安全态势显得格外严峻。

2018年, 国外知名互联网公司Facebook被曝出发生数据泄漏事件, 3月Facebook上5000万名用户个人信息遭一家名为剑桥分析公司的泄露; 在9月Facebook上有大约3000万用户信息泄露, 导致这次是事件的原因是安全系统存在一定的漏洞, 而该漏洞被黑客攻击; 在12月Facebook上有大约有6亿人的私隐照片被泄露。

2019年8月, 美国的大型商业银行Capital One发生数据泄露, 黑客利用他们基础设施中一个特定的配置漏洞, 获得了约1亿美国人以及600万加拿大人的基本信息, 这些信息涵盖了2005年至2019年初的个人信息, 其中包括大约14万个社会安全号码和8万个关联银行账号。

由上述众多数据泄露事件可以看出, 在云计算, 大数据等新的科学技术越来越多的应用在我们平时的生活中, 其所服务的用户数据量回事越来越多, 相应地用户的隐私数据被泄露的可能性也随之增高。随着云计算成为众多企业的首选数据存储处理方案, 其应用场景也是一直在变化, 云计算服务提供者需要保证在任一场景下用户隐私数据都是安全的, 因此云计算不得不面对一个巨大的难题, 即如何在保证用户数据安全的基础上进行相应的数据处理操作。一般地, 我们为了使得数据更加安全, 往往会对用户数据进行加密, 然后将加密后的密文发送给云服务商, 云服务商得到的只会是数据密文, 但是用户数据传输到云服务商处是需要对其进行相应的计算操作, 因为其是密文无法进行相应的操作, 即使进行计算, 也会使得得到的结果无法正确解密亦或得不到预期结果。这种情况下, 云服务只是单单存储数据, 并没有用到其巨大的计算能力。如果使的加密算法具有同态性, 就可以在不解密情况下对密文进行任意的计算操作, 此时对密文的处理等价于对明文的处理, 用户拿到密文结果解密后依然会得到预期明文结果, 通过同态密码算法可以解决目前云计算中的用户隐私数据安全问题。如此云的算力可以很好地发挥出来, 同时又解决了用户隐私数

据安全问题。

## 1.2 研究的主要内容与方法

本课题研究国产加密算法在云计算网络信息传输认证的设计与实现，主要是基于现有的国产密码算法，通过实验判断其是否具有同态特性，再选择合适的密码算法对传输认证系统进行设计与实现。密码算法的同态性是指对于密文进行某种操作，得到相应的密文结果，对密文结果进行解密后得到的结果与直接使用明文进行运算得到的结果是一样的，即对密文的操作等价于在明文上进行相同的操作，满足此类性质的密码算法称为加密算法的同态性。

当在云计算中使用的加密算法具有同态性时，我们可以在客户端对数据进行加密，将密文发送给云服务，云服务根据密文进行相应的计算，返回给我们需要的密文结果，客户端对收到的密文解密。整个过程对于云服务来说，都是在对密文进行操作，不需要进行任何的解密，也无法拿到密文对应的明文，即使数据泄露也可以保证数据的隐私安全，所以密码算法的同态性可以保证我们隐私数据的安全。

研究国产密码算法的同态性，主要采用socket实现客户端和服务端，模拟实际的环境，在客户端加密，将加密后的结果通过socket发送给服务端，服务端只是对密文进行计算，最后返回给客户端，客户端解密拿到结果。

## 1.3 组织结构

本课题主要研究国产密码算法在云计算网络中信息传输认证的原型设计与实现，论文一共分为五个章节。

第一章主要介绍了研究的背景和意义，阐述了目前云计算所存在的问题，研究的主要内容和方法的简要概述以及论文的组织结构等。

第二章研究了密码算法的分类，主要分为对称密码算法，非对称密码算法以及数字摘要算法。同时对密码算法的同态性，分组加密的模式介绍以及网络通信介绍。

第三章主要是对国产密码算法的实现和网络通信的实现做了介绍。

第四章主要阐述使用的SM2，SM4算法的同态性验证方案，实验流程以及对最后的实验得到结果分析。

第五章主要阐述对三种国产密码算法加解密效率的研究以及最后的结果。

第六章主要阐述了传输认证系统的设计与实现。

#### **1.4 本章小结**

本章主要介绍了云计算目前存在的一些问题、研究的背景以及研究的意义。简短阐述了研究的方案。

下一章主要对论文中使用的密码算法做整体的阐述、具体国产密码算法的研究、密码算法的同态性的介绍、分组密码算法工作模式介绍以及网络通信介绍。

## 第二章 密码算法同态性及网络

### 2.1 密码算法简介

密码算法主要就是一种数学函数，将我们需要发送的数据经过相应的数学函数进行计算得到与之前数据不同的数值，将得到的数值发送给通信的另一方，通信的另一方收到该数值后，需要使用该数学函数的反函数进行计算得到原来的数据。上述发送者通过数学函数计算的过程称为加密，接收者使用反函数计算的过程称为解密。一般地，使用的加密算法需要一定的可逆性，否则无法解密得到结果。

明文消息 $M$ 、加密函数 $E$ 、解密函数 $D$ 、密文 $C$ 会满足以下性质：

$C = E(M)$ ，表示对明文消息 $M$ 使用加密函数 $E$ ，得到密文 $C$ 。

$M = D(C)$ ，表示对密文 $C$ 使用解密函数 $D$ ，得到明文 $E$ 。该形式等价于 $M = D(E(M))$ 。

对称密码算法，非对称密码算法，数字摘要算法是目前密码算法主要三大类。

#### 2.1.1 对称密码算法

对称加密算法是指加密的所使用密钥与解密所使用的密钥是相同的，在一般的网络数据传输中，数据的加密都是采用对称加密算法。通过密钥我们可以控制加密和解密的流程，算法相当于数学函数，加密和解密操作都需要通过该数学函数实现。计算量小，加密速度很快，加密效率高是对称加密的优点。但是使用对称加密算法加密数据，如果密钥被第三方得知，那么作为第三方可以获取通信双方传输的内容，所以只使用对称加密的安全性不高。因此对称加密的安全性不能只考虑所使用的加密算法还需要考虑密钥如何去高效管理不会造成密钥泄露。因为加密和解密都使用同一个密钥，所以当时使用非对称加密时，需要考虑如何安全的去分发密钥到解密者手中。密钥管理是对称加密技术的关键，密钥安全性的保证非常重要。

常见的对称加密算法有DES，AES，3DES，IDEA，PBE等。其中DES是数据加密的标准，加密的速度较快，对于需要加密的数据量很大的场景可以使用DES；AES是下一代的加密算法的标准，加密速度很快，并且安全级别高，加密的密钥的位数也是不同的；3DES是三重DES的简称，其基础还是DES，相比于DES，3DES通过三个不同的密钥对一个数据组加密，从而使得其安全性高于普通的DES；IEDA常用于电子邮件的加密中，其工作模式只有ECB；PBE是综合了消息摘要算法和对称加密算

法，工作模式只有CBC，不具有安全性。一般地，对称加密因为其加解密速度快，效率高，常用在对数据进行加密中，但是因为其加密密钥和解密密钥使用的同一个密钥，密码算法的安全性也存在一定的挑战，可以采用多级密钥管理或者配合其他加密算法使用。

### 2.1.2 非对称密码算法

非对称加密是指加密所使用的密钥和解密所使用的密钥是不同的。一般地，在非对称加密中会有公钥和私钥两种密钥，其中公钥是可以被别人得到的，用来对将要传输的数据进行加密，私钥用来对接收到的数据进行解密。非对称加密的安全性很高，但是因为存在公钥和私钥两种密钥，计算量较大，加密速度较慢，加密的效率较低。非对称加密还需要考虑公钥的分发策略，其更多的使用在通信双方的身份认证中。

常见的非对称加密算法有RSA，ECC，DSA等。RSA主要是基于大素数分解，其安全性主要与其密钥长度相关，密钥长度越长，安全性越高，但是需要考虑到程序计算的开销，密钥长度的选择是需要从安全性和程序性能做一个相应的平衡；ECC的原理是在椭圆曲线上的有理点构成Abel加法群上椭圆离散对数的计算非常困难，其主要的优势在于可以使用更短的密钥，提供相当或更高等级的安全性。在给定密钥长度的条件下，ECC是安全性最强的非对称加密算法，在某些对带宽具有要求的场景时可以使用该算法；DSA是基于整数有限域离散对数难题，DSA不仅仅具有私钥和公钥，还具有数字签名。非对称加密算法也可以作为数字摘要算法使用，数字签名通过私钥生成，数据和签名的验证通过公钥实现。数据在网络中传输可能会被修改，当另一方收到数据后，需要通过数字签名来判断数据是否被修改。非对称加密因为其计算的开销非常大，存在公钥和私钥两个密钥，如果单纯使用非对称密码算法加密数据，需要考虑密钥分发策略，程序的性能也会下降。一般地，非对称加密多用在身份认证中，用来确认通信双方的身份。

### 2.1.3 数字摘要算法

数字摘要是通过某个数学函数，以需要传输的明文数据为参数，得到一个固定长度的消息，通过生成的固定长度的消息可以判断数据是否被修改，数字摘要函数也被称为Hash函数。数字摘要所使用的函数是单向不可逆的，并且对数据的修改是敏感的，对于不同的明文数据，通过同一数字摘要函数得到的消息总是不同的。通

过此种性质可以在网络通信中保证数据的完整性，因为其是单向不可逆的，当网络中的密文数据包被第三方获得，纵使第三方拿不到相应的明文数据，但是可以对其进行增减操作，从而改变了数据的内容。通过数字摘要，在发送数据时便对明文进行计算出相应的数字摘要，发送给通信另一方，对方收到数据解密后，使用相同的数字摘要算法计算出相应的数值与传送来的进行比对，如果相同，则可以证明没有修改过，若是不同，那么数据在传输中被截取修改过。

常用的数字摘要算法有md5，SHA-1等。其中md5是一种较为常用的散列函数，其通过相应的摘要函数会产出一个128位的散列值，通过该散列值可以确保信息传输的一致性，但是无法防止碰撞，所以不适用于安全性认证；SHA-1是一种密码散列函数，它可以生成一个被称为消息摘要的160位散列值，通常是由40个十六进制整数表示。数字摘要算法常用在判断数据是否被修改过，主要应用在网络数据传输，下载文件等等。一般地，在网站上的文件可能在下载时会被攻击者篡改，当我们下载网站上的文件打开后，可能会破坏我们的计算机系统，在下载完文件后，应该对比hash值是否与网站上文件显示的hash值相等，相等则证明没有被修改，否则，文件可能被修改了。

### 2.1.4 应用

现如今的网络信息时代，我们常常会使用浏览器或者一些APP来浏览网页，浏览网页的主要内容是HTML，CSS等，但是这些文件是放在远程服务器上，需要通过网络传输到我们的浏览器，浏览器解析后才会显示出我们所看到的内容。现如今，大部分的浏览器都使用了HTTPS协议，因为其可以确保数据在网络中传输的安全性，主要就是通过上述的密码算法来实现网络信息传输的安全。

