**基于SGX的多方数据安全计算密码模块**

**设计与实现**

**作者姓名**

**学校导师姓名、职称 教授**

**企业导师姓名、职称 高工**

**申请学位类别 电子信息硕士**

**基于SGX的多方数据安全计算密码模块**

**设计与实现**

**西安电子科技大学**

**硕士学位论文**

**作者姓名：**

**领　　域：**网络与信息安全

**学位类别：**电子信息硕士

**学校导师姓名、职称：** 教授

**企业导师姓名、职称：** 高工

**学　　院：**广州研究院

**提交日期：**2024年3月

**学　号**

**密　级　 公开**

**学校代码 10701**

**分类号 TP309**

Supervisor: Li Si

Supervisor: Wang Wu

Title: Professor

Title: Senior Engineer

A thesis submitted to

XIDIAN UNIVERSITY

in partial fulfillment of the requirements

for the degree of Master

in Electronic Information

**Cryptographic Module for Multi-Party Data Security Computation based on SGX**

By

Zhang San

March 2024

摘要

在数字化和数据驱动的时代背景下，数据生成的场景日益多样化，数据规模急剧增大。通过采集和深度分析用户相关数据，应用可以为用户提供更加精准化和个性化的服务；企业或组织之间通过数据共享和合作，能够有力推动相关领域的创新和发展。然而，这些数据中往往包含大量敏感信息，对这些数据的不当操作会极大增加数据隐私泄露的风险。近年来，数据隐私泄露事件频发。因此，如何在确保用户隐私信息得到充分保护的前提下，实现多方的数据安全计算，成为当下亟待解决的关键问题。

多方的数据安全计算体系的构建面临以下挑战：首先，由于数据通常分散存储在各个数据采集者的环境中，两方或多方进行数据统筹使用的前提是数据交换。然而多方的数据安全计算体系建立在开放信道之上，实体间通信容易遭受恶意拦截、欺骗攻击、信息窃取等攻击，从而导致多方的数据安全计算体系崩溃、用户信息泄露等危害。其次，如何在海量隐私数据的本地处理和计算中，同时兼顾安全性和效率。保护数据隐私安全的传统方式主要利用密码学方法，如同态加密或对称加密，但同态加密带来的计算开销较大；对称加密不能保证数据明文状态下处理的安全性。

针对上述问题，本文做了如下工作：

（1）设计与实现了一种数据安全交换传输协议，作为多方的数据安全计算体系内实体间数据安全交换的基础，并通过攻击者模型和安全性分析证明协议的安全性。该协议包括认证密钥协商、会话密钥生成和数据传输三个部分。在认证密钥协商阶段，通信实体基于一个两轮密钥协商协议实现相互认证和传输密钥材料的协商；在密钥生成阶段，通信实体使用双阶段密钥派生函数，结合传输密钥材料、系统内预存秘密以及当前的上下文信息生成初始会话密钥，并使用单阶段密钥派生函数更新会话密钥；在数据传输阶段，通信实体使用生成的会话密钥对消息进行加解密，并生成对应的消息认证码，从而充分满足通信的安全需求。

（2）设计与实现了一个增强国密算法执行安全的框架，基于可信执行环境SGX技术保护国密算法运行过程。隐私数据的本地处理包括脱敏、置乱和加解密等操作，均基于国密算法实现。为了防止攻击者在隐私数据以明文形式参与计算时实施非法操作，本文利用SGX的高安全特性，将国密算法应用划分为可信部分和非可信部分，为可信部分构建Enclave安全隔离区域，将密钥预处理、加解密、杂凑值计算、签名生成与验证等关键操作放入Enclave内执行，非可信区域只获取最终计算结果，无法访问或篡改Enclave中的内容，从而形成安全可信框架，有效保护国密算法运行时的密钥、数据以及中间值等信息。通过测试验证，基于SGX的安全增强国密算法展现出良好的实用性。

本文基于上述方案，设计与实现了基于SGX的多方数据安全计算密码模块，该密码模块包括身份认证模块、密钥生成模块和加解密模块，为多方的数据安全计算提供安全增强的密码服务。在实验环境下对密码模块进行了功能和性能测试，证明了方案的可行性和实用性。

**关 键 词**：可信执行环境， Intel SGX， 认证密钥协商， 国密算法， 密钥派生

ABSTRACT

In the context of the digitalization and data-driven era, the scenarios of data generation are becoming increasingly diversified and the scale of data is increasing dramatically. Through the collection and in-depth analysis of user-related data, applications can provide users with more precise and personalized services; and through data sharing and cooperation among enterprises or organizations, innovation and development in related fields can be strongly promoted. However, these data often contain a large amount of sensitive information, and improper manipulation of these data can greatly increase the risk of data privacy leakage. In recent years, there have been frequent incidents of data privacy leakage. Therefore, how to realize data secure computation for multiple parties under the premise of ensuring that users' private information is adequately protected has become a key issue that needs to be solved urgently.

The construction of a multi-party data secure computation system faces the following challenges: First, since data is usually stored scattered in the environments of various data collectors, the premise of data co-ordination and use by two or more parties is data exchange. However, the multi-party data security computing system is built on an open channel, and inter-entity communication is prone to malicious interception, spoofing attacks, information theft, and other attacks, which can lead to the collapse of the multi-party data security computing system, leakage of user information, and other hazards. Second, how to take into account both security and efficiency in the local processing and computation of massive privacy data. Traditional ways to protect data privacy security mainly utilize cryptographic methods, such as homomorphic encryption or symmetric encryption, but homomorphic encryption brings large computational overhead; symmetric encryption does not guarantee the security of data processing in plaintext state.

In response to these issues, this paper has undertaken the following work:

(1) A data security exchange transmission protocol is designed and implemented as a basis for secure data exchange between entities within a multi-party data security computing system, and the security of the protocol is demonstrated through attacker modeling and security analysis. The protocol consists of three parts: authentication key agreement, session key generation and data transmission. In the authentication key agreement phase, the communicating entities realize mutual authentication and agreement of the transmission key material based on a two-round key agreement protocol; in the key generation phase, the communicating entities use a two-phase key derivation function to generate an initial session key by combining the transmission key material, the preexisting secrets in the system, and the current contextual information, and use a single-phase key derivation function to update the session key; in the data transmission phase, the In the data transmission phase, the communication entity uses the generated session key to encrypt and decrypt the message, and generates the corresponding message authentication code, so as to fully satisfy the security requirements of communication.

(2) A framework for enhancing the execution security of national cryptographic algorithm is designed and implemented to protect the operation process of national cryptographic algorithm based on the trusted execution environment SGX technology. The local processing of private data, including desensitization, scrambling and encryption/decryption operations, are implemented based on the national cryptographic algorithm. In order to prevent attackers from illegal operation when the private data are in plaintext form, this paper utilizes the high security features of SGX to divide the application of the national cryptographic algorithm into credible and non-credible parts, and builds the Enclave security isolation area for the credible part, and puts key operations such as key preprocessing, encryption/decryption, hash value calculation, signature generation and verification into the Enclave for execution, while the non-credible area only obtains the final calculation results and cannot be used to protect the operation process of the national cryptographic algorithm. The non-trusted region only obtains the final calculation results and cannot access or tamper with the content in the Enclave, thus forming a secure and trustworthy framework that effectively protects the key, data, intermediate values and other private information during the operation of the national cryptographic algorithm. Through testing and verification, the SGX-based security-enhanced national cryptographic algorithm shows good practicability.

Based on the above scheme, this paper designs and implements a multi-party data security computation cryptographic module based on SGX, which includes an authentication module, a key generation module, and an encryption and decryption module to provide security-enhanced cryptographic services for multi-party data security computation. Functional and performance tests of the cryptographic module are carried out in an experimental environment, which proves the feasibility and practicability of the scheme.

**Keywords**: Trusted Execution Environment, Intel SGX, Authentication Key Agreement,

National Cryptographic Algorithm, Key Derivation

插图索引

[图1.1 研究内容关系 6](#_Toc28763)

[图2.1 SGX内存访问控制 10](#_Toc2310)

[图2.2 SGX密封与解封 10](#_Toc4445)

[图2.3 SGX本地认证流程 11](#_Toc15552)

[图2.4 SGX远程认证流程 12](#_Toc31503)

[图2.5 对称密码体制流程 13](#_Toc5544)

[图2.6 非对称密码体制签名和认证流程 14](#_Toc16510)

[图2.7 消息认证码认证流程 16](#_Toc7819)

[图3.1 加密系统模型 20](#_Toc15724)

[图3.2 Logistic系统分岔图 21](#_Toc30625)

[图3.3 Tent系统分岔图 21](#_Toc32424)

[图3.4 Logistic-Tent 22](#_Toc6672)

[图3.5 Chen系统 22](#_Toc524)

[图3.6 Chen系统迭代图 23](#_Toc30833)

[图3.7 比特填充 24](#_Toc13160)

[图3.8 Chen系统迭代图 30](#_Toc27777)

[图3.9 Chen系统迭代图 30](#_Toc30191)

[图3.10 Chen系统迭代图 30](#_Toc9065)

[图3.11 Chen系统迭代图 32](#_Toc17886)

[图4.1 HMAC-SM3算法在两种环境下性能对比 43](#_Toc1348)

[图4.2 SM4 ECB算法在两种环境下性能对比 43](#_Toc7280)

[图4.3 SGX国密算法安全性测试 44](#_Toc14792)

[图5.1 基于SGX的多方数据安全计算密码模块 47](#_Toc20572)

[图5.2 通信数据包格式 48](#_Toc15164)

[图5.3 客户端密钥协商测试 53](#_Toc16247)

[图5.4 服务端密钥协商测试 53](#_Toc12836)

[图5.5 客户端密钥生成测试 54](#_Toc12848)

[图5.6 服务端密钥生成测试 54](#_Toc26442)

[图5.7 服务端发送消息密文 54](#_Toc12273)

[图5.8 客户端解密得到交换数据明文 55](#_Toc18187)

表格索引

[（4） 常量数组的初始化：该数组中的值来自前64个质数（从2到311）对应立方根的小数部分的前32位。所有常量的16进制表示形式按顺序排列，如表3.1所示。 25](#_Toc29629)

[表3.1 常量数值表 25](#_Toc2067)

[表5.2 可信接口执行时间（单位：ms） 56](#_Toc27261)

符号对照表

|  |  |
| --- | --- |
| 符号 | 符号名称 |
|  | 长期密钥对 |
|  | 临时密钥对 |
|  | 数字证书 |
|  | 身份标识 |
|  | 随机数 |
|  | 随机数基点乘值 |
|  | 数据包序号 |
|  | 椭圆曲线的基点 |
|  | 椭圆曲线基点的阶 |
|  | 传输密钥材料 |
|  | 伪随机密钥 |
|  | 私有字符串 |
|  | 随机盐值 |
|  | 上下文信息 |
|  | 密钥派生函数 |
|  | 会话密钥 |
|  | 消息 |
|  | 加密函数 |
|  | 签名函数 |
|  | 消息认证码函数 |

缩略语对照表

|  |  |  |
| --- | --- | --- |
| 缩略语 | 英文全称 | 中文对照 |
| TEE  SGX | Trusted Execution Environment  Software Guard Extensions | 可信执行环境  软件保护扩展 |
| FE | Functional Encryption | 函数加密 |
| KME | Key Management Enclave | 密钥管理飞地 |
| TCB | Trusted Computing Base | 可信计算基 |
| SAKA | Identity-Based Authentication Key Agreement | 基于身份的认证密钥协商 |
| KGC | Key Generation Center | 密钥生成中心 |
| KDF | Key Derivation Function | 密钥派生函数 |
| MAC | Message Authentication Code | 消息认证码 |
| PBKDF | Password-Based Key Derivation Function | 基于口令的密钥派生函数 |
| PRF | Pseudo-Random Function | 伪随机函数 |
| SCKDF | Stream Cipher-based Key Derivation Function | 基于流密码的密钥派生函数 |
| MFKDF | Multi-Factor Key Derivation Function | 多因子密钥派生函数 |
| RSK | Root Seal Key | 根密封密钥 |
| IAS | Intel Attestation Service | 英特尔认证服务 |
| LA | Local Attestation | 本地认证 |
| RA | Remote Attestation | 远程认证 |
| RSK | Root Seal Key | 根密封密钥 |
| IAS | Intel Attestation Service | 英特尔认证服务 |
| HMAC | Hash-based Message Authentication Code | 基于哈希函数的消息认证码 |
| ECC | Elliptic Curve Cryptography | 椭圆曲线密码学 |
| CMAC | Cipher-based Message Authentication Code | 基于分组密码的消息认证码 |
| EPC | Enclave Page Cache | 飞地页面缓存 |

目录

[摘要 I](#_Toc22763)

[ABSTRACT III](#_Toc12497)

[插图索引 VII](#_Toc21252)

[表格索引 IX](#_Toc29800)

[符号对照表 X](#_Toc10031)

[缩略语对照表 XII](#_Toc32050)

[第一章 绪论 1](#_Toc22164)

[1.1 研究背景及意义 1](#_Toc1244)

[1.2 国内外研究现状 2](#_Toc9387)

[1.2.1 密钥协商协议 2](#_Toc30586)

[1.2.2 密钥派生方案 3](#_Toc4553)

[1.2.3 SGX技术应用 4](#_Toc20052)

[1.3 主要研究内容 5](#_Toc27450)

[1.4 论文组织结构 7](#_Toc32293)

[第二章 背景知识 9](#_Toc9470)

[2.1 Intel SGX关键技术 9](#_Toc31827)

[2.1.1 内存隔离机制 9](#_Toc4773)

[2.1.2 密封机制 10](#_Toc15081)

[2.1.3 本地认证 11](#_Toc32739)

[2.1.4 远程认证 12](#_Toc14678)

[2.2 密码体制 13](#_Toc28359)

[2.2.1 对称密码体制 13](#_Toc30790)

[2.2.2 非对称密码体制 14](#_Toc12485)

[2.3 杂凑算法和消息认证码 15](#_Toc20743)

[2.3.1 杂凑算法 15](#_Toc30959)

[2.3.2 消息认证码 15](#_Toc30045)

[2.4 Dolev-Yao攻击者模型 16](#_Toc14608)

[2.5 本章小结 17](#_Toc17075)

[第三章 基于混沌系统的图像加密方案 19](#_Toc26758)

[3.1 加密系统模型 19](#_Toc11269)

[3.2 混沌模型的选取 20](#_Toc2834)

[3.2.1 一维混沌体系的选取 20](#_Toc22526)

[3.2.2 三维混沌系统的选取 22](#_Toc16114)

[3.3 混沌序列的生成方式 23](#_Toc15644)

[3.3.1 生成唯一标识码 23](#_Toc5377)

[3.3.2 生成混沌序列 27](#_Toc5078)

[3.4 QR码的置乱与扩散 29](#_Toc19129)

[3.4.1 QR码的生成 29](#_Toc30491)

[3.4.2 QR码的置乱操作 31](#_Toc32064)

[3.4.3 QR码的扩散操作 32](#_Toc3493)

[3.5 实验分析 34](#_Toc26090)

[3.6 本章小节 34](#_Toc10989)

[第四章 改进的DCT域图像数字水印 35](#_Toc29662)

[4.1 算法模型介绍 35](#_Toc3275)

[4.2 离散余弦变换（DCT）介绍 35](#_Toc29625)

[4.2.1 DCT变换的原理 35](#_Toc17154)

[4.2.2 DCT的系数介绍 37](#_Toc15712)

[4.2.3 DCT嵌入方式 39](#_Toc30795)

[4.3 结合HVS的分块图像排序算法 39](#_Toc24937)

[4.3.1 人类视觉系统HVS 39](#_Toc20639)

[4.3.2 分块图像排序原理 40](#_Toc23232)

[4.4 图像水印的嵌入算法 42](#_Toc28356)

[4.4.1 分块图像的信息嵌入与提取 42](#_Toc11887)

[4.4.2 安全性分析 43](#_Toc888)

[4.5 实验结果和分析 44](#_Toc29365)

[4.6 本章小节 44](#_Toc6820)

[第五章 基于SGX的密码模块设计与实现 47](#_Toc16340)

[5.1 密码模块设计 47](#_Toc16256)

[5.2 密码模块实现 48](#_Toc24629)

[5.2.1 传输协议数据包定义 48](#_Toc5784)

[5.2.2 身份认证模块 48](#_Toc731)

[5.2.3 密钥生成模块 51](#_Toc17664)

[5.2.4 加解密模块 51](#_Toc2616)

[5.3 密码模块测试 53](#_Toc30279)

[5.3.1 测试环境 53](#_Toc8932)

[5.3.2 功能测试 53](#_Toc7910)

[5.3.3 性能测试 55](#_Toc26816)

[5.4 本章小结 56](#_Toc18239)

[第六章 总结与展望 57](#_Toc22160)

[6.1 工作总结 57](#_Toc19296)

[6.2 未来工作展望 58](#_Toc23318)

[参考文献 59](#_Toc7727)

[作者简介 63](#_Toc2086)

# 绪论

## 研究背景及意义

随着信息时代的蓬勃发展，各行各业正积极推动数字化转型，越来越多的社会资源和生产资料被数字化，这使得当下社会数据量呈现出爆炸式增长的趋势，这种现状的主要特征在于数据的来源多样化、数据产生速度加快、数据规模急剧增大等特点。根据国际数据公司IDC发布的《数字化世界—从边缘到核心》和《2025年中国将拥有全球最大的数据圈》两份报告，全球数据圈将从2018年的33ZB增至2025年的175ZB。其中，2018年，中国共产生了7.6ZB数据，预计2025年我国数据总量有望增至48.6ZB，占全球数据总量的27.8%。这些数据一方面来源于互联网，例如微信、抖音等社交平台，以及支付宝、手机银行等移动支付平台；另一方面源自各种社会活动，例如患者就医生成的医学影像数据、科研工作者实验产生的实验数据等。

随着数据量的快速增长，数据安全问题日益凸显。因系统漏洞、外部网络攻击等造成的数据泄露会造成严重而深远的影响。据《卫报》报道，2018年3月，5000万Facebook用户数据被非法操纵，用于建立复杂的心理学用户画像，并据此针对性地推送政治广告。2022年12月，美国最大的综合医疗服务网站遭遇勒索软件攻击，大量患者的姓名、电话、诊断结果和使用药物等敏感信息遭泄露。据IBM统计，2023年全球数据泄露的平均成本为445万美元，三年内增长了15%。科技公司、银行业、政府部门、教育、医疗等行业已成为数据泄露重灾区。

数据作为信息的载体，蕴含着巨大的潜在价值。通过多方的数据安全计算[1]，企业或相关组织机构可以实现数据的统筹使用，从而增强客户体验、优化产品和服务、驱动商业决策、以及推动科研创新等。然而这些数据通常分散存储在各个数据采集者的环境中。由于数据包含了大量的用户隐私信息，且多方的数据安全计算体系建立在开放信道之上，实体间通信容易遭受恶意拦截、欺骗攻击、信息窃取等攻击，这可能导致多方的数据安全计算体系崩溃、进而引发用户信息泄露等危害。因此，建立安全传输通道对实现数据共享至关重要。

目前，数据的安全主要通过密码学手段保证。多方的数据安全计算过程涉及数据加解密、脱敏、置乱等操作，均基于密码算法实现。然而传统密码算法不能保障数据在明文状态下的处理过程，在面对恶意软件和系统漏洞等复杂威胁时仍显不足。虽然同态加密技术[2]支持数据处于密文状态下进行计算，但会带来极大的计算开销，并不适用于处理海量数据的场景。近年来，可信执行环境[3]（Trusted Execution Environment，TEE）的出现为保护数据在计算过程中的安全提供了新的解决方案。Intel SGX[4]（Software Guard Extensions）作为一种先进的TEE技术，可以保护应用程序内部的敏感数据和代码免受恶意攻击。微软、谷歌和IBM的机密计算服务均采用了Intel SGX技术，通过硬件级别的内存加密和隔离，确保数据在云服务器中的安全计算。可见，SGX作为一种先进的硬件加密技术，在保护数据隐私和安全方面具有显著优势，是响应当前数字化时代安全挑战的重要工具。

