

2023 年全国大学生信息安全竞赛

作品报告



作品名称: MuLPay: 基于格的高效后量子安全区块链钱包系统

电子邮箱: _____

提交日期: 2023.06.13

填写说明

1. 所有参赛项目必须为一个基本完整的设计。作品报告书旨在能够清晰准确地阐述（或图示）该参赛队的参赛项目（或方案）。
2. 作品报告采用 A4 纸撰写。除标题外，所有内容必需为宋体、小四号字、1.5 倍行距。
3. 作品报告中各项目说明文字部分仅供参考，作品报告书撰写完毕后，请删除所有说明文字。（本页不删除）
4. 作品报告模板里已经列的内容仅供参考，作者可以在此基础上增加内容或对文档结构进行微调。
5. 为保证网评的公平、公正，作品报告中应避免出现作者所在学校、院系和指导教师等泄露身份的信息。

目录

摘要	1
第一章 作品概述	2
1.1 背景分析	2
1.1.1 区块链钱包简介	2
1.1.2 区块链钱包发展现状	3
1.1.3 多重签名方案发展现状	4
1.1.4 基于格的多重签名方案背景	5
1.1.5 以太坊智能合约发展现状	6
1.2 核心项目工作	6
1.3 研究目标及特色描述	7
1.3.1 研究目标	7
1.3.2 安全特性	7
1.3.3 产品特色	8
1.4 应用前景分析	9
第二章 作品设计与实现	10
2.1 系统概述	10
2.1.1 设计思路	10
2.1.2 系统功能	11
2.2 技术原理	12
2.2.1 格基本理论	12
2.2.2 多重签名方案 MuSig-L	15
2.2.3 基于格的零知识证明	20
2.2.4 MuSig-L 方案优化	20
2.2.5 国密算法 SM3、SM4	21
2.3 MuLPay 使用流程及其功能	22
2.3.1 登录	22
2.3.2 注册	23
2.3.3 多人签署交易	23
2.3.4 单人签名	23
2.3.5 钱包的社交恢复	24
2.3.6 修改登录口令	25
2.4 评价指标	26
2.4.1 效率	26
2.4.2 安全性说明	26

2.5 GUI 展示.....	27
2.5.1 预备环境.....	27
2.5.2 功能	27
2.5.3 多重签名实例.....	31
第三章 作品测试与分析	36
3.1 测试设备	36
3.2 MuSig-L 性能测试	36
3.2.1 版本 1	37
3.2.2 版本 2	37
3.2.3 版本 3	37
3.2.4 各版本性能对比	37
3.3 SM4 加速	38
3.3.1 SM4 加速比测量	39
3.4 对比其他钱包	39
第四章 创新性说明	40
第五章 总结	42
参考文献	43

摘要

区块链钱包是一种数字钱包，用于管理加密货币的存储、发送和接收。它是区块链技术的核心组成部分之一，可以让用户通过去中心化的方式进行点对点的交易，无需依赖银行或其他第三方机构。用户只需要妥善保管私钥，便可以保证交易的安全性。

由于区块链钱包在电子商务和社交应用等领域中具有重要的作用和潜力，与区块链钱包相关的技术正在蓬勃发展。而目前区块链钱包使用的基于椭圆曲线离散对数问题的多方签名方案存在以下问题：第一，基于椭圆曲线离散对数问题构建的多重签名方案容易受到量子计算机上的多项式时间 Shor 算法 [1] 的攻击；第二，目前已知的多重签名方案无法满足一些特殊的安全性质，包括在朴素公钥模型下可证明安全、满足并发签名安全性等；第三，已知的多重签名方案至少需要进行两轮线上交互，效率相对较低；第四，当前区块链钱包的多人管理大多采用门限签名实现，而门限签名无法对公钥和签名进行聚合，在链上的存储开销较大。

鉴于目前区块链钱包的多方签名方案的局限性，本作品改进了 Cecilia Boschini 等人提出的 MuSig-L 多重签名方案 [2]，并将其创造性地应用于区块链钱包中，提出了一种基于 MuSig-L 的区块链钱包应用——MuLPay，开发了相应的用户界面。该应用具备单人管理、共同支付、社交恢复的功能，并且能够利用助记词注册账户、重置登录口令。与此同时，本作品还对于对于安全性和性能都进行了相应的优化与创新。

本项目的创新性包括但不限于：第一，产品使用的改进的多重签名方案基于格上的困难问题，具备抗量子攻击的能力；第二，改进后的 MuSig-L 多重签名方案在可能发生拒绝采样失败的情况下，签名的效率更高。此外，MuSig-L 算法实现采用多线程优化，相较于朴素实现提升了 252%；第三，通过本地化存储结合登录时的零知识证明，产品能保证用户私钥及登录的安全性；第四，产品实现中的关键技术使用国密算法 SM4 并对其进行了优化，利用 GPU 硬件的高并发性，充分考虑 GPU 硬件各访存层次的因素，最终达到的加解密速率达到 10480Mbps，相较于传统的 CPU SIMD 实现 [3] 提升了 587.30%。

关键词：多重签名；抗量子；区块链技术；MuSig-L；密码货币钱包

第一章 作品概述

本作品 MuLPay 是基于格的一款区块链钱包。具备可靠的共同支付功能，能够使用助记词注册钱包账户，利用助记词恢复钱包，并且在助记词丢失后，能够通过特定验证方式转出对应账户的余额，从而避免货币流失。在本章节中，1.1节介绍本项目的背景，概述了区块链钱包及其发展现状、发展方向，指出目前区块链钱包面临的挑战，接着分析了多重签名方案的发展现状并引入基于格的多重签名方案，简要描述了以太坊智能合约的相关内容。1.2节详细介绍了本项目的核心工作，包括提出背景、创新点、实用性等。1.3节分别介绍了本项目的研究目标、安全特性和产品特色。最后，1.4节分析了本产品的应用前景。

1.1 背景分析

1.1.1 区块链钱包简介

区块链钱包是一种数字钱包，可以存储和管理加密货币，如比特币 [4]、以太币 [5] 或其他加密资产。它采用密码学技术，包括密钥对生成、签名和加密等功能，来保护用户的私钥并验证用户所做的交易，通过公钥和私钥的配对实现身份验证和交易授权。使用区块链钱包需要谨慎保管私钥，以免丢失或泄露。

图1.1简要描述了一笔以太坊交易打包上链的全过程：第一步中，用户对这笔以太坊交易签名；第二步则是用户发送这笔以太坊交易，这两个步骤由区块链钱包完成。在接下来的第三步中，这笔交易被发送到一个以太坊网络节点；在第四步中，这笔交易被以太坊网络节点放入整个以太坊交易池中，第五步和第六步中，则由矿工负责从交易池中选择一些以太坊交易并计算其哈希值，最后将选择的以太坊交易打包上链。

区块链钱包的工作是图1.1的第一步和第二步，完成对交易的签名并把这笔交易发送给以太坊网络节点的工作，剩余的打包上链等工作则由矿工来完成。

市面上已经涌现出多种区块链钱包的应用，包括以太坊钱包 MetaMask、多资产钱包 Exodus、硬件钱包 Ledger Nano S 等等，这些区块链钱包不仅能用于存储和管理加密货币，提供安全的资产储存和交易功能，而且推动了数字资产的普及和应用。

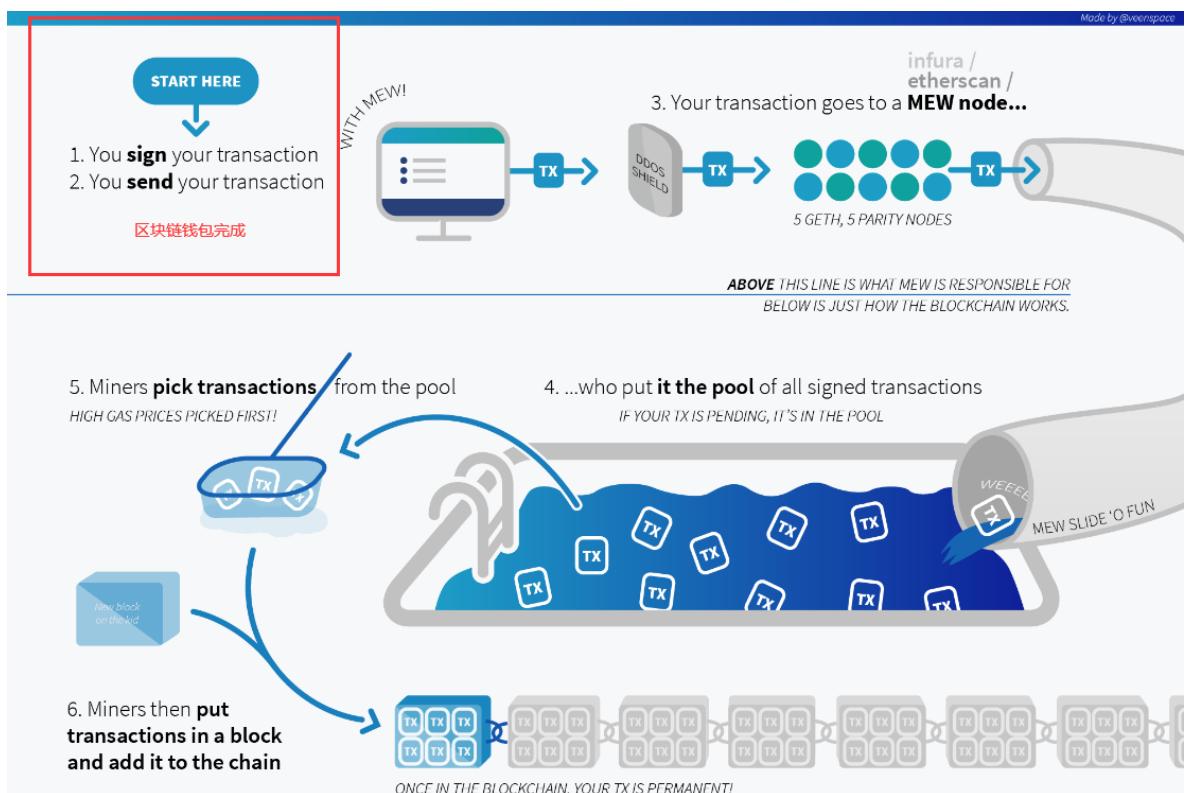


图 1.1 以太坊交易流程 [6]

1.1.2 区块链钱包发展现状

表 1.1 各区块链钱包对比图

	<i>MetaMask[7]</i>	<i>imToken[8]</i>	<i>HyperPay</i>	<i>Binance</i>
是否使用助记词	√	√	√	√
是否支持共同支付	×	√	√	√
基于困难问题	DLP	DLP	DLP	DLP
使用的多方签名	×	MuSig	TSS	TSS

调研发现，目前市面上的区块链钱包主要基于离散对数 [9] (Discrete Logarithm Problem, 简称 DLP) 这一困难问题，并且大多使用门限签名 [10] (Threshold Signature Scheme, 简称 TSS) 或者传统多重签名方案 [11] (Multiple Signatures, 简称 MuSig) 来完成。与传统的基于 DLP 的密码方案相比，基于格的密码方案不仅具备抗量子攻击的特性，还具有很多其他密码学用途，如密钥交换、基于属性的加密、基于身份的加密、全同态加密等，而且基于格的密码学方案往往实现简单，具备最坏情况到平均情况的安全规约。

区块链钱包的发展现状呈现以下几个趋势和特点：

1. 多样化的钱包类型：随着区块链技术的发展和普及，出现了各种类型的区块链钱包。除了传统的纸钱包、软件钱包和硬件钱包外，还涌现了在线钱包、移动钱包、Web 钱包等多种形式。不同类型的钱包在功能和用户体验上有所区别，以满足用户不同的需求。
2. 增强的安全性：安全性一直是区块链钱包的重要关注点。针对私钥的保护和防范钱包被攻击的措施也不断改进。硬件钱包的使用越来越普遍，其离线存储和安全元件可有效保护私钥免受网络攻击。此外，多重签名钱包和生物识别技术（如指纹、面部识别）的应用也提供了额外的安全层。
3. 增加的互操作性：随着不同区块链平台和加密货币的增加，区块链钱包正在朝着更好的互操作性方向发展。现在许多钱包支持多种不同的区块链资产，并通过跨链技术实现资产的转移和交互。这种互操作性有助于提高用户的便利性和自由度。
4. 用户友好的界面和体验：为了吸引更多用户，区块链钱包开始注重用户界面的设计和用户体验的改进。现代的钱包应用通常提供直观的界面、易于操作的功能和友好的用户指南，以使用户更轻松地管理和交易数字资产。
5. 增加的附加功能：区块链钱包不仅提供数字资产存储和交易功能，还增加了一些附加功能。例如，一些钱包提供加密货币市场数据、实时行情、投票权益等功能，以满足用户对于更全面的区块链服务的需求。

总体而言，区块链钱包正处于不断创新和发展的阶段。安全性、互操作性、用户体验和附加功能是当前钱包发展的关键方向。随着区块链技术和数字资产的普及，我们可以预期未来区块链钱包将继续提供更多功能和更好的用户体验。

1.1.3 多重签名方案发展现状

多重签名是一种需要多方协同生成签名的密码学协议。其核心思想是利用 Schnorr 签名算法 [12] 的线性性质，将多个签名方的签名结果聚合起来，最终形成对一个特定消息的签名。

最早的 MuSig 由 Maxwell 等人 [12] 提出。该协议可以由图1.2表示。第一轮交互中，所有参与者发送 Nonce_i 的承诺 $t_i = H(R_i)$ 。第二轮交互中，所有参与者发送 Nonce_i ，其余参与方验证 $t_i = H(R_i)$ 。第三轮交互中，各参与方计算并发送部分签名 s_i 。最后，将 $s_1, \dots, s_i, \dots, s_n$ 聚合，得到多重签名值。

多重签名技术在数字资产交易中得到了广泛的应用：因为此种机制不仅可以防止单个授权方的恶意行为，例如私自冒名顶替，转移资产等，让多方合作实现私钥管

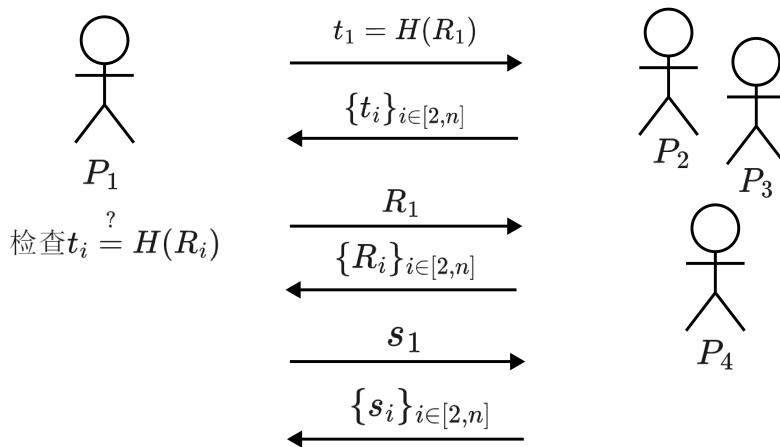


图 1.2 MuSig 三轮交互图

理，保证了加密资产的安全性和灵活性，还能极大地降低链上的存储开销，有助于为加密货币的安全性和可扩展性提供更好的解决方案，作为在区块链技术中广泛采用的安全机制。

MuSig 的发展主要经历了两个阶段。第一个阶段开始于 2018 年。由 Maxwell 等人提出基于 Schnorr 签名的在 Bitcoin Core 上实现的 MuSig 方案 [12]，是 MuSig 方案首次在区块链上亮相，对比特币社区产生了积极的影响。它为比特币提供了一种更高效的多方签名方案，并帮助提高网络的隐私和安全性，使得比特币更具可扩展性和实用性。随着 Schnorr 签名被加入到比特币代码库中，MuSig 也将成为比特币网络中的新标准之一。MuSig 的提出还鼓励了其他加密货币和区块链系统的开发者研究和采用更高效、更安全的签名方案。第二个阶段是 MuSig 方案的不断完善与改进。由于 MuSig 的三轮交互需要更多的交互时间，在 2019 年 Pieter Wuille 等人提出了 MuSig2[13]，成功将三轮交互压缩至两轮。