HTTPS较HTTP更为安全，其安全性主要体现在数据的加密传输，身份认证以及hash值保证数据完整性。数据在网络中传输，可能会被第三方获得传输的数据包，如果对传输的内容使用对称加密算法进行加密，第三方拿到的内容也只会是密文，而无法拿到对应的明文。当网络中的数据被第三方拿到，其拿到的是密文，虽然无法看到具体内容，但是可以对内容做一定程度的破坏，增加、删除或修改任意一段密文，就会造成信息的混乱，这时就需要使用相应的数字摘要算法，在数据发送之前的明文作一次计算获得哈希值，另一端再收到数据解密后使用相同的数字摘要算法计算得到一个哈希值，比较两个哈希值是否相同给，若是不相同，则证明数据在传

输过程中被修改，否则，则说明数据未被修改。网络中通信双方在一开始需要建立连接，确认身份，常用的做法是通过非对称加密实现，一端通过公钥加密一段相应的内容，然后发送给另一端，另一端通过私钥解密，并且相应。对于使用非对称密码算法进行通信的双方，需要考虑公钥的分发策略，如何获取到公钥，一般地公钥都会放在一些可信的机构，比如：公钥存放在CA证书中，请求方拿到CA证书，通过CA证书解析出需要的公钥即可。

HTTPS建立连接的过程如下：

用户端生成一个随机数1，并将自己的所支持的密码算法列表、数字摘要函数列表和该随机数1发送给服务端。

服务端接收之后，选择适当的密码算法和摘要函数，并生成一个随机数2，之后将CA证书，随机数2发送给用户端。

用户端接收之后，先进行CA证书验证，如果CA证书有效，可以从CA中获取其公钥，然后生成随机数3，并使用公钥加密该随机数3，发送给服务端，此时，只有服务端相应的私钥可以解密得到该随机数。

服务端接收到相应的数据后解密得到该随机数，根据双方三个随机数计算确定对称密码的密钥，之后通信双方就可以进行相应的数据传输。

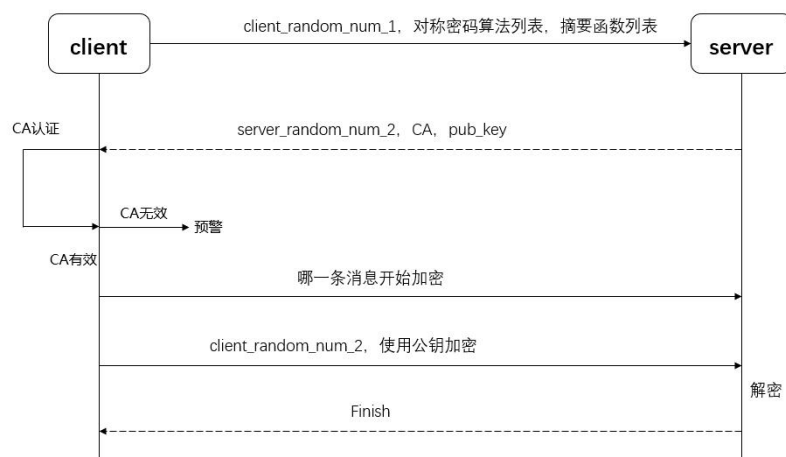


图 2-1 HTTPS 加密流程

HTTPS只可以保证数据在网络中传输的安全性，但是在云环境中，我们需要保证用户的一些隐私数据对于云服务提供商不可见，同时当发生数据泄露，需要保证用户数据隐私性，这就需要通过密码算法的同态性来实现。

## 2.2 加密算法的同态性

目前云计算平台对于用户数据只是简单的以密文的形式存储，无法对密文做任何操作，云服务商对密文的任何操作，都可能导致解密后无法得到预期的结果。同时，由于近年来频繁发生用户隐私数据泄露，云计算的安全问题显得日益严峻。云计算拥有的是强大的计算能力，如果可以对密文进行任意操作的效果等价于对明文的操作，在解密后的结果与直接使用明文计算得到结果一致，就可以完美解决云计算目前存在的问题，这也是同态加密技术的核心。

### 2.2.1 完全同态加密技术

完全同态加密算法是指对于密文进行任何的计算操作，得到的结果解密后与直接使用明文计算拿到的结果一致，主要强调对于执行的计算操作没有限制，任何的计算操作都可以。

完全同态加密系统是一种允许客户端除了进行基本的加密与解密操作之外，还可以在加密数据上进行任何的计算操作而不需要解密数据。假设，我们有两个明文  $m_1$  和  $m_2$ ， $c_1 = \varepsilon(m_1)$  和  $c_2 = \varepsilon(m_2)$  分别是其加密后的密文，完全同态加密可以让我们得到  $c_1 + c_2 = \varepsilon(m_1 + m_2)$  和  $c_1 \cdot c_2 = \varepsilon(m_1 \cdot m_2)$ ，因此，用户可以将自己复杂的计算交给远程的云服务器去完成，云服务在它自己的运行环境中可以提供无限制的计算。

### 2.2.2 部分同态加密技术

部分同态加密算法是指对于密文进行某些特定计算操作，得到的结果解密后与直接使用明文进行该运算拿到的结果一致，主要强调对于某种特定的计算操作，而不是对于所有的计算都满足上述性质。

常见的满足部分同态加密算法有RSA, Paillier, 其中RSA满足乘法的同态性，而Paillier满足加法的同态性。对于未进行填充的RSA加密系统，可以认为是可以相乘的同态，即，对于明文  $m_1$  和明文  $m_2$ ，我们可以得到  $\varepsilon(m_1) = m_1 e \% N$  和  $\varepsilon(m_2) = m_2 e \% N$ ， $N$  是 RSA 的模数， $e$  是 RSA 公钥的指数，因此，可以得到  $\varepsilon(m_1)\varepsilon(m_2) = (m_1 e \cdot m_2 e) \% N = (m_1 \cdot m_2) e^2 \% N = \varepsilon(m_1 \cdot m_2)$ 。RSA 允许我们只是仅仅对密文进行乘法操作而不能进行加法操作，关于加法的同态加密方案是Paillier的加密系统，如果公钥是模  $N$  并且是基于  $g$  的，那么加密的信息  $m$  则是  $\varepsilon(m) = g^m r^N \% N^2$ ，对于  $r$  是区间  $[1, N-1]$  之间任意的数，Paillier 加密算法的同态特性则是  $\varepsilon(m_1)\varepsilon(m_2) = (g^{m_1} r_1^N) \cdot (g^{m_2} r_2^N) \% N^2 = g^{m_1 + m_2} (r_1 r_2)^N \% N^2 = \varepsilon(m_1 + m_2)$ 。



由上可以看出，并不是所有的密码都具有同态性，具有完全同态特性的密码算法是理想化的解决方案，实际很多的密码算法都只可以满足某种计算操作的同态性，也就是部分同态性。

## 2.3 加密模式介绍

一般的对称加密算法根据加密方法的不同又分为分组加密和序列密码两个类。其中分组密码也称为块密码，对明文按一定的位长进行分组，每组称为一个块，例如：按每64位为一组，每个分组也就是一个块，加密的时候对每一个明文块进行加密，明文块经过加密运算后变为密文块，密文块经过解密运算后也会得到相应的明文块，每次以一个块的方式加解密，不需要多次使用加解密函数，减小了加解密过程中的计算量，极大的提高加解密的效率。序列密码也称为流加密，对明文每次都只加密其中的一位，这种加密方式对明文中的每一位都会通过相应的数学函数运算，相比分组加密方式，计算量增大，加解密的效率降低。

ECB, CBC, CFB, OFB是分组密码算法中常见的四种加密模式，本课题研究中主要使用了ECB与CBC模式，所以只是介绍这两种加密模式。

### 2.3.1 ECB 模式

ECB模式也被称为电子密码本模式，是最简单的一种运行模式。它先将明文按64位的长度进行分组，然后对每组分别进行加密，每组加密所使用的都是同一个密钥。所以，在此加密模式下当加密的密钥确定后，对于每一个明文分组，都会有相应的密文与之对应。使用ECB模式进行加密操作，每组独立加密，可以认为存在一个大的密码本，每个明文都会在该密码本中与某个密文惟一对应。因为是分组加密，当明文最后一组的长度小于64位时，需要进行填充至64位。解密过程也是对密文按相同的长度分组，每次解密一个分组，加解密使用的密钥是相同的。

通过上述可以知道，ECB模式下每组加密解密操作都是独立的，不存在组与组之间影响传递的情况，当某一组加密出现错误，不会传递到后面的分组中。分组独立加解密可以并行的加解密，提高加密的速度。但是对于两个明文相同的分组，会被加密成密文相同的两个分组，所以无法抵抗统计分析攻击。所以，ECB模式适于加密小消息，例如：密钥的保护。

### 2.3.2 CBC 模式

CBC 模式也称为加密块链模式，它先将明文按一定的位长进行分组，然后对每组分别加密，加密时，需要考虑前一组加密的结果对本组的影响，前一组加密的结果与本组将要加密的明文分组进行异或操作，再使用密钥对得到的结果进行加密生成相应的密文。当明文分组中第一个明文分组加密时，由于其前面没有相应的密文分组结果，所以需要初始化一个向量与第一个明文分组进行异或操作再加密。CBC 模式加密中，各个分组之间不相互独立，前一个分组会对后一个分组有影响，对于明文分组来说，同一个明文分组会对应多个密文分组，这种加密模式破解起来更加困难，并且通过简单的调换密文分组的顺序无法造成攻击。

通过上述可知，CBC 模式不容易被主动攻击，并且密文分组直接相互依赖，安全性更高，当传输较长的数据时可以使用 CBC 模式。但是由于相互依赖，当某一分组计算错误，会使得误差传播到后面的分组，导致后面分组计算结果不符合预期。同时相互之间的依赖不适合并行加解密，降低了加解密的速度。对于第一个分组的加密，需要确定初始化向量。

## 2.4 网络通信介绍

数据在网络中的传输需要遵守一定的规则，需要考虑传输的介质，如何去发送数据，使用什么协议等等。现在的网络可以使用五层或七层的模型进行描述，七层模型由下到上依次是：物理层，数据链路层，网络层，传输层，会话层，表示层，应用层。本课题实验所采用的 socket 属于网络中的传输层。

物理层。该层主要规定了有关传输介质的特性，一般的物理层规范中包括连接头、帧、编码及光调制等，不同的传输介质承载的数据形式也不相同。

数据链路层。该层定义如果让格式化的数据以帧的形式进行传输，以及如何对物理介质进行访问，还提供了错误检测和纠正来保证数据的可靠传输。

网络层。该层定义了数据包如何在主机之间传输的。通过对网络中路由器抽象，将路由器标识为逻辑节点，标志相应地址，包在网络中如何路由，路由器如何进行主动学习等。因为传输的带宽有限，该层还定义了分包策略，将一个大的数据包进行拆分为更小的数据包，分开发送。

传输层。该层定义了主机在网络中通信的协议，一般的协议有两种，一种是面向连接可靠的协议，会有差错校验、超时重传、拥塞控制以及包排序等；另一种是

用户数据报协议，主要是进行包传输，但是不去关心包是否成功发送、网络是否拥塞等。

会话层。该层定义了对于通信双方一个会话是如何开始，控制和结束的。通过对双向消息的控制，可以在数据传输只完成部分时便通知上层处理，提高处理速度。在某些情况下，如果表示层收到了所有的数据，则用数据代表表示层。示例：RPC，SQL等。