综上所述，本文提出一种数据安全交换传输协议和基于可信执行环境技术保护国密算法运行安全性的方法，基于这两个核心设计原则，设计并实现了基于SGX的多方数据安全计算密码模块。密码模块包括身份认证模块、密钥生成模块和加解密模块。身份认证模块负责实体间的认证与密钥协商；密钥生成模块负责会话密钥的生成和定期更新；加解密模块负责实体间数据安全交换。SGX可信执行环境被应用到密码模块中，保障数据在计算过程中的安全性。密码模块中采用国密算法替代国际密码算法来保障数据的安全性，进一步加强国密算法在多方的数据安全计算体系中的应用，实现自主可控的多方数据安全计算平台。

## 国内外研究现状

### 密钥协商协议

密钥协商协议是两个或多个通信实体之间建立共享密钥，在开放信道上进行安全通信的关键机制。因此，设计一个同时具备高效性和安全性的密钥协商协议一直是数据安全传输领域的研究热点。

1976年，Diffie和Hellman[5]首创了公钥密码学的概念，并提出了著名的Diffie-Hellman协议，然而该协议并没有提供对抗中间人攻击的机制。Matsumoto等[6]在保持Diffie-Hellman协议简洁性和高效性的同时，引入了身份认证机制，提出了身份认证密钥协商方案。MQV协议由Law等[7]提出，充分考虑了包括中间人攻击、重放攻击等多种潜在的攻击方式，其在密钥协商的效率上有着更好的表现。为了证明认证密钥协商协议的安全性，需要定义一个明确的对手模型。Bellare等[8]首次将可证明安全的理论应用于认证密钥协商协议的研究，提出了认证密钥协商协议的第一个对手模型。随后，Canetti和Krawczyk[9]对该模型进行了扩展，提出了著名的CK模型，但该模型不能保证协议抵抗临时秘密泄露等多种攻击。LaMacchia等[10]通过考虑更强大的对手，进一步提出了扩展的eCK模型，在该模型中，对手可以破坏永久私钥或临时秘密密钥。

基于预共享密钥技术和Diffie-Hellman方案，Seo等[11]提出了一个简单的基于身份的认证密钥协商（Identity-Based Authentication Key Agreement，SAKA）协议。Sun[12]和Tseng[13]在他们各自的研究中分别揭示了SAKA方案存在的安全性问题，他们证明了该方案不能抵抗重放攻击和离线密码猜测攻击。Shim[14]提出了一种高效的基于ID的认证密钥协商协议。Oh等[15]证明了Shim[14]的协议不提供密钥生成中心（Key Generation Center，KGC）前向保密性，然后提出了一个提供该安全性的协议，并可以生成两个安全会话密钥来保障消息的安全传输。Abi-Char等[16]提出了一种带有安全认证和密钥确认的三轮认证密钥协商协议，该协议不仅可以抵抗多种攻击，还提供完美的前向保密性，可以保护历史会话和密钥。Cao等[17]提出了一种高效的SAKA协议，在无需额外成本的情况下最大限度减少了消息交换时间。Hou和Xu[18]证明了Cao等[17]的协议不能抵抗已知的特定于会话的临时信息攻击，并提出了一种具有增强安全属性的认证密钥协商协议。Farash和Attari[19]扩展了Cao等[17]的协议，基于椭圆曲线提出了一个更高效的协议，使得独立组织或网络中拥有独立服务器的两个用户可以建立安全会话。高等[20]提出了一种高效的AKA协议，该协议建立在身份加密的基础之上，并且在标准模型下证明了其安全性。赵等[21]提出了一种新的两轮两方认证密钥协商协议，并在eCK模型下对新协议进行了严格的形式化证明。陈等[22]提出了一个基于属性的高效认证密钥协商协议，通过性能分析和实验验证，证明了该协议在计算效率方面的优势。

目前，密钥协商领域已取得了众多研究成果，但研究者们仍致力于该领域的创新与完善工作。近年来，密钥协商协议的研究正逐步与时代的技术趋势相结合，以应对新兴技术带来的挑战和机遇，确保在多变的技术环境中保护数据的安全和隐私。

### 密钥派生方案

密钥派生是指从一个给定的密钥材料中派生出一个或多个新密钥的过程。目前，密钥派生函数（Key Derivation Function，KDF）可以通过诸如哈希函数、分组密码和流密码等密码原语构建。基于哈希函数和分组密码的消息认证码（Message Authentication Code，MAC）算法将可变大小的输入转换为固定长度的输出，而流密码依赖于密钥流生成器将短密钥和公共信息转换为任意长度的输出密钥。

Krawczyk[23]对KDF的研究和工程实现做出了贡献，缩小了理论与实践之间的差距。Krawczyk提供了基于先提取再扩展的KDF详细设计原理，并设计了一个基于哈希函数的实用KDF，可以在多种应用程序中标准化使用。Suresha等[24]利用基于密钥派生的加密技术来增强数据保护，系统的设计涉及从一个主密钥通过安全哈希函数派生三个独立的密钥，分别用于数据修改、数据加密、数据来源验证，该机制为存储在云中的数据提供了可修改性、机密性和身份认证功能。Gueron等[25]提出了一种基于截断分组密码的新密钥派生方案，该方案被证明极大改进了多种操作模式的安全边界，可以显著延长密钥的生命周期。

Chuah等[26]提出了一种安全且高效的基于流密码的密钥派生函数（Stream Cipher-based Key Derivation Function，SCKDF）。在该设计中，随机性提取和扩展被划分为两个独立的子功能，可以分别进行设计，在安全性和性能方面形成良好的先提取再扩展的KDF方案。该方案使用三种密码算法进行了测试，结果表明，当采用高效的流密码进行构造时，SCKDF在软件中的执行速度显著超越了一些基于哈希算法和分组密码的KDF。

基于用户口令生成密钥也是一种非常流行的密钥派生方式。基于口令的密钥派生函数（Password-Based Key Derivation Function，PBKDF）使用用户提供的口令、随机生成的盐值和迭代计数器生成密钥。PBKDF在多种场景中得到广泛使用和部署，如无线网络安全协议、密码管理应用程序以及Windows和iOS操作系统中[27]。Nair等[28]通过扩展PBKDF，提出了首个多因素密钥派生函数（Multi-Factor Key Derivation Function，MFKDF）的通用结构。MFKDF在密钥派生过程中引入身份认证因素，如TOTP、HOTP和硬件令牌。通过这种方式，用户数据首次受到身份认证因素的直接加密保护。相比PBKDF，MFKDF提供了指数级安全改进，在典型的Web浏览器中，其额外的计算开销不到12毫秒。

### SGX技术应用

随着针对数据中心的攻击手段日益增强和攻击类型不断演化，需要更高级别的安全性以避免危险。Intel公司于2013年推出了SGX技术，这是一种前沿的处理器安全技术，该技术通过应用新的指令集和内存访问机制，为用户的应用程序提供可信的执行环境，确保在处理敏感任务时，用户数据的机密性和完整性得到有效的保障。众多研究人员正积极探索将SGX技术与现有成果相结合，以开创更多的安全解决方案，进一步提升数据的安全性。

Fisch等[29]设计了Iron，一个基于SGX的实用函数加密（Functional Encryption，FE）系统。在Iron的高层架构中，系统创建了一个密钥管理飞地（Key Management Enclave，KME），Enclave作为一个安全隔离的区域，其关键作用在于通过加密应用程序中的敏感信息来抵御恶意攻击。KME负责维护主密钥的可信管理，确保标准公钥加密系统和签名方案的安全授权。Priebe等[30]提出，可以使用SGX来保护数据库中的数据安全，他们研发了EnclaveDB，这是一个更安全的数据库引擎，即使在不安全的云主机上运行，或者遇到恶意的数据库管理员时，它也能提供抵御相应攻击的能力。EnclaveDB将所有敏感信息、存储和查询操作、事务处理功能都置于Enclave的可信计算基（Trusted Computing Base，TCB）中。用户仅需创建和管理数据库加密密钥，即可轻松实现与数据库的通信。系统日志不仅记录了计算机系统当前和过去的状态，还具有重要的取证价值，Karande等[31]利用SGX技术重新设计了日志系统，提出了SGX-Log，SGX-Log的关键思想在于利用SGX提供的具有密封和解密封原语的安全区域来保证日志数据的安全性。Peters等[32]应用SGX来保护蓝牙I/O过程中的数据安全，他们提出了BASTION-SGX：SGX上的蓝牙可信I/O架构。该架构将受信任的软件存储在Enclave中，然后创建安全的蓝牙I/O通道，使用对称密钥进行通信。尽管一些应用程序利用SGX对软件进行分区，但如何实现合理的分区却是一个亟待解决的问题。在进行分区时，需要考虑到Enclave的数量和TCB的大小，将软件按需求划分为若干部分，以保证安全区与非安全区的隔离。基于Enclave的完整性和保密性，Atamli和Martin[33]提出了一个包含四种分类方案的框架。但这项研究只是提出了一种策略，没有说明如何使用这些新策略对应用程序进行分区以平衡安全性和性能的完整方法。Glamdring[34]是一个新的自动化程序分区框架，旨在保护使用SGX的C语言程序。Glamdring在源代码级别对程序进行分区，基于最小特权原则，最小化可信区中的代码量，以实现安全性和性能之间的平衡。

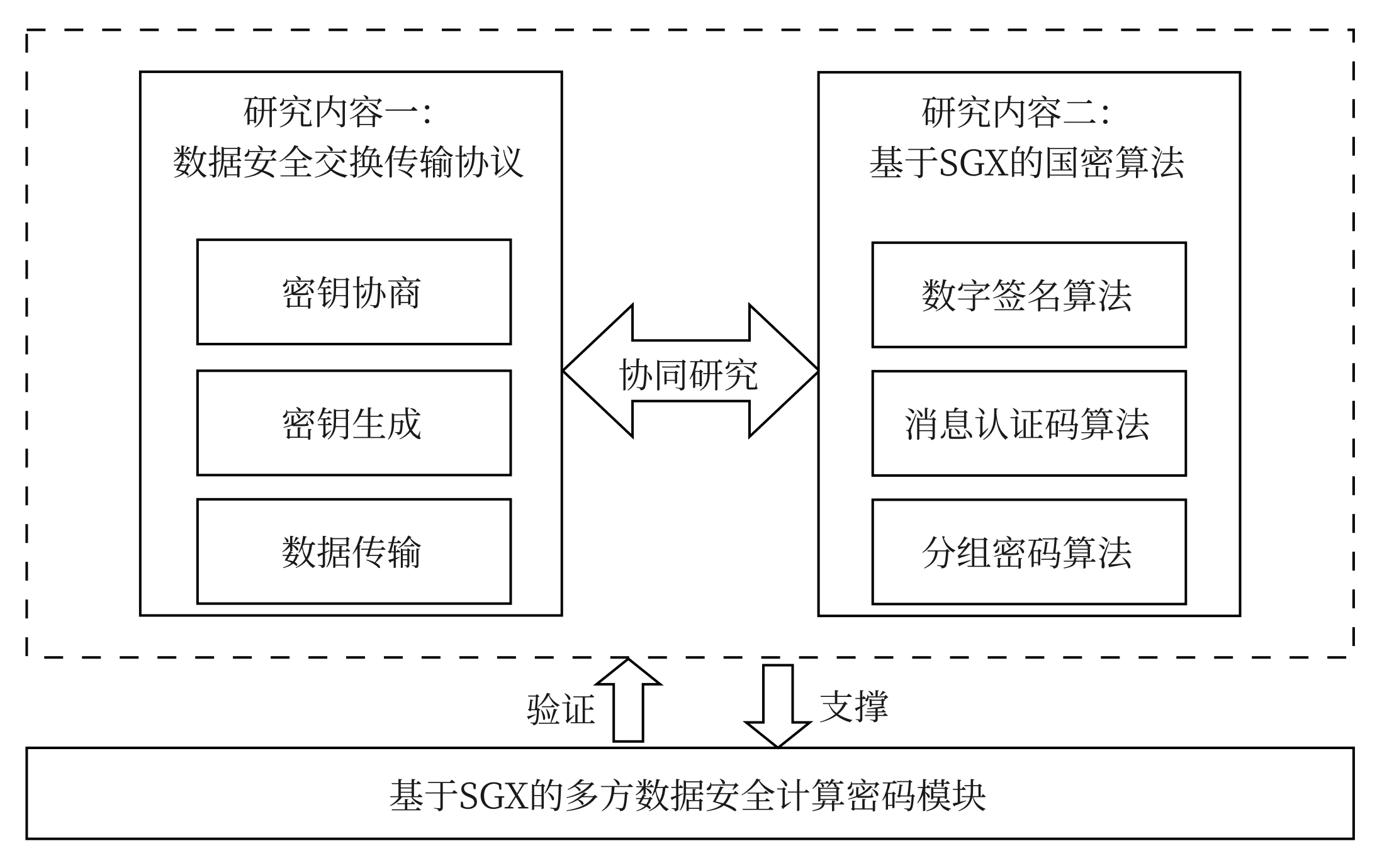
SGX也被用于保护网络通信安全。Coughlin等[35]通过扩展Click模块化路由器来使用SGX执行安全数据包处理，展示了如何在网络设备中使用SGX。然而，Duan等[36]质疑Coughlin等[35]的方法仅保护应用数据（即传输层数据包的有效载荷），而忽略了对底层数据包中各种元数据的保护，如数据包大小、时间戳等，这些信息的泄露也会造成严重的后果。为此，他们设计了一个新的基于SGX的网络中间件系统LightBox，经过测试评估，相较于之前的方案，LightBox展现出了更高的安全性。Sun等[37]提出了一种基于SGX的动态网络身份认证方案DNA-X，它可以有效地防御密钥窃取攻击，并在密钥被窃取时自动进行身份认证。DNA-X通过利用网络通信中数据源和传输过程随机性来持续改变密钥，并利用SGX机制进行本地凭证保护。经过安全性分析和实验验证，DNA-X是一种既安全又高效的网络认证方案。王等[38]利用SGX远程认证提供的安全通道，将主密钥从可信实体传输到边路基站单元，并将身份认证过程中主密钥的使用过程转移到SGX安全区内，利用可信硬件来存储主密钥。唐等[39]利用SGX提供的可信执行环境保证存储在不可信内存数据的机密性和完整性，并在此基础上提出一种保护网络内容包名称隐私的转发方案。

综上所述，SGX作为一种前沿的处理器安全技术，已在多个应用场景中展现了其增强隐私数据本地保护和通信安全的能力，不仅提升了数据的安全性，还为未来的安全解决方案提供了更多的可能性。

## 主要研究内容

本论文受国家自然科学基金项目“泛在共享环境下多方隐私计算关键技术研究”(62332018)支持。

多方数据安全计算平台由前台客户端、业务管理系统、安全计算系统、交换与脱敏系统以及各种软硬件设备组成。为了保证多方数据跨平台统筹使用和数据计算过程中隐私信息的机密性和完整性，需要构建一个基于可信执行环境的多方数据安全计算密码模块。本文针对基于SGX的多方数据安全计算密码模块的设计与实现进行研究，各项研究内容之间关系如图1.1所示。主要研究内容如下。



研究内容关系

（1）针对数据跨域共享问题，本文设计数据安全交换传输协议，作为数据需求方与数据提供方之间数据安全交换的基础。该协议包括认证密钥协商阶段、密钥生成阶段和数据传输阶段三个部分。认证密钥协商阶段，通信双方基于两轮密钥协商协议实现相互认证和传输密钥材料协商；密钥生成阶段，通信双方用双阶段密钥派生函数基于传输密钥材料和系统内预存秘密生成一个伪随机密钥和初始会话密钥，通过计数器实现密钥的定期更新，用单阶段密钥派生函数基于伪随机密钥和上下文信息生成新的会话密钥；数据传输阶段，通信双方使用会话密钥对消息进行加解密，同时计算消息认证码，保证数据交换过程中信息的机密性和完整性。

（2）针对隐私数据计算过程安全问题，本文引入可信执行环境Intel SGX技术，设计了基于SGX的国密算法增强数据安全。方案结合国密算法设计原理和SGX应用设计框架，将SM2数字签名算法、HMAC-SM3算法、SM4分组密码算法的运行环境嵌入到SGX中。使用SGX技术构建隔离的可信安全区域，针对SM2数字签名算法，将杂凑值计算、签名的生成和验证等过程放入可信区执行；针对HMAC-SM3算法，将数据填充、迭代压缩等过程放入可信区执行；针对SM4分组密码算法，将密钥扩展、数据加解密等过程放入可信区执行。在功能正确性方面，三种算法均能提供正确计算的能力，作为独立密码算法使用；性能方面相比原始密码算法虽略有损耗，但在可接受范围内。

（3）基于以上成果，本文设计并实现了基于SGX的多方数据安全计算密码模块，该密码模块包括身份认证模块、密钥生成模块和加解密模块，并通过功能和性能测试对密码模块进行验证。

## 论文组织结构

本文包含六个章节，各章节的具体内容如下。

第一章为绪论。本章主要介绍了基于SGX的多方数据安全计算密码模块的研究背景及意义，调研了当前国内外相关领域的研究现状，并介绍了本文的主要研究内容和论文组织结构。

第二章为背景知识。本章介绍了基于SGX的多方数据安全计算密码模块所需的相关理论基础，包括SGX技术、密码体制、杂凑算法、消息认证码和Dolev-Yao攻击模型等内容。

第三章为数据安全交换传输协议。本章首先说明了数据安全交换传输协议的系统架构，并进行了安全需求分析。其次对协议的具体流程进行了介绍，包括认证密钥协商、密钥生成、数据传输三个阶段。最后对协议进行了正确性和安全性分析。

第四章为基于SGX的国密算法设计与实现。本章首先介绍SGX可信密码库支持的密码算法，针对SGX可信库中缺少国密算法的问题，结合SGX应用设计通用框架和国密算法设计原理，设计并实现了基于SGX的安全增强国密算法。最后对算法功能、性能和安全性进行了测试，并根据测试结果进行了分析。

第五章为基于SGX的密码模块设计与实现。本章首先介绍了密码模块的设计。其次对身份认证模块、密钥生成模块、加解密模块进行了具体的实现。最后对各模块进行了功能和性能测试，针对模块测试结果进行分析。

第六章为总结与展望。本章对本文主要研究内容进行了总结，分析了基于SGX的多方数据安全计算密码模块尚且存在的不足，指明了未来进一步的研究方向。

# 背景知识

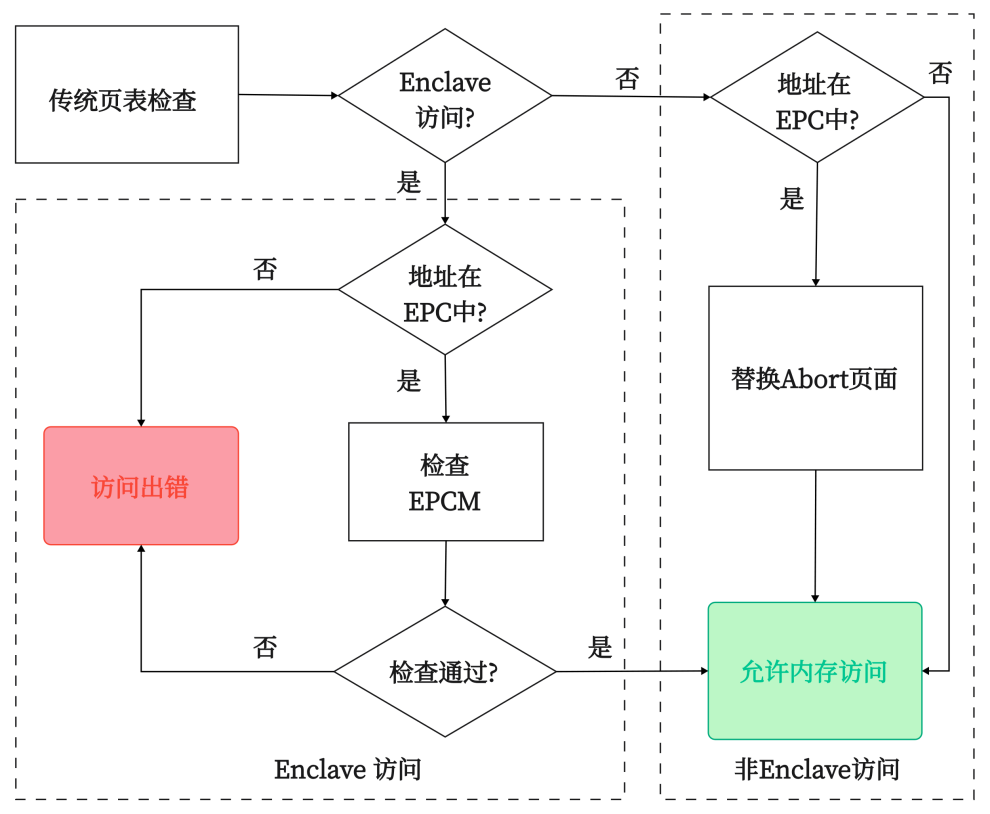
本章主要介绍本文所提出的基于SGX的多方数据安全计算密码模块所需的核心技术及相关理论基础进行介绍，包括SGX技术、密码体制、杂凑算法、消息认证码和Dolev-Yao攻击模型等内容。

