# 面向大规模区块链网络的高效编辑方案

高源芃[1], 冯 哲[1], 刘雪峰[1], 雷 静[1], 裴庆祺[2]

[1]西安电子科技大学 网络与信息安全学院 西安 中国 710126
[2]西安电子科技大学 陕西省区块链与安全计算重点实验室 西安 中国 710071

**摘要** 区块链因其不可篡改性成为去中心化系统的信任基础, 但这一特性也被滥用于存储暴力、色情及恐怖主义信息。变色龙哈希通过利用陷门信息在保持哈希值不变的情况下修改哈希内容, 被视为实现区块链可编辑性的关键。现有方案或依赖中心化的变色龙哈希方法, 使单一实体掌握编辑权限, 削弱区块链可信度; 或采用分布式变色龙哈希方法, 虽保持可信度, 但无法检测编辑过程中的恶意节点信息, 导致系统使用错误信息维持运行, 从而产生错误结果, 浪费计算资源。此外, 这些方案要么仍在区块链上包含被编辑信息, 与区块链编辑理念不符, 要么需全网节点参与安全计算生成陷门信息和掩码, 导致 $O(n^3)$ 的通信复杂度高, 难以适应大规模网络。为此, 本文提出了一种面向大规模网络环境的高效的区块链编辑方案。方案将区块的 Merkle 根生成过程由传统哈希函数替换为变色龙哈希函数, 实现无痕编辑; 通过代理节点聚合信息, 将系统的通信复杂度从 $O(n^3)$ 降至 $O(n^2)$, 并利用同态加密保护陷门和掩码数据, 实现编辑结果的加密聚合, 消除陷门信息和掩码对节点的依赖; 结合零知识证明和承诺机制, 支持恶意节点识别与筛选, 保障系统使用正确信息稳定运行。本文在四个城市部署 211 个区块链节点验证方案性能, 结果表明, 本方案效率优于现有方案, 且性能优势随节点规模扩大更加显著。此外, 方案在理想-现实模型下经过严格的安全性证明, 体现出卓越的安全性和实际应用潜力。

**关键词** 可编辑区块链; 分布式变色龙哈希
**中图法分类号** TP309.1 DOI 号 10.19363/J.cnki.cn10-1380/tn.2026.03.13

# An Efficient Editing Scheme for Large-Scale Blockchain Networks

GAO Yuanpeng[1], FENG Zhe[1], LIU Xuefeng[1], LEI Jing[1], PEI Qingqi[2]

[1] School of Cyber Engineering, Xidian University, Xi'an 710126, China
[2] Shaanxi Key Laboratory of Blockchain and Secure Computing, Xidian University, Xi'an 710126, China

**Abstract** Blockchain's immutability has become the cornerstone of trust in decentralized systems, but this feature has also been exploited to store illicit content such as violent, pornographic, and terrorist information. Chameleon hashing, which enables content modification while preserving the hash value through the use of trapdoor information, is recognized as a key technique for achieving blockchain editability. Existing approaches either rely on centralized chameleon hash schemes, granting editing authority to a single entity and undermining blockchain trust, or adopt distributed chameleon hash schemes that maintain trust but fail to detect malicious node behavior during editing. This results in systems operating on erroneous information, producing incorrect outcomes, and wasting computational resources. Moreover, these schemes either retain edited information on the blockchain, contradicting the principles of blockchain editing, or require all network nodes to participate in secure computations to generate trapdoor information and masking data, leading to high communication complexity ( $O(n^3)$ ) that limits scalability in large networks. To address these issues, this paper proposes an efficient blockchain editing scheme designed for large-scale network environments. The proposed scheme replaces traditional hash functions with chameleon hash functions in the Merkle root generation process to achieve seamless editing. By utilizing proxy nodes for information aggregation, the scheme reduces communication complexity from $O(n^3)$ to $O(n^2)$. Homomorphic encryption secures trapdoor and masking data, enabling encrypted aggregation and eliminating node dependency on these values. Additionally, the integration of zero-knowledge proofs and commitment mechanisms supports the identification and exclusion of malicious nodes, ensuring stable operation with accurate information. Experiments conducted on 211 blockchain nodes deployed across four cities demonstrate that the proposed scheme outperforms existing methods in efficiency, with its performance advantages becoming increasingly pronounced as network size grows. Fur-

**通讯作者:** 裴庆祺, 博士研究生, 教授, Email: qqpei@mail.xidian.edu.cn。
本课题得到国家重点研发计划(No. 2021YFB2700600)、国家自然科学基金(No. 62272365)、国家自然科学基金重点项目(No. 62132013)、陕西省重点研发计划(No. S2024-YF-YBGY-1540)、中央军委科技委基础加强计划项目(No. 2023-JCJQ-JJ-0772-01)资助。
收稿日期: 2024-09-13; 修改日期: 2024-11-21; 定稿日期: 2026-01-26

thermore, rigorous security proofs under the ideal-real model validate the scheme's strong security properties, showcasing its potential for practical application in large-scale blockchain networks.

**Key words** redactable blockchain; distributed chameleon hash

# 1 引言

区块链是一种分布式数据库, 通过共识机制确保多个参与方存储一致的副本, 并利用哈希链式结构保证数据的不可篡改性, 为去中心化系统中的信任构建提供了坚实基础。区块链技术应用广泛, 涵盖金融[1-2]、供应链管理[3]、医疗健康[4]等多个领域。其去中心化的信任机制、透明性和数据安全性, 推动了这些领域的数据共享与协作模式的变革, 显著提升了系统效率, 降低了信任成本。

尽管区块链的不可篡改性保障了数据存储的安全性, 但也为监管与治理带来了挑战。恶意用户可能利用区块链永久存储儿童色情、非法言论等有害数据, 将严重损害区块链的公信力, 削弱用户对整个生态系统的信任[5-6]。与此同时, 智能合约的引入扩展了区块链的应用场景。然而, 受制于区块链的不可篡改性, 智能合约一旦发布便无法修改, 若存在漏洞或需升级, 只能发布新版本, 而无法直接修改现有合约。这不仅浪费存储资源, 还可能引发复杂的迁移和兼容性问题。

因此, 可编辑性成为区块链实际部署中的关键需求之一。具体而言, 是在保持区块链链式结构的前提下, 允许对链上内容进行修改。变色龙哈希[7]是一种重要的解决路径。在变色龙哈希机制中, 存在一个管理中心负责生成系统参数和陷门信息。用户可以对信息进行哈希计算, 生成全新的标签。在保持哈希值不变的前提下, 管理中心使用其陷门信息对数据进行修改。然而, 这种方法依赖于可信中心, 违背了区块链的分布式理念。

为了在维持区块链公信力的同时实现可编辑性, Ateniese 等人提出了使用秘密共享机制, 将变色龙哈希的陷门信息分片分发给区块链的所有全节点。当需要修改链上数据时, 全节点将各自持有的陷门碎片作为私有输入, 进行安全计算以完成数据修改[8-9]。然而, 实用化的区块链编辑方案需要满足以下要求。

(1) 编辑无痕性: 编辑后的数据不应包含任何编辑前的数据信息, 且任意实体无法判断某一区块是否被编辑, 这对于治理区块链上的非法舆论内容至关重要。

(2) 恶意行为标识: 编辑协议由所有区块链全节点共同参与, 要求能标识协议中任何节点的恶意信息, 以防止协议陷入“错误-重启-错误”的死锁循环。

(3) 高效性: 编辑过程涉及所有节点联合执行安全计算, 节点数量的增加会显著降低编辑性能。因此, 需设计一种高效的编辑方案, 以提升编辑过程的计算效率。

现有工作要么采用中心式的变色龙哈希编辑, 使得单一实体拥有区块编辑权限[10-18], 虽然部分工作对该实体的恶意行为进行了追责, 但是降低了区块链的可信度。而另一种采用分布式变色龙哈希修改区块则在不降低区块链可信度的前提下实现区块编辑, 但是此类方案无法对编辑过程中节点的恶意行为进行检测, 只能判断最终编辑结果的正确性, 无法从错误中恢复, 导致浪费诚实节点的通信和计算资源。此外, 这些方案要么是有痕编辑[9], 这与区块链编辑理念相悖, 要么需要在编辑前让每个节点运行可验证秘密共享, 生成与其绑定的陷门信息掩码[8], 该过程需要每个节点向其他节点发送 $O(n^2)$ 数据量, 从而导致 $O(n^3)$ 的通信资源消耗。因此上述方案在大规模网络环境中不实用。