MuSig 目前的发展方向主要有：改进 MuSig 的性能；提高 MuSig 的可扩展性和抗恶意攻击的能力；进一步推广 MuSig 在加密货币、物联网中的应用；探索 MuSig 与其他密码学方案的结合。

1.1.4 基于格的多重签名方案背景

近年来，随着量子计算技术不断发展，2022 年 Boschini 等人首次提出了基于格 (Lattice) 的 MuSig 方案——MuSig-L[2]。与传统的基于 DLP 的密码方案相比，基于格的密码方案不仅具备抗量子攻击的特性，还具有很多其他密码学用途，如密钥交换、基于属性的加密、基于身份的加密、全同态加密等，而且基于格的密码学方案往往实

现简单，具备最坏情况到平均情况的安全规约。而基于 DLP 的方案没有上述优点，因此目前许多研究者正在积极探索基于格的抗量子密码学方案，以满足不断增长的安全需求和更复杂的应用场景，确保系统的整体安全性和鲁棒性。

综合来看，基于格的 MuSig 方案在安全性和可应用性方面都较为优越，将成为比特币和其他加密货币领域未来的重要趋势之一 [14]。

1.1.5 以太坊智能合约发展现状

以太坊智能合约是区块链技术 [15] 中的重要应用之一，也是当前区块链技术发展中最为活跃的领域之一。随着以太坊网络的不断发展和智能合约技术的不断完善，越来越多的开发者和企业开始探索基于以太坊的应用开发和商业应用。

目前，以太坊智能合约在数字资产、去中心化金融（Decentralized Finance，简称 DeFi）[16]、供应链管理、电子商务等领域得到了广泛的应用。在数字资产领域，以太坊智能合约被广泛用于实现代币发行和管理、加密货币交易，包括很多著名的加密货币和数字资产都是基于以太坊智能合约建立的。在 DeFi 领域，以太坊智能合约被用于实现借贷、流动性挖矿、合成资产等多种去中心化金融应用。在供应链管理和电子商务领域，以太坊智能合约被应用于商品溯源、防伪溯源、合同管理等场景，实现网络上的自动化合约执行。

除了智能合约的应用领域不断扩展之外，以太坊智能合约技术本身也在不断发展和完善。目前，以太坊正在逐步完成 ETH 2.0[17] 的升级，将实现更高的交易吞吐量和更低的手续费；以太坊社区也在不断推出新的开发工具和编程语言，如 Solidity、Vyper 等，助力开发者更高效地进行智能合约开发。

因此，可以预见的是，以太坊智能合约在未来将继续成为区块链技术发展的重要驱动力之一，在数字资产、DeFi、供应链管理、电子商务等多个领域产生突破性的应用和创新。

1.2 核心项目工作

区块链钱包在现代金融、电子商务、社交应用等领域中扮演着重要的角色，拥有广阔的应用前景，但目前市面上的区块链钱包主要由基于 DLP 的签名算法构建，容易受到量子计算机上的多项式时间 Shor 算法攻击，而且无法满足四个安全特性，因此本项目创造性地把基于格的多重签名方案应用于区块链钱包上，开发了区块链钱包应用 MuLPay。

本项目改进了 Cecilia Boschini 等人提出的 MuSig-L 多重签名方案 [2]。改进后的

多重签名算法与之前的基于 DLP 的多重签名算法相比，具备抵抗量子攻击的能力，仅有一轮在线交互，计算签名的效率较高，实现也更为简单。

本项目将改进后的 MuSig-L 算法创造性地应用于区块链钱包中，提出了一种基于 MuSig-L 方案的区块链钱包的设想，开发了相应的用户界面，并采用了异构计算和多线程并发的方式对应用性能进行优化，实现了一个具备单人管理、共同支付、社交恢复等功能的区块链钱包（MuLPay）。

特别的，项目在软件实现的时候将国密算法 SM3，SM4 与 MuSig-L 融合，构建出一整套完整且合理的登录、验证、交易过程，同时采用 GPU 加速了 SM4 的实现，为将来可能的产品落地提供了一系列可行的思路。

1.3 研究目标及特色描述

1.3.1 研究目标

本项目的产品 MuLPay 采用基于格的多重签名方案，致力于实现一个简单高效、安全易使用、可拓展性强的区块链钱包，能够使用助记词生成私钥，具备共同支付，助记词恢复等丰富功能。

1.3.2 安全特性

本项目产品 MuLPay 基于改进后的 MuSig-L 方案可以达到以下四个安全特性：

1. 在朴素公钥模型 (Plain Public-Key Model, 简称 PPK 模型) 下可证明安全

PPK 模型要求每个签名者都在没有专门的交互式密钥生产协议的情况下，明确地公开其公钥。为了抵抗流氓密钥攻击 [18]，传统的解决方案需要用户提交他们已知私钥的证据 [19]。然而，本项目基于的 MuSig-L 方案在理想的 PPK 模型下便能够证明抵御此类攻击。

2. 满足并发签名安全性

2019 年, Drijvers 等人 [20] 指出了许多现有交互式方案存在的缺陷，并且提出了一种利用并发地发起多个对话的攻击策略。在此场景中，敌手可能与一个诚实一起发动签名的多个实例，通过组合来自不同会话的签名得到一个新消息的伪造签名。2021 年, Benhamouda 等人 [21] 对攻击方法进行改进，得到了多项式时间的攻击算法。因此，满足并发签名查询下的安全性是至关重要的。本方案可以达到并发签名安全性。

3. 在线交互阶段简单高效

本文所基于的 MuSig-L 方案仅需要一轮在线交互，因此具有简便高效的优点。为了减少在线交互的轮数和通信复杂度，在获取签署消息之前，该方案使用离线预算进行优化。

4. 支持公钥聚合

使得多重签名生成的签名与单人签名生成的签名保持一样的结构和分布。允许验证者用一个普通的单人验签公钥来验证签名，从而使得该方案与现有的验签算法兼容，仅需要一个“聚合的”公钥验证签名的合法性。并且，在区块链上仅需要存储聚合后的签名和公钥，降低了存储的空间开销和验签的时间开销。

5. 抵御量子算法的攻击

本项目的多重签名方案的安全性可以归约到 MSIS 和 MLWE 问题上，这两个困难问题被认为是无法被量子计算机有效解决的计算难题 [22]。

1.3.3 产品特色

本项目设计的区块链钱包 MuLPay 满足以下特色：

1. 支持多人协同管理

本项目运用了多重签名技术实现共同支付的功能，使得多个人能够对于转账交易进行共同签署，也可以进行社交恢复。具体而言，共同支付指多人签署一笔交易，当钱包拥有者要转账时，需要向监管者们发起请求，签名验证成功后才可打包上链；社交恢复功能是指当钱包拥有者忘记助记词时，通过智能合约向预先设置的守护者们发出请求，守护者们协同签署一笔特殊的交易，将原账户中的余额转入一个新的账户，从而实现钱包恢复。

2. 钱包私钥本地化管理

我们实现了一种使用助记词派生公私钥对，并使用口令对公私钥对进行加密后本地存储的私钥管理方法。由于私钥在本地客户端进行私钥存取，加强了密钥管理的安全性和可靠性。

3. 项目设计与实现高效

在方案设计方面，我们使用的 MuSig-L 方案采用离线阶段预算和在线阶段交互相结合，仅需要一轮的在线交互阶段。我们还修改了方案设计，使得各方仅需并发地执行在线阶段，以降低 RejSamp 失败带来的延迟。在方案的工程实现方面，我们综合地使用多线程技术和 GPU 异构计算加速，大大提升了各算法部件的执行效率。

4. 用户界面简单易用

用户界面简单易用。项目 GUI 开发提供了直观的操作界面，用户可以轻松管理加密货币资产、发送和接收交易、社交恢复等。

1.4 应用前景分析

1. 从签名基于的困难问题来看，MuLPay 具有抗量子性。

目前市面上的区块链钱包是基于 DLP 困难问题，不具备抗量子性。而我们实现的 MuLPay 底层采用的多重签名方案是基于格上困难问题，该多重签名方案可以在保证通信安全的前提下，让多个用户共同完成数字签名，并满足四个安全性质。

2. 从使用功能来看，MuLPay 的两个重要功能具有重要意义。

MuLPay 实现了共同支付和钱包恢复。共同支付即多人签署交易，这一功能可以用于加密货币交易所的账户管理，例如需要多个员工签署的交易才能执行资金提取操作，从而提高安全性；可以用于去中心化自治组织的管理和治理，例如需要多个成员签署才能执行某些关键决策或释放资金，并确保所有参与者得到平等的代表和参与。钱包恢复功能允许白名单的用户共同转出忘记私钥的钱包中的钱以避免财产损失，例如社交网络上的朋友、亲属或同事，以协助在忘记密码或私钥等情况下转出钱包的钱。

3. 从应用前景来看，MuLPay 易于扩展，能够管理多种数字货币。

MuLPay 易于扩展的特性不仅表现在它能够管理多种数字货币上，还表现在它未来可以根据需求灵活升级和扩展上。随着格理论的发展，签名大小可以进一步压缩 [23]，除此之外，抗量子区块链系统 [24] 的发展也使得进一步压缩 MuSig-L 的签名大小成为可能。

第二章 作品设计与实现

本作品对 MuSig-L 多重签名方案 [2] 进行了改进，并创新性地应用于区块链钱包中，提出了名为 MuLPay 的基于 MuSig-L 的区块链钱包应用，具备单人管理、共同支付、社交恢复等功能。同时，本作品对安全性和性能进行了优化和创新。

在本章节中，2.1节首先总体上介绍本项目的整体架构，涵盖了底层的密码算法及协议的选择，到编程语言实现，再到应用层如何实现具有共同支付、单人管理、社交恢复功能的区块链钱包等部分。具体而言，2.2节中我们详细介绍了格的基本理论，基于格的多重签名协议 MuSig-L、对 MuSig-L 方案的优化以及本项目使用的其他密码算法的原理与实现。2.3节则介绍了我们依据上述密码算法和协议所构建的区块链钱包的使用流程及其功能。2.4节阐述了本项目在安全性和效率方面达到的效果。2.5节演示了我们所设计的区块链钱包的实际使用场景和操作方法。

2.1 系统概述

2.1.1 设计思路

本项目综合使用了 MuSig-L、SM4 与 SM3 算法，采用了 Python/Sagemath[25]、C++/CUDA 与 Solidity 编程语言实现了具有共同支付、单人管理、社交恢复 [26] 功能的区块链钱包。项目架构如图2.1。

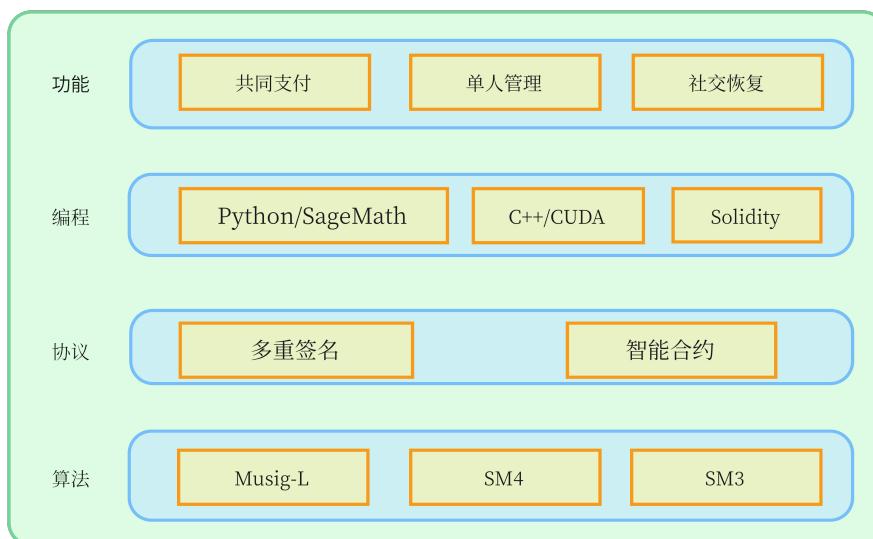


图 2.1 项目架构

2.1.2 系统功能

本项目设计的 MuLPay 是一款利用基于格的多重签名方案实现的区块链钱包，其核心功能不仅包括单人的交易管理，同时借助多重签名协议实现了多人共同支付及社交恢复钱包的功能，具体描述如下：

1. 设计使用助记词派生钱包公私钥，使用口令登录认证，公私钥对本地管理的钱包管理方案。
2. 当用户未设置监管成员时，采用常规的单人签名方式支付。
3. 当用户设置了监管成员后，则利用 MuSig-L 方案实现多人签署一笔交易，即只有账号的 N 个监护人执行签名同意了这笔交易，这笔交易才会被确认，从而实现多人共同管理一个钱包账户的功能。
4. 当用户忘记登录口令时，利用助记词重置登录口令。
5. 当用户忘记助记词时发送验证消息给其监护人，利用 MuSig-L 方案实现的智能合约转出丢失助记词的账户中的余额（但转账的额度收到智能合约的限制）。

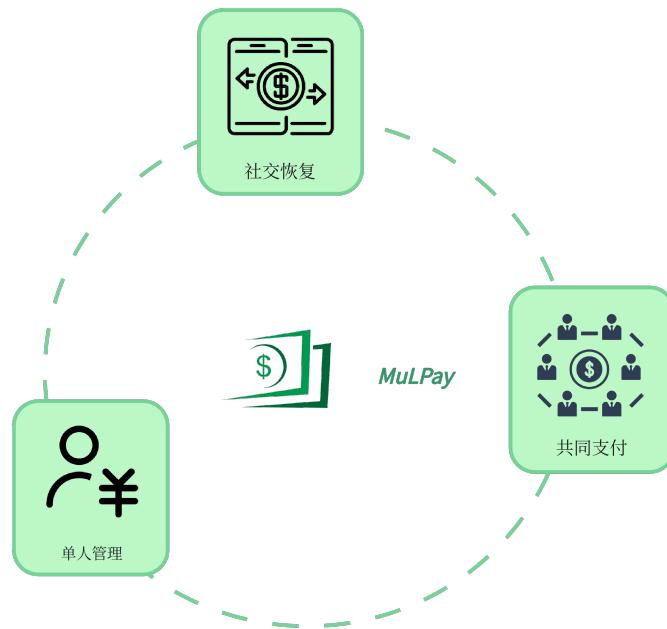


图 2.2 核心功能

2.2 技术原理

2.2.1 格基本理论

定义 1(格) 格是 \mathbb{R}^m 中一类具有周期性结构离散点的集合。严格地说，格是 m 维欧氏空间 \mathbb{R}^m 的 $n(m \geq n)$ 个线性无关向量组 $(\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_n)$ 的所有整系数线性组合构成的集合。即

$$\mathcal{L} = \mathcal{L}(\mathbf{B}) := \mathbf{B} \cdot \mathbb{Z}^m = \left\{ \sum_{i=1}^n z_i \mathbf{b}_i : z_i \in \mathbb{Z} \right\}. \quad (2.1)$$

其中， $\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_n)$ 称作 \mathcal{L} 的一组格基。 m 称为格的维数， n 称为格的秩。对于 \mathcal{L} ，其格基是不唯一的。对于任意的幺模矩阵 $\mathbf{U} \in \mathbb{Z}^{m \times m}$ ， $\mathbf{B} \cdot \mathbf{U}$ 也是 $\mathcal{L}(\mathbf{B})$ 的一组格基，因为 $\mathbf{U} \cdot \mathbb{Z}^m = \mathbb{Z}^m$ 。