## Intel SGX关键技术

可信执行环境（TEE）是处理器中的安全隔离区域，可以为用户在不可信服务器中安全执行应用程序，从而确保数据和代码的机密性和完整性。应用场景包括移动支付、版权管理、身份认证等。目前，主流的TEE技术有AMD SEV[40]、ARM TrustZone[41]和Intel SGX。其中SEV技术专为虚拟化环境的安全问题设计，主要用于保护虚拟机中的数据和代码。TrustZone技术提供系统级别的隔离导致其可信计算基（TCB）过大，增加了系统潜在攻击面。而且TrustZone主要针对的是移动端的安全隔离，在移动设备上更广泛地使用。Intel在其原有CPU架构中引入了新的指令集以支持SGX在CPU内部创建Enclave为应用程序提供隔离执行环境。SGX技术提供开发者精确控制哪些代码和数据包含在Enclave内的能力，这种精细的控制机制有助于最小化TCB。

### 内存隔离机制

Enclave作为SGX的核心概念，是一块受到保护的内存区域，在Enclave中可以安全地执行代码和处理数据。Enclave内的数据对操作系统、虚拟机管理程序或其他更高权限级别的软件不可见，从而确保了数据的隐私性和完整性。Enclave的保护机制主要包含两方面：一是Enclave内存访问语义的改变，二是应用程序地址映射关系的保护。这些功能共同确保了Enclave的机密性和完整性。申请创建Enclave时，应用程序需要执行页面分配、复制程序代码和数据以及度量操作。创建过程的最后一步需要验证Enclave的完整性，以确定特权软件在创建过程中是否伪造了程序数据。测量添加的页面内容，最后将创建序列的结果保存在Enclave的控制结构中。SGX将此结果与Enclave所有者通过初始化指令签署的证书中的完整性值进行比较。如果匹配，则对证书中的所有者公钥进行哈希处理并将其作为密封身份存储在安全区控制结构中。假设不匹配，说明创建过程出现问题，指令返回失败结果。初始化指令执行成功后，进入Enclave执行程序，由SGX提供内存保护和地址映射，使外界无法保护Enclave访问内存，从而保证了Enclave的机密性和完整性。SGX内存访问控制如图2.1所示。



SGX内存访问控制

### 密封机制

密封（Sealing）机制是一种保护机制，将在Enclave内部需要持久化保存的敏感数据安全地保存到外部存储设备上。这样即使Enclave实例析构或系统关闭，这些敏感数据不会被非授权访问，也不会丢失，并且密封的数据可以在需要用到的时候加载回Enclave。密封过程离不开安全的密钥，其可信的本质是CPU在出厂制造过程中生成的唯一根密封密钥（Root Seal Key，RSK）。该密钥内置在CPU硬件中，只有CPU自己知道，不可被外部读取。通过结合RSK和Enclave度量值MRENCLAVE，SGX能够派生出一个唯一对应于该Enclave实例的密封密钥，Enclave使用派生的密封密钥来加密数据，以便安全地保存到外部存储。当Enclave在同一个平台上重新加载时，可以重新派生出密封密钥对之前保存的密封数据进行解密，解密后的数据可再次供Enclave使用，恢复其先前的状态或继续处理。这种持久化数据的加解密被称为SGX的密封机制，流程如图2.2所示。



SGX密封与解封

### 本地认证

本地认证（Local Attestation，LA）机制可以实现在同一系统内的不同Enclave之间建立信任关系。本地认证允许一个Enclave验证另一个Enclave的身份和完整性，确保对方是真实存在的，并且存在于同一SGX平台上。通过本地认证，Enclave实例就可以调用其他Enclave实例提供的服务，而不会调用到恶意程序提供的接口。

Enclave创建与初始化：每个Enclave在创建时都会生成一个唯一度量值MRENCLAVE。MRENCLAVE是Enclave的初始内容经过加密和哈希计算得到的，反映了Enclave创建时的代码和数据状态，任何对Enclave内容的微小更改都会导致MRENCLAVE值的改变。因此，MRENCLAVE代表Enclave的最终身份和完整性状态。如图2.3所示，以下是SGX本地认证流程的详细步骤。



SGX本地认证流程

（1）身份认证请求：当Enclave B（验证者）想要验证Enclave A（被验证者）时，它首先发送自己的MRENCLAVE值给Enclave A。这个值是Enclave B初始化时的度量值，唯一标识了Enclave B。

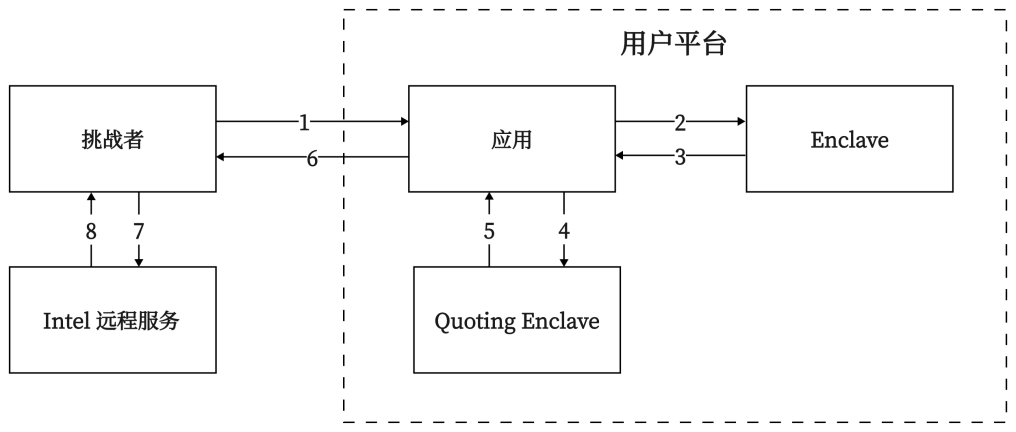
（2）生成并发送Report：Enclave A接收到MRENCLAVE值后，调用EREPORT指令，使用CPU的根签名密钥（RSK）来生成一个针对该MRENCLAVE值的数据结构，即Report B，并将这个报告发送给Enclave B。Report B包括对MRENCLAVE的签名，证明了Enclave A的真实性和未被篡改的状态。

（3）验证Report：Enclave B接收到Report B后，使用EGETKEY指令来生成一个报告密钥（Report Key）来验证报告的MAC值。这个密钥是基于Enclave A的MRENCLAVE值（即Enclave A的身份）和Enclave B自身的身份信息计算得出的。这一步确认了Report B是由在同一SGX平台上运行的真实Enclave生成，从而验证了Enclave A的身份和安全状态。

经过上述步骤后，Enclave B可以确认Enclave A的真实性和安全性。为了建立双向信任，Enclave A也可以采用相同的流程来验证Enclave B。

### 远程认证

远程认证（Remote Attestation，RA）是指硬件实体或硬件实体与软件实体的组合获得远程服务提供商的信任，以便服务提供商可以放心地向客户端提供其所请求的机密信息。当用户写完一个 Enclave程序，想要将其放到云服务器上运行时，用户无法确保Enclave是在真实的SGX平台上运行而不是模拟出来的环境，从而也就无法保证自己的数据安全。Intel SGX 远程认证的提出解决了这一问题。远程认证不仅可以完成对远程服务运行的真实性鉴定，还可以通过密钥交换算法来与服务提供商建立安全通信。如图2.4所示，远程认证过程通常涉及以下步骤。



SGX远程认证流程

（1）挑战发起：远程服务提供者接收到初始请求后，如果想要为其提供服务，则以挑战者的身份向用户平台上的应用程序发送挑战信息，要求证明应用程序Enclave的真实性和安全性。

（2）报告生成：应用程序将挑战信息发送给Enclave，Enclave生成一个内部报告，此报告包括其安全状态和可选的用户数据。

（3）报告传递：Enclave将生成的报告传递给用户平台上的应用程序，应用程序再将Enclave的报告发送给SGX平台上的一个特殊Enclave，称为Quoting Enclave（QE）。

（4）Quote生成：QE验证应用程序Enclave的报告，确保报告是由有效的Enclave在同一平台上生成的。验证通过后会将报告转换成Quote结构。Quote包含应用程序Enclave的身份信息、安全状态信息和签名等。

（5）Quote发送：应用程序从QE接收Quote，并将其发送给远程服务提供者。

（6）Quote验证：远程服务提供者接收到Quote后，将其发送给英特尔认证服务（Intel Attestation Service，IAS），由IAS验证Quote的真实性和有效性，通过后IAS将生成新的验证报告作为返回结果。

## 密码体制

密码体制，亦称为密码系统，构成了信息安全领域的核心，旨在全面解决包括机密性、数据完整性、认证、身份识别、授权管理以及不可否认性等关键安全问题中的一个或多个。密码系统通过一套完整的机制和协议，确保机密信息在存储或传输过程中的安全性和可靠性。根据加密和解密操作的性质，密码体制可以分为两大类：对称密码体制和非对称密码体制。这两种分类基于密钥的使用和分配方式有着根本的区别，各自适用于不同的安全需求和场景。

### 对称密码体制

对称密码体制依赖于单一密钥来执行加解密操作。对称密码体制加解密流程如图2.5所示，发送方和接收方必须事先共享同一个密钥，并且保证该密钥不被未经授权者获取。对称密码体制的优势在于其实现简单、加解密操作通常比较高效，适合于需要处理大量数据的场合。对称密码算法按照其操作方式和结构特点，可以分为两大类：分组密码算法和流密码算法。其中，分组密码算法适用于需要加密大批量数据进行安全传输或存储的场景。常见的分组密码算法有DES、3DES、AES、SM4等。

DES是最早的分组密码算法之一，但由于DES算法被发现存在安全性问题，因此之后出现了AES算法作为替代。AES是一种更现代的对称加密算法，提供了三种不同长度的密钥选项（128、192、256位），成为全球广泛采用的加密标准之一。SM4[42]算法于2012年被国家密码管理局确定为国家密码行业标准，又于2021年被纳入ISO/IEC国际标准正式发布。SM4算法通过32轮的加密过程和复杂的非线性变换设计，提供了高级别的安全性，能有效抵抗差分攻击、线性攻击等多种密码分析方法。在实际应用场景中，SM4适用于多种加密模式，包括ECB、CBC、CFB、OFB、CTR等，满足不同的安全通信需求。SM4使用128位的密钥长度，平衡了安全性与计算资源的需求。综上所述，SM4算法的设计兼顾了安全性、效率和实用性，因此，也是本文主要采用的对称密码算法。

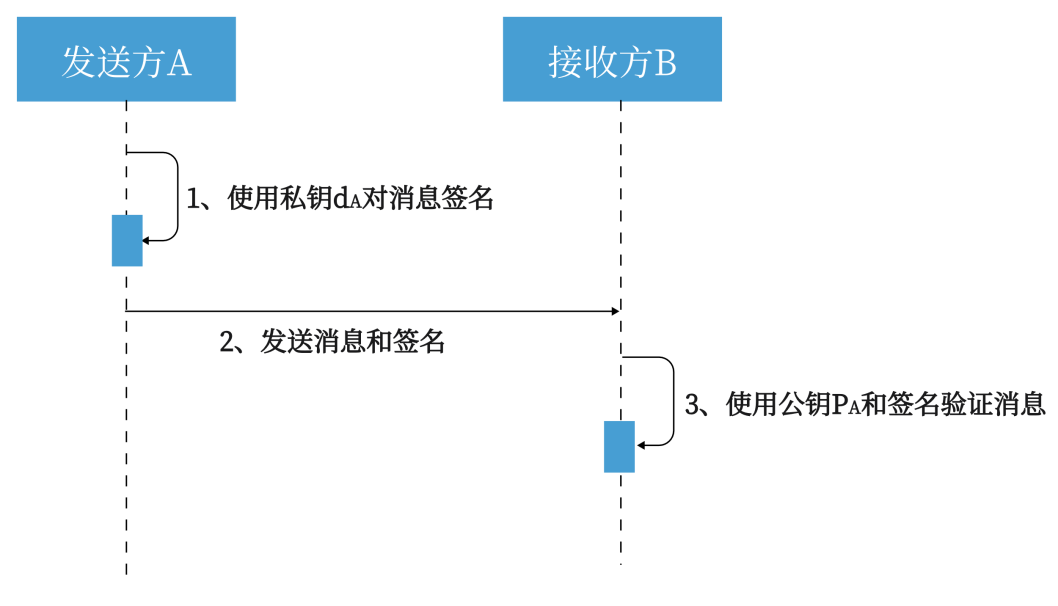


对称密码体制流程

### 非对称密码体制

非对称密码体制，亦称公钥密码体制，旨在解决对称加密体制在密钥分发和管理方面所面临的挑战。与对称密码体制不同，非对称密码体制采用一对密钥：公钥和私钥，这两个密钥在数学上是相关联的，但是根据公钥无法计算得到私钥。在这种体制中，公钥是公开的，任何人均可访问和使用；而私钥则是保密的，仅限于密钥持有者掌握。主流的非对称密码体制包含加密与认证两种算法，公钥和私钥的使用方式也有所不同。在加密算法中，公钥用于加密信息，私钥用于解密信息。在认证算法中，私钥用于生成签名，公钥用于验证签名，非对称密码体制的签名与认证流程如图2.6所示。常见的非对称加密体系主要包括以下几种：基于大整数分解难题的RSA密码机制、基于离散对数难题构建的Elgamal密码体制，以及建立在椭圆曲线数学理论基础上的ECC密码体制。

RSA密码体制基于大数分解问题，是最早的也是最广泛使用的非对称加密算法之一。RSA算法的安全性依赖于大整数分解的难度，适用于加密和数字签名。Elgamal密码体制基于离散对数问题，提供加密和数字签名解决方案。ECC密码体制同样基于离散对数问题，但在椭圆曲线上操作。ECC能够在较短的密钥长度下提供与RSA相同级别的安全性，使其受用于移动设备和资源受限环境中。SM2算法[43]是我国国家密码管理局推荐的公钥加密算法标准之一，并于2017年正式成为ISO/IEC国际标准。SM2算法属于ECC类密码体制，与其他非对称密码体制相比，SM2具有安全性高，计算开销小等优点，因此本文使用SM2作为非对称密码体制。



非对称密码体制签名和认证流程

## 杂凑算法和消息认证码

### 杂凑算法

杂凑算法又称为哈希函数或散列算法，是一种将任意长度的输入数据（消息）压缩为固定长度输出（杂凑值）的算法。杂凑算法可视为一个映射，记为，满足以下基本条件：

（1）易计算：对于任意给定的输入数据，能够容易且快速地计算出其杂凑值。

（2）抗原像攻击：已知杂凑值，通过映射反向得到输入在计算上不可行。

（3）均匀分布：杂凑算法对于任意不同的输入数据，产生的杂凑值应该是均匀分布的。

（4）弱抗碰撞性：给定输入，找到另一个不同的输入，使得在计算上不可行。

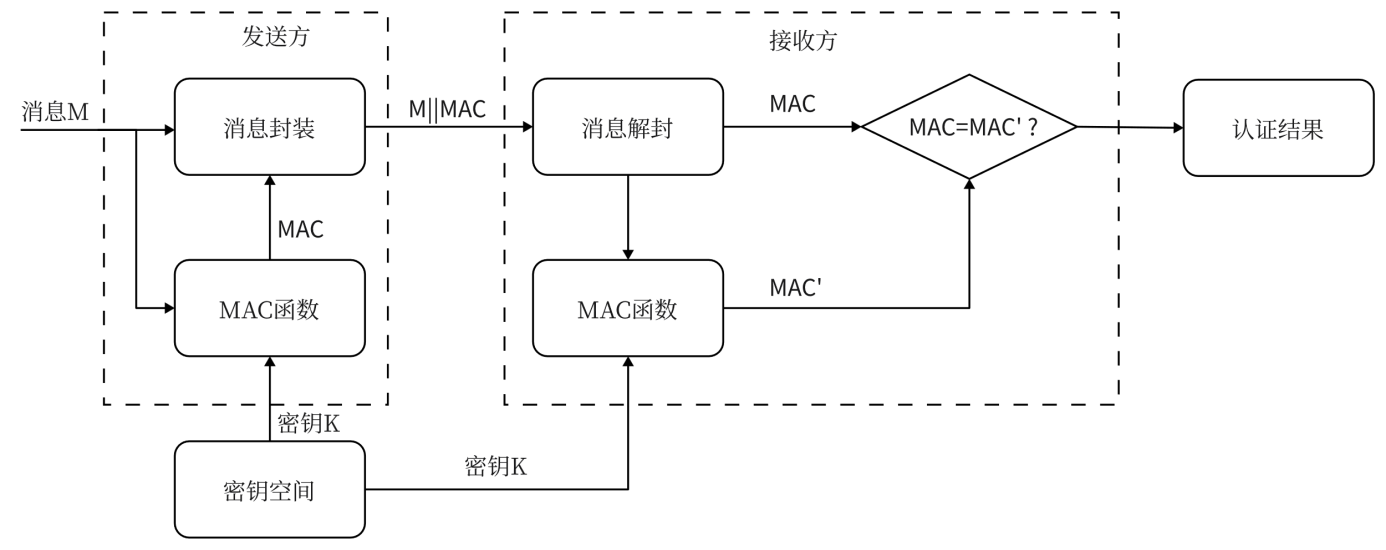
（5）强抗碰撞性：找到任意一对不同的输入和，使得在计算上不可行。

不同长度的杂凑值对抗穷举攻击的性能不同，一般来说杂凑值长度越长的算法抗攻击性更强，计算难度随着长度增加呈指数级增长。MD5算法的输出长度为128位，目前已被广泛认为是不安全的杂凑算法，容易受到碰撞攻击。SHA-1算法的输出长度为160位，与MD5类似，SHA-1也被揭示存在严重的安全问题。SHA-2系列算法包括SHA-256、SHA-384和SHA-512等，输出长度分别为256位、384位和512位。SM3算法[44]是一种公开的密码算法标准，由国家密码管理局于2010年发布。SM3算法输出的杂凑值与SHA-256算法相同，均为256位。相较于SHA-256，SM3在设计上采用了不同的消息扩展、压缩函数和迭代结构，以增强其抗碰撞攻击和抗原像攻击的能力。此外，SM3算法有着更快地发散速度，运算效率更高。本文选择SM3算法作为杂凑算法。

### 消息认证码

消息认证码（MAC）是一种重要的安全机制，允许共享一个秘密密钥的接收方和发送方确信被传输的消息未被篡改。消息认证码工作流程如图2.7所示，发送方首先根据消息M和密钥K计算出MAC值，将MAC值与消息一同发送。接收方使用相同的密钥K计算得到MAC'值，并将其与发送方提供的MAC值进行比较，从而验证消息的完整性和来源的真实性。MAC算法确保了消息在传输过程中未被篡改，并且验证了消息的发送者确实拥有共享密钥，从而提供了认证。消息认证码主要有两种类型：基于哈希函数的HMAC（Hash-based Message Authentication Code）和基于分组密码的CMAC（Cipher-based Message Authentication Code）。

哈希函数可以把较长的消息变换为较短的消息摘要，并且具有抵抗碰撞等良好的性质。为了保证消息的完整性，要在计算摘要时加入密钥，在加入了密钥之后，哈希函数就称为带密钥的哈希函数。但是单独一个带密钥的哈希函数是不能直接作为MAC使用的，必须经过特殊的构造，在具备了较好的性质后才可以用作消息认证码。