## 1.1 本文贡献

为了解决上面的问题, 本文提出了一个面向大规模网络环境的高效区块链编辑方案。本文的主要贡献如下所示。

(1) 无痕编辑: 本方案可实现区块编辑的无痕性, 即除区块链编辑节点, 其他任意实体无法判断此区块链是否被编辑, 且编辑后的区块链无编辑前信息。

(2) 恶意节点标识: 在本方案中, 每个节点可以标识任意节点的错误信息, 通过筛选正确的节点信息, 确保系统的正常运行。

(3) 高效性: 本方案使用代理节点接收其他节点信息, 并进行信息聚合, 从而降低节点的通信量, 达到降低通信复杂度的目的。为了防止不可信代理节点作恶, 本方案使用同态加密对陷门信息、掩码进行加密, 在编辑阶段使用同态操作, 完成编辑结果密文的聚合, 消除了掩码对于节点的依赖。

(4) 为了验证方案的效率, 本文在四个城市部署了 211 个区块链节点进行实验。结果表明, 本文方案在效率上优于现有的区块编辑方案, 且随着节点数量的增加, 性能优势更加明显。同时, 本文在理想-现实模型中对方案进行了严格的安全性证明。

### 1.2 相关工作

现有研究中的可编辑区块链方案主要分为基于平行链和基于变色龙哈希算法两种类型。基于平行链的方案在现有区块链上附加一条区块链或哈希链, 作为修正链存储编辑后的数据, 区块编辑通过节点间的投票决策来完成。而基于变色龙哈希函数的方案, 通过将传统抗碰撞哈希函数替换为具有陷门密钥的变色龙哈希函数, 赋予区块链编辑能力。

基于平行链的可编辑区块链方案主要分为双区块链模式[19]和双哈希链模式。双区块链模式通过侧链存储编辑后的区块数据, 但这一方式偏离了区块链编辑的初衷。双哈希链模式则在相邻区块间引入双向哈希链接, 通过修改哈希软链接, 完成新区块的接入, 但每个区块仅能被编辑一次。上述方案虽然实现了数据的可编辑性, 但在灵活性和实用性上存在局限。

基于变色龙哈希算法的区块链编辑方案最早由Ateniese 等人[8]提出。该方案通过将传统的抗碰撞哈希函数替换为变色龙哈希函数, 允许持有陷门密钥的实体进行区块编辑。方案还提出了全节点共同持有陷门密钥的模式, 要求收集足够数量的陷门密钥片段才能进行区块编辑, 但未能解决节点恶意行为的检测问题。而且在编辑阶段还需要生成大量的密钥掩码, 当节点发生变更时, 预生成的密钥掩码将随之失效, 消耗大量的计算资源。后续研究主要解决陷门密钥分配问题。Derler 等人[11]将变色龙哈希函数与属性加密结合, 授权特定实体进行编辑, 但该方案依赖授权中心, 且无法对恶意编辑行为追责。Tian 等人[14]进一步优化该方案, 通过将区块编辑行为与编辑者绑定, 解决了基于策略的变色龙哈希函数缺少问责机制的问题, 引入黑箱责任制来追踪交易修改者的责任, 防止恶意修改行为的否认。Zhang 等人[18]考虑到节点动态变更的问题, 设计了动态分散的基于策略的变色龙哈希函数。当节点变更或识别出恶意节点时, 该方案可主动更新陷门密钥, 从而保障区块链的前向和后向安全性, 实现更为安全的可编辑区块链方案。

Tian 等人[20]针对编辑权限的时效性和密钥泄露风险, 设计了可撤销的基于策略的变色龙哈希函数, 允许撤销过期或暴露密钥编辑者的权限。Xu 等人[17]通过限制每个区块的编辑次数和引入保证金机制, 从另一个角度减少了编辑权限的滥用, 确保区块编辑的合理性。陈越等人[21]结合陷门撤销和编辑次数限制, 融合了上述两项工作的优势, 提出了支持陷门撤销和编辑次数限制的可编辑区块链方案, 从多方面防范恶意编辑行为。Zhang 等人[18]将陷门撤销和节点动态更新相结合, 利用主动秘密共享和数字签名技术, 设计了支持更新和可追溯的动态变色龙哈希函数, 并在此基础上构建了可编辑区块链方案。

Ma 等人[12]针对基于策略的变色龙哈希函数依赖中心化属性颁发机构的问题, 提出了去中心化的设计, 通过分布式构造实现, 解决了权限过度集中且与区块链去中心化理念相违背的缺陷。Jia 等人[9]也关注到中心化陷门密钥导致编辑权限集中的问题, 采用分布式密钥生成(DKG)协议为每个节点分发私钥分片, 仅当超过门限值的节点联合使用陷门密钥时, 才能计算出哈希碰撞值, 从而完成区块编辑。这种机制有效防止了编辑权限滥用, 提升了区块链系统的信任度, 但是该方案无法在编辑阶段检测任意节点的恶意信息, 而且该方案着重有痕编辑, 这与区块链编辑理念相违背。为解决区块编辑后的验证问题, Shen 等人[13]引入通用累加器和最大序列号原则, 使历史区块失效, 仅允许最新版本通过验证, 确保区块链节点存储的区块信息保持最新。

## 2 基础知识

### 2.1 变色龙哈希算法

变色龙哈希函数[7]是一种可以人为制造碰撞的哈希函数, 通过公钥密码体系, 使得陷门密钥掌握者可以轻易地制造哈希碰撞, 而对于无陷门密钥的用户, 变色龙哈希函数仍具有抗碰撞性。变色龙哈希函数由五个概率多项式时间算法组成, 定义如下。

系统参数生成:该算法输入安全参数 $\lambda$, 输出系统参数 $\mathrm{pp}=(g,q,G,H)$, 其中 $g$ 为 $q$ 阶乘法循环群 $G$ 的生成元, 而 $H$ 为普通的抗碰撞哈希函数。

陷门密钥生成:该算法输入系统参数 pp, 随机选择 $\mathrm{sk}\leftarrow\mathbb{Z}_q$, 计算 $\mathrm{pk}=g^{\mathrm{sk}}$, 输出陷门公钥 pk, 陷门私钥 sk。

哈希: 该算法输入公钥 pk, 信息 $m$, 选择随机数 $r_1,r_2\leftarrow\mathbb{Z}_q$, 计算 $R=g^{r_1}$, $h=R\cdot(\mathrm{pk}^{-H(m\|R)}\cdot g^{-r_2})$, 输出哈希值 $h$ 和随机数 $r=(R,r_2)$。

验证: 该算法输入公钥 pk, 信息 $m$, 随机数 $r$, 哈希值 $h$, 输出验证结果。

编辑: 该算法输入私钥 sk, 新信息 $m'$, 哈希值 $h$, 验证信息 $r$, 选择随机数 $k\leftarrow\mathbb{Z}_q$,计算 $R'=h\cdot g^k$. 和 $r_2'=k-H(m'\|R')\bullet\mathrm{sk}$。输出新随机数 $r'=(R,r_2')$。新随机数也被称为哈希碰撞值。

### 2.2 CL 同态加密

CL 同态加密是一种基于虚二次域类群的加法同态加密方案[22]。该方案包含一种可解离散对数问题的算法 Solve。通过此算法可以完成信息的解密。同时, CL 同态加密算法利用素数阶 $q$ 限制明文空间范围, 避免了 Paillier 加密算法[23]中昂贵的范围证明, 并且其空间范围远大于 ElGamal 加密算法[24]中的明文空间。CL 同态加密算法主要包括以下几个核心算法。

密钥生成: 该算法输入安全参数 $\lambda$, 选择随机数 $x \leftarrow [1, 2^{\lambda-2}]$, 计算 $\mathrm{ek} = g_q^x$, 其中 $g_q$ 为未知阶的乘法循环群 $\mathcal{G}^q$ 的生成元, 输出加密密钥 ek 和解密密钥 $\mathrm{dk} = x$。

加密: 该算法输入信息 $m$, 加密密钥 pk, 选择随机数 $r \leftarrow [1, 2^{\lambda-2}]$, 输出密文 $c = (g_q^r, f^m \mathrm{ek}^r)$, 其中 $f$ 为离散对数可解的 $q$ 阶乘法循环群 $\mathcal{F}$ 的生成元。

解密: 该算法输入密文 $c = (c_1, c_2)$, 输出离散对数解 $m = \mathrm{Solve}(c_2^{\mathrm{dk}} / c_1)$.