表示层。这一层的主要功能是定义数据格式及加密。因为在网络中传输的数据格式多种多样，网络层做了封装，但是对于应用来讲，需要具体的数据格式，通过表示层改变数据呈现的形式。

应用层。该层主要是一些可以直接为应用进程提供服务的协议。常见的邮件协议SMTP，web访问协议HTTP和HTTPS等。

网络的五层协议由下至上分别为物理层，数据链路层，网络层，传输层，应用层。网络中的每一层都会有相应的协议去做一些约定，在物理层主要考虑数据以什么样的形式在网络中传输。数据链路层需要考虑传输的介质，不同的传输介质可以传输的数据形式不一样，有的介质可以传输多种形式的数，但是会存在适配的问题，只有传输某种特定形式的数，传输的效率才会更高。网络层更多的对不同传输介质中的数形式做一种抽象，使得对于上层应用来讲，不需要关心底层数的具体格式，在什么介质中传输，其主要核心功能还是如何进行域内和域间路由。

数在网络中传输，如果使用HTTP，HTTPS，FTP等协议，都会经过网络的五层，自上向下的逐层封装数，最后发送出去。若是使用socket套接字进行简单数的传输，只会经过物理层，数据链路层，网络层和传输层，不会经过会话层，表示层和应用层，整体上速度更快，效率更高。

#### 2.4.1 传输层通信

TCP是一种面向连接的，可靠的数据传输控制协议，其面向连接主要是在数据传输之前会进行三次握手，数据传送完毕之后通过四次挥手来断开连接。其可靠性的保证主要通过ACK确认机制以及超时重传，同时，TCP也提供了流量控制和拥塞控制的功能。

TCP的三次握手主要是客户端与服务端的通信协商。先由客户端发起主动连接，会发送一个相应的数据包给服务端，其中TCP的首部的SYN会被标为1，表示这个数

据包是连接的数据包。服务端收到该数据包后，会返回给客户端一个数据包，其SYN位也被标为1，表示服务端同意与客户端建立连接。客户端收到数据包后，会发送给服务端一个相应的数据包，其中ACK位标为1，表示客户端收到了服务端传输的数据，同时也会商定双方传输的包的序号。

TCP的四次挥手主要是客户端与服务端数据传输完毕后断开连接。先由客户端主动断开连接，发送一个数据包，其FIN位被标为1，服务端收到客户端的数据包后，会先返回给客户端一个数据包，其ACK位标为1，表示服务端已经收到客户端传递的数据包，但是此时服务端可能还会有数据传送给客户端，所以无法立即断开连接。数据传输完毕后，服务端会发送一个数据包，其FIN位标为1，表示这是一个断开连接的数据包，客户端收到后会发送给服务端一个数据包，其ACK位标为1，表示客户端收到了服务端断开连接的包，客户端发送完毕后，会等待2倍的最大报文传送时间，如果考虑到最后一个包丢失，便会重新传送最后一个包来保证断开连接。

TCP的可靠性通过ACK确认和超时重传策略。客户端向服务端发送一个数据包，发送完后，会开启一个计时器，如果在规定的时间内，未收到服务端返回的ACK，那么就会重新传输之前发送的数据包。若是在规定的时间内收到了服务端返回的ACK，就表示这个数据包服务端已经成功接收了，开始传输下一个数据包。

TCP的拥塞控制主要是通过拥塞窗口实现的，如果网络中发生了拥塞，那么在规定时间内发送的数据包太多了，此时就可以动态调整拥塞窗口的大小，当网络中拥塞消失后，可以增加拥塞窗口大小。一般的，最初拥塞窗口的增长是以2倍的方式增长，达到一定阈值时，变为线性增长，当发生拥塞时，将阈值和拥塞窗口变为原来的一半，之后按照线性的方式增长。

本课题实验中所使用的socket通信属于传输层，其是基于TCP实现的，当网络中发生拥塞时，可以进行相应的拥塞控制，因为其是可靠的，传输的数据一定会到达服务端，不会发生数据包的丢失。

#### 2.4.2 数据传输过程

本实验中数据是通过控制台输入的，对于客户端程序来讲需要获取输入，输入的内容是在内核的缓冲区中，内核缓冲区中的数据需要拷贝到我们相应的用户进程中，此时用户进程拿到相应的数据，进行相关处理计算操作后，调用相应的系统调用函数，将数据先拷贝到内核的另一块缓冲区，再由内核线程将数据拷贝到socket

缓冲区中，最后由内核线程将数据发送出去。在用户进程等待数据输入以及数据发送过程中，用户进程是被阻塞的。在客户端发送数据是经过上述五层网络协议中的传输层，网络层，数据链路层和物理层，发送端的顺序由上至下，逐层对数据封装，加上一些相应的首部构成。数据在发送时可能会被分拆成多个数据包，在网络中通过路由器进行路由转发，达到服务端。服务端收到客户端发送的数据后，会先进行由下至上的解包，顺序是物理层，数据链路层，网络层和传输层，最后拿到实际我们传输的数据。

上述操作在网络中的部分不会更改，但是在客户端接收和服务端处理上会存在频繁的数据拷贝以及系统调用时频繁的在用户态和内核态切换，会造成一定的性能消耗。

为了减少数据的拷贝次数以及用户态与内核态频繁切换，socket使用了相应的零拷贝机制。主要的零拷贝技术有直接IO与MMAP的方式。

直接IO允许用户进程不需要通过相应的系统调用，直接可以从磁盘上读取数据，读取后的数据直接存储在用户进程相应的缓冲区域，不需要进行数据的拷贝。这种方式减少了一次用户态到内核态的切换，同时也减少了一次数据从内核缓冲区拷贝到用户缓冲区的操作，降低了CPU的消耗，典型的直接IO的应用有RDMS。

MMAP的方式通过在磁盘开辟一块缓冲区进行映射，此缓冲区为用户进程与内核进程共享的空间，通过共享空间就可以避免频繁的数据拷贝和用户态与内核态切换。客户端程序获取用户输入的内容后，其内容一直存放在缓冲区，该缓冲区被用户进程和内核进程共享，只需要两次系统调用，整体就可以实现数据的输入和发送，不存在数据频繁从内核缓冲区拷贝到用户缓冲区的问题。当用户进程处理完数据后，内核线程直接从共享的区域读取数据，再将数据发送拷贝到socket缓冲区，最后把数据发送到相应主机，此种方式很大程度提高了数据传输的效率。

## 2.5 本章小结

本章主要介绍了密码算法的基本分类、密码算法的同态性，加密的模式以及网络做了介绍。

下一章主要介绍本课题研究中国产密码算法的具体实现以及网络通信的实现。

## 第三章 国产密码算法与通信实现

### 3.1 国产密码算法实现

本课题实现了非对称加密算法 SM2, 对称加密算法 SM4 以及数字摘要算法 SM3。

#### 3.1.1 对称密码算法实现

本课题主要研究了国产对称密码算法中的SM4加密算法。SM4密码算法主要用于对于数据的加解密操作，通过SM4算法来保证数据的安全性。在对称密码算法中，密钥的长度越长，相应的其安全性也就越高。在SM4和AES具有相同的密钥长度分组长度的条件下，AES的安全性是高于3DES的，因此SM4在安全性上高于3DES算法。

SM4的加解密过程如下：

定义反序变换R为：

$$R(A_0, A_1, A_2, A_3) = (A_0, A_1, A_2, A_3), A_i \in Z_2^{32}, i = 0, 1, 2, 3。$$

说明文输入为  $(X_0, X_1, X_2, X_3) \in (Z_2^{32})^4$ ，密文输出为  $(Y_0, Y_1, Y_2, Y_3) \in (Z_2^{32})^4$ ，轮密钥为  $rk_i \in Z_2^{32}$ ， $i = 0, 1, 2, \dots, 31$ 。则本算法的加密变换为：

$$X_i + 4 = F(X_i, X_{i+1}, X_{i+2}, X_{i+3}, rk_i) = X_i \oplus T(X_{i+1} \oplus X_{i+2} \oplus X_{i+3} \oplus rk_i), i=0,1,2,\dots, 31。$$

$$(Y_0, Y_1, Y_2, Y_3) = R(X_{32}, X_{33}, X_{34}, X_{35}) = (X_{35}, X_{34}, X_{33}, X_{32})$$

SM4加密变换与解密变换的结构相同，惟一不同的是轮密钥的使用顺序不同。

加密时轮密钥的使用顺序为： $(rk_0, rk_1, \dots, rk_{31})$ 。

解密时轮密钥的使用顺序为： $(rk_{31}, rk_{30}, \dots, rk_0)$ 。

SM4算法的实现，对于对称加密只是单纯的对数据进行加密来保护数据的安全性。对于SM4的加密解密操作都封装为一个SM4Utils的类，该类是一个SM4算法的工具类。同时实现了SM4加密算法的ECB模式和CBC模式。

输入数据“hello world”，程序的运行结果如下所示：

```
plainText = hello world
ECB模式加密
加密后的密文: 449930d6f9d21f47c999d9727eee2bbc
解密后的明文: hello world
-----
CBC模式加密
加密后的密文: 64be68cb00b44284c3a072b4ab7cdd32
解密明文: hello world
```

图 3-1 SM4 加解密

如图3-1所示，第一行为我们输入的明文数据，后面会使用ECB模式与CBC模式分别加密，将得到的密文展示出来，之后会对相应的密文进行解密，拿到相应的明文数据。

### 3.1.2 非对称密码算法实现

本课题主要研究了国产非对称密码算法中的SM2加密算法。目前较为流行的非对称加密算法是RSA，但基于椭圆曲线上点群离散对数计算难题的SM2算法的安全性是高于RSA的。密钥长度越长，安全性越高，但是当SM2的密钥长度为256位时，其安全性是高于2048位的RSA。

SM2的加密过程如下：

用户的原始数据，椭圆曲线的系统参数，长度为 $k$ 比特的消息 $m$ 以及公钥 $P_B$ 。  
产生一个随机数 $k$ ， $k \in [1, n-1]$ 。

- (1) 计算椭圆曲线上的点  $C_1 = [k]G = (x_1, y_1)$ 。
- (2) 计算椭圆曲线上的点  $S = [h]P_B$ 。
- (3) 判断 $S$ 是否为0，若为0，则报错，否则继续向下执行。
- (4) 计算  $[k]P_B = (x_2, y_2)$ 。
- (5) 计算  $t = KDF(x_2 \parallel y_2, k)$ 。
- (6) 判断 $t$ 是否全0，若是，返回第一步，否则，继续向下执行。
- (7) 计算  $C_2 = M \oplus t$ 。
- (8) 计算  $C_3 = Hash(x_2 \parallel M \parallel y_2)$ 。
- (9) 输出密文  $C = C_1 \parallel C_2 \parallel C_3$ 。