例 1 ① 整数格 \mathbb{Z}^m 。

② “棋盘”格 $\{\mathbf{x} \in \mathbb{Z}^m : \sum_i x_i \text{ 是偶数}\}$ 如图 2.3 所示。

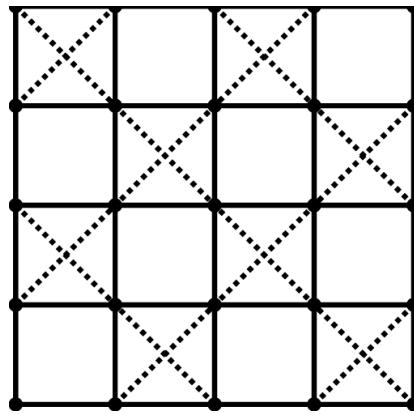


图 2.3 “棋盘”格

定义 2(对偶格 (dual lattice)) 对偶格与原格在同一个线性空间 \mathbb{R}^m 中，定义为 $\mathbf{L}^* = \left\{ \mathbf{x} \in \mathbb{R}^m, \forall \mathbf{v} \in \mathbf{L}, \langle \mathbf{x}, \mathbf{v} \rangle \in \mathbb{Z} \right\}$ 。

定义 3(最短向量长度 (shortest vector length)) 格 \mathcal{L} 上的非零最短向量的长度记为：

$$\lambda_1(\mathcal{L}) := \min_{\mathbf{v} \in \mathcal{L} \setminus \{0\}} \|\mathbf{v}\| \quad (2.2)$$

定义 4(最短向量问题 (Shortest Vector Problem, 简称 SVP 问题)) 给定用格基 \mathbf{B} 描述的格 \mathcal{L} ，寻找格上的最短非零向量，即寻找到一个非零向量 $\mathbf{u} \in \mathcal{L}$ 使得 $\|\mathbf{u}\| = \lambda_1(\mathcal{L})$ 。

定义 5 (近似最短向量问题 (Approximate Shortest Vector Problem, 简称 SVP_γ 问题))

给定用格基 \mathbf{B} 描述的格 \mathcal{L} , 寻找一个格上非零向量 \mathbf{v} , 满足 $\|\mathbf{v}\| \leq \gamma(n) \cdot \lambda_1(\mathcal{L})$ 。

其中 $\gamma = \gamma(m)$ 是近似因子, 当 $\gamma(m) = 1$ 时, 问题还原到 SVP 问题。

定义 6 (判定性近似最短向量问题 (Approximate SVP, 简称 GapSVP_γ 问题)) 给定用格基 \mathbf{B} 描述的格 \mathcal{L} , 判定 $\lambda_1(\mathcal{L}) \leq 1$ 还是 $\lambda_1(\mathcal{L}) > \gamma(n)$ 。

定义 7 (最短线性无关向量问题 (Shortest Independent Vectors Problem, 简称 SIVP))

给定一个秩为 n 的格 \mathcal{L} , 找到 n 个线性无关的向量 $\mathbf{v}_1, \dots, \mathbf{v}_n$, 满足 $\max_i \|\mathbf{v}_i\| \leq \gamma \lambda_n(\mathcal{L})$ 。

定义 8 (最近向量问题 (Closest Vector Problem, 简称 CVP 问题)) 给定用格基 \mathbf{B} 描述的格 \mathcal{L} 和目标向量 $\mathbf{t} \in \mathbb{R}^m$, 寻找距离 \mathbf{t} 最近的格向量 \mathbf{v} , 满足对于任意向量 $\mathbf{u} \in \mathcal{L}$, $\|\mathbf{v} - \mathbf{t}\| \leq \|\mathbf{u} - \mathbf{t}\|$ 。

注记 目前, 在密码学领域尚无归约到 CVP 问题的可证明安全密码方案。

定义 9 (有界距离译码问题 (Bounded Distance Decoding, 简称 BDD 问题)) 给定一个 n 维的格 \mathcal{L} 、目标向量 $t \in \mathbb{R}^m$ 满足 $dist(t, \mathcal{L}) < \gamma \lambda_1(\mathcal{L})$, 找到一个非零的格向量 v , 满足对任意非零向量 $u \in \mathcal{L}$, $\|\mathbf{v} - \mathbf{t}\| < \|\mathbf{u} - \mathbf{t}\|$ 。

定义 10 (绝对距离译码问题 (Absolute Distance Decoding, 简称 ADD 问题)) 给定一个 n 维的格 L 和目标点 $t \in R^n$, 满足 $dist(t, L) \geq d = \mu(L)$ ($\mu(L)$ 为格覆盖半径), 找到唯一的格向量 $v \in L$ 且 $\|t - v\| < d$ 。

定义 11 (离散高斯分布) 令 $\Sigma \in K_{\mathbb{R}}^{m \times m}$ 是一个对称的正定矩阵, 令 $\sqrt{\Sigma} \in K_{\mathbb{R}}^{m \times m}$ 是一个非奇异矩阵, 满足 $\Sigma = \sqrt{\Sigma} \sqrt{\Sigma}^*$ 。格 $\mathcal{L} \subseteq R^m$ 上以 \mathbf{c} 和 Σ 为参数的离散高斯分布 $\mathcal{D}_{\Sigma, \mathbf{c}, \Lambda}$ 定义为:

$$\rho_{\sqrt{\Sigma}, \mathbf{c}}(\mathbf{z}) := \exp\left(-\pi \|\sqrt{\Sigma}^{-1}(\mathbf{z} - \mathbf{c})\|^2\right) \quad \mathcal{D}_{\sqrt{\Sigma}, \mathbf{c}, \mathbf{c}, \Lambda}^m(\mathbf{z}) := \frac{\rho_{\sqrt{\Sigma}, \mathbf{c}}(\mathbf{z})}{\sum_{\mathbf{x} \in \Lambda} \rho_{\sqrt{\Sigma}}(\mathbf{x})} \quad (2.3)$$

定义 12 (L^p 范数) 对于一个模元素 $\mathbf{v} \in R^m$ 的 L^p 范数由系数嵌入定义: 对于 $\mathbf{v} = (\sum_{i=0}^{N-1} v_{i,1} X^i, \dots, \sum_{i=0}^{N-1} v_{i,m} X^i)^T$, 其 L^p 范数为

$$\|\mathbf{v}\|_p := \left\| (v_{0,1}, \dots, v_{N-1,1}, \dots, v_{0,m}, \dots, v_{N-1,m})^T \right\|_p. \quad (2.4)$$

定义 13 (短整数解问题 (Short Integer Solution, 简称 SIS 问题)) 均匀随机选取 m 个向量 $a_i \in Z_q^n$, 并将其按列向量排列, 构建得到矩阵 $A \in Z_q^{n \times m}$, 尝试寻找一个非零整数向量 $z \in Z^m$ 并且 $\|z\| < \beta$, 满足 $Az = \sum_i a_i \cdot z_i = 0 \in Z_q^n$ 。

定义 14 (LWE 分布) 对于一个秘密向量 $s \in Z_q^n$, 处于 $Z_q^n \times Z_q$ 上的 LWE 分布 $A_{s,\chi}$ 被定义为均匀随机选取一个向量 $a \in Z_q^n$, 再从错误分布中选取一个整数 e , 输出 $(a, b = \langle \mathbf{s}, \mathbf{a} \rangle + e \bmod q)$ 。

定义 15 (容错学习问题 (Learning With Errors, 简称 LWE 问题)——搜索版本) 自 LWE 分布 $A_{s,\chi}$ 中选取 m 个独立样本 $(a_i, b_i) \in Z_q^n \times Z_q$, 尝试找到所对应的秘密向量 $s \in Z_q^n$ 。

定义 16 (容错学习问题 (Learning With Errors, 简称 LWE 问题)——决策版本) 给定 m 个独立样本 $(a_i, b_i) \in Z_q^n \times Z_q$, 判定每个样本是取自 LWE 分布 $A_{s,\chi}$ 还是取自均匀随机分布。

定义 17 (Ring-SIS) 给定环 R_q , 均匀随机选取 m 个元素 $a_i \in R_q$, 构成向量 $a \in R_q^m$, 尝试寻找一个非零向量 $z \in R^m$ 并且 $\|z\| < \beta$, 满足 $\langle \mathbf{a}, \mathbf{z} \rangle = \sum_i a_i \cdot z_i = 0 \in R_q$ 。

定义 18 (Ring-LWE 分布) 对于一个秘密元素 $s \in R_q$, 处于 $R_q \times R_q$ 上的 Ring-LWE 分布 $A_{s,\chi}$ 被定义为均匀随机选取一个元素 $a \in R_q$, 再从错误分布中选取一个多项式 e , 输出 $(a, b = s \cdot a + e \bmod q)$ 。

定义 19 (Ring-LWE——搜索版本) 自 Ring-LWE 分布 $A_{s,\chi}$ 中选取 m 个独立样本 $(a_i, b_i) \in R_q \times R_q$, 尝试找到所对应的秘密元素 $s \in R_q$ 。

定义 20 (Ring-LWE——决策版本) 给定 m 个独立样本 $(a_i, b_i) \in R_q \times R_q$, 判定每个样本是取自 Ring-LWE 分布 $A_{s,\chi}$ 还是取自均匀随机分布。

定义 21 (MSIS_{q,k,l, β} 问题) 给定环 R_q , 均匀随机选取 m 个非线性元素 $a_i \in R_q^k$, 尝试寻找 m 个向量 $z_i \in R$ 并且 $0 < \|z_i\| < \beta$, 满足 $\sum_i a_i \cdot z_i = 0 \bmod q$ 。

定义 22 (MLWE_{q,k,l, η} 问题——搜索版本) 给定环 R_q , 均匀随机选取 m 个 $a_i \in R_q^k$, 选取服从某分布 χ^k 的 $s \in R_q^d$, 以及服从某分布 χ 的 $e \in R_q$, 输出 m 组 $(b_i = \langle \mathbf{a}_i, \mathbf{s} \rangle + e_i, a_i)$, 尝试寻找所对应的 s 。

定义 23 (MLWE_{q,k,l, η} 问题——判定版本) 给定环 R_q 与 m 组 (b_i, a_i) , 判断每组样本是取自 $(b_i = \langle \mathbf{a}_i, \mathbf{s} \rangle + e_i, a_i)$ 所对应的分布还是取自均匀随机分布。

定义 24 (MSIS_{q,k,l, β} 假设) 令 $\lambda \in \mathbb{N}$ 是安全参数。对于一个素数 $q(\lambda)$, 一个界 $\beta = \beta(\lambda) > 0$ 和正整数 $k = k(\lambda)$, $l = l(\lambda)$, MSIS_{q,k,l, β} 假设为: 如果对于任何概率多项式时间算法 \mathcal{A} , 下式中的优势在参数 λ 下都是可忽略的。

$$\text{Adv}_{q,k,l,\beta}^{\text{MSIS}}(\mathcal{A}) := \Pr \left[0 < \|x\| \leq \beta \wedge [\mathbf{A} \|\mathbb{I}_k] \cdot x = \mathbf{0} \bmod q : \mathbf{A} \xleftarrow{\$} R_q^{k \times l}; x \leftarrow \mathcal{A}(\mathbf{A}) \right]. \quad (2.5)$$

定义 25 (MLWE_{q,k,ℓ,η} 假设) 令 $\lambda \in \mathbb{N}$ 是安全参数。对于一个素数 $q(\lambda)$, 和正整数 $\beta = \beta(\lambda) > 0$, $k = k(\lambda)$, $l = l(\lambda)$, MLWE_{q,k,ℓ,η} 假设为: 如果对于任何概率多项式时间区分器算法 \mathcal{D} , 下式中的优势在参数 λ 下都是可忽略的。

$$\begin{aligned} \text{Adv}_{q,k,\ell,\eta}^{\text{MLWE}}(\mathcal{D}) := & |\Pr \left[b = 1 : \mathbf{A} \xleftarrow{\$} R_q^{k \times \ell}; \mathbf{s} \xleftarrow{\$} S_\eta^{\ell+k}; \mathbf{t} := [\mathbf{A} \parallel \mathbb{I}_k] \cdot \mathbf{s} \bmod q; b \leftarrow \mathcal{D}(\mathbf{A}, \mathbf{t}) \right] \\ & - \Pr \left[b = 1 : \mathbf{A} \xleftarrow{\$} R_q^{k \times \ell}; \mathbf{s} \xleftarrow{\$} S_\eta^{\ell+k}; \mathbf{t} \xleftarrow{\$} R_q^k; b \leftarrow \mathcal{D}(\mathbf{A}, \mathbf{t}) \right]| \end{aligned} \quad (2.6)$$

注记 在对上述 MSIS 和 MLWE 进行实例化时, 使用二次幂分圆环。令 N 是二的方幂, ζ 为 $2N$ 次本原单位根。 $2N$ 次分圆数域可以表示为 $K := \mathbb{Q}(\zeta) \simeq \mathbb{Q}[X]/(X^N + 1)$, 对应的代数整数环为 $R := \mathbb{Z}[\zeta] \cong \mathbb{Z}[X]/(X^N + 1)$ 。我们定义素数 q 满足 $q = 5 \pmod{8}$ 。令 $R_q := R/qR \cong \mathbb{Z}_q[X]/(X^N + 1)$ 。

定义 26 (Size 条件:) $|u_{i,j}| = \frac{\|v_i \cdot \widehat{v_j}\|}{\|\widehat{v_j}\|^2} \leq \frac{1}{2}, \forall 1 \leq j < i \leq n$ 。

定义 27 (Lovász 条件:) $\|\widehat{v_i} + \mu_{i,i-1} \widehat{v_{i-1}}\|^2 \geq \frac{3}{4} \|\widehat{v_{i-1}}\|^2$ 。

定义 28 (LLL 算法)

通常根据格基正交性的好坏, 即格基中向量的两两垂直程度, 将格基分为“优质基”和“劣质基”。受高斯在二维格中做格基约简的思路启发, Lenstra、LENstra、Lovász 三人将其思路拓展到高维, 并在 1982 年共同提出著名的 LLL 格基约简算法 [27], 任意给定一组格基, 此算法可在多项式时间内将其转化为正交性较好的优质基。

设 $b_1, b_2 \dots b_n \in R^m$ 是格 L 的一组基, 若满足“Size 条件”和参数为 δ 的“Lovász 条件” ($1/4 < \delta \leq 1$) 则称这组有序的基 $(b_1, b_2 \dots b_n)$ 是以 δ 为参数的 LLL 约化基。

2.2.2 多重签名方案 MuSig-L

MuSig-L 多重签名方案 [2] 是一个基于格的需要多方协同生成签名的密码学协议。由于在离线阶段离线信息 off 和状态信息 st 已经完成预算, 所以各方收到待签名的消息后只需要进行一轮在线交互环节, 在各方的输出都通过拒绝采样后, 计算输出最后的聚合签名。

MuSig-L 的具体流程如图2.4所示, 该图以两个参与方为例。具体流程主要分为三个阶段: 公钥聚合阶段、离线阶段和在线阶段。首先是公钥聚合阶段, 在这个阶段参与方 1 和参与方 2 广播各自的公钥, 在本地上计算聚合后的公钥 \tilde{t} 。其次是离线阶段, 在这个阶段参与方 1 和参与方 2 在本地计算各自的离线信息 off 和状态信息 st 。