CL 加密支持的同态操作如下所示。

密文同态加法: 该算法输入两个使用相同加密密钥分别对 $(m, m')$ 进行加密的密文 $E(m, r)$ 和 $E(m', r')$, 则 $m + m'$ 的密文表示为

$$E(m + m', r + r') = E(m, r) \bullet E(m', r')$$

明文同态乘法: 该算法一个密文 $E(m, r)$ 和一个明文 $a \in \mathbb{Z}_q$, 则 $a \bullet m$ 的密文表示为

$$E(a \bullet m, a \bullet r) = E(m, r)^a$$

### 2.3 零知识证明

假设 $x$ 为一个论断, $w$ 为对应的秘密信息。零知识证明(Zero-Knowledge Proof, ZKP)可以使得证明者能够说服验证者相信 $x$ 是正确的, 同时不泄露 $w$ 的额外信息[25]。本方案使用 Sigma 协议来构造零知识证明, 这是因为 Sigma 协议生成证明的高效性以及具备结构化的交互方式。并且, Sigma 协议还可以通过 Fiat-Shamir 启发式[26]使用哈希函数生成协议中挑战, 从而将协议转化为非交互式零知识证明, 具体包含两个算法。

$\mathrm{Prove}_{\mathcal{R}}(x, w) \to \pi$: 该算法输入 $(x, w)$, 输出一个证明 $\pi$, 此证明可以表明 $(x, w)$ 符合某种关系 $\mathcal{R}$。

$\mathrm{Verify}_{\mathcal{R}}(x, \pi) \to 0/1$: 该算法输入论断 $x$ 与证明 $\pi$, 输出 0 或 1, 来表明 $(x, w)$ 是否符合某种关系 $\mathcal{R}$。

下面介绍本方案使用零知识证明的三种关系。

(1) CL 加密证明: 该关系证明了对于一个密文 $c$ 是通过加密密钥 ek 对信息 $m$ 加密得到, 此信息被承诺在群元素 $Q$ 中:

$$R_{\text{Enc-DL}} = \{((c, \mathrm{ek}, Q), (m)) : Q = g^m \wedge c = \mathrm{Enc}(m, \mathrm{pk})\}$$

(2) CL 解密证明: 该关系证明了对于密文 $c = (c_1, c_2)$ 是被解密密钥 dk 正确解密的, 其中 $c_2' = c_2 / f^m$。

$$R_{\text{Dec}} = \{(\mathrm{ek}, (c_1, c_2')), \mathrm{dk} : \mathrm{dk} \in [0, S] \wedge c_2' = c_1^{\mathrm{dk}} \wedge \mathrm{ek} = g_q^{\mathrm{dk}}\}$$

### 2.4 可验证秘密共享

秘密共享[27]指在多方之间进行一个秘密数据的分享时, 可以在不泄露秘密数据本身的情况下, 将秘密数据拆分成多个子秘密, 交由多方共同保管。只有收集超过一定数量的子秘密, 才可以恢复完整的秘密。为了让协议参与者可以公开验证收到的子秘密是否正确, Pedersen[28]提出实用的可验证秘密共享, 可以通过离散对数完成对多项式系数的绑定, 从而让各参与者对子秘密进行验证。此协议主要包含 3 个算法。

$\mathrm{Share}(s) \to (\{s_i\}_{i \in [1,n]}, \{F_k\}_{k \in [0,t]})$: 该算法输入 $s$, 选择随机多项式 $f(x) = a_0 + a_1 x + \cdots + a_t x^t$, 其中 $a_0 = s$, 输出多项式值 $\{s_i = f(i)\}_{i \in [1,n]}$ 和多项式系数的离散对数 $\{F_k = g^{a_k}\}_{k \in [0,t]}$。

$\mathrm{Verify}(s_i, \{F_k\}_{k \in [0,t]}) \to 0/1$: 该算法输入子秘密 $s_i$, 验证 $g^{s_i} \stackrel{?}{=} \prod_{k=0}^{t} (F_k)^{i^k}$。

$\mathrm{Recover}(\{s_i\}_{i \in [1,t+1]}) \to s$: 该算法输入 $t+1$ 个子秘密, 计算 $s = \sum_{i=1}^{t+1} s_i L_i(x)$, 其中 $L_i(x)$ 是拉格朗日插值系数, 输出秘密 $s$。

## 3 问题定义

### 3.1 系统模型

本文提出了一种支持大规模区块链节点联合持有密钥的高效可编辑区块链方案。系统由三类实体组成: 区块链节点、代理节点和区块链用户, 如图 1 所示。

区块链节点: 在本方案中, 区块链节点负责区块编辑和区块链系统的正常运行。超过门限值数量的区块链节点可以联合使用陷门信息对区块内容进行编辑。

区块链用户: 区块链用户是区块链系统的使用者, 可以将原区块信息、新区块信息以及编辑原因打包为编辑请求并发送给全体区块链节点, 以完成编辑操作。

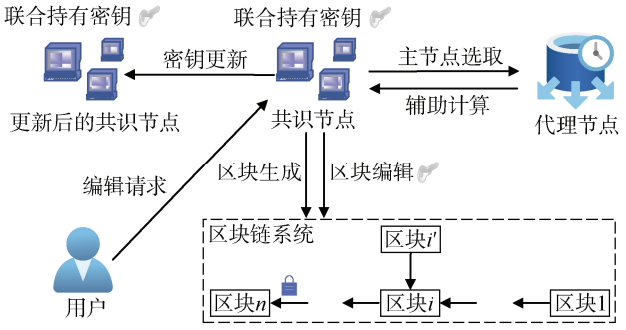


**图 1　系统模型**

**Figure 1　System Model**

代理节点: 代理节点为不可信实体, 负责私钥分片信息的验证、聚合与转发, 降低系统通信复杂度。代理节点由 PBFT 共识算法确定的主节点指定。

### 3.2　假设

(1) 威胁模型假设: 假设系统中存在 $n$ 个区块链节点与 1 个代理节点, 其中最多有 $t$ 个区块链节点与代理节点被敌手攻破, 并执行破坏系统的行为。

(2) 广播信道假设: 假设存在公开可验证的广播信道, 通过该广播信道, 所有节点接收到相同的信息, 并确定信息来源。

### 3.3　设计目标

针对现有可编辑区块链的方案漏洞和上述的威胁模型, 本文提出了大规模环境下高效的可编辑区块链方案。

(1) 无痕编辑: 支持对区块链上恶意数据或交易进行删除或替换, 且用户只能验证区块的正确性, 而无法判断其是否已被编辑。

(2) 恶意信息识别: 支持对协议运行中的恶意信息进行标识, 具体为代理节点可以检测任意区块链节点的恶意信息, 反之亦然。

(3) 高效性: 降低生成陷门信息及其掩码的通信复杂度, 消除掩码与节点的绑定。

### 3.4　安全定义

本文将采用理想-现实模型[29], 通过在理想世界中存在的可信第三方来验证现实环境中协议的执行来定义方案的安全性。

**理想模型**: 假设存在一个可信第三方(Trusted Third Party, TTP)。参与方将隐私输入通过安全信道发送给 TTP。TTP 根据输入计算函数, 并将计算结果返回给各参与方。对于协议 $\mathcal{F}$ 定义理想模型, 协议 $\mathcal{F}$ 的输入来自 $n$ 个区块链节点 $\{P_k\}_{k=1}^n$ 以及代理节点 $\mathcal{P}$, 值得注意的是代理节点并不提供输入但为了保证涉及实体的完整性, 理想模型中将包含代理节点。

作为应用于区块链系统上的协议, 协议参与方与区块链中区块链节点概念对等, 因此协议可容忍的被恶意敌手 $\mathcal{S}$ 控制的腐败节点数量应与共识算法的容错量相等。以 PBFT 算法为例, $t \le n/3$。为表述清晰, 本文假设一组腐败的区块链节点和一个腐败的代理节点为 $\mathcal{I}=\{P_1,\cdots,P_t,\mathcal{P}\}$。理想协议执行方式如下。