SM2的解密过程如下：

用户的原始数据，椭圆曲线系统参数，密文  $C = C_1 \parallel C_2 \parallel C_3$ ，私钥 $d_B$ 。

- (1) 从密文中取出 $C_1$ 。

- (2) 验证 $C_1$ 是否满足曲线方程，若不满足，则报错退出；否则，继续向下执行。
- (3) 计算椭圆曲线点 $S = [h]C_1$ 。
- (4) 判断 $S$ 是否等于0，若是，则报错退出；否则，继续向下执行。。
- (5) 计算 $[d_B]C_1 = (x_2, y_2)$ 。
- (6) 计算 $t = KDF(x_2 \parallel y_2, k)$ 。
- (7) 判断 $t$ 是否全0，若是，则报错退出；否则，继续向下执行。
- (8) 计算 $M' = C_2 \oplus t$ 。
- (9) 计算 $u = Hash(x_2 \parallel M' \parallel y_2)$ 。
- (10) 判断 $u$ 是否等于 $C_3$ ，若不相等，则报错退出；否则，数据明文 $M'$ 。

SM2算法实现，对于非对称加密既可以用于签名也可以用于对数据的加密使用，本课题主要研究了对于数据的加密，对于SM2的加密解密操作都封装为一个SM2EncDecUtils的类，该类是一个SM2算法的工具类。

输入字符串“hello world”，程序运行结果如下所示：

```

输入的明文数据: hello world
加密使用的公钥: 04B834D657EE7E8490E66EF577E6B3CEA28B739511E787F84F71B7F38F241D87F18A5A93DF74E90FF94F4EB907F271A36B2958851F971DA5418F4915E2C1A23D6E
解密使用的私钥: 0B1CE43098BC21B8E82B5C065EDB534CB86532B1900A49D49F3C53762D2997FA
加密的结果: 045408d7f2f9c44ef26d68913bea5e1c4eca3d8deab8f9c32d242ccff3dcd5e3c9b93aa0ee36afcd5b91947fdacb0aea0c9d301b859d8d74170e7f27a550185d47060ae81aca
解密后的结果: hello world

```

图 3-2 SM2 加解密结果

如图3-2所示，程序会将输入的数据进行加解密，第一行为输入的明文；第二行为加密的公钥，此公钥在代码中已经确定；第三行是解密的私钥，此密钥在代码已经确定好的；第四行是输入数据通过公钥加密后的结果；第五行是通过私钥解密后的结果。

### 3.1.3 数字摘要算法实现

本课题主要研究了国产的数字摘要算法中的SM3算法。数字摘要算法需要保证数据的完整性，而SM3算法可应用在数字签名和验证消息认证码的生成与验证等众多场景需要中，满足应用的安全需求。对于数字摘要算法，其安全性与最后生成的散列值长度有关，所以，一般的产生的散列值不能太短，否则无法确保安全性。相比于MD5和SHA-1，SM3计算得到的散列值长度为256比特，其安全性会更高。

SM3的加解密过程如下：



(1) 填充。假设消息 $m$ 的长度为 $l$ 比特，首先将比特“1”添加到消息的末尾，再添加 $k$ 个“0”， $k$ 是满足 $l+1+k \equiv 448 \bmod 512$ 的最小非负整数。然后添加一个64位的比特串，该比特串是长度为1的二进制表示。填充后的消息 $m'$ 的比特长度是512的倍数。

(2) 迭代。将填充后的消息 $m'$ 按512比特进行分组： $m' = B^{(0)}B^{(1)}\dots B^{(n-1)}$ ，其中 $n = (l+1+k+65)/512$ 。对 $m'$ 按下列方式迭代：

FOR  $i = 0$  TO  $n-1$

$$V^{(i+1)} = CF(V^{(i)}, B^{(i)})$$

ENDFOR

其中 $CF$ 是压缩函数， $V^{(0)}$ 为256比特初始值 $IV$ ， $B^{(i)}$ 为填充后的消息分组，迭代压缩的结果为 $V^{(n)}$ 。

(3) 消息扩展。将消息分组 $B^{(i)}$ 按以下方法扩展生成132个字 $W_0, W_1, \dots, W_{67}, W'_0, W'_1, \dots, W'_{63}$ ，用于压缩函数 $CF$ 。

(4) 压缩函数。令 $A, B, C, D, E, F, G, H$ 为字寄存器， $SS1, SS2, TT1, TT2$ 为中间变量，使用压缩函数 $V^{i+1} = CF(V^{(i)}, B^{(i)})$ ,  $0 \leq i \leq n-1$ 。

(5) 杂凑值。 $ABCDEFGH \leftarrow V^{(n)}$ ，输出256比特的杂凑值 $y = ABCDEFGH$ 。

SM3算法的实现，对于数字摘要算法是需要生成一个长度固定的散列值，并且对于数据的改变是敏感的。

输入数据“hello world”，程序的运行结果如下图：

```

输入的明文数据: hello world
计算得到的散列值: 44F0061E69FA6FD3C290C494654A05DC0C053DA7E5C52B84EF93A9D67D3FFF88
-----
修改明文为: I love the world
计算得到的散列值: 9195748EBF6D1E243931966C1539A829BA0B46032426F02CA69756C65A093860
    
```

图 3-3 SM3 生成散列值

如图3-3所示，第一行显示了我们输入的数据，第二行是SM3算法根据我们输入数据计算得到的散列值，当输入的数据改变时，计算的散列值与前面计算得到的散列值不相同。

### 3.2 网络通信实现

用户端使用Java语言进行开发，使用Java的socket，在创建socket对象时，传入服务端程序监听的端口和所在机器的IP，需要连接服务端的IP和端口均采用配置化的方式编写，将IP和端口写在一个配置文件中，在用户端启动时，需要从配置文件中获取相应的IP和端口。

客户端的socket使用单例的类主要防止每次在对数据进行加密和解密时需要频繁的创建该类的对象，造成CPU会频繁的调度垃圾回收线程导致程序的性能下降。在初始化单例对象时进行了加锁，保证操作的原子性，在多个线程同时访问时，也只会初始化一个该类的对象，得到预期的效果。

服务端使用Java语言进行开发，服务端启动时需要绑定主机的某个端口，监听的端口也是配置化的方式，通过在配置文件里写入监听的端口，在创建服务端socket对象时，传入相应的端口即可。服务端采用线程池来对客户端的请求响应，使用的阻塞队列是链表阻塞队列，拒绝策略使用的是当任务过多，线程数达到最大线程数时，会让调用线程池的线程执行该任务。使用线程池进行响应避免了每次响应都创建线程，造成频繁的线程创建与销毁，提高程序的运行效率。对于每个客户端的连接都会有一个变量进行统计，表示这是第几次连接，该变量使用了Java中原子类，在多线程并发访问下是线程安全的，通过该变量就可以实现正确的数值统计，不会存在多个客户端同时连接，计数错误的情况。服务端的socket对象也是单例的，当有用户端请求与其进行连接时，会返回一个相应的socket对象，通过该对象与用户端进行通信。

对于服务端使用的线程池，其工作流程是当某个任务提交给线程池后，先判断当前的线程数是否大于线程池的核心线程数，如果小于核心线程数，那么便创建一个线程去执行该任务，否则，将该任务加入到所设置的任务队列中，如果任务队列中的任务数量没有达到最大的限制，那么便将该任务加入任务队列，否则，判断线程池中的线程数是否大于最大线程数，如果小于最大线程数，则创建一个线程去执行该任务，当线程执行完该任务后，便会存活一定的时间，在规定的时间内，该线程没有执行其他任务，那么该线程对象会被回收。如果大于最大线程数，则进行执行相应的任务拒绝策略。创建线程池所使用的是默认的工厂创建。本实验中核心线程数设置为CPU数量加1，因为服务端的任务主要是对客户端的请求响应，属于CPU

密集型的任务，不需要创建过多的线程，造成频繁的进行线程切换影响程序性能。最大线程数设置的100，线程存活时间设置为60秒，采用链表的阻塞队列，设置了队列的最大长为1000。为了让每一个任务都会被执行得到一个结果，采用的拒绝策略是让调用线程池的线程执行该任务。使用原子类操作主要是为了保证多个客户端连接计数的正确性，其主要是通过CAS操作实现。

### **3.3 本章小结**

本章主要介绍了网络的七层和五层模型，同时对传输层的两个主要的协议TCP和UDP做了详细的介绍，对于网络数据发送做了阐述。

下一章主要介绍本课题研究的国产算法的实验方案、实验过程和实验结果。

## 第四章 国产密码算法同态性分析

### 4.1 方案设计

本课题研究了国产对称密码算法SM4，国产非对称密码算法SM2以及国产数字摘要算法SM3的同态性，使用socket通信模拟数据网络传输。

对于SM2算法，在用户端使用SM2在代码中确定的公钥进行加密操作，然后将加密后的密文通过socket传输到服务端，服务端接收到密文后，对密文进行字符串的拼接操作，返回整体拼接的结果并且需要返回之前传递的数据，客户端收到服务端返回的数据后使用相应的私钥解密数据，判断是否与直接使用明文进行操作得到的结果相同。若是相同，则证明SM2存在部分同态性，否则，则说明SM2不满足同态性。

对于SM4算法，实现了其加解密的ECB模式与CBC模式。将需要传输的数据在用户端进行加密，然后通过网络将加密后的数据发送到服务端，在服务端对密文进行一个相加的操作。服务端最后将处理完的密文发送给客户端，客户端接收到密文后进行解密操作，拿到解密的结果再与客使用明文进行相同操作得到的结果比对，如果两个结果是一致的，那么说明满足同态性，否则，则不满足同态性。分别对SM4的两种加密模式进行实验，判断其在哪种模式下的会更加接近同态加密的特性。

因为SM3加密算法是数字摘要算法，其具有不可逆性，无法根据生成的密文解密生成对应的明文，所以不适合研究其同态性。

实验开发工具使用IDEA，JDK版本是1.8。

### 4.2 SM2 同态性分析

#### 4.2.1 SM2 实验

第一步，配置好服务端监听的端口，此实验配置的端口是7878，然后启动服务端程序，若成功启动，会提示“服务器启动成功，等待客户端发送数据”，如图4-1所示。



```
服务器启动成功，等待客户端发送数据...
```

图4-1 服务端启动

第二步，配置好客户端需要连接的服务端的IP和通信的端口，本实验配置的IP是127.0.0.1，端口7878，然后启动一个客户端应用程序，若是启动成功会在客户端提示输入字符串，同时，服务端会提示本次是第几次连接，并且等待客户端输入数据，如图4-2所示。




图 4-2 客户端连接成功

客户端第一次连接到服务端，此时服务端会打印出第1次连接，若是此时有新的客户端连接会体现是后续的连接次数，如图4-3所示。




图 4-3 连接成功服务端显示

第三步，客户端通过控制台输入相应的数据，先输入一个hello，在客户端会显示出相应的明文以及加密后的结果。

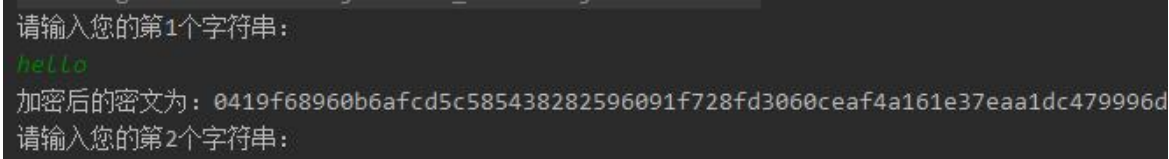


图 4-4 客户端输入数据后变化

图4-4显示的即将客户端加密前的明文以及加密后的密文都打印出来。当客户端每输入一个数据，便会进行加密，然后通过socket发送给服务端，服务端接收到对应的密文，该密文也会在服务端展示出来。

```
第1次连接
客户端发送的信息: 0419f68960b6afcd5c585438282596091f728fd3060ceaf4a161e37eaa1dc479996d5f9814a30c23fbda1d309899f543117b4b76644
```