消息认证码认证流程

## Dolev-Yao攻击者模型

Dolev和Yao[45]于1983年提出了验证公钥加密协议的数学模型，正式定义了攻击者的行为，并提出了Dolev-Yao攻击者模型，对攻击者可能的攻击行为进行了精确描述。Dolev-Yao 攻击者模型包含以下两个假设。

（1）假设加密系统是完美的，安全协议的执行序列是严格按照协议规范定义的消息步骤的交替序列。

（2）攻击者可以控制整个通信网络，应假定攻击者具有相应的知识和能力。攻击者可以窃听、重播、拦截所有通过网络的消息，也可以对原始信息进行重组，伪造信息内容。

与其他安全协议分析模型相比，Dolev-Yao攻击模型有三个显著的优点和特性，使其成为分析密码学协议安全性的强有力工具。首先，Dolev-Yao模型假设攻击者具有非常强大的能力，包括监听、拦截、修改和伪造网络通信。这种假设确保设计的协议能够在极端条件下保持安全，提供了一种“安全上限”评估方式。其次，通过抽象化攻击者的能力，研究者可以集中精力分析协议的逻辑结构。最后，Dolev-Yao威胁模型的形式化定义提供了系统化的方法来描述攻击者可能采取的行动，以及攻击者如何利用拦截到的信息来进行攻击。这种形式化描述使得安全协议的分析和验证可以更加严谨和系统化。其形式化定义如下。

（1）包含：若攻击者已经得到的消息集合中包含信息。则攻击者直接掌握了信息，并可以用其构造新的消息。

（2）拆分：若攻击者得到包含信息项和的消息集合，则攻击者直接掌握了信息项和，因为攻击者有从一组信息中提取单个信息项的能力。

（3）解密：若攻击者既有加密信息，也有对应的解密密钥，则攻击者可以通过解密掌握信息。

（4）约减：若攻击者得到的消息集合中包含带有形式为“”的消息，则攻击者掌握了信息。

（5）组合：若攻击者同时掌握了信息和，则攻击者可以构造新的消息。

（6）加密：若攻击者同时掌握了信息和加密密钥，则攻击者可以使用密钥对信息加密，构造新消息。

## 本章小结

本章首先详细介绍了SGX的关键技术，包括内存隔离机制、密封机制、本地认证和远程认证。其次，对密码学中的重要概念进行了阐述，包括密码体制、杂凑算法和消息认证码。最后，还介绍了Dolev-Yao攻击者模型，用以理解协议安全性。这些知识为基于SGX的多方数据安全计算密码模块的设计与实现提供了理论基础。

# 基于混沌系统的图像加密方案

本章节将介绍存证系统获取元数据后，进行的主要操作包括：利用元数据生成原始图像的唯一标识，利用唯一标识码生成置乱密钥，将唯一标识编码成QR码，用置乱密钥生成混沌序列，将QR码利用混沌序列进行混沌置乱和扩散操作得到置乱和扩散后的混沌图片。然后测试对混沌图片的反向提取操作，解密混沌图像得到QR码，最后进行安全性测试，测试混沌加密方案的有效性。

## 加密系统模型

加密系统的加密流程如图 3.1 所示。下面是对详细步骤的解释：

步骤一：通过明文图像提供的特征信息采用SHA256算法算出特征信息的哈希值。

步骤二：将生成的哈希值转换成QR码图像作为秘密信息。

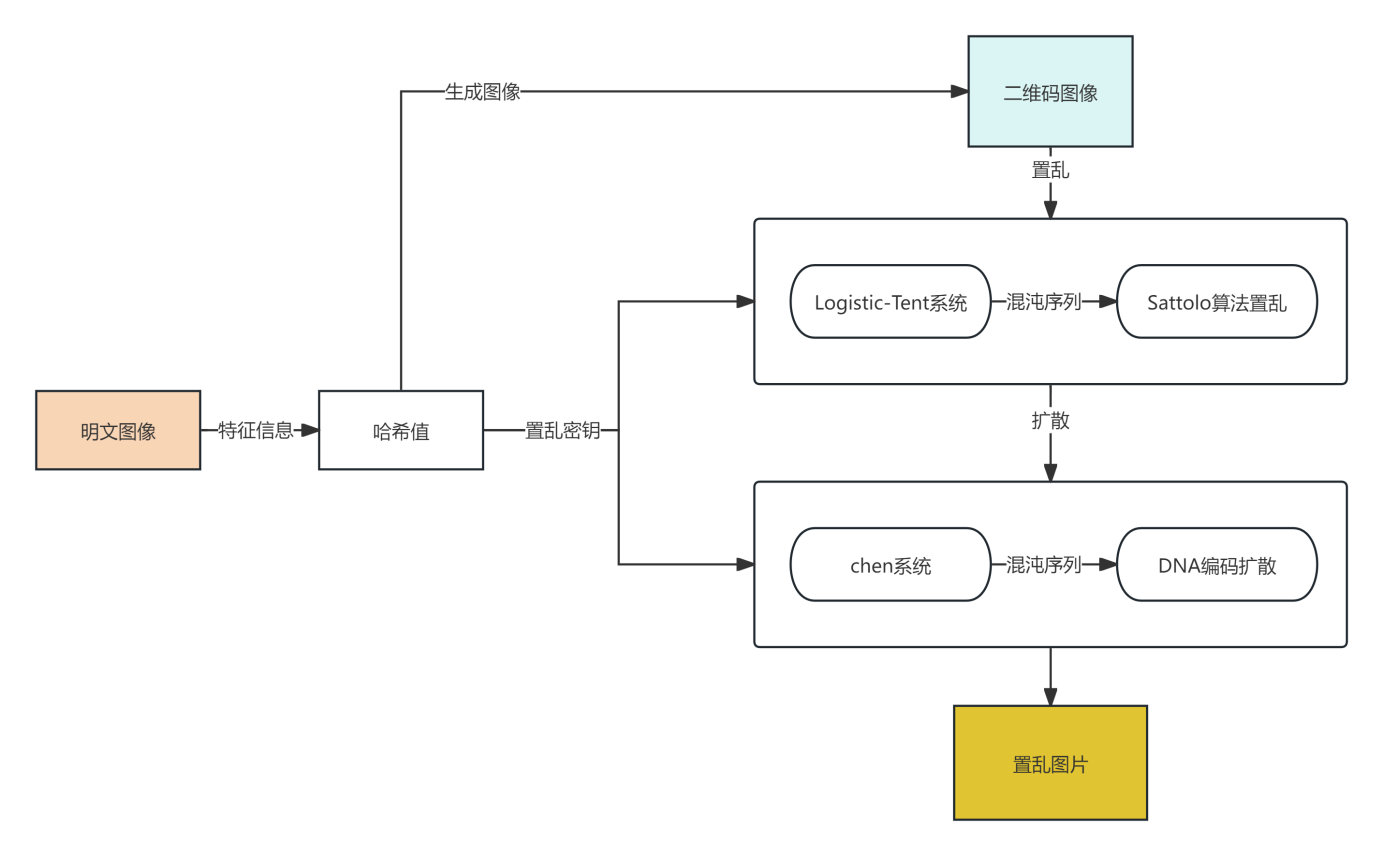
步骤三：将哈希值作为加密密钥，对该散列值进行运算后得到混沌系统的初始值和参数值。

步骤四：混沌系统带入步骤三生成的初始值和参数值，生成用于置乱的混沌序列和用于DNA编码扩散的混沌序列。

步骤五：将sattlolo随机置乱算法和置乱混沌序列结合，对QR码图像进行置乱操作。

步骤六：将DNA循环编码扩散机制和扩散混沌序列结合，对步骤五得到的置乱图像进行扩散操作。

步骤七：经过步骤三，四，步骤五两个关键步骤后，即可完成对原始图像的加密处理并得到混沌图像。



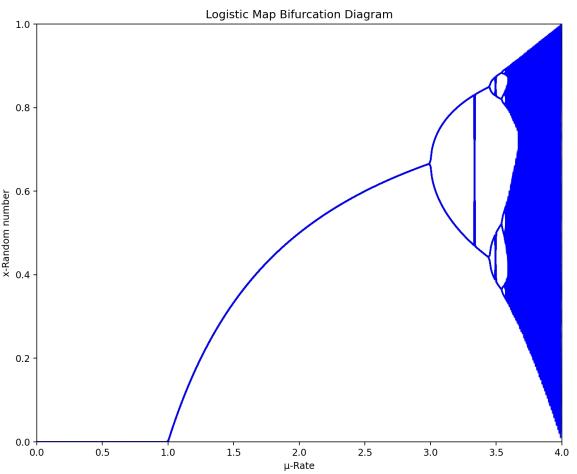
加密系统模型

## 混沌模型的选取

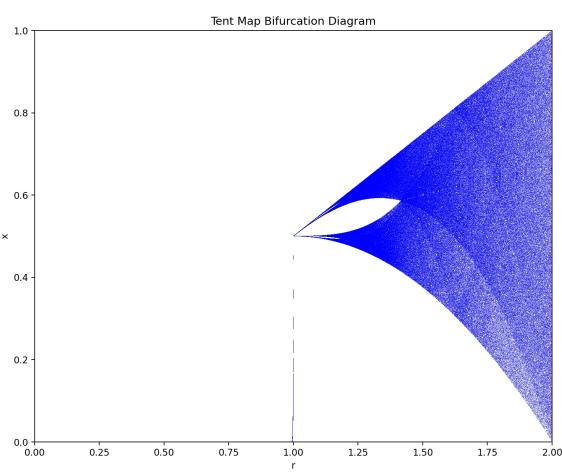
### 一维混沌体系的选取

在第二章的混沌系统中介绍了两个经典的混沌映射系统 Logistic系统和Tent系统。Logistic系统和Tent系统都是常见的混沌系统，但它们在动态特性上有所不同。Logistic系统具有较简单的数学模型和较好的稳定性，但对于一些复杂的非线性现象，它的表现较为有限。Tent系统则在产生更为复杂的混沌行为方面表现得更为优异，适用于一些需要更高灵敏度和更强动态行为的应用。但是两者都不是满映射系统，这直接导致了他们的参数的选取范围收到了限制。

如图3.2和图3.3所示：Logistic系统在当 3.5699 < *μ* ≤ 4 时，系统处于混沌状态。Tent系统在当 3.7500 < *μ* ≤ 4 时，系统处于混沌状态。本章介绍的加密系统需要参数 *μ* 在当作密钥参数，Logistic系统和Tent的参数选取范围使得加密系统的密钥空间变小，不利于系统安全性和复杂性。



Logistic系统分岔图



Tent系统分岔图

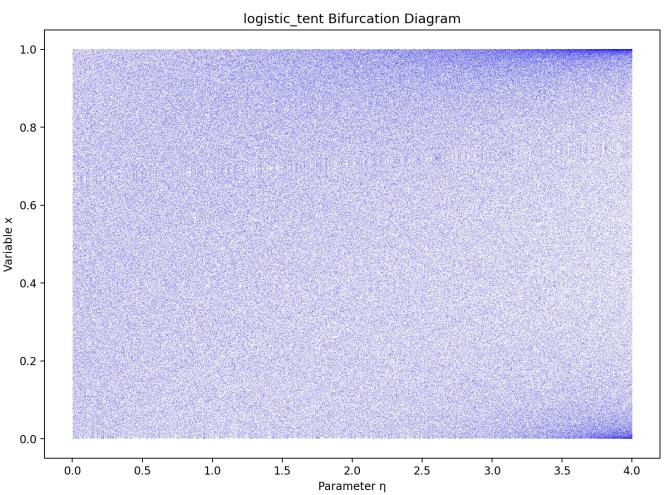
本文采用Logistic-Tent系统，Logistic-Tent系统的优点在于它结合了两者的特点，既保留了Logistic系统的简单性和稳定性，又引入了Tent系统的高灵敏度和多样性，使得在处理一些具有较高复杂度的混沌问题时更加高效和灵活，解决了单一系统在稳定性与复杂性之间的权衡问题。

Logistic-Tent映射表示为公式：



其中 *μ* ∈(0, 4] 。

Logistic-Tent的分岔图3.4所示：Logistic-Tent是满映射系统和Logistic，Tent系统相比，有更加广阔的密钥空间。对加密模型的安全性和复杂性有正向的提升。

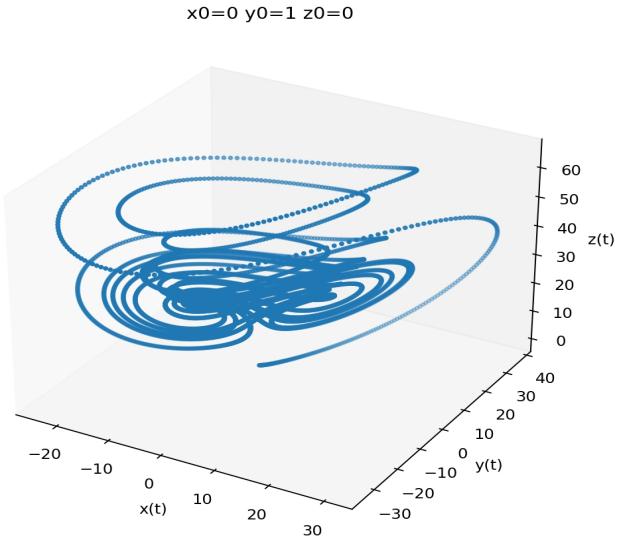


Logistic-Tent

### 三维混沌系统的选取

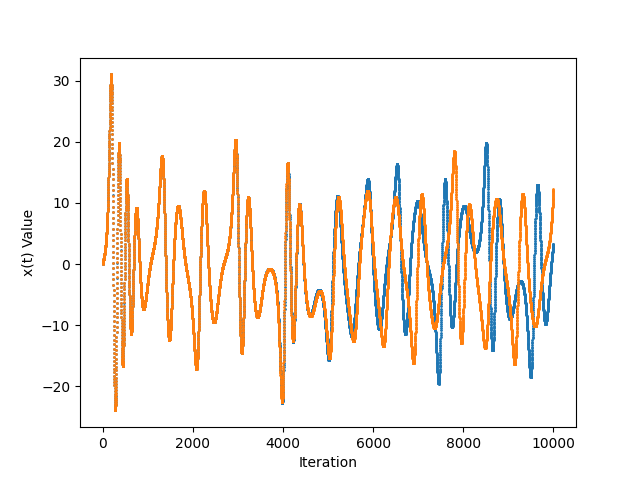
根据第二章内容，Chen系统和Lorenz系统都属于经典的三维混沌系统，Chen系统通常能比Lorenz系统产生更均匀、随机性更强的序列。更复杂的混沌行为使得系统对初始条件更加敏感，进而提升密钥敏感性和抗攻击性。

由图3.5可见，Chen参数设置为a = 35 ，b = 3 ，c = 28，经过长时间运行后，系统只在三维空间的一个有限区域内运动，系统在此区域中的运动是混沌状态。



Chen系统

从两个靠的很近的初值条件出发（y只相差0.0001）给出了x(t)轨道的演化图3.6如下：橙色线条的y初始值位1，蓝色线条的y初始值为1.0001。



Chen系统迭代图

由图3.6可见，随着时间的演化，可以看到原本靠得很近的轨道，在迭代7000次左右后x的值开始出现较大的区分，最后两条轨道变得毫无关联。

因为Chen是个三维的混沌系统，所以可以用chen系统的三维混沌序列来操控DNA编码扩散情景下的DNA编码，解码操作和DNA运算操作操作。用Chen系统作为DNA编码扩散操作的混沌系统可以增加加密系统的混沌性。

## 混沌序列的生成方式

在双端协同存证模型中，唯一标识码作为原始图片的数字指纹，同时是置乱密钥的生成依赖，需满足全局唯一性、抗篡改性与高效生成需求。混沌序列作为加密载体QR码的动态控制参数，需要完成对QR码的置乱和扩散，必须具备强随机性和初值敏感性。

本节下采用一种元数据驱动的“标识-序列协同生成机制”，通过哈希函数确保标识码唯一性，结合一维和三维混沌系统生成动态密钥和多维混沌序列。因为每张图片的元数据不同，生成的散列值不同，置乱密钥也不同所以可以实现“一图一密”，确保载密QR码的信息安全。

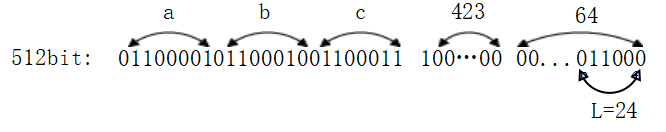
### 生成唯一标识码

NIST标准化哈希算法历经多代迭代，形成SHA-0、SHA-1、SHA-2、SHA-3四大分支。其中SHA-2系列包含六种子类（SHA-224/256/384/512/512-224/512-256）[44]，其核心优势体现在长哈希值设计（224-512位）与抗碰撞强度提升，相较SHA-1与MD5具备显著安全性优势（SHA-1碰撞攻击复杂度2^63次，SHA-256达2^128次）。尽管SHA-512通过增加迭代轮数（80 vs 64）进一步强化安全性，但由此产生的计算开销导致吞吐率不如SHA-256。综合考量安全性基线（满足128位抗碰撞）、运行效率（单位时间处理量）及软硬件兼容性（广泛支持AES-NI指令加速），SHA-256成为标识码生成函数的优化选择。使用SHA-256 算法的算法生成唯一标识主要步骤如下：

（1）元数据转换：输入的元数据包括，用户编号，本地存证编号，文件提交时间，文件名，文件大小，文件类型，图片像素长度，文件像素宽度，文件创建时间，本地存证生成32位随机英文和数字编码，将这些数据都按照unicode编码，然后将编码乱排，将乱排结果作为散列函数的输入。

（2）比特填充：对于输入的元数据字符串，其长度为L，需要在消息的末尾添加填充比特。填充的具体方法如下：首先，在消息末尾添加一个1位的比特。接着填充K个0，其中K是满足方程L+K+1=448mod512的最小非负整数。最后，附加消息原长度L的二进制表示。经过这种填充后，最终的消息长度将是512的整数倍。

填充比特的具体规则请参见图3.7。为了进一步说明这一过程，假设输入比特串为 "a, b, c"。



比特填充

1. 第三步是算法输出的初始化：在每次执行SHA-256计算时，首先需要进行输出初始化。该过程使用8个32位的寄存器来保存SHA-256在每个计算步骤中的中间结果。根据算法的协议标准，初始化值由前8个质数平方根的小数部分的前32位组成。这些初始值可以通过16进制表示如下：

H=0x8a46e667

H=0xbb67ae85

H=0x3c6ef372

H=0xa54ff53a

H=0x510e527f

H=0x9b05688c

H=0x1f83d9ab

H=0x5be0cd 19

1. 常量数组的初始化：该数组中的值来自前64个质数（从2到311）对应立方根的小数部分的前32位。所有常量的16进制表示形式按顺序排列，如表3.1所示。

常量数值表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **常量数值表** | | | | | | | |
| 428a2f98 | 71374491 | b5c0fbcf | e9b5dba5 | 3956c25b | 59f111f1 | 923f82a4 | ab1c5ed5 |
| d807aa98 | 12835b01 | 243185be | 550c7dc3 | 72be5d74 | 80deb1fe | 9bdc06a7 | c19bf174 |
| e49b69c1 | efbe4786 | 0fc19dc6 | 240ca1cc | 2de92c6f | 4a7484aa | 5cb0a9dc | 76f988da |
| 983e5152 | a831c66d | b00327c8 | bf597fc7 | c6e00bf3 | d5a79147 | 06ca6351 | 14292967 |
| 27b70a85 | 2e1b2138 | 4d2c6dfc | 53380d13 | 650a7354 | 766a0abb | 81c2c92e | 92722c85 |
| a2bfe8a1 | a81a664b | c24b8b70 | c76c51a3 | d192e819 | d698aa4a | f40e3585 | 106aa070 |
| 19a4c116 | 1e376c08 | 2748776c | 34b0bcb5 | 391c0cb3 | 4ed8aa4a | 5b9cca4f | 682e6ff3 |
| 748f82ee | 78a5636f | 84c87814 | 8cc70208 | 90befffa | a4506cbe | bef9a3f7 | c67178f2 |

1. 数组的计算，~的值分别由 512比特输入块按32比特从高到低分割得到，数组中的~的获取方式如式所示：



式中的参数 ，的获取方式如式和所示：





压缩函数的迭代计算：SHA-256算法进行64次迭代计算。在每次迭代中，8个常数变量a, b, c, d, e, f, g, h都是 32 位的数据变量。首先，需要初始化这8个迭代常数变量，其初始化公式如式所示：