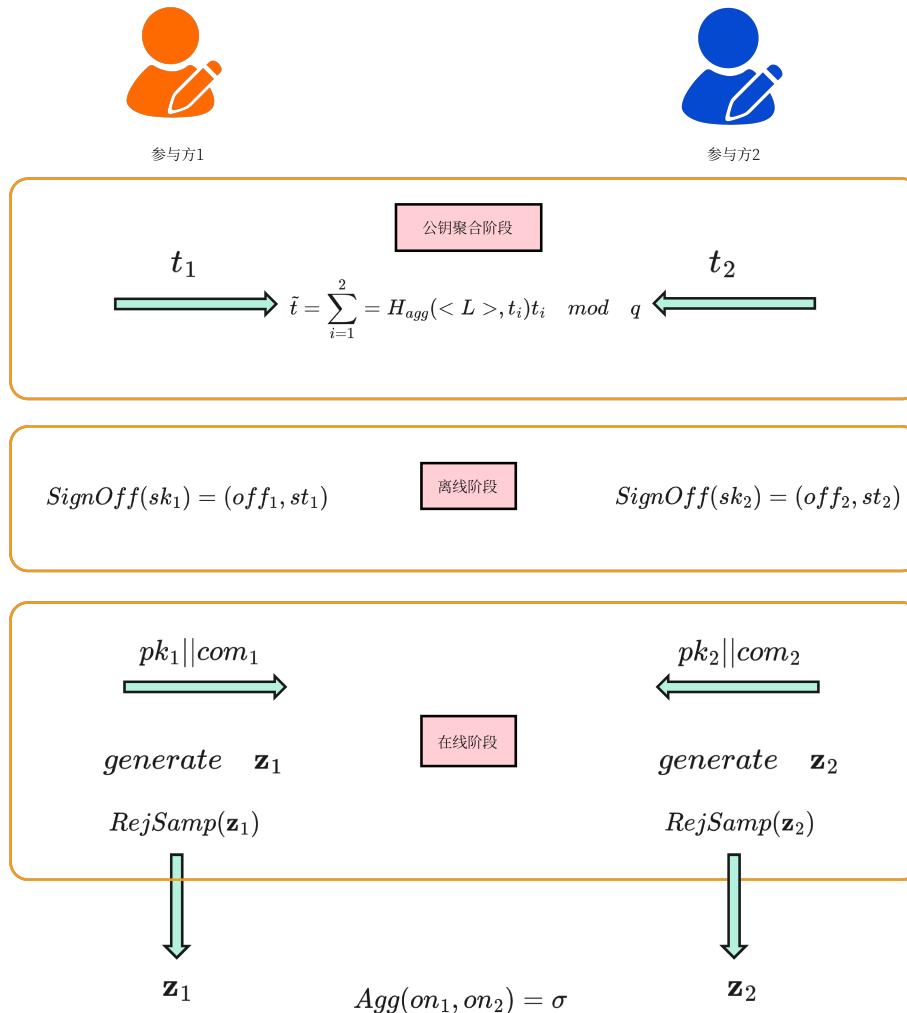


图 2.4 MuSig-L 的基本流程

最后是在线阶段，在这个阶段参与方 1 和参与方 2 广播各自的 $pk_i||com_i$ ，广播各方生成的通过拒绝采样后的签名，然后将各方的签名聚合得到最后的聚合签名 σ 。在这三个阶段中，由于参与方 1 和参与方 2 的公私钥保持不变，因此公钥聚合阶段和离线阶段都能在预算里完成，极大减轻在线交互的工作量，提高了方案的效率。

该方案由 $(\text{Setup}, \text{Gen}, \text{KAgg}, \text{SignOff}, \text{SignOn}, \text{Agg}, \text{Ver})$ 七个算法构成，描述如下：

① $\text{Setup}(1^\lambda)$ 阶段：输出公开参数 pp ，公开参数 pp 作为之后所有函数的隐形输入。

Algorithm 1 初始化算法 $\text{Setup}(1^\lambda)$

- 1: $\mathbf{A} \leftarrow R_q^{k \times l}$
 - 2: $\hat{\mathbf{A}} := [\mathbf{A} | I_k]$
 - 3: $pp := \hat{\mathbf{A}}$
 - 4: **return** pp
-

② 在密钥生成算法 Gen() 中，输出各方的公私钥。

Algorithm 2 密钥生成算法 Gen()

```

1:  $s_1 \leftarrow \mathcal{S}_\eta^{l+k}$ 
2:  $\mathbf{t}_1 := \hat{A}s_1 \mod q$ 
3:  $(pk, sk) := (\mathbf{t}_1, s_1)$ 
4: return  $(pk, sk)$ 
```

③ 在签名聚合算法 Agg(on_1, on_2, \dots, on_n) 中，把所有签名聚合成最终的唯一签名 σ 。

Algorithm 3 签名聚合算法 Agg(on_1, on_2, \dots, on_n)

```

1: if  $\exists i \in [1, n] : \mathbf{z}_i = \perp$  then
2:   return  $\perp$ 
3:  $\mathbf{z} := \sum_{i=1}^n \mathbf{z}_i$ 
4:  $\sigma = (\tilde{\mathbf{w}}, \mathbf{z})$ 
5: return  $\sigma$ 
```

④ 密钥聚合算法 KAgg(L) 把各方公钥作为输入，将其聚合成一个公钥。

Algorithm 4 密钥聚合算法 KAgg(L)

```

1:  $\{\mathbf{t}_1, \dots, \mathbf{t}_n\} := \mathcal{L}$ 
2: for  $i \in [1, n]$  do
3:    $a_i := H_{agg}(<\mathcal{L}>, \mathbf{t}_i)$ 
4:    $\tilde{\mathbf{t}} := \sum_{i=1}^n a_i \mathbf{t}_i \mod q$ 
5: return  $\tilde{\mathbf{t}}$ 
```

⑤ 验签算法 Ver(pk, σ, μ) 以消息、聚合公钥、签名作为输入，输出验签的结果。

Algorithm 5 验签算法 Ver(pk, σ, μ)

```

1:  $(\tilde{w}, \mathbf{z}) := \sigma$ 
2:  $\tilde{t} := pk$ 
3: if  $\bar{A}\mathbf{z} - c\tilde{\mathbf{t}} = \tilde{\mathbf{w}} \mod q \wedge \|\mathbf{z}\|_2 \leq \mathbf{B}_n$  then
4:   return 0
5: else return 0
```

⑥ 离散高斯采样算法 Samp(r): 以 r 作为种子传入，从 \mathcal{D}_{σ_b} 采样。

Algorithm 6 离散高斯采样算法 Samp(r)

```

1: Sample  $b \sim \mathcal{D}_{\sigma_b}$  using randomness  $r$ 
2: return  $b$ 
```

- ⑦ 拒绝采样算法 $\text{RejSamp}(\mathbf{v}, \mathbf{z}, (b^{(j)})_{j \in [m]})$: 判断各方在现阶段生成的签名是否满足要求, 一旦有一方拒绝采样不予通过, $SignOn$ 阶段终止。

Algorithm 7 拒绝采样算法 $\text{RejSamp}(\mathbf{v}, \mathbf{z}, (b^{(j)})_{j \in [m]})$

```

1:  $\sum := (\sigma_1^2 + \sigma_y^2 \sum_{j=2}^m (b^{(j)}) * b^{(j)}) \cdot \mathcal{I}_{l+k}$ 
2:  $\rho \leftarrow \$[0, 1]$ 
3: if  $\rho \geq \min\left(\frac{\mathcal{D}^{l+k}(\mathbf{z})}{M \cdot \mathcal{D}^{l+k} \sqrt{\sum_v}}, 1\right)$  then
4:     return 0
5: return 1

```

- ⑧ 离线签名算法 $\text{SignOff}(\text{sk}_1)$: 各方各自在本地运行该算法, 以用户私钥 sk_1 作为输入, 输出各方离线阶段的预算结果。

Algorithm 8 离线签名算法 $\text{SignOff}(\text{sk}_1)$

```

1:  $s_1 := \text{sk}_1$ 
2:  $\mathbf{y}_1^{(1)} \leftarrow \mathcal{D}_{\sigma_1}^{\ell+k}$ 
3: for  $j \in [2, m]$  do
4:      $\mathbf{y}_1^{(j)} \leftarrow \mathcal{D}_{\sigma_y}^{\ell+k}$ 
5: for  $j \in [1, m]$  do
6:      $\mathbf{w}_1^{(j)} := \bar{\mathbf{A}}\mathbf{y}_1^{(j)} \bmod q$ 
7:  $\text{com}_1 := (\mathbf{w}_1^{(1)}, \dots, \mathbf{w}_1^{(m)})$ 
8:  $\text{off}_1 := (t_1, \text{com}_1)$ 
9:  $\text{st}_1 := (\mathbf{y}_1^{(1)}, \dots, \mathbf{y}_1^{(m)}, \text{com}_1)$ 
10: return ( $\text{off}_1, \text{st}_1$ )

```

- ⑨ 在线签名算法 $\text{SignOn}(\text{st}_1, \text{msgs}, \text{sk}_1, \mu, (\text{pk}_2, \dots, \text{pk}_n))$ 各方交互共同完成对消息的签名。

Algorithm 9 在线签名算法 $\text{SignOn}(\text{st}_1, \text{msgs}, \text{sk}_1, \mu, (\text{pk}_2, \dots, \text{pk}_n))$

```

1:  $(\mathbf{t}_i, \text{com}_i)_{i \in [2, n]} := \text{msgs}$ 
2: if  $\langle (\mathbf{t}_i)_{i \in [2, n]} \rangle \neq \langle (\text{pk}_i)_{i \in [2, n]} \rangle$  then return ⊥
3: if  $\exists i \geq 2 : \mathbf{t}_i = \mathbf{t}_1$  then return ⊥
4:  $L := \{\mathbf{t}_1, \dots, \mathbf{t}_n\}$ 
5:  $a_1 := \mathsf{H}_{\text{agg}}(\langle L \rangle, \mathbf{t}_1)$ 
6:  $\tilde{\mathbf{t}} := \mathsf{KAgg}(L)$ 
7:  $W := \{\mathbf{t}_i || \text{com}_i\}_{i \in [n]}$ 

```

```

8:  $(r^{(j)})_{j \in [2, m]} := \mathsf{H}_{\text{non}}(\langle W \rangle, \mu, \tilde{\mathbf{t}})$ 
9:  $b^{(1)} := 1$ 
10: for  $j \in [2, m]$  do
11:    $b^{(j)} := \text{Samp}(r^{(j)})$ 
12: for  $j \in [1, m]$  do
13:    $\mathbf{w}^{(j)} := \sum_{i=1}^n \mathbf{w}_i^{(j)} \bmod q$ 
14:    $[w_1^{(m)}, \dots, w_k^{(m)}]^T := \mathbf{w}^{(m)}$ 
15:   if  $w_1^{(m)} \notin R_q^\times$  then return  $\perp$ 
16:    $\tilde{\mathbf{w}} := \sum_{j=1}^m b^{(j)} \cdot \mathbf{w}^{(j)} \bmod q$ 
17:    $\tilde{\mathbf{y}}_1 := \sum_{j=1}^m b^{(j)} \cdot \mathbf{y}_1^{(j)}$ 
18:    $c := \mathsf{H}_{\text{sig}}(\tilde{\mathbf{w}}, \mu, \tilde{\mathbf{t}})$ 
19:    $\mathbf{v} := c \cdot a_1 \cdot \mathbf{s}_1$ 
20:    $\mathbf{z}_1 := \mathbf{v} + \tilde{\mathbf{y}}_1$ 
21:   if  $\text{RejSamp}(\mathbf{v}, \mathbf{z}_1, (b^{(j)})_{j \in [m]}) = 0$  then
22:      $\mathbf{z}_1 := \perp$ 
23:    $\text{on}_1 := (\mathbf{z}_1, \tilde{\mathbf{w}})$ 
24: return  $\text{on}_1$ 

```

安全性证明 文献 [2] 中，首先证明了 MuSig-L 方案在敌手没有问询过签名预言机的情况下，敌手获胜的优势，即在 MS-UF-KOA 游戏中获胜的优势为公式2.7。证明思路采用文献 [12] 中提出的”双重分叉技术”将安全性归约至 MSIS 问题和 MLWE 问题的困难性之中。

$$\text{Adv}_{\text{MuSig}-L}^{\text{MS-UF-KOA}}(\mathcal{A}) \leq \frac{Q(2Q+3)}{|C|} + \frac{2^{k+1}}{q^{kN/2}} + \text{Adv}_{q,k,\ell,\ell,\eta}^{\text{MLWE}}(\mathcal{B}') + \sqrt{\frac{Q^2}{|C|} + Q \sqrt{Q \cdot \text{Adv}_{q,k,\ell+1,\beta}^{\text{MSIS}}(\mathcal{D})}} \quad (2.7)$$

进而，证明了敌手在 UF-CMA 实验中，获胜的概率满足公式2.8。

$$\begin{aligned} \text{Adv}_{\text{MuSig}-L}^{\text{MS-UF-CMA}}(\mathcal{X}) &\leq 2(Q_h + Q_s)^2 \cdot \left(\frac{1 + 2^{-\Omega(N)}}{q^{kN}} \right)^m + \frac{(2Q_h + Q_s)^2}{\rho_{\sigma_b}(R)} \\ &\quad + e \cdot (Q_s + 1) \cdot \left(Q_s \cdot \epsilon_s + \text{Adv}_{\text{MuSig}-L}^{\text{MS-UF-KOA}}(\mathcal{A}) \right) \end{aligned} \quad (2.8)$$

2.2.3 基于格的零知识证明

我们所使用的零知识证明需要证明由随机矩阵 $A \in R_q^{n \times m}$ 和像 $t \in R_q^n$ 所定义的单向函数的小型原像。即须完成满足关系 $(A \cdot s = t, \|s\| \leq B)$ 的 s 的证明 [28]，

流程如图2.5：

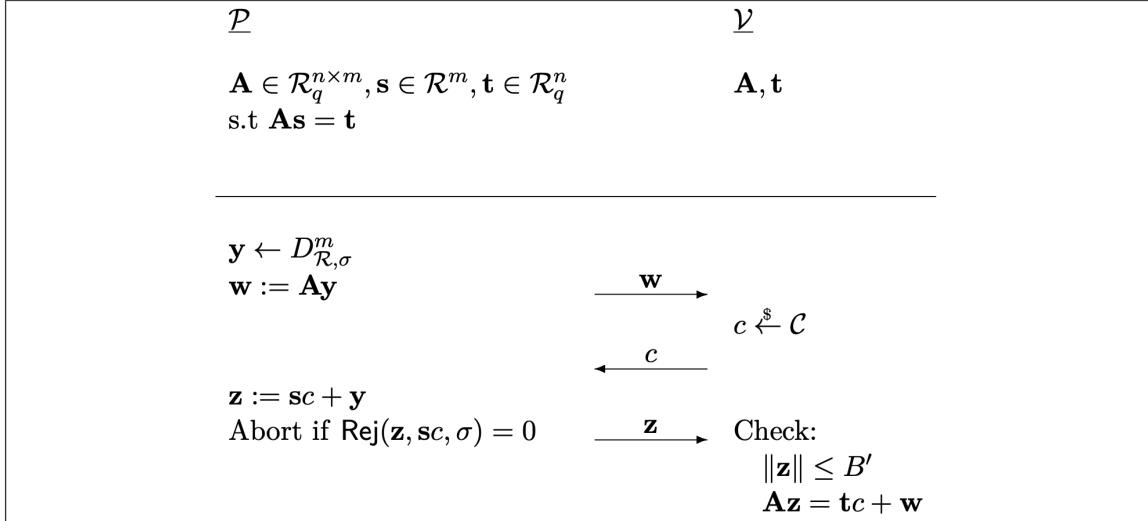


图 2.5 基于格的零知识证明流程

2.2.4 MuSig-L 方案优化

正确性分析 签名方案可能因如下原因被打断：

1. 公钥没有被正确编码： $bad_1 := (<(t_i)_{i \in [2,n]}> \neq <(pk_i)_{i \in [2,n]}>)$ 。
2. 公钥存在碰撞： $bad_2 := (\exists i_1, i_2 \in [1, n] : t_{i_1} = t_{i_2})$ 。由生日悖论可以得到该情况发生可能性 $Pr[bad_2] \leq \frac{n(n-1)}{|S_\eta^k|^2} = \frac{n(n-1)}{\eta^{kN}} \leq 2^{-poly(\lambda)}$ 。
3. 不满足可逆条件： $bad_3 := (\exists i \in [1, n] : \omega_1^{(m)} \notin R_q^\times)$ 。可证明地，此情况发生概率 $Pr[bad_3] = \frac{2}{q^{\frac{N}{2}}} - \frac{1}{q^N} = 2^{-poly(\lambda)}$ 。
4. 未通过拒绝采样： $bad_4 := (\exists i \in [1, n] : RejSamp(v, z_1, (b^{(j)})_{j \in [m]}) = 0)$ 。可证明地，该情况发生概率 $Pr[bad_4] \leq 1 - [\frac{1}{M} + \frac{\epsilon + \delta_2 - \epsilon\delta_2}{M}]^n = 1 - \frac{1}{M^n} + negl(\lambda)$ 。
5. 聚合签名未通过验证： $bad_5 := (Ver(KAgg(L), \mu, \sigma_j) = 0)$ 。在该情况下， $Pr[bad_5] \leq n \cdot 2^{-195}$ 。