图 4-5 服务端接收输入

据图4-5可知服务端接收到的知识客户端发送给他的加密后的数据，客户端并没有发送相应的明文信息，服务端完全是对密文进行操作。

第四步，输入一个world，同时结束输入，本实验中当输入的字符串符号为“#”时，客户端默认用户输入完毕，后面便不会提示用户继续输入，只需要等待服务端处理后返回结果，如图4-6所示。

```
请输入您的第2个字符串:
world
加密后的密文为: 04d7e934ad6b33cfdbf89615b316b99cb102fa0ca2ac314c9314160d6d980d6cd2e0223a5bdf7735524b398c1b2
请输入您的第3个字符串:
#
```

图 4-6 客户端输入完毕

第五步，服务端会根据客户端输入的密文进行一个拼接操作，然后将密文传递给客户端，客户端拿到服务端传递的数据后，会进行整体以及分隔字符串的方式解密，对比结果。

```
客户端发送的信息: 0419f68960b6afcd5c585438282596091f728fd3060ceaf4a161e37eaa1dc479996d5f9814a30c23fbda1d309899f543117b4b76644
客户端发送的信息: 04d7e934ad6b33cfdbf89615b316b99cb102fa0ca2ac314c9314160d6d980d6cd2e0223a5bdf7735524b398c1b2
客户端发送的信息: #
服务端发送给客户端的数据:
0419f68960b6afcd5c585438282596091f728fd3060ceaf4a161e37eaa1dc479996d5f9814a30c23fbda1d309899f543117b4b76644
```

图 4-7 服务端收到的全部数据

图4-7是客户端传递到服务端的所有密文，每次都会将发送的密文打印出来，当服务端完成相应的操作后，会将结果返回给客户端，同时，也会把每次传递的密文以列表的形式返回给客户端。

```
开始接收服务端返回的数据...
总的解密的结果是: hello 0419f68960b6afcd5c585438282596091f728fd3060ceaf4a161e37eaa1dc479996d5f9814a30c23fbda1d309899f543117b4b76644
分开解密的结果是: hello world
```

图 4-8 客户端收到的数据

据图4-8可知，客户端会对接收的数据进行解密，有两个解密结果。其中第一个是整体解密结果，其含义是服务端发送给客户端一个字符串列表，客户端使用列表的最后一个字符串进行解密，解密得到的结果如上图展示，如果整体解密，密文中间会存在一些解析错误的字符，也就是乱码，可以看到整体解密的结果与我们预期的结果不符合；第二个是分组解密的结果，从服务端返回给客户端的数据可以看出，服务端返回的列表中不仅仅只返回了处理的结果，还返回了客户端传递给服务端的数如数据，通过返回输入的数据，客户端可以根据这些输入的数据对服务端处理结果进行分隔字符串，因为返回的是密文，所以是根据列表中前面N-1个字符串对最后一个密文字符串进行逐个的拆分操作，拿到每个拆分的结果后再逐个解密，最后将解密结果拼接起来，得到一个分开解密的结果，这种解密方式拿到的结果与我们预期的结果是一致的，通过次种方式解密是可以得到正确的结果。

第六步，重新启动一个客户端程序，会在服务端显示是第二次连接，客户端逐次输入数据“I love the world!!!”，客户端每次对于输入的数据都是会进行加密，然后将数据发送给服务端，所以用户输入的数据没有长度的限制。服务端收到数据后，进行相应处理后返回给客户端列表，客户端根据返回的列表进行整体解密和分开解密操作做对比。

```
第2次连接
客户端发送的信息: 0412f51e163b540bec595a9aecb26ab88d877825b4fc43dd008b13a4eb71a761d06daa1b3a2659f1345b6
客户端发送的信息: 04f17c1542e3d650d03e505c8bf2115a21ff1383bbe6554c25c61890b23bd087106879bad9270f491c729
客户端发送的信息: 045068f446e4a5ca2c3a1f1151dbd6e858903ab0f088899fa49ad862be8e07022d9d5f0ba8c90e931abff
客户端发送的信息: 048f5c1968093e5486ec7de17ad611b7ca71499b2514f86b9e9118e787f0436d9c5a35b1ca166525ea837
客户端发送的信息: #
服务端发送给客户端的数据:
0412f51e163b540bec595a9aecb26ab88d877825b4fc43dd008b13a4eb71a761d06daa1b3a2659f1345b6ac69e4d421783bcd
```

图 4-9 服务端收到的数据

图4-9展示服务端与第二个客户端连接后，客户端输入数据加密后的结果以及服务端处理后返回给客户端的数据。



```
开始接收服务端返回的数据...
E;T\Z0"j&f!@)HvA3?:
E
Ecn,ELV59kE@dV(#?鰐t
分开解密的结果是: I love the world!!!
```

图 4-10 客户端收到数据

如图4-10所示,客户端会收到服务端传递的数据,其中整体解密的结果全是乱码,解密得到的结果与我们的预期不符合,所以SM2不满足同态性。

#### 4.2.2 SM2 结果分析

通过上述实验过程可以看出,客户端接收用户的输入然后对数据进行加密,将加密后的密文发送给服务端,服务端对密文进行了拼接操作,结果返回给客户端,客户端解密后得到了乱码的结果,拼接密文会影响字符串的识别,这与直接使用明文操作得到的结果不一致,说明了国产密码算法SM2不具有同态性。

### 4.3 SM4 同态性分析

第一步,配置好服务端监听的端口,此实验配置的端口是7878,然后启动服务端程序,若成功启动,会提示“服务器启动成功,等待客户端发送数据”,如图4-11所示。

```
服务器启动成功,等待客户端发送数据...
```

图4-11 服务端启动

第二步,配置好客户端需要连接的服务端的IP和通信的端口,本实验配置的IP是127.0.0.1,端口7878,然后启动一个客户端应用程序,若是启动成功会在客户端提示输入字符串,同时,服务端会提示本次是第几次连接,并且等待客户端输入数据,如图4-12所示。

```
请输入您的第1个字符串:
```

图 4-12 客户端连接成功



客户端第一次连接到服务端，此时服务端会打印出第1次连接，若是此时有新的客户端连接会体是后续的连接次数，如图4-13所示。

```
服务器启动成功，等待客户端发送数据...
-----
第1次连接
```

图 4-13 连接成功服务端显示

以上便是两次实验操作相同之处，后面加解密操作分别使用不同的加解密模式即可。

### 4.3.1 ECB 模式实验

第三步，在客户端分别输入数据“hello world”，使用SM4的ECB模式加密输入的数据，通过socket将数据发送到服务端，服务端对接收到的数据进行拼接操作，并将操作后的结果返回给客户端，客户端展示结果即可。

```
请输入您的第1个字符串：
hello
加密后的密文为：dda57cde66c3d19d71fb06219784f9b7
请输入您的第2个字符串：
world
加密后的密文为：7bdc30499125d8c065ea6bb64e89ce4d
请输入您的第3个字符串：
#
```

图 4-14 客户端输入的数据

如图4-14所示，客户端输入数据“hello world”，并且将输入的数据加密后的密文发送给服务端。

```
第1次连接
客户端发送的信息： dda57cde66c3d19d71fb06219784f9b7
客户端发送的信息： 7bdc30499125d8c065ea6bb64e89ce4d
客户端发送的信息： #
服务端发送给客户端的数据：
dda57cde66c3d19d71fb06219784f9b7,7bdc30499125d8c065ea6bb64e89ce4d,dda57cde66c3d19d71fb06219784f9b77bdc30499125d8c065ea6bb64e89ce4d
```

图 4-15 服务端收到以及发送的数据

如图4-15所示，服务端接收到客户端传输的密文，然后对密文进行拼接操作，最后将拼接的结果以列表的形式返回给客户端。

```
-----  
开始接收服务端返回的数据...  
总的解密的结果是: hello ❖ world  
分开解密的结果是: hello world
```

图 4-16 客户端接收数据解密后的结果

如图4-16所示，客户端接收到服务端的数据后，对数据进行了分开解密与整体解密，其中整体的结果可以解密出hello world，但是在两个单词之间存在不能被识别的空格以及乱码符号，结果于预期不符合；对于分开解密的结果与我们的预期相符合。

### 4.3.2 CBC 模式实验

第三步，在客户端分别输入数据“hello world”，使用SM4的CBC模式加密输入的数据，通过socket将数据发送到服务端，服务端对接收到的数据进行拼接操作，并将操作后的结果返回给客户端，客户端展示结果即可。

```
请输入您的第1个字符串:  
hello  
加密后的密文为: ac6c1aee7157c9ca1c46b509465e225b  
请输入您的第2个字符串:  
world  
加密后的密文为: 2e01a4d2ef0abaa752dfa45b8d500b6c  
请输入您的第3个字符串:  
#
```

图 4-17 客户端输入的数据

如图4-17所示，客户端输入数据“hello world”，并且将输入的数据加密后的密文发送给服务端。

```
服务器启动成功，等待客户端发送数据...  
-----  
第1次连接  
客户端发送的信息: ac6c1aee7157c9ca1c46b509465e225b  
客户端发送的信息: 2e01a4d2ef0abaa752dfa45b8d500b6c  
客户端发送的信息: #  
服务端发送给客户端的数据:  
ac6c1aee7157c9ca1c46b509465e225b,2e01a4d2ef0abaa752dfa45b8d500b6c,ac6c1aee7157c9ca1c46b509465e225b2e01a4d2ef0abaa752dfa45b8d500b6c
```

图 4-18 服务端收到以及发送的数据

如图4-18所示，服务端接收到客户端传输的密文，然后对密文进行拼接操作，最后将拼接的结果以列表的形式返回给客户端。

```
-----  
开始接收服务端返回的数据...  
总的解密的结果是: hello  
分开解密的结果是: hello world
```

图 4-19 客户端接收数据解密后的结果

如图4-19所示，客户端接收到服务端的数据后，对数据进行了分开解密与整体解密，其中整体解密的结果只会显示出hello，对于传输数据会有缺失，无法解析出后面的数据，结果于预期不符合；对于分开解密的结果与我们的预期相符合。

#### 4.3.3 SM4 结果分析

通过上述实验过程可以看出，客户端接收用户的输入然后对数据进行加密，将加密后的密文发送给服务端，服务端对密文进行了拼接操作，结果返回给客户端，客户端解密后得到的结果与直接使用明文操作得到的结果一致，则说明该密码算法具有同态性，上述实验对于分开解密的方法两者都会具有一定的同态特性，但对于整体解密两者都会存在一定的瑕疵，使得解密的结果不完整。