**输入**: 腐败节点 $P_i \in \mathcal{I}$ 的输入为 $x_i$, 诚实节点 $P_j$ 的输入为 $x_j$, 敌手 $\mathcal{S}$ 的辅助输入为 $z$。

**发送输入至 TTP**: 每个诚实节点 $P_j$ 发送输入 $x_j$ 至 TTP, 腐败节点 $P_i \in \mathcal{I}$ 发送输入 $x_i$ 以及与 $x_i$ 相同长度的 $x_i'$ 或中断信息 $\text{abort}_i$ 至 TTP。

**中断**: 在收到来自腐败节点 $P_i$ 的 $\text{abort}_i$ 后, TTP 向所有诚实节点 $P_j$ 发送 $\text{abort}_i$ 并终止协议, 否则继续执行协议。

**返回输出**: TTP 利用输入计算 $\mathcal{F}$, 并发送 $\mathcal{F}$ 至腐败节点 $P_i \in \mathcal{I}$。

**继续或终止协议**: 每个腐败节点 $P_i \in \mathcal{I}$ 发送 $\text{abort}_i$ 或 $\text{continue}_i$ 至 TTP。如果 TTP 收到 $\text{continue}_i$ 则将输出 $y_k$ 发送给节点 $\{P_k\}_{k=1}^n$。当 TTP 收到 $P_i$ 发送的 $\text{abort}_i$ 后, 将 $\text{abort}_i$ 广播给所有参与节点。

**输出**: 诚实节点输出从 TTP 接收到的信息, 腐败节点不输出信息。敌手 $\mathcal{S}$ 输出任意 PPT 算法, 输入为辅助输入 $z$ 并从 TTP 获得输出。作为输出值发送给节点 $\{P_k\}_{k=1}^n, k \ne j$。

设 $\mathcal{S}$ 为控制 $t$ 个腐败节点 $P_i$ 的非均匀 PPT 敌手, 设 $\text{IDEAL}_{\mathcal{F},\mathcal{S}(z),i}(x_1,x_2,\cdots,x_n)$ 表示节点输入 $\{x_1,x_2,\cdots,x_n\}$, 敌手 $\mathcal{S}$ 输入 $z$ 后诚实节点和 $\mathcal{S}$ 在执行理想协议 $\mathcal{F}$ 后的输出。

**真实模型**: 协议 $\pi$ 涉及 $n$ 个区块链节点和一个代理节点间的交互。在网络中执行的每轮协议攻击者 $\mathcal{A}$ 可以腐化 $t$ 个区块链节点和1个代理节点。$\mathcal{A}$ 可以在每轮协议中收到其他节点发送的信息后立刻发出信息。

设 $\mathcal{A}$ 为控制 $t$ 个腐败节点 $P_i$ 的非均匀 PPT 敌手, 设 $\text{REAL}_{\pi,\mathcal{A}(z),i}(x_1,x_2,\cdots,x_n)$ 表示节点输入 $\{x_1,x_2,\cdots,x_n\}$, 敌手 $\mathcal{A}$ 输入 $z$ 后诚实节点和 $\mathcal{S}$ 在执行真实协议 $\pi$ 后的输出。

**定义**: 设 $\mathcal{F}$ 是一个 $n$ 方函数, $\pi$ 是一个 $n$ 方计算 $\mathcal{F}$ 的协议。若协议 $\pi$ 在恶意敌手存在的情况下安全地使用 $\text{abort}_i$ 计算 $\mathcal{F}$, 则对于真实世界中非均匀

PPT 敌手$(\hat{c},\pi_{\hat{c}})$，总存在理想世界的非均匀 PPT 敌手$\mathcal{S}$，使得

$$\mathrm{IDEAL}_{F,S(z),i}(x_1,...,x_n)\equiv\mathrm{REAL}_{\pi,A(z),i}(x_1,...,x_n)$$

## 4 方案设计与安全分析

在现有全节点编辑方案中[8]，每次区块编辑均需使用新随机掩码来隐藏陷门信息。为确保所有节点对编辑结果的一致性，系统要求各节点运行可验证秘密共享协议生成掩码碎片，以保证掩码碎片与私钥信息碎片的接口一致性。然而，该协议需要各节点向其他每个节点发送 $t$ 个多项式系数承诺与一个多项式值(其中 $t$ 为多项式阶，与区块链节点数量 $n$ 成正比)。每个节点发送数据总量为 $O(n^2)$，导致系统通信复杂度为 $O(n^3)$。同时，系统在编辑过程中无法识别恶意节点的信息，因而无法使用正确信息维持系统的正常运行，导致编辑失败，并造成诚实节点的通信和计算资源大量消耗。此外，因为随机掩码依赖于节点数量及其顺序，当节点发生变更时，随机掩码也需要重新生成，否则会导致区块编辑失败。

为解决各节点运行可验证秘密贡献带来的通信复杂度过高的问题，本文通过引入代理节点，使得各节点只需将 $t$ 个多项式和 $n$ 个多项式值发送给代理节点，而无需重复向其他节点发送。代理节点对接收到的多项式系数承诺和多项式值分别进行聚合，然后将聚合后的多项式值与多项式系数承诺发送回各节点，使得每个节点发送的数据量从 $O(n^2)$ 降低至 $O(n)$，从而将整个协议的通信复杂度从 $O(n^3)$ 降低至 $O(n^2)$。为了防止代理节点泄露信息并确保其聚合功能不受影响，节点对多项式值进行同态加密，并提供零知识证明以保证多项式值的正确性，从而防止节点发送错误信息。同时，当各节点收到代理节点信息后，可对其解密，并通过比对多项式系数承诺验证多项式值是否正确，以判断代理节点是否存在恶意行为。

为解决陷门信息掩码与节点间的耦合问题，本方案通过同态加密算法加密陷门信息和掩码，使得每个节点只需选择陷门信息碎片和掩码碎片，并使用联合公钥对其进行加密，然后将加密数据广播至全网。各节点在本地分别聚合所有的陷门信息碎片和掩码碎片密文，从而得到完整陷门信息密文和掩码密文。在编辑阶段，通过对陷门信息密文和掩码密文的同态操作完成编辑结果的密文计算，然后各节点联合使用解密密钥对编辑结果密文进行解密，即可得到编辑结果。该方法仅要求各节点联合持有解密私钥，而无需各节点分别持有大量掩码碎片，从而降低了由于节点变更带来的掩码碎片重复计算所需的通信和计算开销。

此外，为实现无痕编辑，本文对区块生成方式进行了调整：在生成 Merkle 树根时，用变色龙哈希替代传统哈希，并在区块头中增加一个随机数字段，存储变色龙哈希的验证辅助信息。该字段不参与区块头哈希计算。编辑区块时，各节点按照常规方式构建新区块，唯一不同的是在生成 Merkle 根时使用变色龙哈希的陷门信息计算出验证辅助信息。通过调整验证辅助信息，保证 Merkle 根不变。这样，外部实体只能通过变色龙哈希验证区块的合法性，无法判断是否进行过编辑。

### 4.1 系统初始化

系统初始化是一个多方执行协议，由全体区块链节点和代理节点共同执行，主要包括门限加密密钥生成算法、变色龙哈希密钥生成算法。其中，门限加密密钥生成算法用于 CL 加密门限加密的公私钥对。变色龙哈希密钥生成算法用于生成变色龙哈希公私钥对，其中变色龙哈希私钥以密文形式公开存储。而且在本方案中默认各节点已经具备其他节点的加密密钥。

#### 4.1.1 分布式 CL 密钥生成

在本方案中，分布式密钥不再用于椭圆曲线群上的去中心化变色龙哈希计算，而是应用于 CL 门限加密。CL 加密运行于虚二次域类群上，这是一种未知阶群，因此椭圆曲线群上的传统秘密共享方案无法适用，需要采用整数秘密共享以适应未知阶群的运算特性。由于该群阶未知，各参与方无法对计算结果进行取模操作，所有运算必须在整数范围内进行。在秘密恢复阶段，传统的有限域秘密共享方案通过模逆运算计算拉格朗日系数。然而，在未知阶群上，由于阶数未知，模逆运算不可行。为解决这一问题，通常的做法是将拉格朗日系数与 $\Delta=n!$ 相乘，其中 $n$ 表示参与方的数量，从而消除分母并避免模逆运算。通过使用整数秘密共享方案，门限加密密钥生成协议在输入参与方集合 $P=P_1,\cdots,P_n$、门限值 $t$ 和安全参数 $\lambda$ 的情况下，输出虚二次域类群上的公钥 pk 和各参与方的私钥 $\mathrm{sk}_{i,i\in[1,n]}$。这种设计有效支持了在未知阶群上的密钥分发和后续的门限加密操作。下面是具体的方案设计。