综上，该签名方案出错概率 $Pr = \prod_{j=1}^r \sum_{i=1}^5 Pr[bad_i] = (1 - \frac{1}{M^n} + n \cdot 2^{-195} + negl(\lambda))^r$ 。

在软件的具体实现中为了应对以上可能存在的问题，我们对 MuSig-L 方案进行了一定程度的优化

随机预言机实例化 在实例化随机预言机时，我们核心杂凑算法采用国密算法 SM3，主要需要解决的是由 $\{0, 1\}^*$ 到 C 的编码问题。

$$C = \{c \in R : \|c\|_\infty = 1 \& \|c\|_1 = \kappa\} \quad (2.9)$$

我们希望寻求一个合适映射。一个简单的想法为将 C 中的各项有序排列。将 SM3 的结果先映射到整数 $n \in Z_C$ ，然后再映射到 C 中的第 n 个元素。

首先考虑 C 中多项式的个数，即计数 N 次多项式，其中 κ 位为 ± 1 ，其余位为 0。故 $|C| = C_N^\kappa * 2^\kappa$ 。

接着，构造由序号 $Z_{|C|}$ 到 C 的映射，即到 C 中第 n 个多项式的映射 f 。

$$\begin{aligned} f &: \mathbb{Z}_{|C|} &\longrightarrow R \\ n &&\longrightarrow h \end{aligned} \quad (2.10)$$

将 h 看成 N 长 01 序列，若 h 首位为 1 则后续有 $C_{N-1}^{\kappa-1} \times 2^{\kappa-1}$ 种可能，利用 n 与 $C_{N-1}^{\kappa-1} \times 2^{\kappa-1}$ 的大小关系决定 h 的每一个比特，容易验证此映射唯一且为多项式时间算法。至此我们完整定义了 MuSig-L 方案实现上的随机预言机。

在线阶段优化 MuSig-L[2] 中当在线阶段的签名生成失败时将重新进行线下线上阶段的采样，按照本项目所设计的软件流程，将线下预算部分存储于云端，重新进行线下部分的采样意味着重新进行云端的信息更新，这样操作的效率较低。从效率上考虑我们重新设计了算法中的 Hnon 部分，通过引入轮常量 round，对可能造成签名失败的部分引入了额外的随机性，对此我们在签名失败时只需要进行 round 上的改变而不需重新进行 SignOff 阶段，这就大大节省了云端更新的开销，并且我们还可以对不同的 round 进行并行，只需选择第一个能全体通过 SignOn 的 Online 作为最终结果。

2.2.5 国密算法 SM3、SM4

在本方案中的对称加密算法采用国密算法 SM4，哈希函数算法采用国密算法 SM3 进行实例化。

本文中采用的 SM4 算法实现，参照国家标准 GM/T 0002-2012[29]。SM4 分组密码算法是一个迭代分组密码算法，由加解密算法和密钥扩展算法组成。SM4 分组密码算法采用非平衡 Feistel 结构 [30]，分组长度为 128 比特，密钥长度为 128 比特。加密算法与密钥扩展算法均采用 32 轮非线性迭代结构。加密运算和解密运算的算法结构相同，解密运算的轮密钥的使用顺序与加密运算相反。其加密算法流程如图2.6所示，其中轮函数的示意图如图2.7所示。

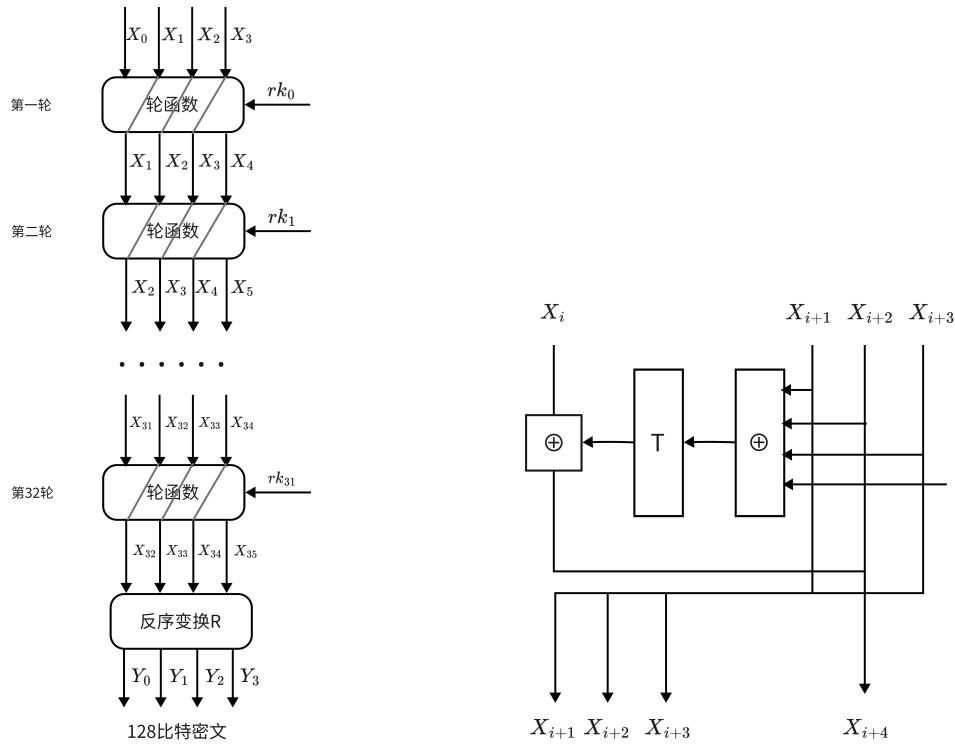


图 2.6 SM4 加密流程图

图 2.7 SM4 轮函数流程图

本项目中的哈希函数采用 SM3 算法与自主设计的编码函数实例化。2004 年，王小云院士提出的比特追踪法成功对 MD5、SHA-0 和 SHA-1 算法实施了碰撞攻击 [31]。因此，在本方案中，我们使用更加安全可靠的由我国自主研发的国密哈希函数算法——SM3。代码实现时参照中国国家密码管理局 2010 年公布的中国商用密码杂凑算法标准 [32]。SM3 算法采用 Merkle-Damgard 结构 [33]，与 SHA-256 的压缩函数具有相似的结构，但是 SM3 算法的设计更加复杂。

2.3 MuLPay 使用流程及其功能

2.3.1 登录

登录时我们需要进行公私钥匹配来完成登录功能，由于检查机制在服务器，为了不泄漏私钥信息，我们采用了基于格的零知识证明算法，通过完成远程服务器的挑战进行验证。远程的检查机制相较于本地检查机制，其无法通过修改本地数据进行登录绕过，具有更好的安全性。

2.3.2 注册

注册时，用户自定义一个强口令发送给本地客户端。本地客户端随机生成助记词 $\{W_i\}_{i \in [12]}$ ，将助记词哈希得到 $seed$ ， $seed$ 作为种子随机生成用户的签名私钥 sk ，进而生成验签公钥 pk 。将口令的哈希值作为密钥，调用 SM4 算法加密公私钥对，并将密文存至本地。取公钥的前 80 位作为钱包的地址。本地客户端将钱包地址、用户公钥和预计算结果（其中，预计算结果均分为两份，一份使用明文形式存储，另一份则使用加密后的密文形式存储）上传至服务器，将助记词返回给用户。用户需将助记词妥善保存。

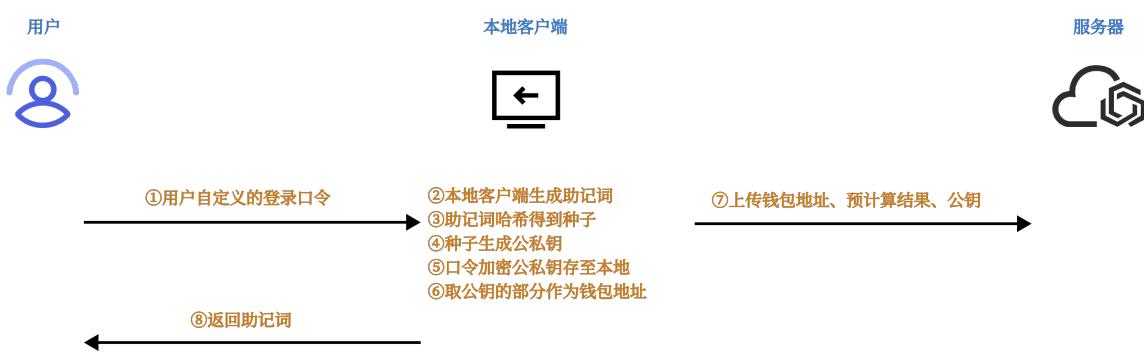


图 2.8 注册流程图

2.3.3 多人签署交易

若用户群组不为空，则用户在发起一笔交易时，执行流程如2.9所示，执行 MuSig-L 多重签名协议完成交易的签署。

用户在本地客户端填写目标交易地址和转账金额，发起一笔交易。客户端向服务器数据库查询，验证用户余额是否充足。若用户余额充足，服务器将该交易内容、所有组用户的公钥及第一轮离线签名预计算结果打包至各组用户的待验证交易池中，并将该交易上传至全局的待处理交易池。其他组用户将会在待签署交易列表中看到这笔交易，签署交易后，将在线签名所得结果发送给服务器。服务器将签名添加到待验证交易池中，检查签名数目是否达到阈值。若达到阈值，将签名聚合并验证签名的合法性。若通过验证，将交易及信息打包上链。

2.3.4 单人签名

若用户群组为空，则用户在发起一笔交易时，执行流程如2.10所示。用户使用本地客户端发起一笔交易，本地客户端生成签名并将交易内容和签名发送给服务器。服

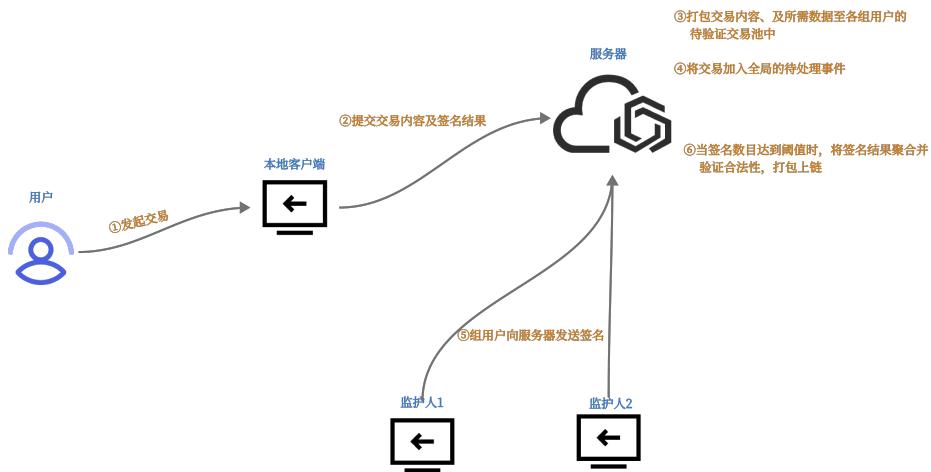


图 2.9 多人签署交易流程图

务器验证签名合法性通过后，将交易打包上链。

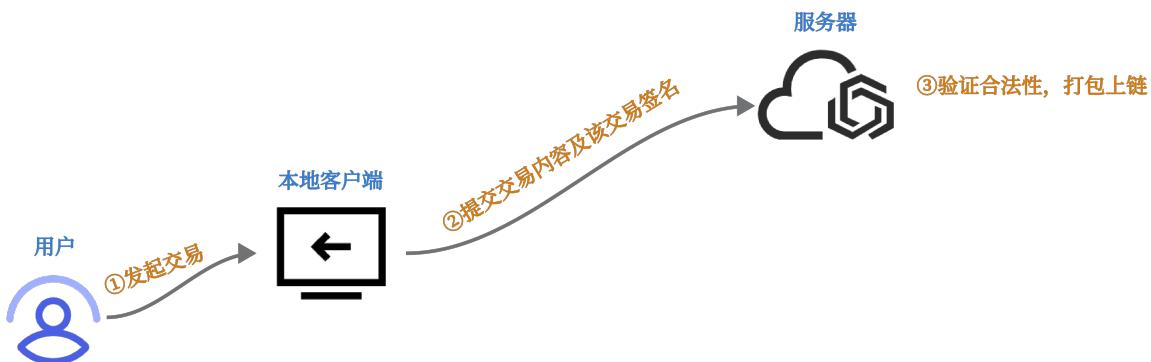


图 2.10 单人签署交易流程图

2.3.5 钱包的社交恢复

若用户忘记口令，可以填入助记词恢复公私钥。但是，若用户同时忘记口令和助记词，则可以使用社交恢复功能，发起账户丢失请求，其详细流程如2.11所示。

当服务器接收到用户的账户丢失请求，查找用户组成员，向他们发送随机文件，用户组成员登录账户提交文件的签名返回给服务器。服务器将随机文件和签名交由智能合约验证合法性，若验证通过，则提取合约中的余额。