接下来，需要对这八个常数进行迭代更新。八个常数的迭代公式如下式所示：



其中函数CH，Maj，，的运算公式如下所示：









最后，计算每一步的Hash值时，需要使用上一步（i-1）的Hash值以及更新后的常数变量。具体的计算方法如式所示：



在处理完最后一个512比特的消息块后，最终的SHA-256输出将是最后一次迭代的运算结果。该结果的输出形式如公式所示：



在上面各个公式中，>>表示的是循环右移，→标识的是右移操作，符号|表示的是位的拼接，符号⊕表示的是异或，┐表示的是取反运算，&表示的是按位与运算。

### 生成混沌序列

1. 一维置乱混沌矩阵的生成过程

在上一节中，使用SHA-256哈希算法对元数据进行了处理，从而生成了相应的哈希值H。这一节中，将十六进制格式表示哈希值转换为一个由256位二进制数字组成的唯一标识码key。随后，将key按每32位一组进行划分，将每组二进制数转换为十进制大数，并通过逐项相加，再通过式进行计算，得到混沌序列的混沌初值。SHA-256哈希算法的一个重要优点是，输出序列与明文之间有着紧密的关联，因此，当元数据发生变化时，生成的哈希值也会随之不同。又由于每个图片的元数据都不同，所以每张图片生成的哈希值不同，因此每张图片的根据唯一标志码生成的置乱密钥都不同，做到了一图一密，增加了加密模型的安全性。



接下来根据上文得到的8个大数字，根据式生成一维混沌序列的混沌初值。就是logistic-tent混沌映射其中一个置乱密钥，也是混沌映射系统的混沌初值。



logistic-tent混沌映射还有一个置乱参数∈(0, 4]，那么的生成方式如下式所示，参数由本地存证产生。



由此可以得到一个值在(0, 4]之间的混沌参数，这是一维混沌序列的第二个置乱密钥。

带入初始值和迭到式，迭代生成和M×N（M和N分别是生成二维码的厂像素长度和像素宽度）长度的序列S。该S序列就是和Sattolo算法结合的置乱序列。

1. 三维混沌矩阵的生成过程

三维序列的chen系统选取的参数是a = 35 ，b = 3 ，c = 28，在3.2.2节看到了，在这个参数体系下，系统呈现出良好的混沌性以及初值敏感性。那么只要确定x，y，z三个初值就可以通过迭代获得三维的混沌序列了。首先可以将求一维混沌序列的值和作为x和y的初始值。z的初始值如式所示：



其中，-20的目的是将z的初始值控制在-20到20之间。所以综上所述，chen系统的初始值*u*0 如式所示：



假设图像大小为 M×N，系统生成 10000+M×N 长度的伪随机序列，并丢弃前 10000 个值以获得更好的随机效果，得到 M×N 长度的混沌序列 X ，Y ，Z。



混沌序列X,Y,Z就是三维的混沌序列，用于DNA编码扩散和DNA运算控制。

## QR码的置乱与扩散

在上一节介绍了一维混沌序列和三维混沌序列的产生步骤。有了混沌序列之后就可以对QR码进行混沌置乱和扩散的操作。这一届主要介绍QR码的混沌置乱的具体步骤：

第一步：利用唯一标识码生成唯一的QR码

第二步：将Sattolo 算法和logistics-tent混沌映射系统生成的混沌序列结合进行像素的置乱。

第三步：将DNA编码运算和chen系统的混沌序列结合，将第二部生成的置乱图像进一步像素扩散。

QR码经过置乱+扩散后很好的隐藏了QR码的原始信息

### QR码的生成

根据生成的散列值H，进一步生成相应的二维码。现如今，互联网上存在多种二维码生成工具，但是他们定制化能力很低，基于开源的C++项目qrencode，结合常用的Base64编码技术，开发了一个二维码生成软件，能够将图片或文本信息转换为二维码。

qrencode是一个广泛应用的C++开源库，专门用于生成QR Code二维码，qrencode能够高效地生成二维码，并支持多种平台。在生成二维码时，它会将输入的数据（如文本或URL）转换为QR Code格式。

此外，Base64编码是一种常见的数据传输编码方法，通常用于将二进制数据转换为可打印字符的形式。它使用64个字符表示任意二进制数据，这样可以有效地在网络上传输，并且具有加密和解密的优势，广泛应用于各种网络协议和数据传输中。

下来展示的是通过自定义的二维码生成软件生成的二维码效果：

（1）简单英文：

Xidian University is a science and engineering university 生成二维码图像：



Chen系统迭代图

（2）长英文：

Abstraction ability helps improve work efficiency and expand applications Range; Critical thinking is crucial for ensuring the reliability and fairness of AI applications Key; Innovative thinking ability drives the boundary expansion of AI applications, while ethics Consciousness ensures that the development of AI is in line with social values.

生成二维码图像：



Chen系统迭代图

1. 二维码内容：“hello”由SHA256算法得出的散列值

2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa7425e73043362938b9824

生成二维码图像：



Chen系统迭代图

由以上三种二维码对比可以看出，sha256算法生成的256位的散列值用16进制转换后的二维码复杂度适中。

### QR码的置乱操作

置乱操作的目的式将一个序列打乱，但是什么才是真正的乱？对于包含有n个元素的序列，由于这个序列的排列方式有n!种，所以意味着，将这个元素完全打乱后有n!种可能，如果序列组足够混乱则产生的每一种排列的可能都是相同的。以上是从序列整体来分析。对于序列中的每一个元素来说，如果序列足够的混乱，则某个元素出现在序列中任何位置的可能性都是相同的，即任何一个元素，出现在任意一个位置的概率都是1/n。

Fisher-Yates洗牌算法已经在第二章详细介绍，Fisher-Yates 算法的核心思想是逐步缩小随机选择的范围，确保每个位置的元素仅与未固定的位置交换。但是该算法的时间复杂度为O(),空间复杂度为O(*n*)。所以选择经过优化后的洗牌算法，Knuth-Durstenfeld算法，该算法不需要删除元素和额外空间，时间复杂度为O(*n*),空间复杂度为O(1)，Knuth-Durstenfeld算法的流程：

1. 输入：长度为n的数组A = []
2. 遍历方向：从后向前遍历，索引从n-1到0
3. 步骤：
   1. 对于每个位置i（从n-1到0），生成随机整数j∈[0,i]
   2. 交换A[i]和A[j]
   3. 循环步骤(1)(2)指导到达第一个位置
4. 输出：一个完全打乱的数组

由上述的算法步骤可以看出Knuth-Durstenfeld算法整体非常简单，我们可以很容易发现，算法中的“随机生成整数”步骤非常关键，因为这直接决定了洗牌算法的随机性。这里可以将洗牌算法和上一节生成的混沌序列结合。用混沌序列的值代替随机函数生成的j∈[0,i]。步骤如下：

步骤1：将元数据生成的二维码图像信息转变成比特流序列J。

步骤2：利用密钥和，结合logistic-tent混沌产生与比特流长短相等的混沌序列S（上一节已经详细介绍）。

步骤3：从比特流序列P的最后一位像素开始循环

步骤4：每次循环的像素位置为i，每次循环i都减小一位。

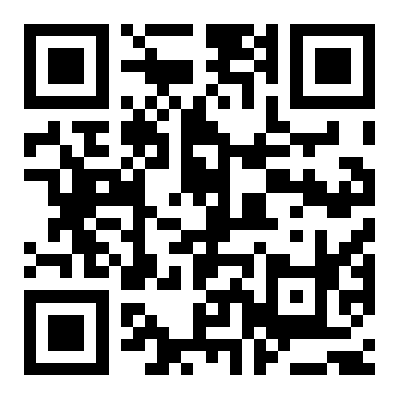
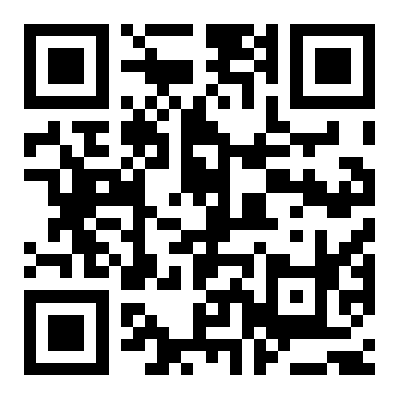
步骤5：利用混沌序列S生成随机下标j，如式所示。down()函数表示的是向下取整操作。



步骤 6：交换J(i)和J(j)

步骤 7：一直重复步骤3到步骤6到第一个元素，就得到置乱后的图像P。

经过上面的步骤就可以得到了置乱后的QR码，用 Knuth-Durstenfeld算法来保证理论上置乱的混乱性，又用logistic-tent混沌体系产生的混沌系统来保证算法中最关键的随机性。下面是经过置乱前的QR码和置乱后的QR码的对比。



Chen系统迭代图

### QR码的扩散操作

上一节已经对二维码进行了置乱操作，置乱操作通过改变像素的位置和排列方式，破坏图像的空间相关性。但是这样的置乱并没有改变像素原有的值，原始图片只是发生了像素的位置变化，并没有发生像素值的变化。这样置乱图片的统计学特征并没有和原始图像产生区别，比如置乱图像的直方图，像素值分布等。如果攻击者获得了部分的明文信息可能通过对比来推测置乱的规则，也可以利用统计学特征来还原信息。

本节的扩散操作就是要改变置乱图像的像素值，目的是通过扩散操作使像素值的分布趋于平均，这样直方图的分布就平坦化，隐藏了原始图像的统计学特征。而且扩散操作需要扩散密钥，扩散密钥和置乱密钥结合，使得攻击者及时破解了其中一个密钥也不能复原原始图像的内容。下面将详细介绍基于chen系统的DNA编码扩散操作。

DNA编码扩散操作需要确定原始图像和扩散矩阵的编码规则，原始图像和扩散矩阵的DNA运算规则，还有DNA解码规则。以上三种规则的选择会影响最终的扩散效果，本文将混沌序列和DNA编码扩散结合，用混沌序列的无序性来选择DNA编码扩散步骤中的各种规则选择。下面是DNA编码扩散算法的实现。

步骤 1：将混沌序列X的元素值映射到[0,1]内，将映射完毕的矩阵当作一个M\*N的扩散矩阵。如式所示，shape()函数是将混沌序列构造成一个M×N的矩阵C。式表示的是矩阵C每个元素的值的确定过程，i∈[1, MN]，函数down()是向下进行取整操作，这么做的目的是将混沌序列转成成一个M\*N的二值矩阵。





步骤 2：对扩散矩阵C和原始图像矩阵P进行压缩操作，每8位压缩成一个十进制的整数，每行不足8位的补零，这样做的目的式方便进行DNA的编码，因为每个DNA编码至少需要2位二进制的数。进行压缩后，矩阵每个元素就可以编码成3位DNA序列。这样就得到了混沌矩阵和原始图像矩阵

步骤 3：开始对和进行编码操作：将和分成16块，这样可以用多线程编码来提高DNA编解码的速度，同时将步骤1产生的矩阵C也分成 4×4的矩阵。需要对这两个矩阵进行DNA 编码，对矩阵的DNA编码方式由式决定。



其中表示置乱图像和扩散矩阵各块的编码方式，得到的数字分别对应着第二章给出的DNA编码的8种编码方式，在第二章已经给出了每个号码对应的编码方式。

接下来要确定矩阵C和扩散矩阵之间的DNA编码的运算规则了，在第二章种我们介绍了DNA编码的运算规则有三个，分别是加法、减法、异或和同或。可以用数字0到3来代表四种DNA的运算规则。通过式和混沌矩阵Y配合，确定了原始图像每一块区域和混沌矩阵C。



最后要确定DNA的解码规则，通过式和混沌矩阵Z配合，可以获得解码的方式。同编码方式一样，DNA解码方式有8种。



DNA 扩散过程如图 4 所示。

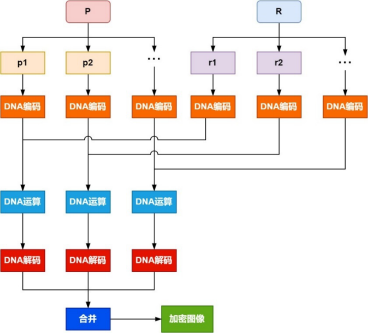


图 3.5 DNA 分块扩散过程

将扩散矩阵*C*和图像矩阵*P*执行压缩操作后，再将矩阵 和分成块矩阵，对分块后的矩阵依次进行 DNA 编码以及运算，再将运算后的结果进行DNA解码，所获的图像像素值与原始图像将完全不相同。合并后即可获得加密图像。

步骤 4 ：将步骤 3 循环 9 次运算，即可得到更加难以预测的加密图像。

## 实验分析

## 本章小节

# 结合HVS的DCT域二值图像水印算法

按照第三章的方法，将原始图像的特征信息生成了二维码，并且将二维码经过了置乱与扩散处理。得到了一张无序混乱的二值图像。第四章要做的就是将该二值图像当作数字水印隐藏到原始图像中去，并且载密图像要有一定的抗剪切，污损，难以察觉的特点。基于以上要求，本文提出结合HVS的DCT域的二值图像数字水印算法，该算法将载体图像分块分析，量化的分析图像每块不同区域的是否适合嵌入信息，为每块区域自适应设置不同的嵌入强度和嵌入数据量，然后根据混沌序列将秘密信息完全随机的嵌入到不同图像块DCT域中去，最后通过逆DCT变换恢复原始灰度图，加上原始载体图像的颜色通道，完成嵌入流程。通过加入高斯噪声、椒盐噪声以及JPEG压缩和剪切等处理操作，结果表明了该算法具有很好的视觉掩蔽特性和鲁棒性。

## 算法模型介绍

中在介绍DCT域嵌入算法前，我们已经做了一些准备工作，包括：

1. 对二维码信息进行混沌置乱；
2. 对混沌置乱后的水印信息进行混沌扩散，实现二次加密；

本章设计的DCT域的数字水印算法的流程如下：

步骤1：将载体图像转换成灰度图像，按照8×8分块。

步骤2：计算每一块图像的均值、方差、熵值，并给图像块打分，分数越高，越适合隐写数据，相应的嵌入强度越高。

步骤3：将8×8块每一块都进行DCT变换。

步骤4：按照第三章产生的混沌序列随机选择插入图像的DCT系数位置。

步骤5：将秘密信息按照步骤4选择的插入位置，已经步骤2计算得到的嵌入系数对图像进行嵌入操作。

步骤6：重复步骤4，5，直到所有的秘密元素都被隐藏到载体图像中。

步骤7：将嵌入完成的载体图像恢复RGB通道色彩。完成整个嵌入流程。

## 离散余弦变换（DCT）介绍

### DCT变换的原理

DCT的全称是[离散](https://so.csdn.net/so/search?q=%E7%A6%BB%E6%95%A3&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/yanceyxin/article/details/_blank)余弦变换(Discrete Cosine Transform)，DCT可以将空域上的信号映射到频域上。从而可以让我们将对信号的研究和操作从空域空间转到了频域空间。由于DCT域的水印算法发生在频域空间，他的水印嵌入和空域常用的LSB算法比起来具有更高的稳定性，对常见的攻击有优秀的鲁棒性，同时DCT水印技术具有良好的可逆性，所以我们可以靠DCT的逆变换将频域信息重新转换成图像信息，并且对图像几乎无损。

DCT可以将一维或者二维的离散序列转换成一组余弦函数的系数序列。这组余弦函数表示了原始数据中的频谱特征。基于以上的原理，我们可以将图像的信号分解成一系列的不同频率成分的系数。式就是一维的DCT的定义。



在上式中，*f(x)*就是一维的信号序列，*C(u)*如式所示：



DCT同时可以进行可逆变换，这也方便了我们对信号进行还原操作，DCT的逆变换如式所示：



我们发现了，DCT无论正向变换还是逆向变换，都有式在



我们将他提取出来可以得到式。其中的*M*可以表示成为式一个矩阵。





了解到完一维的DCT变换，可以将定义推广到二维的DCT变换中，将之前的一维序列*f(x)*推广到二维序列*f(x,y)*。*f(x,y)*的DCT变换为式*：*



二维的DCT逆变换如式所示：



### DCT的系数介绍

一般进行DCT变换首先要做的就是对图像进行分块，一般选取的分块是8×8分块，意思是将原始图像分成8×8像素大小的小块，然后对每一块进行DCT变换，这样每一个小块都会得到64个DCT系数。为了使DCT系数矩阵能够呈现出规律性，会将系数矩阵进行Zig-Zig排序，如图4.1所示。

DCT系数分为直流分量（DC系数）和交流分量（AC系数）。其中DC系数位于系数字水印的数矩阵的左上角(0,0)位置，他反映了图像块的平均亮度或能量，是其他所有像素值的加权平均。AC系数则是除了DC系数外的其他系数，他反映了图像中不同频率的细节信息。DCT变换后大部分的能量集中在低频区域，对应着图像的平滑区域，高频区域对应的是图像的边缘，纹理等细节。图4.1就是经过DCT变换后的一个系数矩阵。

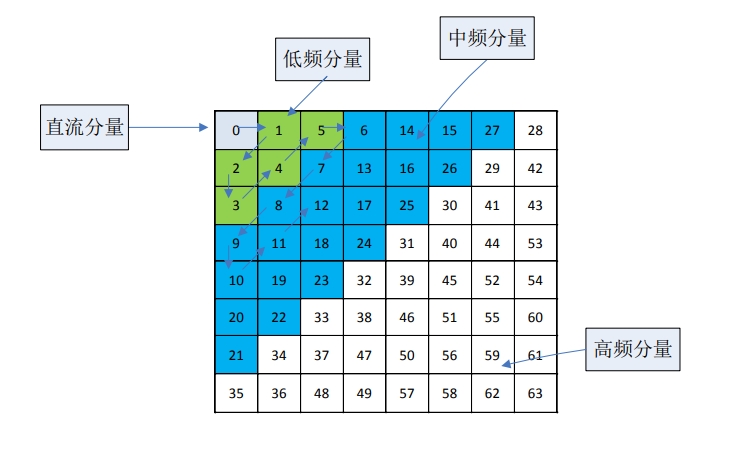


图 4.1 DCT系数矩阵

使用python对lema.bmp图像做DCT变换，可以更加直观的看到DCT系数代表的能量分布：

|  |  |
| --- | --- |
| lena512  Lena原始图像 | DCT变换后的lena图  DCT变换后能量分布 |

图 4.2 DCT变换

为了让图片在人类视觉中的尽量小，同时还要兼顾隐写效果的鲁棒性，一般选择在中频系数中隐藏数据。因为低频系数的改变会引起图篇画面的显著变化，高频会使隐写的鲁棒性会大大降低，因为压缩算法会破坏图片的高频信息，使隐写的数据丢失。所以要在隐蔽性和鲁棒性之间建立一个平衡，兼顾隐蔽性和鲁棒性，本文根据人类视觉系统的特点设计的水印算法就是为了平衡隐蔽性和鲁棒性。

### DCT水印嵌入方式

主要介绍三种的水印嵌入方式

1. 加法嵌入

如式所示，加法嵌入中是DCT系数矩阵中的第i个元素，现在要在这个元素中嵌入信息，就给元素上加上一个权重为r的的隐藏信息w。



可以通过控制r的大小来控制嵌入强度，r越大嵌入的强度越大，但是对原始图像的影响越明显。

1. 乘法嵌入

如式所示，加法嵌入中是DCT系数矩阵中的第i个元素，现在要在这个元素中嵌入信息，就给元素上乘上一个一加权重为r的的隐藏信息w。



1. 对比嵌入

如式所示，加法嵌入中是DCT系数矩阵中的第i个元素，现在要在这个元素中嵌入信息，就给元素上乘上一个一加权重为r的的隐藏信息w。



## 结合HVS的图像打分算法

前文介绍过，图像的DCT算法一般先将原始图像分成8×8的小块，秘密信息就隐藏到这些图像小块中。本节将介绍一种和HVS结合的分块图像的打分算法，目的是将所有的8×8小块按照是否适合隐写进行打分。从而实现分数越低的的（不适合隐写）排序越靠后，分数大的图像块（适合隐写）排序越靠前。通过排序来挑选隐藏秘密信息的图像块。