区块链节点 $P_i$：选择秘密值 $\alpha\leftarrow\mathbb{Z}_p$，选择随机数 $r_1,\ldots,r_t$。构造多项式 $f_i(X)=\tilde{\alpha}+r_1\cdot x+\cdots+r_t\cdot x^t$，

计算多项式值 $f_i(1),\cdots,f_i(n)$ 和系数承诺 $C_{i,0},\cdots,C_{i,t}$，其中 $f_i(j)$ 为节点 $P_i$ 发送给节点 $P_j$ 的多项式值。广播 $C_{i,0}$ 到全体区块链节点。为了防止代理节点获取秘密碎片信息，区块链节点还需计算秘密碎片承诺 $S_{i,j}=g_q^{f_i(j)}$ 和秘密碎片密文 $c_{i,j}=\mathrm{Enc}(f_i(j),\mathrm{ek}_j)$。但是为了保证每个节点的行为是正确的，可以被其他节点进行纠错，区块链节点还需要对秘密碎片进行零知识证明 $\psi_i=\mathrm{Prove}(c_{i,j},S_{i,j},\mathrm{ek}_j)$，从而保证代理节点可以对区块链节点的秘密碎片密文进行验证。最后发送 $\{c_{i,j},\psi_i\}_{j\in[1,n]}$ 和 $\{C_{i,k}\}_{k\in[0,t]}$ 至代理节点。

代理节点：首先通过多项式系数承诺计算各节点的多项式值承诺 $S_{i,j}=\prod_{k=0}^{\mathrm{t}}C_{i,k}^{j^k}$。然后验证各节点的多项式值与密文是否正确 $0/1\leftarrow\mathrm{Verify}(c_{i,j},S_{i,j},\mathrm{ek}_j)$。如果验证失败，则对该节点进行投诉，从而对所有在第一步骤中作恶的节点进行标识。将验证通过的节点视为诚实节点，构建集合 $Q$，并将此集合广播至全部区块链节点。利用同态加密的性质，完成节点解密密钥密文碎片的拼接和多项式系数承诺的聚合：$C_k=\prod_{i\in Q}C_{i,k}$，$c_j=\prod_{i\in Q}c_{i,j}$。最后发送 $(\{C_k\}_{k\in Q},c_j)$ 到对应的区块链节点 $P_j$。

区块链节点 $P_i$：聚合有效集合 $Q$ 的多项式首项系数承诺 $C_0=\prod_{j\in Q}C_{j,0}$ 并与代理节点发送的多项式首项系数承诺对比，从而防止代理节点恶意删除部分节点的多项式和密文数据。然后解密多项式值密文 $m_i=\mathrm{Dec}(c_i,\mathrm{dk}_i)$，并通过多项式系数承诺验证该信息是否正确 $g_q^{m_i}\stackrel{?}{=}\prod_{k\in[0,t]}C_k^{i^k}$。如果上面验证结果存在问题，则广播一个对代理节点的投诉。超过 $t$ 个区块链节点广播了对代理节点的投诉说明代理节点为恶意节点，将恶意代理节点移除区块链节点集合 $P$，重新选举新的代理节点并执行后续密钥生成步骤。最后输出加密公钥和解密密钥碎片 $(\mathrm{pk}=C_0,\mathrm{sk}_i=m_i)$。

#### 4.1.2 变色龙哈希密钥生成

此步骤，区块链节点并行选择随机数作为变色龙哈希密钥陷门信息，并使用加密公钥进行加密，最后聚合，即可得到变色龙哈希密钥的密文，并存储于链上。

区块链节点 $P_i$：选择随机数 $x_i\leftarrow\mathbb{Z}_p$，计算陷门公钥分片 $\mathrm{tp}_i=g^{x_i}$，以及陷门私钥碎片的密文 $c_{ts_i}=\mathrm{Enc}(x_i,\mathrm{pk})$。并且为了证明此陷门私钥碎片是正确加密的，还需要提供零知识证明 $\varphi_i=\mathrm{Prove}(c_{ts_i},\mathrm{pk},\mathrm{tp}_i)$。最后广播 $(\mathrm{tp}_i,c_{ts_i},\varphi_i)$ 给所有诚实节点集合中的节点。

区块链节点 $P_i$：首先验证各节点的陷门私钥碎片密文，通过验证零知识证明 $0/1\leftarrow\mathrm{Verify}(c_{ts_i},\mathrm{pk}_i,\varphi_i)$ 保证各节点的信息正确性。如果验证通过，则聚合陷门公钥分片和陷门私钥密文：$\mathrm{tp}=\prod_{i\in Q}\mathrm{tp}_i$，$c_{ts}=\prod_{i\in Q}c_{ts_i}$。

### 4.2 区块生成

传统区块链使用抗碰撞哈希函数，导致区块内数据无法修改，难以满足对可编辑性的需求。为解决此问题，本文对区块结构进行了最小化调整：将区块的 Merkle 根生成方式由传统哈希函数替换为变色龙哈希函数，并在区块头部增加了随机数字段。编辑区块时，通过变色龙哈希函数的陷门密钥重新计算修改后的 Merkle 根并更新随机数，无需更改区块整体哈希值。由于随机数字段不计入区块哈希计算，编辑前后区块的整体哈希值得以保持一致。具体实现过程中，由区块链节点首先构建 Merkle 树，并在生成 Merkle 根时由传统的抗碰撞哈希函数改用变色龙哈希函数，生成对应的哈希值和随机数。然后，将生成的哈希值填入原区块的 Merkle 根位置，并将 Merkle 根与随机数一并放入区块头的指定位置(如图 2 所示)。最后，按照正常的区块生成流程完成区块构建，并在区块链网络中进行共识。

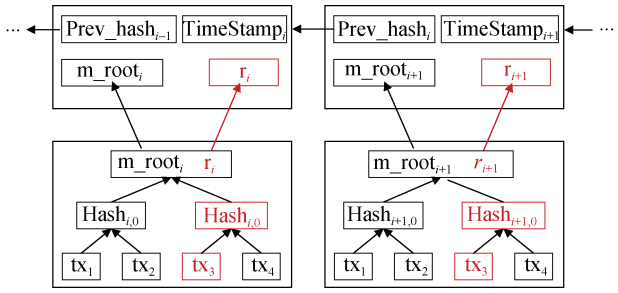


**图 2 可编辑区块链构造**

**Figure 2 Construction of Redactable Blockchain**

### 4.3 区块编辑

与现有工作相同，本方案也需要为区块编辑生成变色龙哈希函数的随机掩码，从而防止陷门密钥泄露，但是生成方式不再需要执行 DKG 协议，而是采用和变色龙哈希公私钥对相同的生成方式生成随机掩码的承诺 $K=g^k$ 和随机掩码的密文 $c_k=\mathrm{Enc}(k,\mathrm{pk})$。

当区块链用户或节点请求对区块高度为 $\tau$ 的交易 tx 进行编辑时，首先由区块链节点构造一个替代交易 tx′，并将 tx 替换为 tx′。然后重建 Merkle 树，在生成 Merkle 树根时，由区块链节点联合执行变色龙

哈希函数的编辑算法重新计算随机数字段, 使得编辑后的区块可通过验证。因此, 未参与编辑的节点只能判断该区块是否可以通过验证, 而无法判断该区块是否经过编辑, 从而实现无痕编辑。具体操作如下所示:

区块链节点 $P_i$: 计算修改随机数 $R'=\text{Merkel}\bullet K$, 然后计算 $c_{r'}=c_k\bullet c_{ts}^{-H(m'\|R')}$, 其中 $m'$ 为构造 Merkle 根的两个树子节点。然后使用解密密钥对随机数密文进行解密, 得到解密随机数碎片 $r_i'=\text{Dec}(c_k\bullet c_{ts}^{-H(m'\|R')},\text{sk}_i)$, 需要超过阈值 $t$ 个区块链节点的解密随机数碎片, 才可以恢复完整的, 并对此随机数碎片进行零知识证明 $\chi_i=\text{Prove}(r_i',c_{r'};\text{sk}_i)$ 确保解密随机数碎片的正确性, 然后将 $(c_i',\chi_i)$ 发送给其他区块链节点。