对于分开解密的结果，两种模式都具有一定的部分同态性，但是对于整体解密的结果显示两者不具有部分同态性。

本课题研究的国产密码算法有SM2，SM3，SM4，但是SM3不具有可逆性，无法根据密文对解析出明文，所以对于同态性的研究在密码算法SM2，SM4上。SM2非对称性使得整体解密完全都是乱码的结果，而对于SM2算法，其两种模式下，ECB模式可以解密出完整的结果，但是会数据之间会存在少部分的乱码，但是效果是优于SM4的CBC模式，该模式下解密的数据会存在丢失。而最终选择使用SM4，因为其加密解密的速度都较快，加解密的效率高。SM4属于对称密码算法，相比于非对称的密码算法，数字摘要算法，其本身的加解密的效率都是要高于其他两种加密算法，同时因为使用ECB模式进行加密操作，每组加解密所使用的密钥都是相同的，所以

使用SM4来进行加解密操作可以使程序的运行速度更快，效率更高。

#### 4.4 密码算法同态性结果对比分析

对于SM2国产非对称密码算法，对整体进行解密，其结果出现了无法解析的字符，解密之后完全无法得到预期的数据，无法得到正确的结果，不满足同态性。故该算法不具备同态性。

对于SM4国产对称密码算法，在CBC模式下对整体进行解密，结果显示其会缺失部分数据，与预期相差较大。在ECB模式下，整体解密的结果显示其可以解密出预期的数据，但是中间会存在一些无法解析的字符，导致得到的结果与预期不完全相符，但使用分组解密，可以完整的解密出结果，符合预期。

#### 4.5 本章小结

本章主要介绍了SM4实验的方案，对于其不同的加解密模式都进行实验，实验的流程以及最后的实验结果。

下一章主要对本课题所使用的三种密码算法的加解密时间消耗进行对比，得到性能最优的密码算法。

## 第五章 密码算法的性能对比

对于一个密码算法，安全性是需要首先考虑的因素，但是其加解的效率也是衡量密码算法的性能的一个重要指标，本课题研究中我们希望尽可能找到安全性高，具有同态性，加解密时间较小的密码算法。

对加解密效率的衡量指标可以通过加解密所消耗的时间体现，加解密消耗的时间越短，则说明该算法的效率越高；相反，则说明其加解密的效率较低。

### 5.1 性能方案设计

本课题主要计算SM2, SM3, SM4三个国产密码算法的加解密时间，对于每种密码算法需要固定为同一个输入的数据，保证此输入的变量的数值相同，该变量被固定为字符串“hello world”，再对该变量进行加解密操作，得到加解密时间，最后对它们做相应的比较，根据时间判断得到最优的密码算法。

在对于每个密码算法消耗时间的计算中，因为操作系统的整个资源都是在动态变化的，当某一个密码算法进行时间计算时，系统此时给应用程序分配了更多的内存，或是CPU此时有多个空闲，都会对计算结果造成影响，所以本实验使用对每个密码算法的加解密操作时间计算 $n$ 次，然后求解这 $n$ 个时间的平均值作为最终得到的结果，以此可以更大程度的减小动态系统资源对结果的影响。

在SM2实验中，先计算出一次加解密时间，然后逐渐的增加 $n$ 大小，对于每一个 $n$ 计算加解密时间，最后当确定下 $n$ ，每次计算 $n$ 次加解密时间时，都会计算出一次加解密时间，当两者相差不大时，可以认为 $n$ 的取值为最佳，此时计算得到的加解密时间可认为是该算法的加解密时间。

本实验时间的单位均为毫秒， $n$ 次和一次计算两者的误差小于0.5ms以内即可。

### 5.2 SM2 加解密时间分析

SM2属于非对称加密算法，加密操作依赖于其公钥，解密操作依赖于其私钥，通过得到其加密与解密时间判断其加解密的性能。

第一步，固定明文变量为“hello world”，调用该算法的加密函数和解密函数对其进行一次加解密操作，得到一次加解密的所花费的时间。

```
输入的明文数据: hello world
加密的结果: 04f623cd8feca0e74686e36916145d71665088257e6ad8dcd1831fdb69b12602f3f6
加密操作花费的时间为: 1223ms
解密后的结果: hello world
解密操作花费的时间为: 20ms
```

图 5-1 SM2 一次加解密时间

如图6-1所示，SM2算法一次加密的时间大约为1223ms，加密所消耗的时间明显比较长。其加密时间大约为20ms，相对于加密时间，其解密时间消耗时间较短。

第二步，将n取值为10，计算SM2算法运行10次加解密操作所得到的平均时间。

```
10次平均的加密时间为: 192
10次平均的解密时间为: 11
输入的明文数据: hello world
加密的结果: 0443840c1446cf41863b92192452e18dc75ee1068a0b27baffc3c62d6f6cc3002f8a425
加密操作花费的时间为: 14ms
解密后的结果: hello world
解密操作花费的时间为: 9ms
```

图 5-2 SM2 运行 10 次加解密时间

如图6-2所示，当n为10的时候，平均加密时间为192ms，平均解密时间为11ms，此时再计算一次加密时间为14ms，一次的解密时间为9ms，这个数据与之前单独进行一次加解密时间相差较大，原因可能由于一次加解密时，计算量较小，系统分配资源较少，而当需要多次计算时，系统会分配更多的内存，此时再计算一次加解密时，得到的时间会较短。10次计算得到的时间与一次计算得到的时间差距仍然很大，所以需要继续增大n。

第三步，将n取值为50，计算SM2算法运行50次加解密操作所需要的平均时间。

```
50次平均的加密时间为: 42
50次平均的解密时间为: 8
输入的明文数据: hello world
加密的结果: 04bfdceb2937bd6203d56e5eab95934be2fdd4642512d1d9af90f15b3f515c95ccd56ba3097490e
加密操作花费的时间为: 12ms
解密后的结果: hello world
解密操作花费的时间为: 7ms
```

图 5-3 SM2 运行 50 次加解密时间

如图6-3所示，当n为50时，计算得到的50次平均加密时间为42ms，50次平均解密时间为8ms，此时一次加密的时间为12ms，一次解密的时间为7ms。50次和一次差距仍然较大，继续增加n的大小。

第四步，将n取为100，计算SM2算法运行100次加解密操作所需要的平均时间。

```
100次平均的加密时间为：26
100次平均的解密时间为：6
输入的明文数据：hello world
加密的结果：04dd5e8ae5b6b9069487af0622b39a64b7ed1c0a0bbe1012673c22bc67a3096e90a
加密操作花费的时间为：8ms
解密后的结果：hello world
解密操作花费的时间为：3ms
```

图 5-4 SM2 运行 100 次加解密时间

如图6-4所示，当n取值为100时，100次平均的加密时间为26ms，100次平均的解密时间为6ms，此时一次加密的时间为8ms，一次解密的时间为3ms。100次和一次运行得到的时间差距仍然较大，继续增大n。

第五步，将n取为1000，计算SM2算法运行1000次加解密操作所需要的平均时间。

```
1000次平均的加密时间为：8
1000次平均的解密时间为：2
输入的明文数据：hello world
加密的结果：0425a18b0cfb35474df33997ee4b97a8c541e7665e8298818d5ba3bec74ca7d1890592cb99005760ed43
加密操作花费的时间为：5ms
解密后的结果：hello world
解密操作花费的时间为：2ms
```

图 5-5 SM2 运行 1000 次的加解密时间

如图6-5所示，SM2运行1000次的加密时间为8ms，解密时间为2ms。此时运行一次的加密时间为5ms，解密时间为2ms。加密时间相同，但此时解密时间不相同，可以适当增加n获得准确的加密时间。

第六步，将n取为5000，计算SM2算法运行5000次加解密操作所需平均时间。



```
5000次平均的加密时间为：5  
5000次平均的解密时间为：2  
输入的明文数据：hello world  
加密的结果：04a637ede98545abc8523f1bb142c0ac360dd9b9347082fe7a42dbca061060dcd2e6aaca2f  
加密操作花费的时间为：5ms  
解密后的结果：hello world  
解密操作花费的时间为：2ms
```

图 5-6 SM2 运行 5000 次的加解密时间

如图6-6所示，SM2算法运行5000次的加密时间为5ms，解密时间为2ms。此时，运行一次的加密时间为5ms，解密时间为2ms。两者时间相同。

综上所述可以得到，SM2算法的加密时间为5ms，解密时间为2ms。

### 5.3 SM3 加解密时间分析

因为SM3算法是数字摘要算法，具备单项不可逆性，所以对其生成的散列值无法反向计算得到相应的明文信息，所以，只可以计算其加密的时间。

第一步，固定明文变量为“hello world”，计算一次SM3算法消耗的时间。

```
输入的明文数据：hello world  
计算得到的散列值：44F0061E69FA6FD9C290C494654A05DC0C053DA7E5C52B84EF93A9D67D3FFF88  
一次散列值计算的时间为：17ms
```

图5-7 SM3一次计算时间

如图6-7所示，单次计算散列值的时间消耗在17ms左右，此时需要增加n的数值。

第二步，将n取值为10，计算10次SM3算法执行的平均时间。

```
10次计算散列值的平均时间为：1.8ms  
输入的明文数据：hello world  
计算得到的散列值：44F0061E69FA6FD9C290C494654A05DC0C053DA7E5C52B84EF93A9D67D3FFF88  
一次散列值计算的时间为：0.1249ms
```

图5-8 SM3计算10次的时间

如图6-8所示，将n取值为10，10次计算平均消耗的时间为1.8ms，但此时单次计算的时间在0.1249ms，误差大于0.5，需要继续增加n的数值。

第三步，将n取值为100，计算100次SM3算法执行的平均时间。



```
100次计算散列值的平均时间为：0.13ms  
输入的明文数据：hello world  
计算得到的散列值：44F0061E69FA6FDFC290C494654A05DC0C053DA7E5C52B84EF93A9D67D3FFF88  
一次散列值计算的时间为：0.0769ms
```

图5-9 SM3计算100次的时间

如图6-9所示，100次计算平均消耗的时间为0.19ms，一次计算的平均时间为0.0676ms，误差小于0.5ms，此时不需要继续增加n的数值。

综上所述，当n取值为100时，此时SM3加密的时间约为0.0676ms~0.19ms。

## 5.4 SM4 加解密时间分析

本课题研究的对称密码算法是SM4，实现了其两种加密模式，一种是ECB，另一种是CBC模式，所以需要计算出在两种模式下分别的加解密时间。

### 5.4.1 ECB 模式加解密时间分析

第一步，固定明文为“hello world”，先计算出一次加解密操作的时间。

```
plainText = hello world  
ECB模式加密  
加密后的密文：449930d6f9d21f47c999d9727eee2bbc  
ECB模式加密所花费的时间：3ms  
解密后的明文：hello world  
ECB模式解密花费的时间：23ms
```

图5-10 SM4一次加解密时间

如图6-10所示，ECB模式下，SM4一次加密时间约为3ms，解密时间约为23ms，此时增大n的值。

第二步，将n取值为10，计算10次SM4算法加解密的平均时间。

```
ECB模式加密  
10次加密的平均时间：0.3ms  
10次解密的平均时间：1.7ms  
plainText = hello world  
ECB模式加密  
加密后的密文：449930d6f9d21f47c999d9727eee2bbc  
ECB模式加密所花费的时间：0.1793ms  
解密后的明文：hello world  
ECB模式解密花费的时间：0.3998ms
```