### 人类视觉系统HVS

人类的视觉系统的特点在数字图像处理中有广泛的应用，这要是利用了人类视觉系统对图像频率、亮度、纹理、对比度有不同的感知特性。从而可以通过调整人类视觉系统不敏感的一些特性来隐藏信息并且几乎不会引起视觉系统的察觉。比如人类视觉系统对纹理密集，亮度高的图像区域的细微变化不敏感，所以当嵌入载体图像的秘密信息低于人类视觉系统的对比门限(Contrast Sensitivity Threshold, CST)，眼睛就不会察觉到图像有修改的痕迹。本文主要利用了人类视觉系统的一些特性：

1. 人类视觉系统对图像的边缘信息敏感，所以当图像区域有丰富的边缘信息，就要尽量少的去修改。
2. 人类视觉系统对图像的平滑区域的变化十分敏感，所以在图像的平坦区域尽量少的嵌入信息。相反，如果图像区域有丰富的细节纹理，那么隐写数据带来的图像噪声和失真问题就会引起视觉系统的注意。
3. 人类视觉系统对不同的灰度有不同的敏感性。人眼对中度的灰度最为敏感，然后对灰度高或者灰度低的部分不敏感，而且这种敏感度的下降是非线性的下降，所以要将秘密信息隐藏在图像的灰度高或者灰度低的区域。

基于上述的人类视觉来设计的嵌入方案，主要是通过分析图像的纹理，灰度，是否含有边缘信息来确定该图像区域是否适合嵌入秘密信息。

### 分块图像排序原理

之前介绍了人类视觉系统的一些特性，比如果图像的纹理越复杂，背景的灰度偏高或者低，那么人类的视觉系统对图像的变化越不敏感，所以要嵌入秘密信息到图像中去就要嵌入到这种特征的图像块中去。可以通过分析图像的方差可以关联到图像的纹理，当方差大时，图像应该包含着比较复杂的纹理图案；分析图像的均值可以得到图像的整体亮度信息；可以通过Sobel边缘检测方式算子计算图像块的边缘纹理。下面将综合这三个条件对分块的图像进行排序，排序越靠前的越适合嵌入信息。

1. 计算分块图像的边缘纹理

图像的边缘检测方式有很多，这里选用了Sobel边缘检测，Sobel边缘检测比Candy检测的准确性差但是效率却高。Sobel边缘检测主要的原理是使用两个3x3的卷积核，分别检测每个像素水平方向梯度和垂直方向的梯度。通过将这两个方向的梯度幅值结合，可以得到边缘的整体强度。如式所示，对所有分块图像进行式的量化，得到的边缘复杂度*E*，*E*越高的边缘越复杂。



计算出单个图像块的边缘复杂度后要进行归一化处理，这样才能确定该分块图像在所有图像分块中的复杂程度，也方便后续对每个图像块进行排序，归一化公式也很简单如式所示，表示归一化后的分数，*max(E)*表示所有分块中最大的边缘复杂度，*min(E)*为最小的边缘复杂度，表示当前图像块的边缘复杂度。



1. 计算分块图像的均值

HVS对图像边缘纹理敏感，其次就是对亮度敏感。HVS对于过亮或者过暗区域敏感度不高，针对这个特点，计算图片块的均值*μ*。如式所示，其中*M*和*N*都是8，和表示的式图像中第*i*行第*j*列的灰度值。



同样的给均值也要做归一化处理，如式所示，表示归一化后的分数，表示当前图像块的均值。由于人眼对过暗或者过亮的区域不敏感，所以越靠近平均值越不适合隐写信息。



1. 计算分块图像的方差

图片块的方差反映了图片的平滑程度，越大则图片块的包含这较多的纹理或边缘信息，这些地方适合隐写信息，当过小则反映了图块较为平滑，不适合隐藏信息。式表示了图块的计算过程，其中*M*和*N*都是8，和表示的式图像中第*i*行第*j*列的灰度值，*μ*表示图块的均值。



同样对方差进行归一化处理。如式所示，表示归一化后的分数，*max()*表示所有分块中最大的方差，*min()*为最小的方差，当最大和最小方差相同时=0，表示当前图像块的方差。



1. 计算分块图像的得分

上面介绍了给分块图片排序的三个重要指标，按重要程度分别是边缘纹理，亮度，纹理细节，并且已经对他们进行了归一化处理，现在要综合这三个评价指标，按照重要程度给每个参数设置权重，按照重要程度分别设置为=0.5，=0.3，=0.2。这样就得到了式，数值越大则越适合嵌入图像。



根据式，就可以计算所有分块图片的分数，将图片按照分数从大到小排列，我们就得到了一个分块图片序列*P*。在序列前面的图块会优先被选择当作隐藏秘密信息的图像块。

## 图像水印的嵌入算法

前文已经介绍了结合HVS的图像块打分算法，通过这个算法我们可以量化每个图像块的隐写能力。下面本节将从嵌入和提取一比特信息开始，介绍图像水印嵌入流程和其中的原理。

### 一比特信息的嵌入与提取

拿图像lena举例，lena是512×512的灰度图像，将lena图像当作载体图像。对lena图像进行8×8的分块，那么lena图像一共被分成了64×64块，每一个小块隐藏1比特的秘密信息，那么lena图片一共可以隐藏4096比特的秘密信息。我们的秘密信息是50×50的二值图像，可以将秘密图像转成2500比特的0，1序列，白色像素为0黑色像素为1。可以从lena图像中选取2500块来隐藏二维码序列的信息。 接下来将介绍如何在8×8的图像分块中隐藏1比特的信息，和如何将信息提取出来。

1. 一比特信息的嵌入

嵌入过程如图4.2所示，首先判断嵌入的信息是0还是1，当我们想在这个8×8的图像块嵌入0时，就不在这个8×8图像块嵌入任何信息；当我们想在这个图像块嵌入1时，我们就按照式，即，将22个随机0，1序列（随机0，1序列的生成规则会在章节4.4.2介绍）按照一定的嵌入强度r（r的选取规则会在下面的章节介绍）嵌入到图块的22个中频系数中，图4.1已经标注好了DCT变换后的中频系数位置。选接下来要介绍如何从图像块中提取刚刚嵌入的1比特信息。

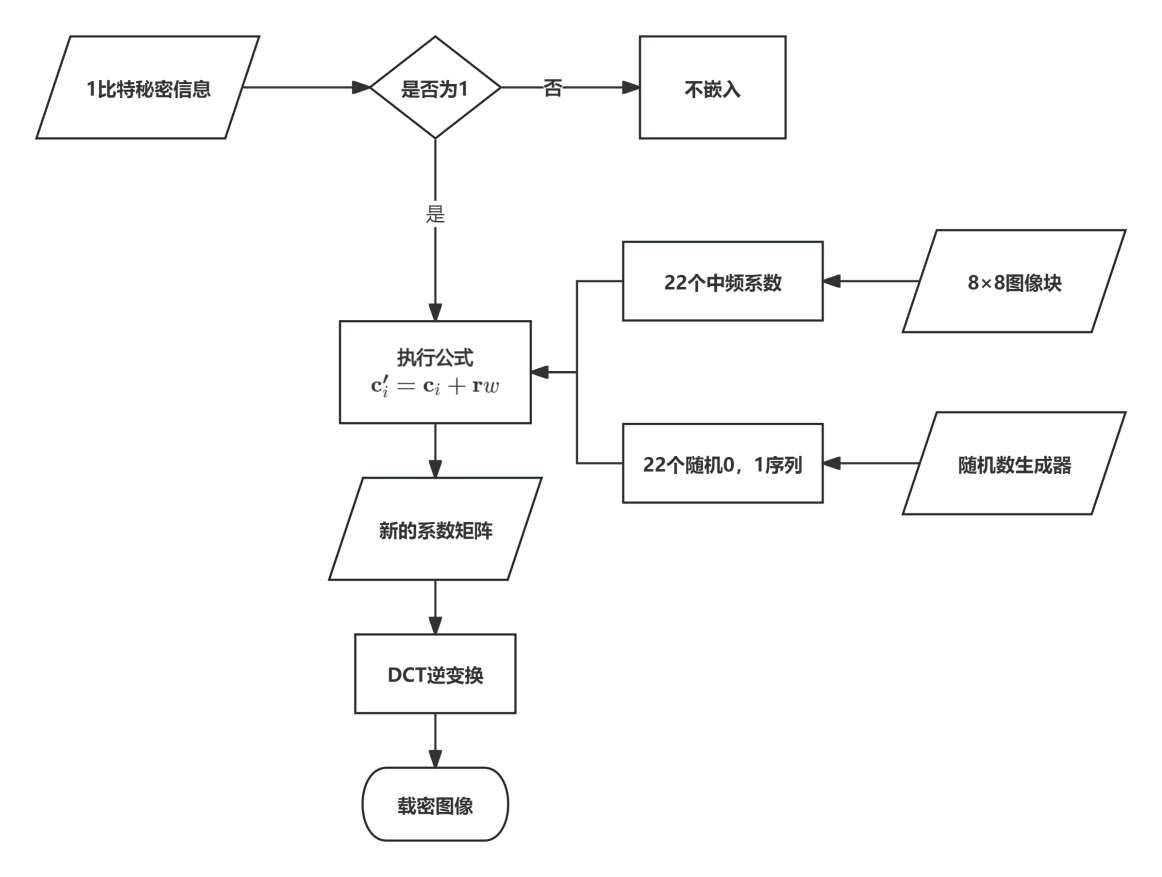


图 4.3 1比特信息的嵌入

1. 一比特信息的提取

信息的提取过程就是嵌入过程的逆过程。大体过程如图4.3所示，首先从原始图像和载密图像选出要提取1比特信息的8×8图像块，然后对选出的两个图像块做DCT变换，将两者变换后的DCT系数矩阵中的中频系数相减。如果所有的差值为0到Δ：没有嵌入信息，代表着秘密信息为0；如果差值大于Δ：嵌入了信息，代表秘密信息为1；这里的Δ取值为5。

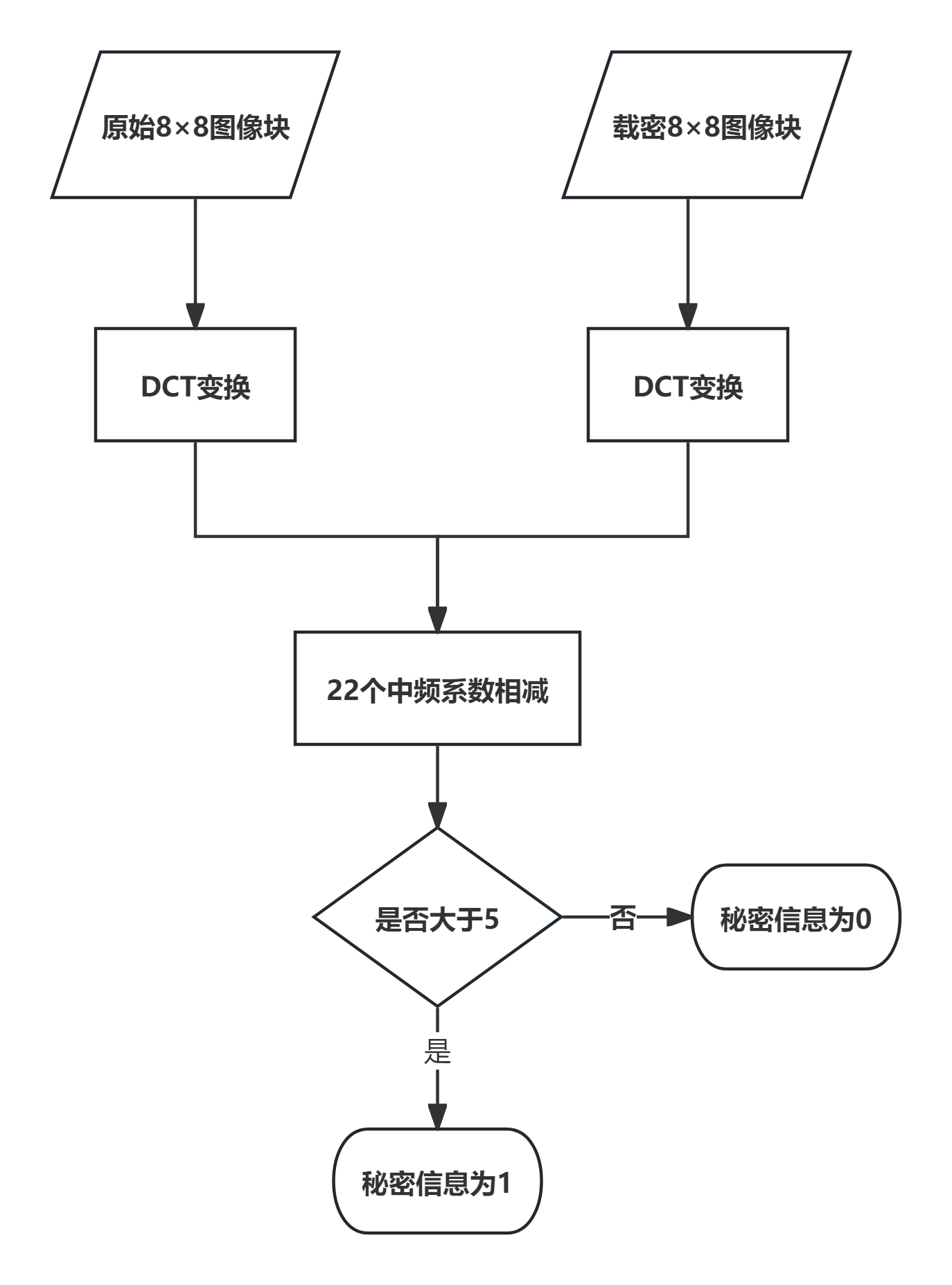


图 4.4 1比特信息的读取

### 确定秘密信息嵌入位置

在4.2.1中介绍了一个比特的秘密信息是如何在8×8的图像块中嵌入和提取的。现在要解决多个比特秘密信息如何确定每个比特嵌入到哪个图像块中。对于要隐藏的秘密信息：一个50×50的二值图像。我们需要2500个图像块来嵌入秘密消息，通过4.3.2节，已经得到了排序过后的，包含64×64个图形块的序列P。我们选择前2500个图像块嵌入消息。下面将介绍如何确定每个比特信息在哪个图像块中嵌入。

本节要达到的效果是，2500个比特完全随机的散布在挑选出来的2500个图像块中，达到完全混乱的问题。这种需求和之前提到过的二维码的完全随机置乱几乎一样。两者都是要求，每个元素落在每个位置上的概率都是相同的。因此将采用同3.4.2节相同的方式来确认

### 水印的嵌入和提取步骤

前面小节我们已经介绍了如何将1比特信息在8×8图像分块中隐藏和提取。然后介绍了隐藏和提取过程中随机0，1序列的生成过程。下面将详细介绍如何将秘密二值图像隐藏到载体图像中，以及如何将秘密图像从载体图像中提取。秘密二值图像选择50×50的二维码图像，载体图像选择512×512的lema彩色图像。

1. 水印的嵌入流程。

图像水印的嵌入流程如图4.5所示：

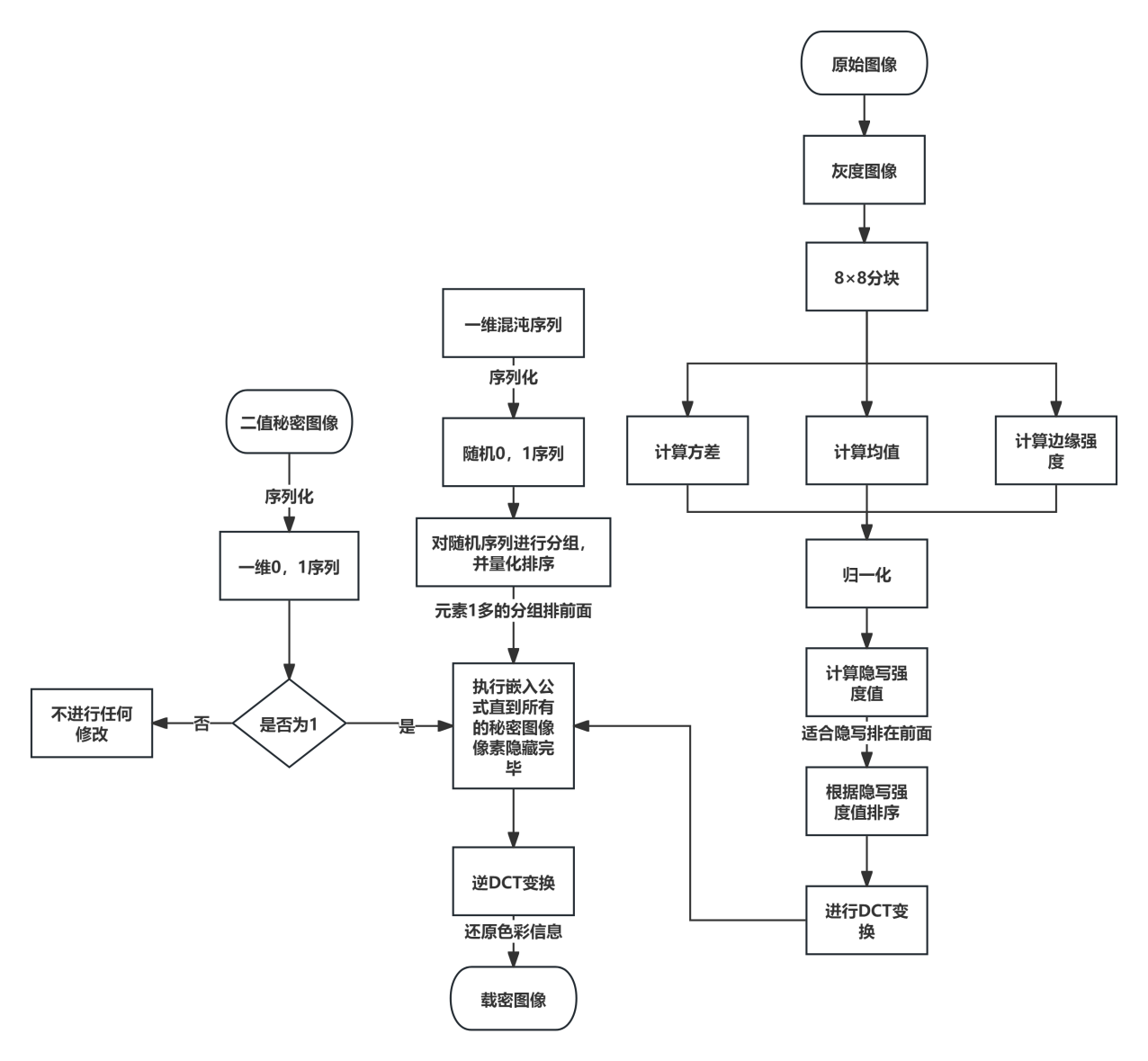


图 4.5 水印嵌入流程图

图4.5展示了秘密图像如何隐写到载体图像的全流程，下面将详细介绍所有步骤：  
 步骤一：对载体图像结合HVS进行排序。

（1）将载体图像按8×8分成互不重合的图像子块。

（2）计算每个图像子块的方差，均值，边缘强度，并且都做归一化处理。详细步骤已经在4.3.2详细介绍。

（3）对每个子块计算适合隐写的分数，按分数排序，分数越高越适合隐写，排序越靠前。得到排序后的图块序列*P*。

步骤二：将二值秘密图像进行序列化。

（1）将二值图像转成一维序列，白色像素块代表0，黑色像素块代表1。

（2）得到二值图像的一位序列*C*。

步骤三：将混沌序列映射成随机0，1序列。

1. 将3.2.2计算出的混沌序列*S*进行映射，生成符合要求的新的随机0，1序列。具体步骤已经在4.4.2中详细给出。

步骤四：对步骤一已经排序好的图块序列*P*所有的图块做DCT变换。

步骤五：查看*C(i)*是否为0，进入步骤六；如果*C(i)*为1，则进入步骤七。

步骤六：*P(i)*无需做任何操作，查看*i*是否为*C*序列的最后一个元素。若不是，则*i*增加一位，重复步骤五；若是最后一位，执行步骤八。

步骤七：对图块*I(i)*执行1比特的嵌入操作，嵌入操作在4.4.1已经详细介绍。并查看*i*是否为*C*的最后一个元素。若不是则i增加一位，重复步骤五；若是最后一位，执行步骤八。