区块链节点 $P_i$: 验证各节点的随机数碎片是否正确 $0/1\leftarrow\text{Verify}(r_i',c_{r'})$, 如果不正确则对该节点进行投诉, 最终筛选出正确的 $t+1$ 个节点的随机数解密碎片。然后 $t+1$ 个随机数解密碎片进行随机数恢复: $r'=\prod_{j\in[1,t+1]}r_j'^{L_i(x)}$。最后使用新的随机数 $r'$ 替换原随机数, 即可完成区块编辑。

### 4.4 密钥更新

密钥更新方案与密钥生成阶段方案基本一致, 每个区块链节点使用解密私钥再次进行一轮秘密共享, 并对多项式值进行加密和零知识证明, 然后发送给代理节点。代理节点对信息进行验证, 然后通过同态操作, 对多项式值密文和多项式系数承诺分别进行聚合从而恢复门限解密密钥: $c_j=\prod_{i\in[1,t+1]}L_i(x)c_{i,j}$, $C_k=\prod_{i\in[1,t+1]}L_i(x)C_{i,k}$。然后将多项式值密文和多项式系数承诺发送给对应节点。区块链节点在收到代理节点的信息后同样对信息进行验证, 即可更新各节点的解密密钥。

### 4.5 安全证明

#### 4.5.1 分布式 CL 密钥生成的安全分析

**定理 1.**分布式 CL 密钥生成协议在少于 $t$ 个恶意节点的模型中安全。

证明. 图 3 展示了理想世界中分布式 CL 密钥生成协议的执行。假设敌手已经控制了区块链节点 $\mathcal{N}_1,...,\mathcal{N}_t$ 和一个代理节点。在理想世界中, 模拟器同样控制了相同的区块链节点 $\mathcal{N}_1,...,\mathcal{N}_t$ 和代理节点。然后使用模拟器来模拟诚实节点, 这对于敌手来说是与现实协议是不可区分的。模拟器有被控制节点的输入 $x_1,\cdots,x_t$, 以及最终的输出 pk。

每个共识节点 $N_i$ 发送随机数 $s_i$ 到 TTP。TTP 运行如下所示:

1) 计算 $s=\sum_{i=1}^{n}s_i$, $\text{pk}=g^s$, 选择 $a_1,\cdots,a_t\leftarrow\mathbb{Z}_q$, 并设置 $f(x)=s+a_1x+\cdots a_tx^t$。

2) 向节点 $N_i$ 发送 $(\text{pk},f(j))$。

**图 3 分布式 CL 密钥生成协议的理想协议**

**Figure 3 Ideal Protocol of Distributed CL Key Generation Protocol**

为了模拟诚实节点, 模拟器首先选择随机数 $\hat{x}_{t+1},\cdots,\hat{x}_n$ 作为诚实节点的输入, 然后分别为每个诚实节点 $\mathcal{N}_i$ 执行下面操作:

(1) 运行算法 $\text{Share}(\hat{x}_i)\rightarrow(\{\hat{s}_{i,j}\}_{j\in[1,n]},\{F_{i,k}\}_{k\in[0,t]})$: 由于 $\hat{x}_{t+1},\cdots,\hat{x}_n$ 是随机选择的, 而且多项式系数也是随机选择的, 因此对于敌手来说 $(\{\hat{s}_{i,j}\}_{j\in[1,n]},\{F_{i,k}\}_{k\in[0,t]})$ 与现实世界是不可区分的。

(2) 加密多项式值 $\hat{c}_{i,j}=\text{Enc}(\hat{s}_{i,j},\text{ek}_j)$: 由于密文 $\hat{c}_{i,j}$ 与随机数不可区分, 因此这一步骤与现实世界的诚实节点也是不可区分的。

(3) 计算 $\pi_{i,j}\leftarrow\text{Prove}_{\text{Enc}-\text{DL}}(\hat{c}_{i,j},\text{ek}_j,g^{\hat{s}_{i,j}})$, 进行零知识证明: 此处, 零知识证明是正常执行的, 而且也可以被正常验证, 因此对于敌手而言也不可区分。

最后模拟器发送 $\{\hat{c}_{i,j},\pi_{i,j}\}_{i,j\in[1,n]}$ 给敌手, 计算 $F'_{n,0}=\text{pk}/\sum_{i=1}^{n-1}x_i$ 和 $F'_{n,k}=(F_{n,0})^{\lambda_{k,0}}\cdot\prod_{i=1}^{t}(g^{\hat{s}_{n,i}})^{\lambda_{k,i}}$。发送 $(\{F_{i,k}\}_{i\in[t+1,n-1]},F'_{n,k})_{k\in[0,t]}$ 给敌手, 此处 $\{F'_{n,k}\}_{k\in[0,t]}$ 是根据给每个被控制节点发送密钥碎片以及公钥 pk 计算而来的, 因此对于敌手来说也是不可区分的。

敌手在得到模拟器发来的信息后对多项式值密文进行零知识证明验证, 而零知识证明是正常构造的, 因此验证可以通过。然后, 敌手解密多项式值密文, 并对多项式值明文进行多项式系数承诺验证, 由于多项式系数承诺 $\{F_{n,k}\}_{k\in[0,t]}$ 为特殊构造, 也可以通过验证。因此在理想模型中, 模拟器的运行与现实协议的执行对于敌手而言是不可区分的。因此对于敌手而言, 存在理想世界的非均匀 PPT 敌手 $\mathcal{S}$, 使得

$$\text{IDEAL}_{F,S(z),i}(x_1,...,x_n)\equiv\text{REAL}_{\pi,A(z),i}(x_1,...,x_n).$$

#### 4.5.2 分布式变色龙哈希修改的安全分析

**定理 2.**分布式变色龙哈希修改协议在少于 $t$ 个恶意节点的模型中安全。

证明. 图 4 展示了理想世界中分布式变色龙哈

希修改协议的执行, 假设敌手已经控制了区块链节点 $\mathcal{N}_1,...,\mathcal{N}_t$ 与一个代理节点。在理想世界中模拟器同样控制相同的区块链节点 $\mathcal{N}_1,...,\mathcal{N}_t$ 和代理节点。然后使用模拟器来模拟诚实节点, 这对于敌手来说是不可区分的。模拟器有被控制节点的输入 $\{\mathrm{sk}_j\}_{j\in[1,t]}$ , 以及最终的输出 $(r_1',r_2')$。模拟器执行下面操作来模拟第 $t+1$ 个节点的输入:

每个共识节点 $N_i$ 发送解密密钥碎片 $\mathrm{sk}_i$ 到 TTP。
TTP 运行如下所示:
1) 计算 $\mathrm{sk}=\sum_{i=1}^{t+1}L_i(x)\mathrm{sk}_i$, $r'_1=h\cdot g^k$, $\tilde{c}=c_k\cdot c_{ts}^{-H(m''||r'_1)}$, 解密 $r'_2=\mathrm{Dec}(\tilde{c},\mathrm{sk})$。
2) 向节点 $N_j$ 发送 $(r'_1,r'_2)$。

**图 4 分布式变色龙哈希修改的理想协议**

**Figure 4 Ideal Protocol of Distributed Chameleon Hash Adaptation Protocol**

(1) 计算 $r_1'=h\cdot g^k$, $\hat{c}'_{t+1}=(\mathrm{pk}/(g^{\sum_{i=1}^{t}\mathrm{sk}_i\cdot L_i(x)}))^{\frac{1}{L_{t+1}(x)}}$,和 $\hat{c}''_{t+1}=\tilde{c}/\hat{c}'_{t+1}$。此步骤直接通过最终输出以及各节点输入算得, 因此对于敌手来说是不可区分的。

(2) 调用零知识证明模拟器得到对 $\hat{c}=(\hat{c}'_{t+1},\hat{c}''_{t+1})$ 进行解密的零知识证明 $\pi_{\hat{c}}$, 然后发送 $(\hat{c},\pi_{\hat{c}})$ 到敌手。由于零知识证明模拟器使得零知识证明对于敌手而言也是不可区分的。