图5-11 SM4运行10次加解密时间

如图6-11所示，SM4算法的ECB模式下，10次加密的平均时间为0.3ms，解密的时间为1.7ms，而单次加密的时间为0.1793ms，单次解密时间为0.3998ms，解密的时间差大于0.5ms，需要增大n的值。

第三步，将n取值为100，计算100次SM4算法加解密的平均时间。

```
ECB模式加密
100次加密的平均时间: 0.06ms
100次解密的平均时间: 0.28ms
plainText = hello world
ECB模式加密
加密后的密文: 449930d6f9d21f47c999d9727eee2bbc
ECB模式加密所花费的时间: 0.061ms
解密后的明文: hello world
ECB模式解密花费的时间: 0.2378ms
```

图5-12 SM4运行100次加解密时间

如图6-12所示，SM4算法的ECB模式下，100次加密的平均时间为0.06ms，解密的时间为0.28ms，单次加密的时间为0.061ms，单次解密的时间为0.2378ms，此时，两者加密与解密的时间差值均小于0.5ms，符合要求。

综上所述，SM4的ECB模式下，加密的时间为0.06ms~0.061ms，解密的时间为0.2378ms~0.28ms。

#### 5.4.2 CBC 模式加解密时间分析

第一步，将明文固定为“hello world”，计算单次加解密的时间。

```
plainText = hello world
CBC模式加密
加密后的密文: 64be68cb00b44284c3a072b4ab7cdd32
CBC模式加密花费的时间: 4.6356ms
解密明文: hello world
CBC模式解密花费的时间: 28.7323ms
```

图5-13 SM4单次加解密时间

如图6-13所示，SM4算法的CBC模式下，单次加密时间为6.6356ms，单次解密时

间为28.7323ms，此时需要增加n的值。

第二步，将n取为10，计算10次SM4的平均加解密时间。

```
CBC模式加密
10次加密的平均时间: 0.3ms
10次解密的平均时间: 2.6ms
plainText = hello world
CBC模式加密
加密后的密文: 64be68cb00b44284c3a072b4ab7cdd32
CBC模式加密花费的时间: 0.2044ms
解密明文: hello world
CBC模式解密花费的时间: 0.6312ms
```

图5-14 SM4运行10次加解密时间

如图6-14所示，SM4算法的CBC模式下，10次加密的平均时间为0.3ms，解密的平均时间为2.6ms。此时，单次加密的时间为0.2044ms，单次解密的时间为0.6312ms，两者的解密时间的差值大于0.5ms，所以需要增加n的值。

第三步，将n取为100，计算100次SM4的平均加解密时间。

```
CBC模式加密
100次加密的平均时间: 0.06ms
100次解密的平均时间: 0.28ms
plainText = hello world
CBC模式加密
加密后的密文: 64be68cb00b44284c3a072b4ab7cdd32
CBC模式加密花费的时间: 0.0783ms
解密明文: hello world
CBC模式解密花费的时间: 0.2123ms
```

图5-15 SM4运行100次加解密时间

如图6-15所示，SM4算法的CBC模式下，100次加密的平均时间为0.06ms，解密的平均时间为0.28ms。此时，单次加密的时间为0.0783ms，单次解密的时间为0.2123ms，加密与解密的差值都小于0.5ms，符合要求。

综上所述，SM4算法CBC模式下，其加密的时间为0.06ms~0.0783ms，解密的时间为0.2123~0.28ms。

## 5.5 密码算法性能分析

本课题研究的三种密码算法加解密所需要的时间如下表所示：

表 5-1 加解密时间与 n 取值

	加密时间	解密时间	n取值
SM2	5ms	2ms	5000
SM3	0.0676ms~0.19ms		100
SM4	0.06ms~0.0783ms	0.2123~0.28ms	100

如表6-1所示，三种密码算法中，SM3因为其单向不可逆，所以不存在解密时间，SM4的两种模式下，选择加解密时间最大的数值作为表格数据。对于加密时间，SM4<SM3<SM2；对于解密时间，SM4<SM2。所以，对于密码算法的计算效率，SM2的加解密效率都要低于SM4。

对于SM4的两种模式的加解密，结果如下图所示：

表 5-2 SM4 两种模式的加解密时间

	加密时间	解密时间
ECB	0.06ms~0.061ms	0.2378ms~0.28ms
CBC	0.06ms~0.0783ms	0.2123~0.28ms

如表6-2所示，SM4算法ECB模式的加密时间是小于CBC模式，但是CBC模式的解密时间是小于ECB模式的解密时间。但是使用CBC模式下，同态性确定中，对于结果解密会出现数据丢失，所以最优的算法应使用SM4的ECB模式。

## 5.6 本章小结

本章主要研究了国产密码算法SM2，SM3，SM4三种密码算法的加解密效率，以及对它们进行了横向对比确实出最优的算法和加密模式。

下一章主要介绍传输认证系统的架构设计与实现原理。

## 第六章 传输认证系统设计与实现

### 6.1 实现方案

通过第五章实验可以看出，对于非对称密码算法效率较低，不适合用于对于数据的加密与解密操作，但因为其是非对称密码算法，使用公钥对数据加密，只有特定具有私钥的人才可以解密数据，所以可以用于对于通信双方的身份验证。

对于云计算环境下数据的传输，使用的加密算法必须具有一定的同态性，并且加解密算法会被频繁调用，所以需要使用性能较高的密码算法，可以使用对称密码算法SM4的ECB模式。

数据在网络中传输，不可避免的会被劫持和修改，为了避免此类情况的发生可对明文数据使用SM3算法计算散列值在加密，服务端收到后，解密数据拿到明文，最后对比散列值即可。服务端解密操作完全是基于内存的，不会将数据持久化到磁盘，也可以保护用户隐私数据的安全性。

非对称密码算法涉及到公钥的分发，对于客户端主动验证时，需要拿到相应的公钥，一般的公钥都是放在CA证书中，本课题研究可直接将公钥放在客户端，即客户端直接使用该公钥加密数据，再把数据发送给服务端。

### 6.2 系统架构

系统整体分为两个部分，客户端与服务端，客户端先与服务端建立TCP连接，然后再进行身份认证，当身份认证通过后，可以开始数据传输，用户输入的每个数据都会先计算哈希值，再加密，把密文与哈希值一起发送给服务端，服务端接收到数据后，先进哈希值校验，只有通过哈希值校验的数据才可以被处理，最后将处理的结果发送会客户端，客户端解密即可。

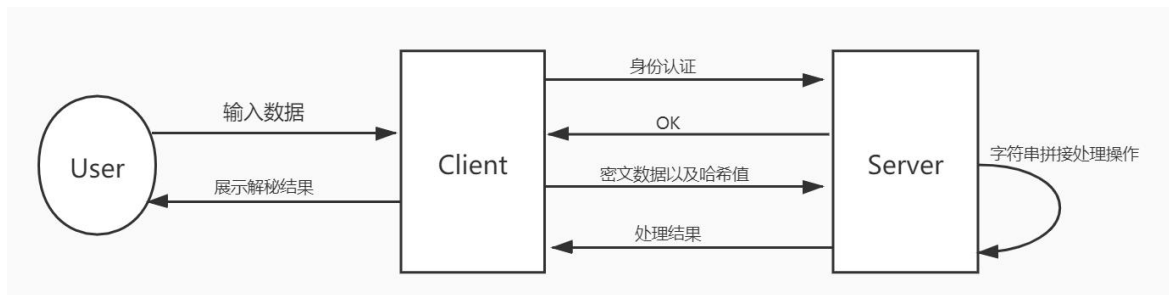


图6-1 系统架构图

图6-1展示了本系统的架构，可以看到，先启动了客户端，然后会与服务端进行身份认证，当服务端认同了客户端的身份后，会发送一个“OK”，客户端之后接收用户的输入拿到相应的密文数据，将数据加密以及哈希值发送给服务端，服务端处理后返回给客户端，客户端解密拿到明文结果。

### 6.3 实验分析

先启动服务端程序，在启动客户端程序，会自动进行身份认证，并且将认证结果打印出来。

```
开始进行身份认证...  
接收到服务端的数据为：OK  
身份认证成功!!!  
开始接收用户输入数据...
```

图6-2 客户端认证展示

如图6-2所示，客户端进行身份认证，使用SM2算法加密一个固定的消息“connect to server”发送给服务端，服务端收到后解密该消息做对比，判断是否同意客户端的身份。

```
服务器启动成功，等待客户端发送数据...  
-----  
第1次连接  
开始进行身份认证....  
通知客户端认证成功  
开始接收客户端输入的数据...
```

图6-3 服务端认证展示

如图6-3所示，服务端同意客户端身份后，会发送一个“OK”，客户端接收到后，判断是否为“OK”，如果是，则可以开始进行数据传输，否则，无法进行数据传输，即身份认证失败。

以传输“hello world”为例，用户每输入一个字符串，客户端均会对其加密并计算哈希值，然后拼接在一起发送给服务端。

```
请输入您的第1个字符串:  
hello  
生成的hash数值为: becbbfaae6548b8bf0cfcad5a27183cd1be6093b1cceccc303d9c61d0a645268  
加密后的密文为: a432c83e2fb356d420c9a674959b8832
```

图6-4 客户端密文以及哈希值展示

如图6-4所示，客户端对输入的hello计算了其哈希值并加密，最后将密文以及哈希值拼接在一起发送给服务端。

```
-----  
客户端发送的信息: a432c83e2fb356d420c9a674959b8832|becbbfaae6548b8bf0cfcad5a27183cd1be6093b1cceccc303d9c61d0a645268  
hash值校验成功
```

图6-5 服务端校验哈希值

如图6-5所示，服务端接收会对接收到的哈希值进行校验，对于可以成功校验的哈希值会打印出hash值校验成功。

之后可任意输入字符，以“#”作为结束标志，服务端对接收到的数据拼接处理，客户端解密后拿到结果。

```
world  
生成的hash数值为: ffaf99bfdde43053469fdbbd579bca72f83c6027ef9d2691398dbccd1f885c43  
加密后的密文为: 7bdc30499125d8c065ea6bb64e89ce4d  
请输入您的第3个字符串:  
#  
-----  
开始接收服务端返回的数据...  
总的解密的结果是: hello world  
分开解密的结果是: helloworld
```

图6-6 客户端接收到服务端结果展示

如图6-6所示，客户端接收到服务端的数据后解密展示给用户的结果。

## 6.4 本章小结

本章主要介绍了传输认证系统的架构涉及与实现原理，并且展示了系统具体的运行。

## 结 论

对于传输认证系统的设计可以使用国产密码算法SM4的ECB模式对数据进行加密，使用SM2在客户端启动时与服务端进行身份认证，通过SM3算法计算哈希值，最后来判断数据的完整性。

本课题研究的创新点在于云计算加密算法需要使用同态特性，通过SM4的ECB模式进行分组解密。随着国产密码技术的发展，国产密码算法的应用场景会越来越多，相应密码算法需要更多的特性来支撑各种场景。在云计算领域，密码算法的同态性至关重要，用户希望自己的数据传递上去不被云服务提供商所获得，云服务提供商只是提供相应的计算和存储能力，这个场景下，就需要云服务提供商只是对密文进行下相应的计算操作，将计算结果返回给用户，解密操作在用户端进行。同时，如果云服务提供商没有相应的解密方案，对于数据泄露问题也就可以得到有效的解决，即使数据泄露，但是数据是以密文的形式呈现的，无法获取到相应的明文信息，保护了用户隐私数据的安全。