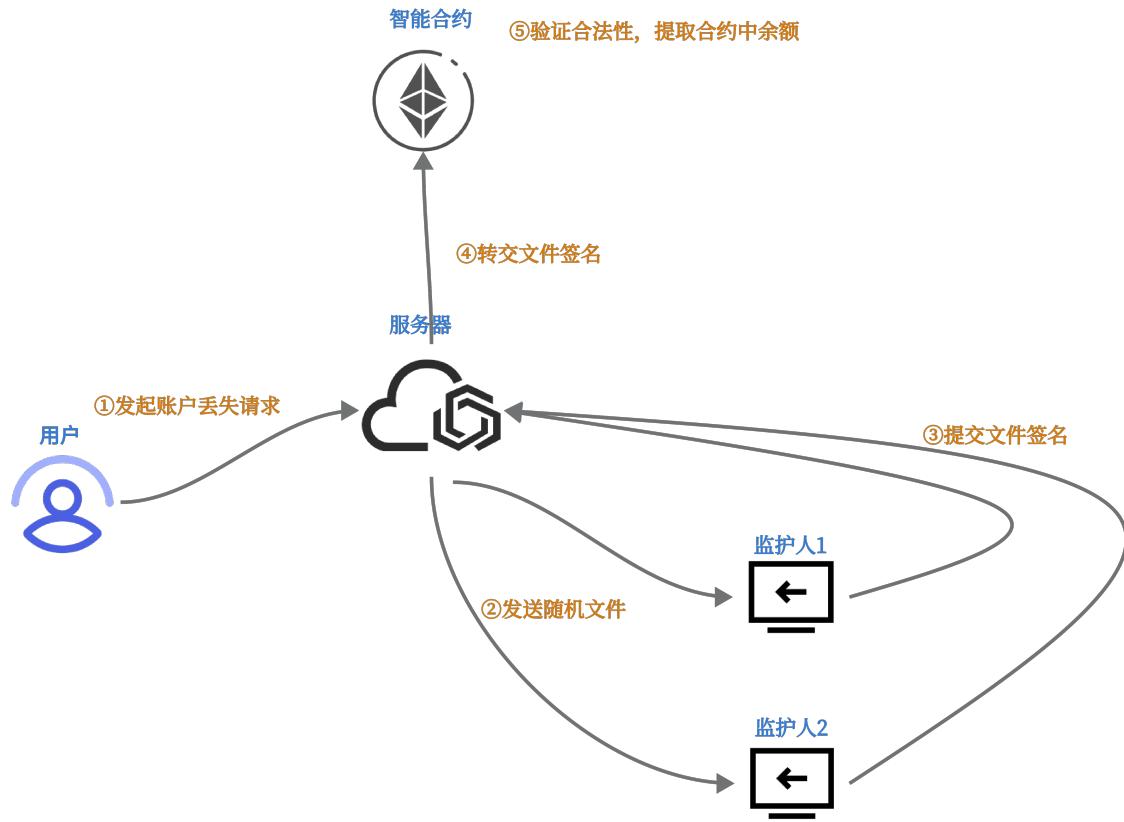


图 2.11 社交恢复钱包流程图

2.3.6 修改登录口令

当用户修改登录口令时，执行如2.12的流程。用户登录本地客户端，输入助记词与新口令。客户端将助记词哈希得到种子 $seed$ ，以 $seed$ 为随机数生成器的种子生成公私钥对，然后将新的口令作为对称加密密钥加密公私钥对存至本地。

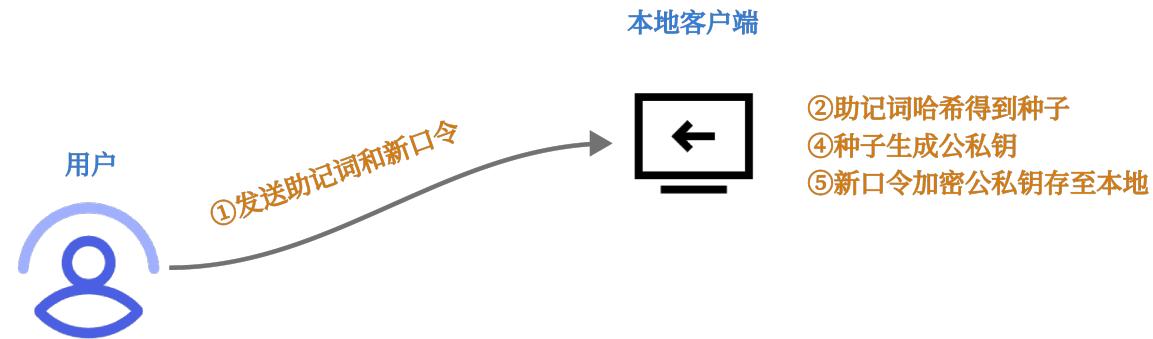


图 2.12 修改登录口令流程图

2.4 评价指标

2.4.1 效率

我们在方案的设计和证明上，都对 MuSig-L 方案进行了优化和改进。

1. 在方案设计方面，本项目底层采用的 MuSig-L 多重签名方案分成一轮线上计算和一轮线下计算。线下预算算的引入减少了交互轮数和通信复杂度。
2. 由于方案 MuSig-L 中签名方以一定的概率会拒绝采样失败，若一方签名失败，则各方均要重新执行离线签名和在线签名算法，对多重签名的执行效率带来了极大的影响。针对这一问题，我们提出了两个优化：
 - ① 工程上，采用多线程技术，使签名的各方同时并发地执行多个签名的实例。
 - ② 理论上，我们往在线阶段引入了新的随机量，使得当签名生成失败时，各方只需执行在线阶段的协议，节省了离线阶段的开销
3. 在方案实现方面，对 SM4 算法使用 Cuda 编程，实现 GPU-CPU 异构计算的硬件加速；对 MuSig-L 多重签名算法，实现多线程并行计算优化。

2.4.2 安全性说明

钱包私钥本地化管理 MuLPay 区块链钱包用助记词哈希的结果得到种子，再用种子生成钱包私钥，私钥派生出公钥。用登录口令的哈希值做对称密钥加密，公私钥对的加密结果存储在本地。

从使用体验的角度，MuLPay 为用户提供了助记词，简化操作的同时不丢失安全性。

从私钥管理的角度，MuLPay 钱包登录基于一个本地存储的文件，这个本地文件是一个经过对称加密后的公私钥文件，需要使用登录口令才能解密。从这个角度而言，MuLPay 创造性地完成私钥本地化存储的实践，用户无需把自己的私钥交给云服务器，提高了安全性。

此外，由于一笔交易需要多人签署，即使有管理员的私钥被盗，只要不超过数量 N，资金仍然是安全的。

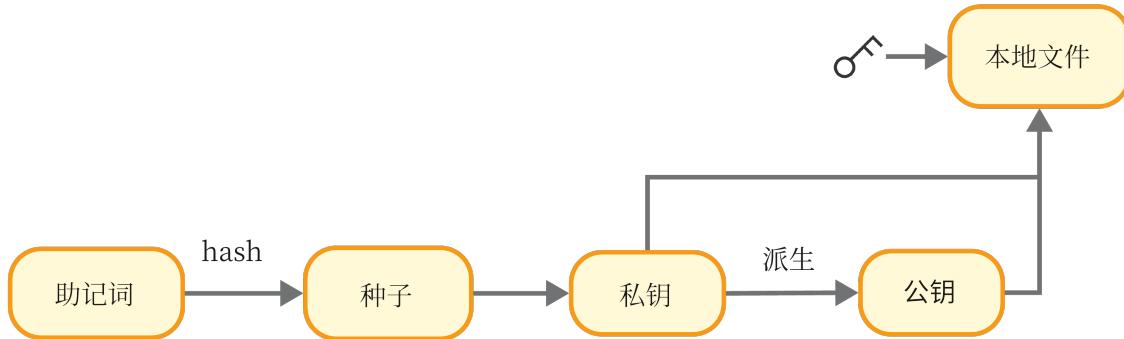


图 2.13 私钥本地管理

算法安全性说明：我们实现的 MuSig-L 在 PPK 和并发查询安全模型下，基于 MSIS、MLWE 假设，可以证明在选择消息攻击下存在不可伪造。具体的安全性证明见 2.2.2 小节。并且，该方案基于格上困难问题，还能够满足抗量子攻击的特性。

方案使用的其他密码算法均为国密算法，其中哈希函数使用 SM3 算法，对称加密算法使用 SM4 算法。国密算法由我国自主研发，安全可控，更适合我国国情和法规。

2.5 GUI 展示

2.5.1 预备环境

在运行该作品之前，需要安装 Sagemath 9.2、Python 3.9.7 版本与 CUDA 12.0.1 及以上版本，其中 Python 环境需要搭配 Flask 框架与 Pycryptodome 库运行，并可选定在 Windows 系统下运行。

安装好前置环境之后，进入作品文件夹，在命令行下输入 `sage app.sage` 命令，在浏览器下进入 `http://127.0.0.1:5000/` 网址即可。

2.5.2 功能

1. 欢迎界面

访问根目录时，若本地无账户文件跳转至欢迎界面：

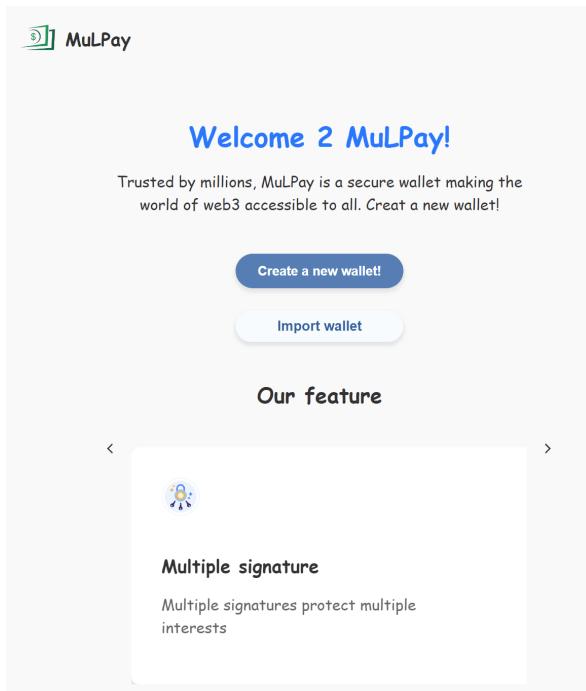


图 2.14 欢迎界面

2. 创建一个新的钱包设置登录口令，后端将利用该口令加密公私钥文件，存储在本地。

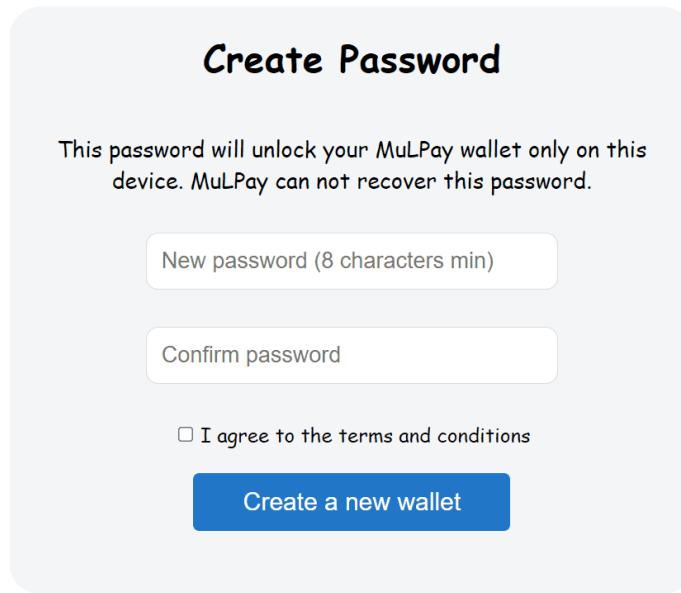


图 2.15 创建新钱包

3. 助记词界面后端随机产生助记词，并根据助记词生成公私钥，对口令哈希后结果作为密钥加密公私钥文件，存储于本地。

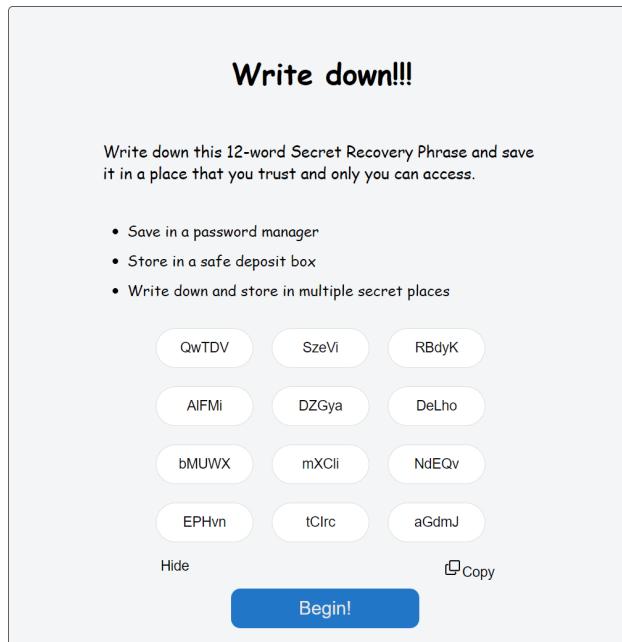


图 2.16 助记词

4. 登录界面输入口令登录，后端解密本地文件获取公私钥信息，并对公私钥正确性进行验证，通过后登录至公钥所对应的地址账户。

Welcome 2 MuLPay!

Password

login

Forget the password?

图 2.17 登陆界面

5. 用户界面

- +: 未设置
- →: 发起交易
- Transaction History: 交易历史
- Group List: 辅助验证成员列表
- Pending Transaction: 待验证交易

后端实现的效果，若 Group List 中有成员，交易不立即结算，存储于数据库等待组内其他成员签名，并将交易信息发送给 Group List 中的所有成员，若 Group List 中没有成员，交易可直接上链。