因为第 $t+1$ 个节点的信息是特殊构造的, 而且有随机数, 因此对于敌手而言不可区分。而零知识证明由于零知识证明模拟器的存在, 因此也不可区分。因此对于敌手而言, 存在理想世界的非均匀 PPT 敌手 $\mathcal{S}$。使得

$$\mathrm{IDEAL}_{F,S(z),i}(x_1,...,x_n)\equiv\mathrm{REAL}_{\pi,A(z),i}(x_1,...,x_n).$$

## 5 性能分析

为了在真实环境中评估本方案的性能, 本文在四个城市的 13 台云服务器上模拟部署了 211 个区块链节点。每台服务器配置 8 核 CPU、32GB 内存, 运行 64 位 Ubuntu 22.04 操作系统, 并通过多进程模拟多个区块链节点的运行。方案实现分为密码模块和通信模块。密码模块利用 BICYCL 库[30]进行虚二次域类群的算术运算、CL 同态加密和零知识证明; 该库在相同安全级别下将 CL 加密方案的效率提升至与 Paillier 加密相当。本文对该库进行了封装, 增加了多种零知识证明和加解密功能, 接口设计为字符串类型, 使编译后的动态库可供其他高级编程语言调用。椭圆曲线群的算术运算在C++中使用OpenSSL库实现, 在 Rust 中则采用 Curv 库。通信模块使用 Rust 的异步框架 Tokio 构建, 包含广播信道和 P2P 信道: 广播信道负责向所有参与方发送信息, P2P 信道则用于点对点通信。该架构确保了多节点环境下的高效通信和运算性能。

### 5.1 系统初始化

在现有去中心化可编辑区块链方案中, Jia 等人[9]的方案是有痕编辑, 即编辑过后仍然可以在链上发现错误信息, 因此与本方案研究目标不符。而 Ateniese 等人[8]的方案通过分布式变色龙哈希算法完成对区块内容的重编辑, 与本方案相关度较高, 因此选择 Ateniese 等人的方案进行对比。为了评估两种方案的性能, 本文部署了 211 个节点进行实验。

如图 5 所示, 本方案的耗时高于 Ateniese 等人[8]的方案, 主要原因在于本方案中涉及大量虚二次域类群的运算, 这些运算效率较低。此外, 由于 CL 加密密钥在未知阶群上进行操作, 所有计算只能在整数域中完成, 增加了计算复杂性。同时, 为生成变色龙哈希的陷门密钥, 还需要进行额外一轮通信, 进一步增加了耗时。相比之下, Ateniese 等人的方案假设所有节点都是诚实的, 因此采用 Shamir 秘密共享, 不具备对恶意节点的检测能力。如果有节点发送错误信息, 系统无法恢复, 必须重启协议, 且在存在恶意节点时, 系统可能无法达成一致结果。

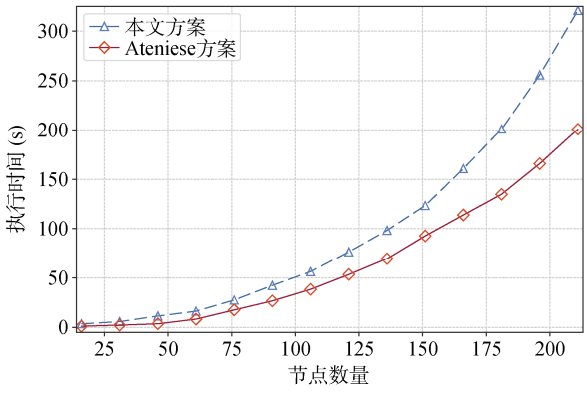


**图 5 分布式密钥生成性能对比**

**Figure 5 Performance Comparison of Distributed Key Generation**

### 5.2 区块生成

可编辑区块链相较于传统区块链, 将抗碰撞哈希函数替换为变色龙哈希函数, 从而引入了额外的计算开销。为了评估这种开销对区块生成阶段运行效率的影响, 本文在 FISCO BCOS 开源区块链平台[31]

上, 将底层哈希函数替换为变色龙哈希函数, 并测试了传统区块链、Ateniese 方案和本文方案的性能表现。

图 6 展示了区块链系统在接收 100,000 笔交易后打包区块的效率, 主要体现引入变色龙哈希并未对区块链系统造成影响。横轴表示每个区块内的最大交易数, 纵轴则表示区块链系统的吞吐量, 即每秒处理的交易数。在可编辑区块链系统的区块生成过程中, 仅在生成 Merkle 根时将传统哈希函数替换为变色龙哈希算法, 其他步骤与传统区块链完全相同。实验结果表明, 本方案的区块生成吞吐量与传统区块生成相比几乎没有差异。

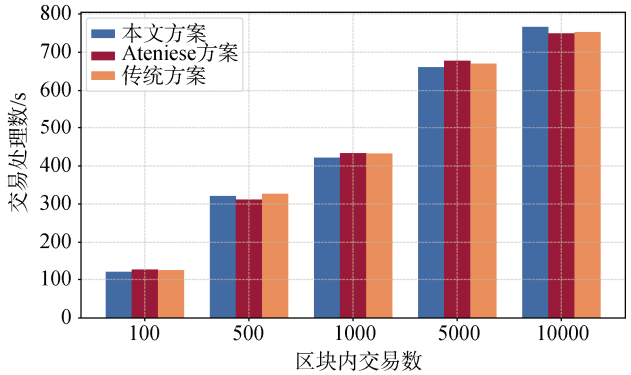


**图 6 区块生成效率对比**

**Figure 6 Comparison of Blockchain Generation Efficiency**

### 5.3 区块编辑

区块链的编辑操作分为离线阶段和在线阶段。离线阶段主要用于在编辑前生成用于保护密钥的随机掩码, 可进行大量预计算, 而在线阶段则是在节点接收到编辑请求后, 联合执行变色龙哈希的同态操作及联合解密。

如图 7 所示, 本方案在离线阶段的耗时仅为 Ateniese 方案的 37%。此外, 本方案的掩码生成与节点解耦, 即使节点发生变动, 掩码也无需重新生成。而 Ateniese 方案中的掩码与节点绑定, 节点动态变化时, 无法继续使用之前生成的掩码。

如图 8 所示, 在线阶段, 本方案在节点数量小于 61 时, 耗时较高, 因为相比于 Ateniese 的方案, 本方案需要额外的时间来生成和验证零知识证明, 当节点数量超过 61 时, 本方案的效率更高, 耗时仅为 Ateniese 方案的 78%。因此, 在大规模网络环境中, 本方案的优势得以体现, 且随着节点数量的增加, 本方案的性能优势还在进一步扩大。

实验证明, 本方案不仅具备安全性, 而且在整体性能上表现优异。虽然分布式密钥生成阶段效率较低, 但因该步骤仅需在系统初始化时执行一次, 其开销是可以接受的。

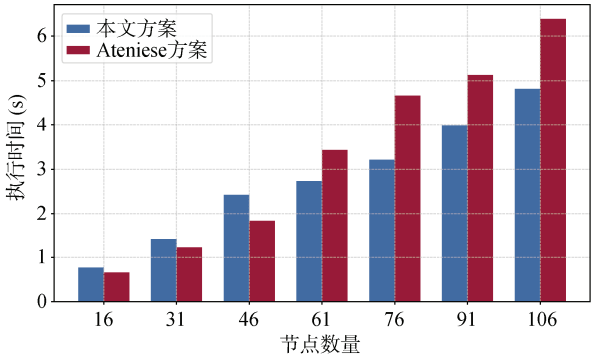


**图 7 离线随机掩码生成效率对比**

**Figure 7 Comparison of Offline Random Mask Generation Efficiency**

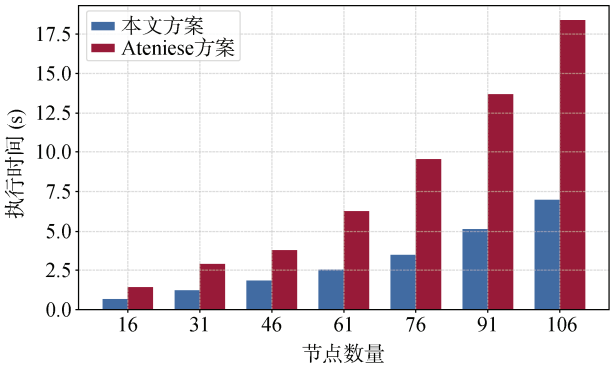


**图 8 在线编辑效率对比**

**Figure 8 Comparison of Online Redacting Efficiency**

## 6 结论

现有的可编辑区块链方案在编辑阶段需要生成随机掩码来保护变色龙哈希函数的陷门私钥, 这通常需要各节点运行可验证秘密共享来实现, 导致高通信开销, 并且在编辑阶段无法验证节点信息的正确性。为此, 本文提出了一种适用于大规模网络的可编辑区块链方案, 仅需将随机数加密存储于链上, 在编辑时对链上的加密数据进行同态运算, 最终联合解密得到编辑结果, 大幅减少了通信资源的消耗。同时, 本方案通过零知识证明与承诺绑定节点信息, 保证节点间信息的正确性, 使得系统可以从错误信息中筛选正确信息, 并维持系统运行。此外, 本方案改进了现有区块链结构, 实现最小化改动和无痕编辑, 维护了可编辑区块链系统的可信度。

## 参考文献

[1] Ullah F, Al-Turjman F. A Conceptual Framework for Blockchain Smart Contract Adoption to Manage Real Estate Deals in Smart Cities[J]. *Neural Computing and Applications*, 2023, 35(7): 5033-5054.