本课题研究的密码算法只有SM2，SM3，SM4三种国产密码算法，密码算法选择的范围较小，今后可以选择更多的国产密码算法研究其同态性，在一个更大的范围去寻找全局最优解，也会去考虑将机器学习等与密码算法同态行进行结合。



## 参考文献

- [1] 杨竞. 同态加密关键技术研究[J]. 电子科技大学, 2019.
- [2] 李婷. 浅析云计算安全技术[J]. 机电信息, 2019 (33): 113-114.
- [3] 李曾鹏,马春光,周红生. 全同态加密研究[J]. 密码学报, 2017(06): 561-578.
- [4] 徐海霞. 云计算环境中改进的整数上全同态加密算法研究[J]. 科技通报, 2019(06): 87-92.
- [5] 杨浩淼,金保隆,陈诚,吴新沿. 一种高效的同态加密方案及其应用[J]. 密码学报, 2019(06): 611-619.
- [6] 洪家军,陈俊杰. 一种基于全同态加密的密文检索算法[J]. 廊坊师范学院学报(自然科学版), 2018 (04): 15-18+30.
- [7] 巩林明,李顺东,郭奕旻. 同态加密的发展及应用[J]. 中兴通讯技术, 2016(01): 611-619.
- [8] 程晋格. 国密算法在数据存储及码流数据传输中的应用[J]. 中国集成电路, 2018(07):15-18.
- [9] 王栋,李国春,俞学豪,陈智雨,葛冰玉,谢磊,谭静. 基于量子保密通信的国产密码服务云平台建设思路[J]. 电信科学,2018(07):171-178.
- [10] 刘悦,贾忠田,张波.结合国产密码算法的应用密码学课程教学探讨[J]. 计算机教育,2018(03):10-13.
- [11] 鲍海燕. 基于同态加密算法的网络信息安全保护[J], 现代计算机. 2019(24):22-25.
- [12] 桑杰,许雪姣,刘硕,蔡子凡. 基于国密算法的分布式加密存储研究[J], 数据通信.2020(01):9-12.
- [13] 卢希. 国产密码算法的安全、可信之路[J], 智能建筑与智慧城市, 2019(03):11.
- [14] 吴红英. 云计算下数据安全存储技术研究[J], 计算机产品与沟通, 2020(07):10
- [15] 黄延伟. 云计算背景下计算机安全问题策略[J], 网络安全技术与应用,2020(04):90-91.
- [16] 李鹏,李华,石永红. 云计算下的信息安全体系研究[J], 机械工程与自动化,2020(02):225-226.
- [17] 王佳,张远,刘超,杨婷婷.探讨云计算安全问题及其技术对策[J],科学技术创新,2020(10):52-53.
- [18] 贾凌杉,关艳. 同态加密技术在物联网中的应用[J], 科技风, 2018(17):90.
- [19] 王全福,宋文爱,杨顺民. 云环境中数据安全的同态加密方法[J], 计算机工程与设计,2017(01):42-46.
- [20] 柳玉东,王绪安,高忠石. 基于同态加密算法的欧式距离外包计算协议[J], 计算机工程与应用,2019(05):110-116.

## 致 谢

值此论文完成之际，首先向我的导师闫怀志老师表示感谢。我的论文是在老师的指导下修改完成的，正是因为他细心帮助和耐心的指导，我才会完成本论文，在此，对于闫老师表示由衷的感谢。

感谢所有在大学期间传授我知识的老师，不积跬步无以至千里，本论文能够顺利完成，也归功于各位任课老师的认真负责，使我能够很好的掌握和运用专业知识，并在实验中得以体现。正是有了你们的悉心帮和支持，才使我的毕业论文工作顺利完成，在此向北京理工大学计算机学院的全体老师表示由衷的谢意，感谢你们四年来的辛勤栽培。

感谢我的爸爸妈妈，在日常交流中您们用的最多的两个字就是加油，感谢父母一直以来给我不断的鼓励，这些鼓励转化为我前进的动力，学无止境，我将努力做一个对社会有用的人，感谢亲人的关心，愿爸爸妈妈永远健康快乐。

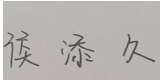
最后，我要感谢专业的同学们，感谢他们的鼓励和支持，感谢他们和我一路走来，让我在此过程中倍感温暖！

感谢所有关心和帮助过我的老师、同学、朋友，谢谢你们！

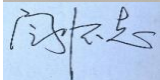
# 北京理工大学本科生软件工程专业 毕业设计（论文）毕业要求达成度评价表

学生姓名：侯添久

学号：1120161912

学生签字：

导师姓名：闫怀志

导师签字：

毕业要求	毕业要求指标点	分值	学生自评	指导教师评价
毕业要求6.工程与社会	6.1 能够了解应用领域背景知识，完成复杂软件系统的需求分析，说明其合理性	15	4	4
毕业要求6.工程与社会	6.3 能够采用适当的方法评价工程实践对社会、健康、安全、法律以及文化的影响，并理解应承担的责任	20	4	4
毕业要求7.环境和可持续发展	7.1 能够了解软件工程及相关行业的政策和法律法规，了解国内外行业标准、规范和技术发展趋势	10	4	4
毕业要求7.环境和可持续发展	7.2 能够理解复杂软件工程问题的专业实践和对环境以及社会可持续发展的影响	20	3	3
毕业要求10.沟通	10.2 能够具备一定的国际视野，掌握一门外语，能够了解和跟踪软件工程专业的最新发展趋势，具有跨文化交流和沟通能力	25	3	3
毕业要求12.终身学习	12.2 能够养成主动学习习惯，运用科学的学习方法管理知识和处理信息，表现出不断探索的成效，能够自我评价	10	4	4
最终评分	89			

说明：

1. 成绩计算方法为： $\Sigma$ （分值\*学生自评等级分数）/8+ $\Sigma$ （分值\*教师评价等级分数）/8

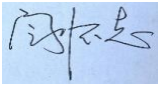
2. 学生自评和教师评价采取四级评分制，评定等级为“非常满意、满意、不满意、非常不满意”，分别对应分数为4、3、2、1，请填写分数。

**北京理工大学本科生软件工程专业  
毕业设计（论文）过程及质量跟踪评价表**

学生姓名：侯添久

学生学号：1120161912

导师姓名：闫怀志


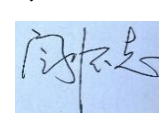
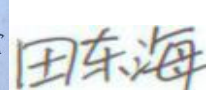
导师签字：

序号	毕业设计过程及论文质量监控点	分值	评价
1	完成了任务书的各项要求，软件运行正常或算法达到预期效果	25	3
2	论文结构合理，内容充实、结论可信	20	3
3	能够掌握一门外语，外文翻译准确流畅，具有跨文化交流和沟通能力	15	4
4	充分应用了软件工程相关理论和技术，并能够用形式化模型和文档等形式呈现软件系统解决方案和成果	10	4
5	学生积极接受指导老师指导，至少2周进行一次直接指导，表现出主动学习和不断探索的习惯，能够自我评价	20	3
6	论文结果有创新性结果或新颖之处，表现出进行深入研究的能力	10	4
最终评分	84		

说明：

- 1. 成绩计算方法为： $\Sigma（分值*评定等级分数）/4$
- 2. 指导教师对学生以上各项做出评价，采取四级评分制，评定等级为“非常满意、满意、不满意、非常不满意”，分别对应分数为4、3、2、1，请填写分数。

北京理工大学计算机学院本科生毕业设计（论文）软件验收表

学生姓名	侯添久	学号	1120161912	导师姓名	闫怀志
论文题目	基于国产密码算法的云计算网络信息传输认证系统设计与实现				
验收组成员名单	姓 名	专业技术职务	单 位		
组 长	戴银涛	副教授	北京理工大学		
组 员	闫怀志	副教授	北京理工大学		
组 员	田东海	讲师	北京理工大学		
验收时间:	2020 年 6 月 18 日				
验收资料清单	项目代码, 任务书, 软件运行视频				
源语言/开发环境	Java, Windows 10		运行环境/ 系统配置	Linux/JAVA_HOME	
总代码行数/字节数 (KB)	2978/440KB		手工编写代码行数	853	
软件运行状况	正常运行, 可使用相应的国产密码算法进行加解密操作以及可以通过 socket 发送任意数据。				
软件特点及应用情况	软件分为客户端与服务端, 客户端使用国产密码算法对用户输入的数据加解密操作, 服务端步进行解密并且对密文操作后将结果返回给客户端。				
验收结论 (明确说明是否运行正常, 功能是否与论文一致, 最后须给出定量的百分制结论)					
运行正常, 功能与论文一致, 与论文阐述功能一致性百分比为 98%。					
验收组长签字: 					
组员签字:  					
2020 年 6 月 18 日					

# 软件工程专业本科毕业设计（论文）形式审查表（一）

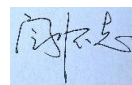
学生姓名：侯添久

学号：1120161912

电话：13051172966

导师初审

导师签字：

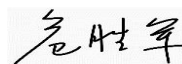


论文封皮	√	答辩委员会名单及签字	√	正文格式	√
任务书首页	√	提问及回答	√	不使用人称代词	√
题目类别	√	代表签字	√	字体字号	√
题目性质	√	答辩评语	√	图表正确编号 X-Y	√
论文题目	√	答辩委员签字	√	图表正确引用	√
题目内容	√	成绩及时间	√	单页空白少于 3 行	X
任务要求	√	委员会主任签字	√	页眉页脚页码	√
具体内容	√	论文起止时间	√	参考文献	√
进度安排	√	答辩日期	√	外文翻译原文	√
指导教师签字	√	论文理论引用<20%	√	翻译中文及页眉	√
教学单位签字	√	论文抄袭	√	翻译内容及质量	√
责任教授签字	√	中文摘要	√	翻译封皮	√
指导教师评语	√	英文摘要	√	形式审查表	√
评阅教师评语	√	目录及格式		软件验收表	√
评语表（二）	√	正文前各页页码	√	资料袋	—

注：√表示该项核查后符合要求，X表示该项核查不符合要求

各答辩组复审

审查人签字：



论文封皮	√	答辩委员会名单及签字	√	正文格式	√
任务书首页	√	提问及回答	√	不人称代词	√
题目类别	√	代表签字	√	字体字号	√
题目性质	√	答辩评语	√	图表正确编号 X-Y	√
论文题目	√	答辩委员签字	√	图表正确引用	√
题目内容	√	成绩及时间	√	单页空白少于 3 行	X
任务要求	√	委员会主任签字	√	页眉页脚页码	√
具体内容	√	论文起止时间	√	参考文献	√
进度安排	√	答辩日期	√	外文翻译原文	√
指导教师签字	√	论文理论引用<20%	√	翻译中文及页眉	√
教学单位签字	√	论文抄袭	√	翻译内容及质量	√
责任教授签字	√	中文摘要	√	翻译封皮	√
指导教师评语	√	英文摘要	√	形式审查表	√
评阅教师评语	√	目录及格式	√	软件验收表	√
评语表（二）	√	正文前各页页码	√	资料袋	—