步骤八：对图片序列*P*所有的图像块做逆DCT操作，并将所有的图像块放回原始位置，组合为图像*E*。

步骤九：将图像*E*的色彩通道重新融合，得到载密图像。

1. 水印的提取流程。

图4.6表示了提取水印的所有流程，如图所示：

## 实验结果和分析

## 本章小节

本章介绍了基于SGX的国密算法。首先介绍了SGX可信库中可直接调用的密码算法及其在SGX中的功能，针对缺少国密算法的问题，提出基于SGX的数据安全增强国密算法。其次从SGX应用设计出发，阐明引入SGX技术实现密码模块所面临的挑战，强调设计过程需要注意的关键问题，并详细介绍了SGX应用设计通用框架。然后结合国密算法设计原理和SGX设计通用框架设计了基于SGX的国密算法，包括SM2数字签名算法、HMAC-SM3算法、SM4算法，并详细说明了调用每个算法时，可信区与非可信区的运行交互流程。最后对SGX国密算法进行了功能、性能和安全性测试与分析，测试结果表明基于SGX的国密算法可以完成相应功能；SGX保护国密算法运行环境带来的开销在可接受范围内；基于SGX保护国密算法运行过程的设计，能够有效阻止关键信息被非法访问和篡改，为国密算法运行态提供隔离保护。

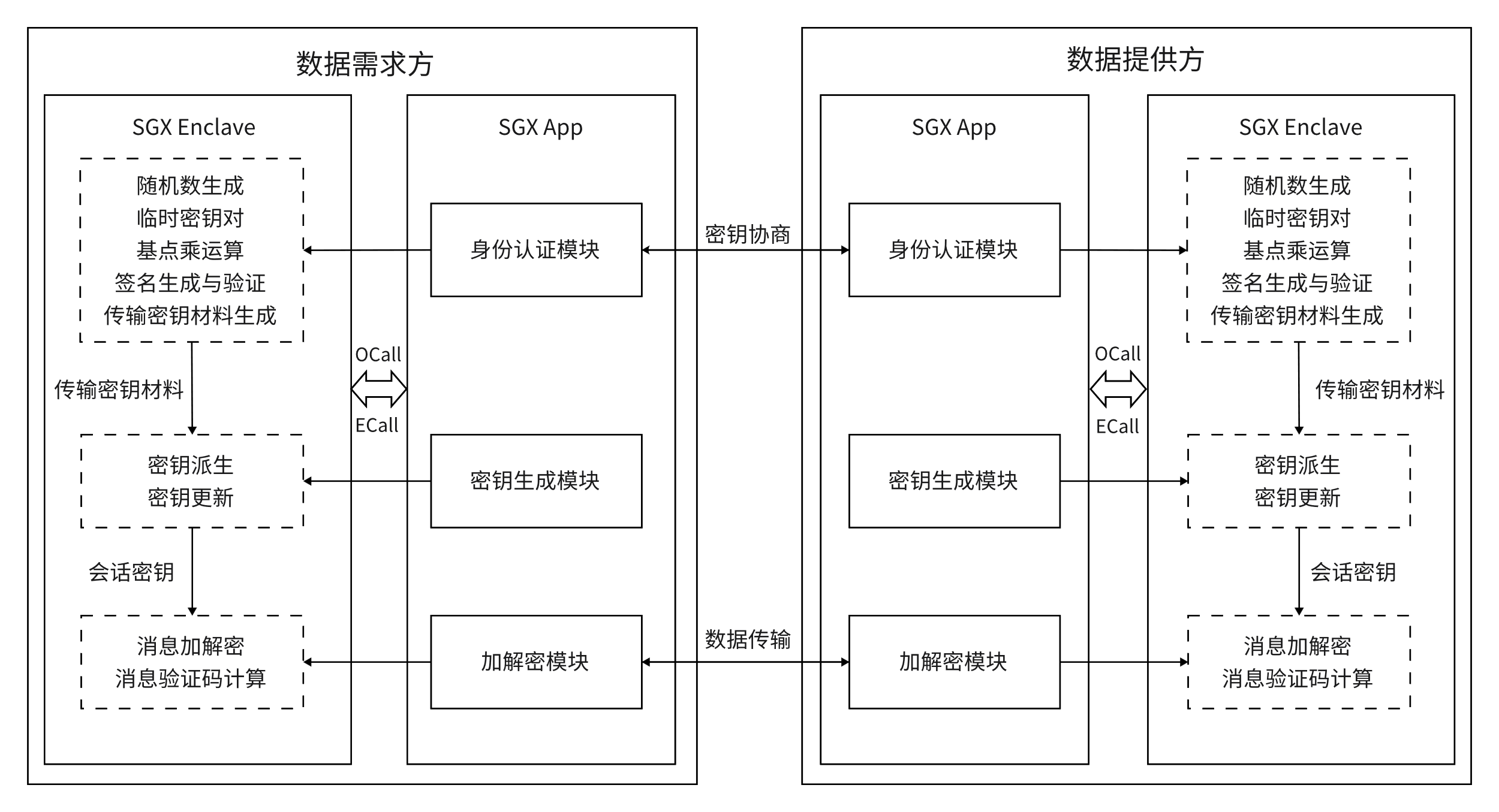
# 基于SGX的密码模块设计与实现

本章基于第三章和第四章的内容，对基于SGX的多方数据安全计算密码模块进行了设计和实现，其中阐述了各个密码模块的功能，详细介绍了实现细节，包括协议数据包格式、数据在可信区和非可信区的交互以及可信接口的具体形式等。最后对密码模块进行了功能和性能测试，并对测试结果做了总结。

## 密码模块设计

根据数据安全交换传输协议的设计，基于SGX的多方数据安全计算密码模块设计如图5.1所示，密码模块包含身份认证模块、密钥生成模块和加解密模块。

身份认证模块负责数据需求方和数据提供方之间的认证密钥协商，根据认证密钥协商协议的设计，数据需求方和数据提供方涉及秘密信息的处理过程包括随机数生成、随机数在椭圆曲线上的基点乘运算、数字签名生成、数字签名验证、传输密钥材料生成等。因此将这些计算过程放入可信区。传输协议消息头的填充以及数据的接收和发送在非可信区完成。密钥生成模块负责基于传输密钥材料派生出传输密钥，此模块不涉及数据的发送和接收。加解密模块负责传输数据的加解密和消息认证码的生成，因此将加解密操作和MAC值计算过程放入可信区，传输协议消息头的填充以及数据的接收和发送在非可信区完成。

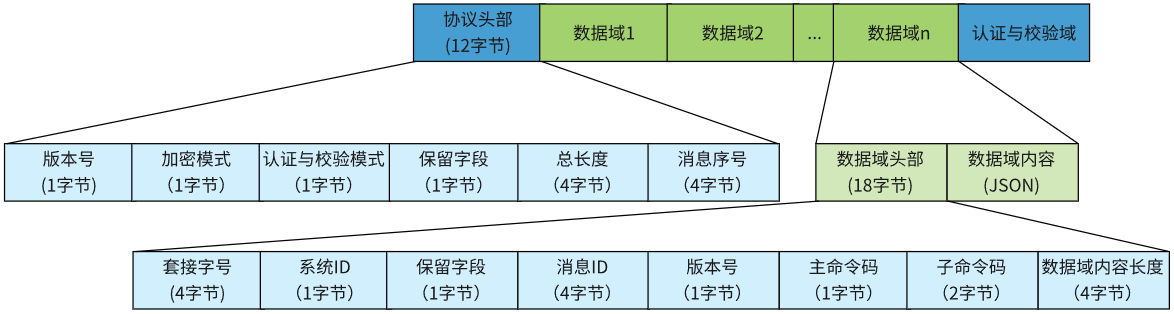


基于SGX的多方数据安全计算密码模块

## 密码模块实现

### 传输协议数据包定义

多方的数据安全计算跨平台通信采用了统一的数据包格式，有助于不同参与方之间的数据交换和解析，使得平台在处理大量数据时能够保持高效和稳定。格式具体定义如图5.2所示。



通信数据包格式

数据包头部字段解释如下。

（1）版本号：表示满足规范的协议版本。

（2）加密模式：标识是否加密，用何种加密算法加密。

（3）认证与校验模式：标识是“签名”还是“MAC”。

（4）保留字段：暂时保留，后续开发使用。

（5）总长度：报文的总长度，包括协议头部、数据域、认证与校验域的长度之和。

（6）消息序号：密钥协商后，发送报文通过消息序号来抵抗重放攻击。

数据域头部字段解释如下。

（1）套接字号：通信引擎自动填充。

（2）系统ID：用于标识接收或发送消息的系统类型。

（3）保留字段：暂时保留，后续开发使用。

（4）消息ID：后台根据此消息序号判断是否为重放消息。

（5）版本号：表示业务协议的版本。

（6）主命令码：用于区分密钥协商、数据交换、安全计算。

（7）子命令码：标识数据域的具体内容。

（8）数据域内容长度：数据域JSON的长度。

### 身份认证模块

数据需求方和数据提供方基于身份认证模块进行认证密钥协商时，流程如下。

（1）数据需求方首先调用initialize\_enclave函数创建并初始化Enclave，在非可信区调用可信接口ecall\_sgx\_gen，程序运行上下文从非可信区切换到可信区。在可信区内首先调用sgx\_read\_rand函数生成随机数，sgx\_read\_rand函数是SGX SDK提供的用于生成随机数的接口，通过Intel处理器内部的硬件随机数生成器来实现，该生成器基于物理过程产生随机数，提供高质量的随机性。然后调用SM2椭圆曲线基点乘运算函数sgx\_sm2\_base计算得到随机数的基点乘结果。最后调用sgx\_sm2\_keygen函数生成临时公私钥对。随机数和临时私钥不出可信区，随机数的基点乘结果和临时公钥作为输出值返回非可信区。

（2）数据需求方调用可信接口ecall\_sgx\_sm2\_sign，程序运行上下文从非可信区切换到可信区，在可信区用数据需求方预存在可信区的长期私钥对消息签名，签名结果返回非可信区，在非可信区确认数据包序号。通过transmit\_client\_main\_key\_step1将消息发送给数据提供方。

（3）数据提供方通过transmit\_server\_main\_key\_step1在非可信区响应数据需求方的请求，创建并初始化一个Enclave，先提取出证书进行验证，验证失败则结束会话。验证成功则调用可信接口ecall\_sgx\_sm2\_verify进入可信区用数据需求方公钥对签名进行验证，验证结果以0或1的形式返回到非可信区，非可信区通过返回值判断签名验证失败还是通过，值为1即验证通过，值为0则验证失败并报错。验证通过则继续下一步骤。

（4）数据提供方在非可信区调用可信接口ecall\_sgx\_gen，程序运行上下文从非可信区切换到可信区。在可信区内首先调用sgx\_read\_rand函数生成随机数，然后调用sgx\_sm2\_base函数计算得到随机数的基点乘结果。最后调用sgx\_sm2\_keygen函数生成临时公私钥对。随机数和临时私钥不出可信区，随机数的基点乘结果和临时公钥作为输出值返回非可信区。

（5）数据提供方调用可信接口ecall\_sgx\_main\_key，程序运行上下文从非可信区切换到可信区。在可信区内完成传输密钥材料的生成，传输密钥材料保留在可信区。

（6）数据提供方在非可信区从接收到的消息中提取出需求方数据包序号进行加一操作，并确定提供方数据包序号，最后调用可信接口ecall\_sgx\_sm2\_sign，在可信区用数据提供方预存在可信区的长期私钥对消息签名，签名结果返回非可信区，在非可信区通过transmit\_client\_main\_key\_step2发送消息给数据需求方。

（7）数据需求方在非可信区通过transmit\_server\_main\_key\_step2接收到数据提供方的消息后，首先提取提供方的证书进行身份认证。认证通过则提取签名，调用可信接口ecall\_sgx\_sm2\_verify函数进入可信区用数据提供方公钥对签名进行验证，验证结果以0或1的形式返回到非可信区，非可信区通过返回值判断签名验证失败还是通过。认证通过则继续进行下一步骤。

（8）数据需求方调用可信接口ecall\_sgx\_main\_key，程序运行上下文从非可信区切换到可信区。在可信区内完成传输密钥材料的生成，传输密钥材料保留在可信区。程序运行上下文从可信区切换回非可信区。

|  |
| --- |
| //初始化一个安全区  int initialize\_enclave(void)；  //生成随机数、计算点乘等（可信接口）  void ecall\_sgx\_gen([out,count=64]unsigned char \*pucRandG，[out,count=64]unsigned char \*pucPubkey);  //基点乘运算  int sgx\_sm2\_base(unsigned char \*pucK,unsigned char \*pucOutX,  unsigned char \*pucOutY);  //生成临时密钥对  void sgx\_sm2\_keygen(unsigned char \*pucPri,unsigned char \*pucPubX,  unsigned char \*pucPubY);  //生成签名（可信接口）  void ecall\_sgx\_sm2\_sign( [in,count=uiDataLen] unsigned char \*pucData,  unsigned int uiDataLen,[out] unsigned char aucR[32],  [out] unsigned char aucS[32]);  //数字签名算法中杂凑值计算  void sgx\_sm3\_hash(unsigned char \*pucData,unsigned int uiDataLen,  unsigned char aucDigest[32]);  //基于签名者杂凑值生成签名  int sm2\_sign(const unsigned char \*pucPri,unsigned char \*pucDigest,unsigned char \*pucRand,unsigned char \*pucR,unsigned char \*pucS);//验证签名（可信接口）  int ecall\_sgx\_sm2\_verify([in,count= uiDatalen] unsigned char \*pucData,  unsigned int uiDatalen,[in] unsigned char aucR[32],[in] unsigned char aucS[32]);  //基于签名者杂凑值验证签名  int sgx\_sm2\_verify(unsigned char \*pucR,unsigned char \*pucS,unsigned char \*pucDigest,unsigned char \*pucPubX,unsigned char \*pucPubY);  //生成传输密钥材料（可信接口）  void ecall\_main\_key();  //数据需求方密钥协商第一步，发送消息给数据提供方  int transmit\_client\_main\_key\_step1(void \*pucSend );  //数据提供方密钥协商第一步，接收需求方消息  int transmit\_server\_main\_key\_step1(void \*pucRecv );  //数据提供方密钥协商第二步，发送消息给需求方  int transmit\_server\_main\_key\_step2(void \*pucSend );  //数据需求方密钥协商第二步，接收提供方消息  int transmit\_client\_main\_key\_step2(void \*pucRecv ); |

### 密钥生成模块

数据需求方和数据提供方基于密钥生成模块生成初始会话密钥和定期更新密钥。首先，双方调用可信接口ecall\_sgx\_two\_kdf，程序运行上下文从非可信区切换到可信区，在可信区内，基于传输密钥材料和公共信息生成中间值伪随机密钥和初始会话密钥，伪随机密钥和初始会话密钥都留存在可信区。当需要定期更新会话密钥时，双方可以通过定期更新计数器，调用可信接口ecall\_sgx\_single\_kdf，程序运行上下文从非可信区切换到可信区，在可信区内，基于伪随机密钥和公共信息生成新的会话密钥，会话密钥保留在可信区。

|  |
| --- |
| //双阶段密钥派生(可信接口)  void ecall\_sgx\_two\_kdf();  //单阶段密钥派生(可信接口)  void ecall\_sgx\_single\_kdf(); |

### 加解密模块

加解密模块在功能上以4.3.3小结的内容为基础，增加了多种分组加密模式，包括OFB模式、CTR模式、CFB模式和CBC模式。基于密钥生成阶段，会话密钥在可信区生成，且不出可信区。因此加解密模块中，SM4分组密码算法和HMAC-SM3算法的可信接口的设计中，密钥不再作为参数从非可信区传入可信区。数据需求方和数据提供方基于加解密模块加解密消息以及计算消息认证码。当需要使用输出反馈模式加解密时，调用ecall\_sgx\_sm4\_ofb函数；当需要使用计数器模式加解密时，调用ecall\_sgx\_sm4\_ctr函数；当需要使用密文反馈模式加解密时，调用ecall\_sgx\_sm4\_cfb函数；当需要使用密码分组链接模式加解密时，调用ecall\_sgx\_sm4\_cbc函数；以上四种加密模式都不提供消息认证功能，因此进入可信区后使用会话密钥执行加解密操作，结果返回非可信区。数据传输阶段还需要调用ecall\_sgx\_hmac\_sm3函数进入可信区使用会话密钥计算消息的MAC值，MAC值返回非可信区。发送方将消息认证码和密文一起发送给对方；接收方通过比对生成的MAC值和接收到的消息认证码判断消息的完整性。

|  |
| --- |
| //SM4算法CBC模式加解密（可信接口）  void ecall\_sgx\_sm4\_cbc(int iMode, unsigned int uiInputLen, [in,count =uiLength] unsigned char \*pucInput, [out]unsigned int \*uiOutputLen, [out,count=3000] unsigned char \*pucOutput);//SM4算法CTR模式加解密（可信接口）  void ecall\_sgx\_sm4\_ctr(int iMode, unsigned int uiInputLen, [in,count =uiLength] unsigned char \*pucInput, [out]unsigned int \*uiOutputLen, [out,count=3000] unsigned char \*pucOutput);//SM4算法OFB模式加解密（可信接口）  void ecall\_sgx\_sm4\_ofb(int iMode, unsigned int uiInputLen, [in,count =uiLength] unsigned char \*pucInput, [out]unsigned int \*uiOutputLen, [out,count=3000] unsigned char \*pucOutput);//SM4算法CFB模式加解密（可信接口）  void ecall\_sgx\_sm4\_cfb(int iMode, [in,count=16] unsigned char \*pucIv, unsigned int uiInputLen, [in,count =uiLength] unsigned char \*pucInput, [out]unsigned int \*uiOutputLen, [out,count=3000] unsigned char \*pucOutput);//HMAC-SM3算法（可信接口）  void ecall\_hmac\_sm3(unsigned int uiDataLen, [in,count=uiDataLen] unsigned char \*pucData, unsigned int uiHmacLen, [out] unsigned char aucHmac);  //数据需求方发送消息  int transmit\_client\_send (void \*pucSend );  //数据需求方接收消息  int transmit\_client\_recv (void \*pucRecv );  //数据提供方发送消息  int transmit\_server\_ send (void \* pucSend );  //数据提供方接收消息  int transmit\_server\_recv (void \* pucRecv ); |

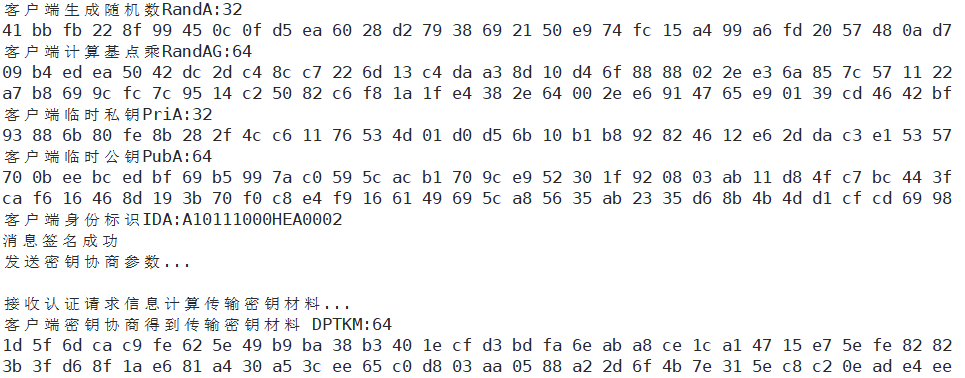
## 密码模块测试

### 测试环境

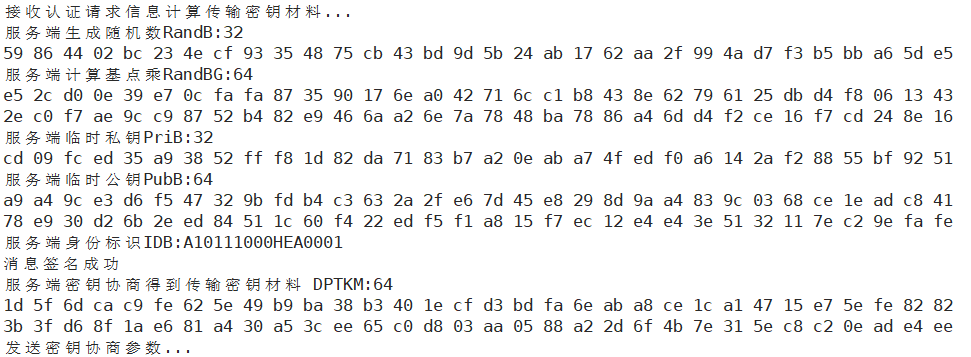
本节实验在两台配置相同且启用Intel SGX的云服务器上进行，云服务器CPU版本是Intel(R) Xeon(R) Platinum 8269CY CPU @ 2.50GHz，内存大小为8G。实验在CentOS 7.7的64位操作系统中进行，内核版本为3.10.0-1062。实验使用的SGX SDK版本为sgx\_linux\_x64\_sdk\_2.10.100.2，gcc编译器版本是gcc 8.3.1，GmSSL版本为GmSSL 3.1.1。