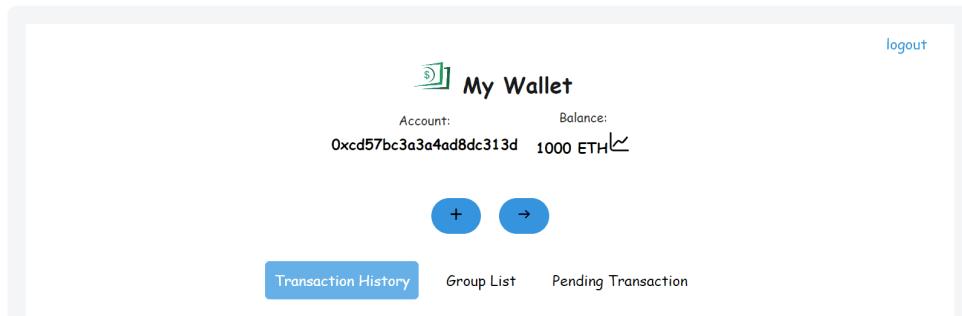


图 2.18 用户界面

组内成员管理如下：

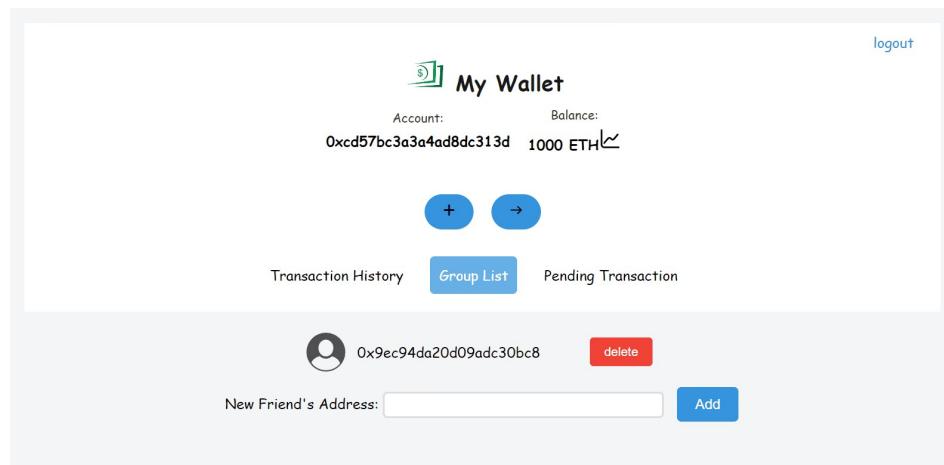


图 2.19 群组成员管理

6. 交易界面如图2.20所示：

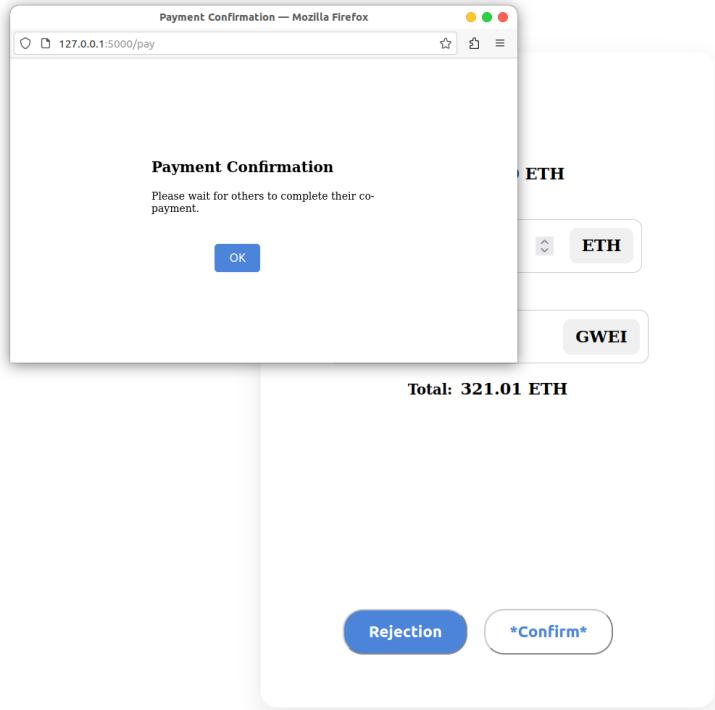


图 2.20 交易界面

7. 重置口令界面如图2.21所示：

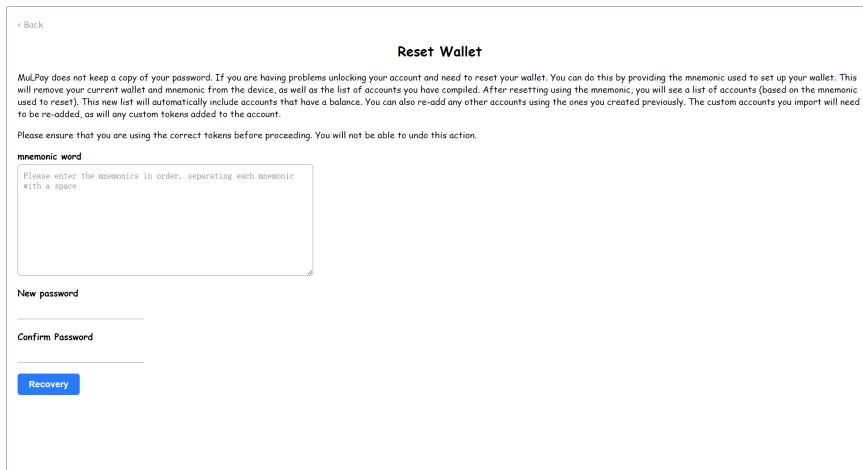


图 2.21 重置口令界面

2.5.3 多重签名实例

下面对一个具体应用场景进行描述，在这个例子中假设某公司使用一个公司账户进行公司总资产的管理

该场景共涉及三个账户，公司公共账户 C，公司股东账户 S1，S2

公司账户 C 如图2.22所示：

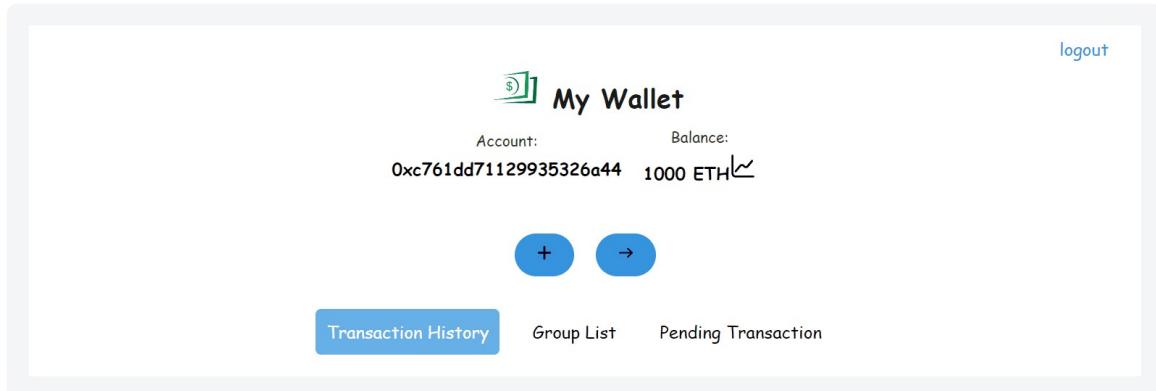


图 2.22 公司账户 C 的用户界面

股东账户 S1 如图2.23所示：

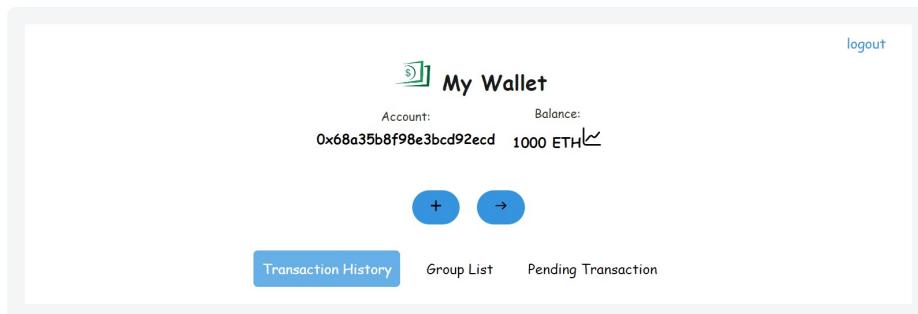


图 2.23 股东账户 S1 的用户界面

股东账户 S2 如图2.24所示：

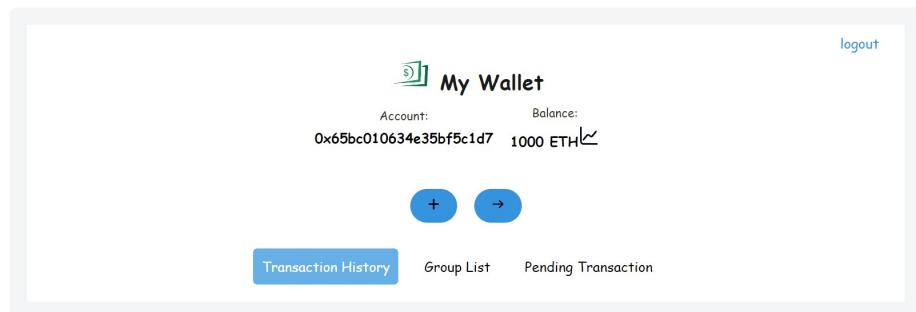


图 2.24 股东账户 S2 的用户界面

在公司账户 C 下添加股东账户 S1, S2, 如图2.25所示:

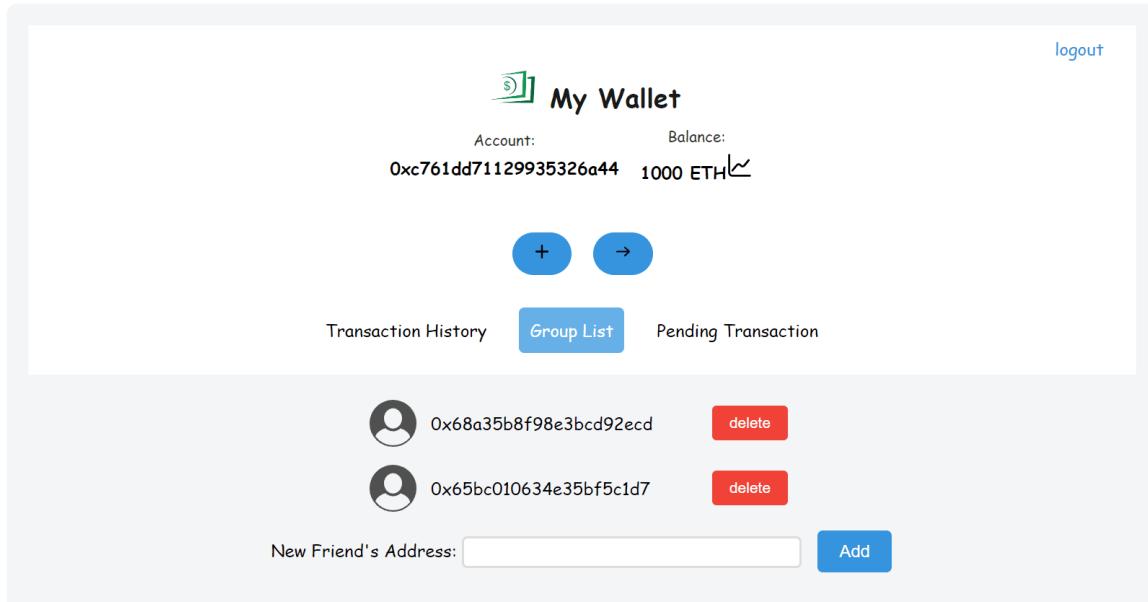


图 2.25 群组成员管理

当公司需要进行一笔重要投资时，比如建立新的生产线或者购买新的设备，该决策需要得到两个股东账户的共同授权。公司发起了该交易，如图2.26所示：

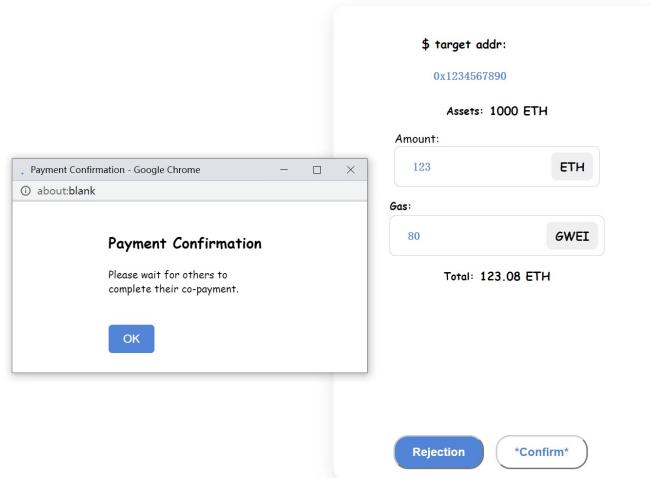


图 2.26 交易界面

股东 1 的带签署列表中将出现这笔交易，若股东 1 支持该交易则利用其股东账户 S1 对该交易进行签名，股东 2 同理。

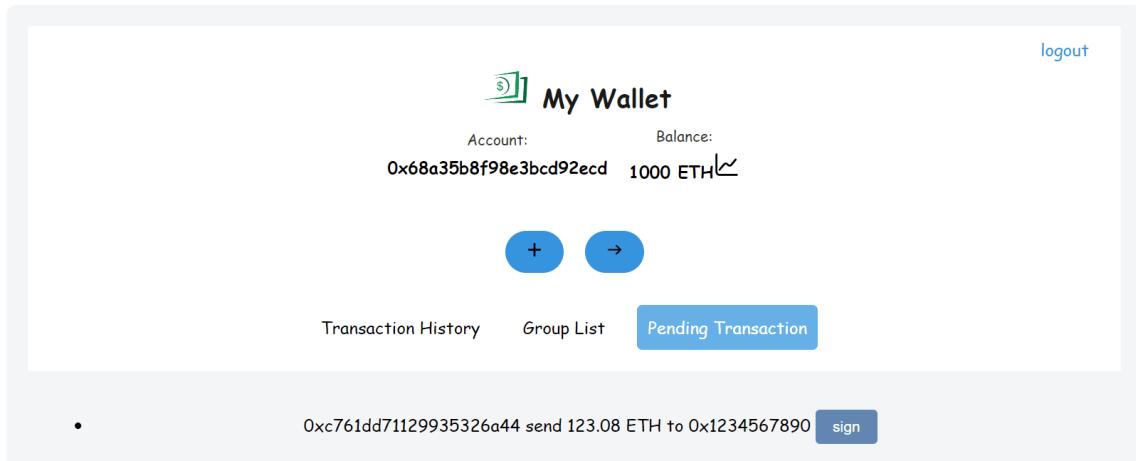


图 2.27 股东 S 签名界面

所有待签用户完成签名后，即两个股东均支持了此交易时，后台打印已完成交易消息，后续是交易具体上链的过程：

```
transaction [0xc761dd71129935326a44 sent 123.08 ETH to 0x1234567890] has on the chain
127.0.0.1 - - [24/May/2023 21:44:56] "POST /mulsign HTTP/1.1" 302 -
127.0.0.1 - - [24/May/2023 21:44:56] "GET /usr_state HTTP/1.1" 200 -
```

图 2.28 控制台输出

此时，待签事件在服务器的待签池中清除，同时公司账户 C 进行余额的扣除。

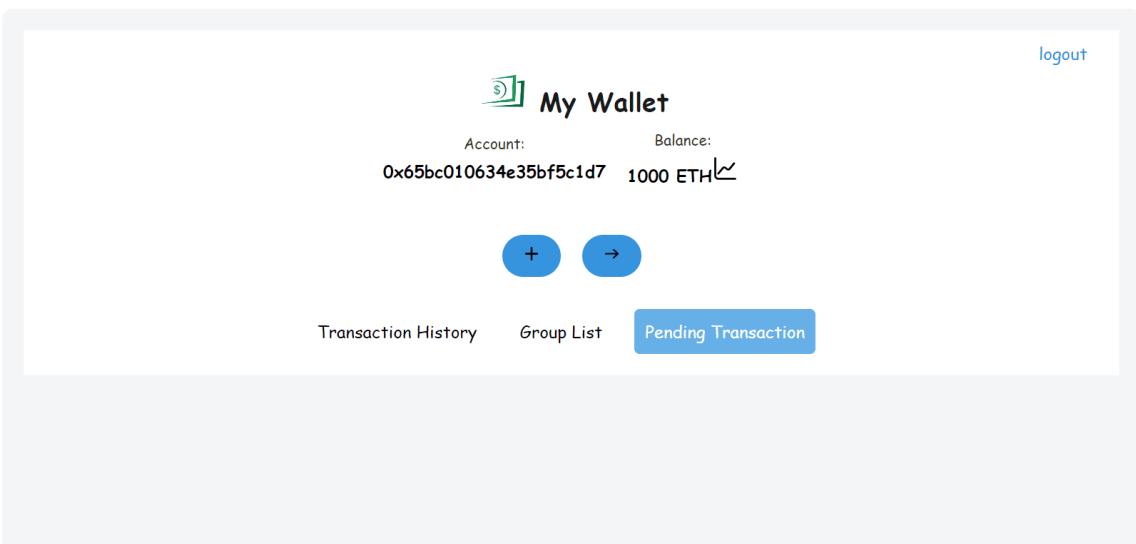


图 2.29 待签署交易列表

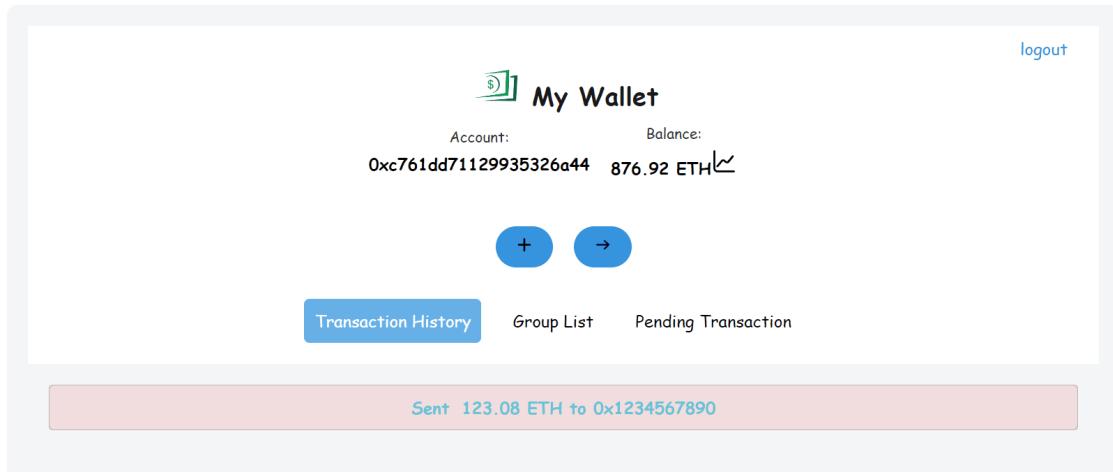


图 2.30 公司账户 C 的用户界面

第三章 作品测试与分析

在本章节中，主要介绍本作品各关键部分的性能测试结果。3.1节介绍了测试数据时采用的测试设备。3.2节简要描述 MuSig-L 多重签名方案的性能，并通过不同实现版本的测试效率展现本作品对于 MuSig-L 实现过程的逐步优化。3.3节详细介绍了利用 GPU 硬件加速 SM4 对称加密算法的过程和优化结果，与现有商密软件工具相比有较大提升。最后的3.4节简要描述了本作品相较于其它钱包的优势所在。