[2] Yang X H, Li W J. A Zero-Knowledge-Proof-Based Digital Identi-

ty Management Scheme in Blockchain[J]. *Computers & Security*, 2020, 99: 102050.

[3] Gai K K, Zhang Y, Qiu M K, et al. Blockchain-Enabled Service Optimizations in Supply Chain Digital Twin[J]. *IEEE Transactions on Services Computing*, 2023, 16(3): 1673-1685.

[4] Bhattacharya P, Tanwar S, Bodkhe U, et al. BinDaaS: Blockchain-Based Deep-Learning As-a-Service in Healthcare 4.0 Applications[J]. *IEEE Transactions on Network Science and Engineering*, 2021, 8(2): 1242-1255.

[5] Al Sayid M M R N E. The Fusion of Blockchain, Pornography and Human Trafficking in a Global Digital Dragnet that Forms the Online Child Sex Trafficking[J]. *Russian Law Journal*, 2023, 11(5s): 1-19.

[6] Whyte C. Cryptoterrorism: Assessing the Utility of Blockchain Technologies for Terrorist Enterprise[J]. *Studies in Conflict & Terrorism*, 2023, 46(7): 1126-1149.

[7] Ateniese G, de Medeiros B. On the Key Exposure Problem in Chameleon Hashes[M]. Security in Communication Networks. Berlin, HeidelbergSpringer2005: 165-179.

[8] Ateniese G, Magri B, Venturi D, et al. Redactable Blockchain – or – Rewriting History in Bitcoin and Friends[C]. *2017 IEEE European Symposium on Security and Privacy*, 2017: 111-126.

[9] Jia M, Chen J, He K, et al. Redactable Blockchain from Decentralized Chameleon Hash Functions[J]. *IEEE Transactions on Information Forensics and Security*, 2022, 17: 2771-2783.

[10] Dai W Q, Liu J K, Zhou Y, et al. PRBFPT: A Practical Redactable Blockchain Framework with a Public Trapdoor[J]. *IEEE Transactions on Information Forensics and Security*, 2024, 19: 2425-2437.

[11] Derler D, Samelin K, Slamanig D, et al. Fine-Grained and Controlled Rewriting in Blockchains: Chameleon-Hashing Gone Attribute-Based[C]. *Proceedings 2019 Network and Distributed System Security Symposium*, 2019.

[12] Ma J H, Xu S M, Ning J T, et al. Redactable Blockchain in Decentralized Setting[J]. *IEEE Transactions on Information Forensics and Security*, 2022, 17: 1227-1242.

[13] Shen J, Chen X F, Liu Z L, et al. Verifiable and Redactable Blockchains with Fully Editing Operations[J]. *IEEE Transactions on Information Forensics and Security*, 2023, 18: 3787-3802.

[14] Tian Y G, Li N, Li Y J, et al. Policy-Based Chameleon Hash for Blockchain Rewriting with Black-Box Accountability[C]. *The 36th Annual Computer Security Applications Conference*, 2020: 813-828.

[15] Tian Y G, Liu B W, Li Y J, et al. Accountable Fine-Grained Blockchain Rewriting in the Permissionless Setting[J]. *IEEE Transactions on Information Forensics and Security*, 2024, 19: 1756-1766.

[16] Xu S M, Huang X Y, Yuan J M, et al. Accountable and Fine-Grained Controllable Rewriting in Blockchains[J]. *IEEE Transactions on Information Forensics and Security*, 2023, 18: 101-116.

[17] Xu S M, Ning J T, Ma J H, et al. K-Time Modifiable and Epoch-Based Redactable Blockchain[J]. *IEEE Transactions on Information Forensics and Security*, 2021, 16: 4507-4520.

[18] Zhang Y Q, Ma Z F, Luo S S, et al. Dynamic Trust-Based Redactable Blockchain Supporting Update and Traceability[J]. *IEEE Transactions on Information Forensics and Security*, 2024, 19: 821-834.

[19] Thyagarajan S A K, Bhat A, Magri B, et al. Reparo: Publicly Verifiable Layer to Repair Blockchains[M]. Financial Cryptography and Data Security. Berlin, HeidelbergSpringer2021: 37-56.

[20] Tian Y G, Miyaji A, Matsubara K, et al. Revocable Policy-Based Chameleon Hash for Blockchain Rewriting[J]. *The Computer Journal*, 2023, 66(10): 2365-2378.

[21] Chen Y, Hao Z H, Wei J H, et al. Redactable Blockchain Supporting Trapdoor Revocation and Limited Number of Redactions[J]. *Journal on Communications*, 2023, 44(7): 100-113.
(陈越, 郝增航, 魏江宏, 等. 支持陷门撤销和编辑次数限制的可编辑区块链[J]. *通信学报*, 2023, 44(7): 100-113.)

[22] Guilhem Castagnos, Fabien Laguillaumie. Linearly homomorphic encryption from ddh. Cryptology ePrint Archive, Paper 2015/047, 2015. https://eprint.iacr.org/ 2015/047.

[23] Paillier P. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes[M]. Advances in Cryptology — EUROCRYPT '99. Berlin, HeidelbergSpringer2007: 223-238.

[24] Elgamal T. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms[J]. *IEEE Transactions on Information Theory*, 1985, 31(4): 469-472.

[25] Goldreich O, Micali S, Wigderson A. Proofs that Yield Nothing but Their Validity and a Methodology of Cryptographic Protocol Design[C]. *ACM Books*, 2019: 285-306.

[26] Fiat A, Shamir A. How to Prove Yourself: Practical Solutions to Identification and Signature Problems[M]. Advances in Cryptology — CRYPTO' 86. Berlin, HeidelbergSpringer2007: 186-194.

[27] Shamir A. How to Share a Secret[J]. *Communications of the ACM*, 1979, 22(11): 612-613.

[28] Pedersen T P. Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing[M]. Advances in Cryptology — CRYPTO '91. Berlin, HeidelbergSpringer2007: 129-140.

[29] Canetti R. Security and Composition of Multiparty Cryptographic Protocols[J]. *Journal of Cryptology*, 2000, 13(1): 143-202.

[30] Bünz B, Fisch B, Szepieniec A. Transparent SNARKs from DARK Compilers[C]. *Advances in Cryptology – EUROCRYPT 2020*, 2020: 677-706.

[31] FISCO BCOS. https://www.fisco-bcos.org/. 2020.

**高源芃** 黑龙江牡丹江人，硕士研究生，主要研究方向：区块链技术，应用密码学。 Email: ypgao.xidian@gmail.com

**冯哲** 陕西延安人，博士研究生.主要研究领域为区块链上隐私保护、分布式身份管理。Email: fz@stu.xidian.edu.cn

**刘雪峰** 安徽亳州人, 副教授. CCF 会员。主要研究领域为数据可用性与隐私、应用密码学。Email: liuxf@mail.xidian.edu.cn

**雷静** 甘肃泾川人, 博士后, 主要研究领域数据安全与隐私保护, 应用密码学, 后量子密码迁移等。Email: leijingxd@163.com

**裴庆祺** 广西玉林人, 教授。CCF 会员。主要研究领域为区块链技术、数据安全与隐私、可信算网融合等。Email: qqpei@mail.xidian.edu.cn