### 功能测试

在密码模块功能测试中，模拟银行征信场景下，数据需求方（客户端）向数据提供方（服务端）发起数据交换请求。首先，客户端与服务端基于身份认证模块通过密钥协商得到传输密钥材料。客户端测试结果如图5.3，服务端测试结果如图5.4，双方能够基于身份认证模块协商出一致的传输密钥材料。

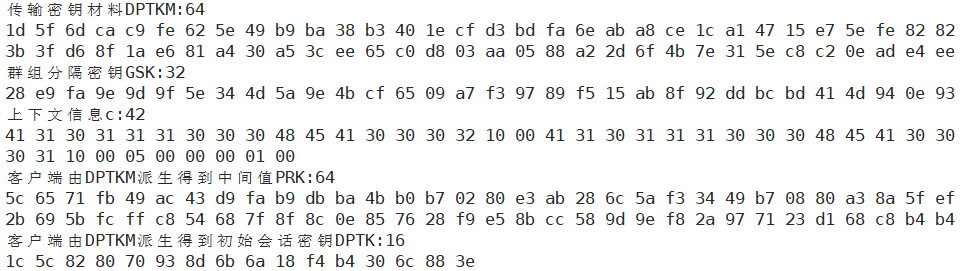


客户端密钥协商测试

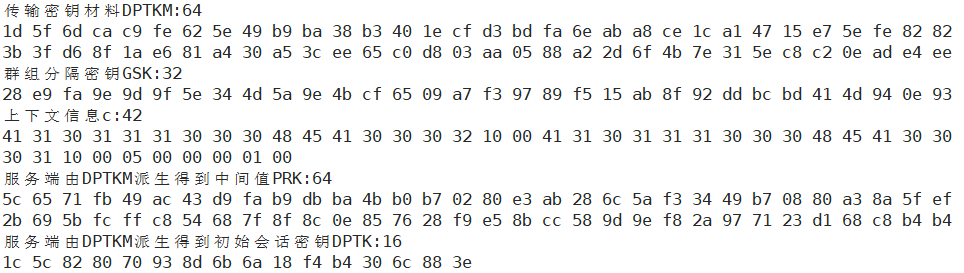


服务端密钥协商测试

客户端与服务端各自基于密钥生成模块派生会话密钥。客户端测试结果如图5.5，服务端测试结果如图5.6，双方能够基于传输密钥材料、群组分隔密钥和上下文信息派生出一致的传输密钥材料。

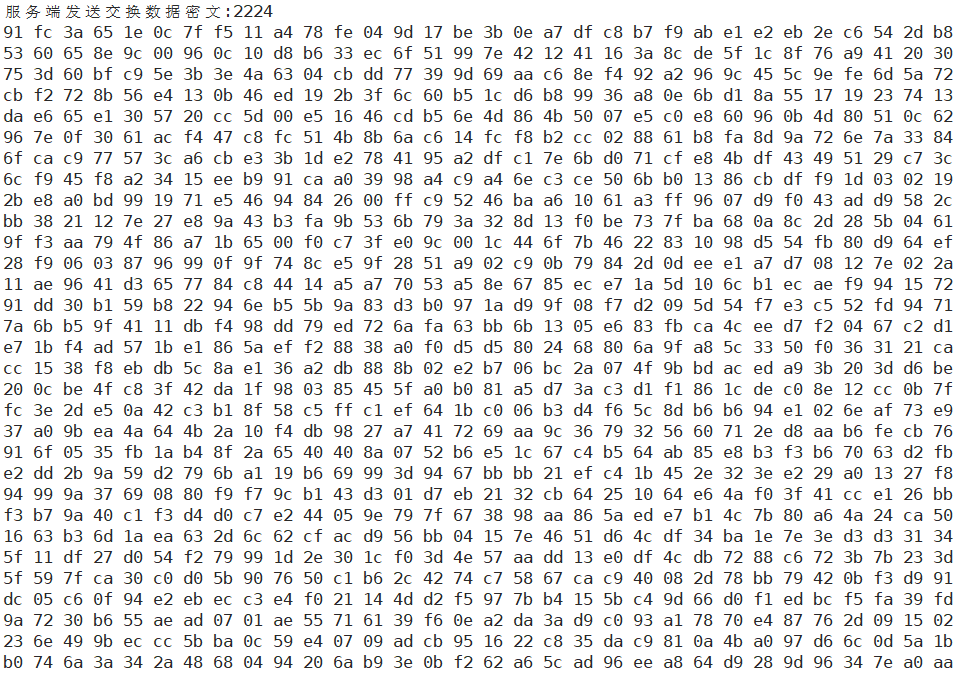


客户端密钥生成测试



服务端密钥生成测试

客户端与服务端基于加解密模块进行消息的传输。服务端测试结果如图5.7，客户端测试结果如图5.8。服务端使用会话密钥计算得到密文和MAC值发送给客户端，客户端使用会话密钥解密得到数据交换结果，并计算MAC值验证消息的完整性和真实性。



服务端发送消息密文



客户端解密得到交换数据明文

### 性能测试

由基于SGX的多方数据安全计算密码模块的设计与实现可知，三个模块的核心功能均通过调用可信接口实现，因此针对实际应用场景数据量，测试了每个可信接口执行一次所消耗的时间。测试结果如表5.1所示，其中，SM4算法的CTR、OFB、CFB模式的运行时间和CBC模式接近。在此基础上，可以分析每个密码模块调用可信接口的计算开销。通信双方基于身份认证模块进行一次密钥协商时，调用可信接口计算开销为23.223ms；基于密钥生成模块生成初始会话密钥时，调用可信接口计算开销为0.005ms，密钥更新时，调用可信接口计算开销为0.003ms。基于加解密模块进行一次消息的发送或接受时，调用可信接口计算开销为0.06ms。

由4.4.3小结测试结果可知，基于SGX保护算法运行过程会有一定的性能开销，这种性能开销包括运行上下文在可信区和非可信区之间的切换，以及边界值的复制等。因此，在设计可信接口时，本文通过将尽可能多的操作封装在一个可信接口中实现，从而减少因SGX技术特性带来的性能损耗。

可信接口执行时间（单位：ms）

|  |  |
| --- | --- |
| 可信接口 | 时间 |
| ecall\_sgx\_gen | 8.987 |
| ecall\_main\_key | 4.496 |
| ecall\_sgx\_sm2\_sign |  |
| ecall\_sgx\_sm2\_verify |  |
| ecall\_sgx\_sm4\_cbc | 0.040 |
| ecall\_hmac\_sm3 | 0.020 |
| ecall\_sgx\_single\_kdf | 0.003 |
| ecall\_sgx\_two\_kdf | 0.005 |

## 本章小结

本章对基于SGX的多方数据安全计算密码模块的设计与实现进行了介绍。首先介绍了密码模块的总体结构设计。其次介绍了数据安全交换传输协议的数据包格式定义以及每个模块的可信接口实现细节。最后介绍了测试环境、测试方案和测试结果。测试结果表明基于SGX的多方数据安全计算密码模块能够满足需求，具有可行性。

# 总结与展望

## 工作总结

在当今的数字化时代，数据量的爆炸性增长使得数据安全问题愈发凸显。数据泄露事件频发，给个人、企业乃至社会带来了严重的影响。多方的数据安全计算为提升数据价值提供了新的可能。然而，这些数据通常分散存储在各个数据采集者环境中，形成数据孤岛。多方数据的共享在开放信道通中进行，这使得数据传输过程极易遭受各种形式的攻击。此外，传统密码算法不能保障数据在明文状态下处理过程的安全性，在面对恶意软件和系统漏洞等复杂威胁时仍显不足。

因此本文设计了基于SGX的多方数据安全计算密码模块，以解决上述多方的数据安全计算所面临的问题。具体包括以下工作。

（1）针对多方数据在开放信道上安全共享的需求，本文设计并实现了数据安全交换传输协议。该协议包括认证密钥协商阶段、密钥生成阶段和数据传输阶段。在认证密钥协商阶段，通过设计一个两轮的密钥协商协议提高整体效率，通信实体基于该协议协商得到传输密钥材料；在密钥生成阶段，通过设计两种密钥派生函数实现初始会话密钥的生成和密钥的定期更新；在数据传输阶段，通信实体使用会话密钥进行安全通信。最后通过攻击者模型和安全性分析证明了协议的安全性。

（2）针对隐私数据计算过程的安全问题，本文设计并实现了基于SGX的安全增强国密算法，包括SM2数字签名算法、HMAC-SM3算法和SM4算法。方案结合国密算法设计原理和SGX应用设计框架，将国密算法应用逻辑划分为可信部分和非可信部分。本文使用Intel SGX技术构建可信安全区域Enclave，并将国密算法的核心步骤放入Enclave内执行，形成可信模块，对国密算法运行时数据进行隐私性保护。测试结果表明，三种算法均具有正确计算的能力，引入SGX机制所带来的性能开销在可接受范围内，基于SGX保护国密算法运行过程的设计，能够有效阻止关键信息被非法访问和篡改，为国密算法运行态提供隔离保护。

（3）本文基于上述方案，设计并实现了基于SGX的多方数据安全计算秘密模块，包括身份认证模块、密钥生成模块和加解密模块。方案利用SGX的内存隔离机制确保核心代码、数据和密钥的安全。在模块设计中，将密钥生成、加解密等过程放入可信区，传输协议消息头的填充以及数据的接收和发送在非可信区完成。方案通过将尽可能多的操作封装在一个可信接口中实现，从而减少因SGX技术特性带来的性能损耗。最后在征信场景下验证了三个模块的功能，测试结果表明方案满足需求。

## 未来工作展望

尽管本文已经完成了部分工作，但还存在缺点和不足。在后续的研究中，可以从以下方向进行优化。

（1）数据安全交换传输协议认证密钥协商阶段存在计算量较大的缺陷，虽然密钥协商协议通过减少通信轮次提高整体效率，但协商过程中双方需要进行多次点乘运算，所以本方案还需进一步完善和提高。

（2）虽然安全增强国密算法的性能测试结果显示性能损耗在可接受范围内，但持续探索更高效的算法实现和优化技术仍然是未来工作的重点。这包括对SGX执行环境的深入优化，以及国密算法在特定应用场景下的性能调优，以期达到更高的计算效率和响应速度。

（3）本文仅对基于SGX的身份认证模块、密钥生成模块、加解密模块进行了设计和实现。在多方的数据安全计算体系中，数据脱敏、置乱等模块也有待结合可信执行环境技术进一步开发。

参考文献

1. 李凤华,李晖,谢绒娜,等.多方的数据安全计算方法、装置、电子设备和存储介质[P].北京市:CN114866317B,2023-01-10.
2. 李凤华,李晖,牛犇,等.隐私计算的学术内涵与研究趋势[J].网络与信息安全学报,2022,8(06):1-8.
3. Jauernig P, Sadeghi A R, Stapf E. Trusted execution environments: properties, applications, and challenges[J]. IEEE Security & Privacy, 2020, 18(2): 56-60.
4. Zheng W, Wu Y, Wu X, et al. A survey of Intel SGX and its applications[J]. Frontiers of Computer Science, 2021, 15: 1-15.
5. Diffie W, Hellman M. New directions in cryptography[J]. IEEE Transactions on Information Theory, 1976, 22(6): 644-654.
6. Matsumoto T, Takashima Y, Imai H. On seeking smart public-key-distribution systems[J]. IEICE TRANSACTIONS (1976-1990), 1986, 69(2): 99-106.
7. Law L, Menezes A, Qu M, et al. An efficient protocol for authenticated key agreement[J]. Designs, Codes and Cryptography, 2003, 28: 119-134.
8. Bellare M, Pointcheval D, Rogaway P. Authenticated key exchange secure against dictionary attacks[C]//International conference on the theory and applications of cryptographic techniques. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000: 139-155.
9. Canetti R, Krawczyk H. Analysis of key-exchange protocols and their use for building secure channels[C]//International conference on the theory and applications of cryptographic techniques. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001: 453-474.
10. LaMacchia B, Lauter K, Mityagin A. Stronger security of authenticated key exchange[C]//International conference on provable security. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007: 1-16.
11. Seo D H, Sweeney P. Simple authenticated key agreement algorithm[J]. Electronics Letters, 1999, 35(13): 1073-1074.
12. Sun H. On the security of simple authenticated key agreement algorithm[C]//Proc. Management Theory Workshop 2000. 2000: 223-227.
13. Tseng Y M. Weakness in simple authenticated key agreement protocol[J]. Electronics Letters, 2000, 36(1): 1.
14. Shim K. Efficient ID-based authenticated key agreement protocol based on Weil pairing[J]. Electronics Letters, 2003, 39(8): 1.
15. Oh J B, Yoon E J, Yoo K Y. An efficient ID-based authenticated key agreement protocol with pairings[C]//Parallel and Distributed Processing and Applications: 5th International Symposium, ISPA 2007 Niagara Falls, Canada, August 29-31, 2007 Proceedings 5. Springer Berlin Heidelberg, 2007: 446-456.
16. Abi-Char P E, Mhamed A, El-Hassan B. A secure authenticated key agreement protocol for wireless security[C]//Third International Symposium on Information Assurance and Security. IEEE, 2007: 33-38.
17. Cao X, Kou W, Du X. A pairing-free identity-based authenticated key agreement protocol with minimal message exchanges[J]. Information Sciences, 2010, 180(15): 2895-2903.
18. Hou M, Xu Q. A one-round id-based authenticated key agreement protocol with enhanced security[C]//2011 2nd International Conference on Intelligent Control and Information Processing. IEEE, 2011, 1: 194-197.
19. Farash M S, Attari M A. An id-based key agreement protocol based on ECC among users of separate networks[C]//2012 9th International ISC Conference on Information Security and Cryptology. IEEE, 2012: 32-37.
20. 高志刚,冯登国.高效的标准模型下基于身份认证密钥协商协议[J].软件学报,2011,22(05):1031-1040.
21. 赵建杰,谷大武.eCK模型下可证明安全的双方认证密钥协商协议[J].计算机学报,2011,34(01):47-54.
22. 陈燕俐,杜英杰,杨庚.一种高效的基于属性的认证密钥协商协议[J].计算机科学,2014,41(04):150-154+177.
23. Krawczyk H. Cryptographic extraction and key derivation: The HKDF scheme[C]//Annual Cryptology Conference. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010: 631-648.
24. Suresha D, Karibasappa K. Enhancing data protection in cloud computing using key derivation based on cryptographic technique[C]//2021 5th International Conference on Computing Methodologies and Communication (ICCMC). IEEE, 2021: 291-299.
25. Gueron S, Lindell Y. Better bounds for block cipher modes of operation via nonce-based key derivation[C]//Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. 2017: 1019-1036.
26. Chuah C W, Dawson E, Simpson L. Key derivation function: the SCKDF scheme[C]//Security and Privacy Protection in Information Processing Systems: 28th IFIP TC 11 International Conference, SEC 2013, Auckland, New Zealand, July 8-10, 2013. Proceedings 28. Springer Berlin Heidelberg, 2013: 125-138.
27. Kodwani G, Arora S, Atrey P K. On Security of Key Derivation Functions in Password-based Cryptography[C]//2021 IEEE International Conference on Cyber Security and Resilience (CSR). IEEE, 2021: 109-114.
28. Nair V, Song D. Multi-factor key derivation function (MFKDF) for fast, flexible, secure, & practical key management[C]//Proceedings of the 32nd USENIX Conference on Security Symposium. 2023: 2097-2114.
29. Fisch B, Vinayagamurthy D, Boneh D, et al. Iron: functional encryption using Intel SGX[C]//Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. 2017: 765-782.
30. Priebe C, Vaswani K, Costa M. EnclaveDB: A secure database using SGX[C]//2018 IEEE Symposium on Security and Privacy (SP). IEEE, 2018: 264-278.
31. Karande V, Bauman E, Lin Z, et al. SGX-Log: Securing system logs with SGX[C]//Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security. 2017: 19-30.
32. Peters T, Lal R, Varadarajan S, et al. BASTION-SGX: Bluetooth and architectural support for trusted I/O on SGX[C]//Proceedings of the 7th International Workshop on Hardware and Architectural Support for Security and Privacy. 2018: 1-9.
33. Atamli-Reineh A, Martin A. Securing application with software partitioning: A case study using sgx[C]//International Conference on Security and Privacy in Communication Systems. Cham: Springer International Publishing, 2015: 605-621.
34. Lind J, Priebe C, Muthukumaran D, et al. Glamdring: Automatic application partitioning for intel SGX[C]//2017 USENIX Annual Technical Conference (USENIX ATC 17). 2017: 285-298.
35. Coughlin M, Keller E, Wustrow E. Trusted click: Overcoming security issues of NFV in the cloud[C]//Proceedings of the ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization. 2017: 31-36.
36. Duan H, Wang C, Yuan X, et al. LightBox: Full-stack protected stateful middlebox at lightning speed[C]//Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. 2019: 2351-2367.
37. Sun H, Xiao S. DNA-X: Dynamic network authentication using SGX[C]//Proceedings of the 2nd International Conference on Cryptography, Security and Privacy. 2018: 110-115.
38. 王冠,张倩倩.基于SGX的车联网身份认证方案研究[J].计算机技术与发展,2023,33(11):99-105.
39. 唐雨,张驰.一种基于Intel SGX的信息中心网络隐私保护方案[J].信息网络安全,2023,23(06):55-65.
40. Sev-Snp A M D. Strengthening VM isolation with integrity protection and more[J]. White Paper, January, 2020, 53: 1450-1465.
41. Pinto S, Santos N. Demystifying arm trustzone: A comprehensive survey[J]. ACM computing surveys (CSUR), 2019, 51(6): 1-36.
42. GM/T 0002-2012. SM4分组密码算法[S].2012.
43. GM/T 0003-2012. SM2椭圆曲线公钥密码算法[S].2012.
44. GM/T 0004-2012. SM3密码杂凑算法[S].2012.
45. Cook A, Vigano L. A Game Of Drones: Extending the Dolev-Yao Attacker Model With Movement[C]//2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW). IEEE, 2020: 280-292.
46. Grimes R, Yao J F. Development of a novel key-derivation function for speed, security and simplicity in picture-passphrase encryption[J]. Journal of Computing Sciences in Colleges, 2015, 31(2): 194-200.
47. Wu M, Li Z, Chen H, et al. Toward an SGX-Friendly Java Runtime[J]. IEEE Transactions on Computers, 2024, 73(01): 44-57.
48. Weisse O, Bertacco V, Austin T. Regaining lost cycles with HotCalls: A fast interface for SGX secure enclaves[J]. ACM SIGARCH Computer Architecture News, 2017, 45(2): 81-93.

作者简介

##### 攻读硕士学位期间的研究成果

**3.1参与科研项目及获奖**

1. 国家自然科学基金项目：泛在共享环境下多方隐私计算关键技术研究（62332018），2022.5-至今，初期负责业务管理系统开发，后期负责基于可信执行环境的多方数据安全计算密码模块开发
2. xxx.基于SGX的多方数据安全计算密码模块.2023.3-2023.11.已完成.负责模块设计与开发实现
3. 2022年中国国际“互联网+”大学生创新创业大赛省银奖
4. 2022校级一等奖学金、2023校级二等奖学金
5. 2022校级优秀研究生、2023校级优秀研究生