3.1 测试设备

测试中使用的硬件环境配置如表3.1所示。

表 3.1 硬件环境配置

类型	参数
CPU 型号	AMD Ryzen 7 5800H with Radeon Graphics
CPU 架构	x86 架构
CPU 主频	3.20GHz (基准速度)
CPU 逻辑处理器数	16
CPU 核心数	8
内存	Hynix 8G × 2 DDR4, 3200MHz
GPU	NVIDIA GeForce RTX 3060 Laptop GPU

3.2 MuSig-L 性能测试

所选取的参数标准参考了美国国家标准与技术研究院(NIST)所推行的 CRYSTALS-Dilithium 签名参数标准 [34]，其关键参数如表3.2所示：

表 3.2 MuSig-L 所应用的关键参数标准

Parameters	Parameters Description
N (the degree of $f(X)$)	256
$f(X)$	$X^N + 1$
$\lceil \log_2(q) \rceil$	23 and $q \equiv 5 \pmod{8}$

κ

60 ± 1's and 196 0's

相应地，MuSig-L 方案各阶段的测试时间如表3.3所示：

表 3.3 MuSig-L 方案各阶段的时间开销

	线下签名阶段	线上签名阶段	密钥聚合阶段	验证签名阶段
时间 / ms	100 ~ 110	165 ~ 175	20 ~ 30	45 ~ 55

在具体应用 Sagemath 开源数学语言实现 MuSig-L 基础方案时，按照时间顺序总共产生了 3 个代码版本，其内容主要是对线下签名与线上签名两个阶段逐步做了速度上的优化。

3.2.1 版本 1

朴素版的 MuSig-L 方案实现，未应用任何优化技术。

3.2.2 版本 2

将 MuSig-L 方案包装为 Class 对象，实现面向对象的编程，并添加多线程技术，使得每个用户内部生成线下签名的过程实现并行计算。

具体地，SiganOff 阶段的时间开销主要来自于 y_i^j 的多次高斯采样过程，因而在此处插入了多线程进行并行化的高斯采样，减少了串行采样所带来的时间开销。

3.2.3 版本 3

继续添加多线程技术，使得每个用户内部生成签名的线上部分实现并行计算。

具体地，SiganOn 阶段的时间开销也有一部分来自于 b^j 的多次高斯采样过程，因而仍在此处插入了多线程进行并行化的高斯采样，减少了串行采样所带来的时间开销。

3.2.4 各版本性能对比

各实现版本的运行时间与速度如图3.1所示。

可以看出，线下部分与线上部分所消耗的时间资源整体呈下降趋势，实现了性能上的优化。线上签名加速了 252%，线下签名加速了 218%。

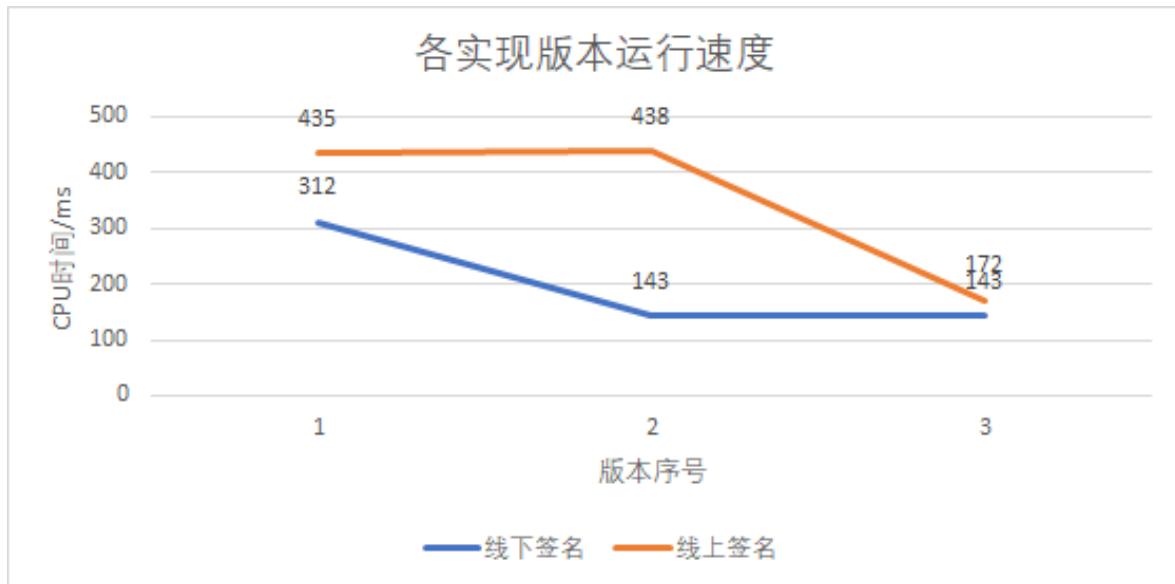


图 3.1 MuSig-L 各版本性能

3.3 SM4 加速

在实现本作品时应用到了国密对称密码算法——SM4，并为了达到优化 SM4 加解密性能的目的，采用了 CUDA 语言在 GPU 硬件上对 SM4 算法进行加速。为了充分发挥 GPU 的高并发性，选取的分组模式为 CTR 模式，因为在此模式下的加解密过程均可以做到并行化实现。

在利用 CTR 模式本身的并行性之后，还要考虑 CUDA 语言本身的特点，这些语言特性的考量同样会影响到 SM4 算法的执行速度。其具体的特性包括：

1. 存储器层次的选择和使用：GPU 内部也存在着类似于 CPU 寄存器至外存的存储器结构层次，执行速度由快到慢可以依次划分为：寄存器；共享存储器；本地存储器；全局存储器；常量存储器与纹理存储器。SM4 算法中存在着对于 S 盒的频繁访问，若 S 盒放置于 DRAM 或 GPU 外部的其它存储器层次上，会导致 GPU 与外部存储器的频繁通信，降低执行效率，因而在具体实现 SM4 加速时，应将 S 盒这类的常量数据放置于常量存储区中，减少 GPU 访存的延迟。
2. 线程束的分配：目前 CUDA 所规定的一个线程束大小为 32 个线程，每次线程调度时以一个线程束作为单位，因而应该保证线程块中的线程数目为 32 的整数倍，保证每一次线程调度后的 GPU 能够充分利用核心资源。
3. 线程块大小与线程数目的选择：一个线程块占用了一个 SM，因而因尽量使得一个线程块中的线程数目足够多，做到充分利用 SM 的核心，与此同时，线程束分配部分的所述，还应尽量保证一个线程块中的线程数目为 32 的整数倍，保证每个

核心的计算资源得到充分分配。

3.3.1 SM4 加速比测量

目前，传统 CPU SIMD 的 SM4 实现加解密速率达到了 1524.8Mbps[3]，而商密软件栈 SIG 则最高到达了 4364Mbps，本作品所实现的 SM4 在充分考虑 GPU 核心资源与访存速度的情况下，加解密速率达到了 10480Mbps。

表 3.4 SM4 加解密速率

	传统 CPU SIMD 实现 [3]	商密软件栈 SIG	GPU 硬件实现
加解密速率 / Mbps	1524.8	4364	10480

可见，GPU 硬件最终达到的加速效果相较于传统的 CPU SIMD 实现 [3] 提升了 587.30%，而相较于商密软件栈 SIG 的实现效率提升了 140.15%。

3.4 对比其他钱包

表 3.5 各区块链钱包对比图

	MetaMask[7]	imToken[8]	HyperPay	Binance	MuLPay
是否使用助记词	√	√	√	√	√
是否支持共同支付	×	√	√	√	√
基于困难问题	DLP	DLP	DLP	DLP	<small>MSIS MLWE</small>
使用的多方签名技术	×	MuSig	TSS	TSS	MuSig-L

第四章 创新性说明

本作品创新性地提出将格密码与格上签名应用于区块链相关技术中，在量子计算机的时代到来之前为区块链相关应用实现抗量子攻击的升级。

本作品的主要创新点包括以下几个方面：改进了 MuSig-L 方案的拒绝采样；将传统的基于离散对数的以太坊多重签名方案转换成基于格上的困难问题；通过基于格的 ZK-Proof 完成远程验证的功能；具体实现中，采用了多线程技术来加速每个用户的内部签名生成；与其他区块链钱包相比，有着更广阔的应用前景。

改进 MuSig-L 方案 针对 MuSig-L 在 SignOn 阶段可能出现的一方拒绝采样不通过导致的整个 SignOn 失败的情况，我们在 SignOn 阶段引入随机量，客户端同时运行多个 SignOn 程序，取通过的第一个版本作为最后的 SignOn 结果。这样的改进能极大减少 SignOn 拒绝采样不通过的概率。

格上困难问题的引入 现在以太坊中的签名方案普遍采用基于 ECDLP 的 MuSig-/MuSig2 方案，因 Shor 攻击算法的存在 [1]，此问题无法抵抗量子计算机的攻击。而上述的作品实现已说明 MuSig-L 方案可以规约到 MSIS 问题上，而 MSIS 问题目前还无有效的量子攻击方案，能够在量子计算的时代提供安全性的保障。

基于格的零知识证明 对于用户登录而言，这里采用了基于格的零知识证明流程，由于其与 Musig-L 的签名方案思路一致，在实现上可以共用一部分组件。并且远程的验证相较于本地验证具有更好的安全性和鲁棒性，使得钱包登录过程更加健壮而可靠。

多线程技术加速 参与多重签名的每个用户在生成各自的线下签名或线上签名时，存在着各种操作，彼此之间并无依赖关系，可以并行化处理，以多项式采样为例，采出的多项式彼此之间并无关系，可以在此处插入多线程部分。同时进行多个多项式的采样，实现加速的目的。

GPU 硬件加速 SM4 执行 为了实现自主可控，在对称加密部分引入了 SM4 国密算法，并且为了加速 SM4 的执行速度，利用 GPU 硬件的高并发性重新实现了 SM4 算法，加解密速率已做到比商密算法实现更快的效果。

全周期的后量子安全区块链钱包 本作品方案中使用的多重签名方案是基于格上困难问题的，具备抗量子特性。除此之外，本作品方案中相关的的哈希函数和分组密码

在实际实现时也分别选用自主可控的国密算法 SM3 和 SM4，在实现产品自主可控的同时能够抵御量子攻击。由于查阅相关资料发现暂时没有公开的密钥长度为 256 比特的国产分组密码算法，为了抵抗 Grover 攻击 [35] 所带来的对称密码密钥长度减半问题，我们在实际实现的时候类比于 3DES 算法，采用了 3SM4 的方法，以此来提高实际使用时的安全强度，但此情况下的具体量子安全性强度还待未来的进一步分析结论。

对比其他钱包 总体而言，我们的 MuLPay 有以下的优势：

1. 从签名基于的困难性问题及上述创新点来看，MuLPay 具有抗量子性。目前市面上的区块链钱包是基于 *DLP* 困难问题的，不具备抗量子性。而我们实现的 MuLPay 采用了基于格的多重签名方案，该多重签名方案可以在保证通信安全的前提下，让多个用户共同完成数字签名，并满足四个安全性质。
2. 从私钥存储的角度来看，MuLPay 创造性地完成私钥本地化存储的实践。钱包登录基于一个本地存储的文件，这个本地文件是一个经过对称加密后的存储了公私钥的文件，需要使用自己的登录口令才能解密。从这个角度而言，用户无需把自己的私钥交付给云服务器，提高了安全性。
3. 从应用前景来看，MuLPay 易于扩展，能够管理多种数字货币，未来可以根据需求灵活升级和扩展上。随着格理论的发展，MuLPay 签名大小可以进一步压缩 [23]，除此之外，抗量子区块链系统 [24] 的发展也使得进一步压缩 MuSig-L 的签名大小成为可能。

第五章 总结

从区块链钱包的发展现状来看，它是保护数字资产安全的重要工具，广泛应用于加密货币交易、数字资产发行与管理、身份验证和数字身份管理等领域。但随着多项式时间量子攻击 Shor 算法的提出 [1] 及量子计算的不断发展 [22]，如何抵御量子计算的攻击成为区块链钱包等加密应用的一个紧迫问题。在此背景下，基于格上困难问题的多重签名方案，因其能抵抗量子攻击、实现简单高效、拥有困难情况到一般情况的归约等一系列优点，逐渐进入人们的视野。

为了使区块链钱包达到抗量子攻击的目的，本项目实现的区块链钱包采用了基于格上困难问题的多重签名方案。具体而言，本项目改进了 Cecilia Boschini 等人提出的 MuSig-L 多重签名方案 [2] 并将其创造性地应用于区块链钱包应用中，基于改进后的方案开发了一款名为 MuLPay 的区块链钱包。

从本项目产品的设计来看，MuLPay 应用了后量子安全的、优化后的 MuSig-L 多重签名方案，在原方案的基础上取得了一定的提速，并使用我国自主研发的哈希算法 SM3 和对称加密算法 SM4，减少了对于国外技术的过度依赖，避免了未知的安全隐患，在完成保密目标的同时实现了技术的自主可控。

从本项目产品的功能来看，MuLPay 具有单人管理、共同支付、社交恢复等功能，实现了私钥本地化存储。在具备安全性的同时，我们还注重了用户体验的优化。钱包的界面友好且操作简便，用户能够通过其灵活、高效、安全的特点轻松管理数字资产，实现多人监管，并保证资产的安全性。

从本项目的创新性来看，本产品是基于格的区块链钱包，具备抗量子攻击的特点，实现简单高效；改进后的 MuSig-L 多重签名方案在可能发生拒绝采样失败的情况下，签名的效率更高；此外，MuSig-L 算法实现采用多线程优化，线上签名阶段相较于朴素实现提升了 252%，线下签名阶段提升了 218%；用户私钥的本地化存储且登录时通过零知识证明来验证公私钥匹配，可保证账号的安全性；在实现中加速了产品中使用的国密算法 SM4，采用了国密算法 SM4 与 SM3，通过异构计算的手段，将 SM4 算法置于 GPU 之上，利用 GPU 硬件的高并发性，充分考虑 GPU 硬件各访存层次的因素，最终达到的加解密速率相较于传统的 CPU SIMD 实现 [3] 提升了 587.30%。

从本项目的发展前景来看，MuLPay 具备良好的开放性和可扩展性，未来可以根据需求灵活升级和扩展，满足不断变化的区块链应用需求。在量子计算机快速发展的今天，MuLPay 所采用的基于格上多重签名方案的设计理念具有良好的前瞻性，为未来基于格的区块链应用提供了有效的参考和借鉴。

参考文献

- [1] SHOR P W. Algorithms for quantum computation: Discrete logarithms and factoring [C/OL]//35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994. IEEE Computer Society, 1994: 124-134. <https://doi.org/10.1109/SFCS.1994.365700>.
- [2] BOSCHINI C, TAKAHASHI A, TIBOUCHI M. Musig-l: Lattice-based multi-signature with single-round online phase[J/OL]. IACR Cryptol. ePrint Arch., 2022: 1036. <https://eprint.iacr.org/2022/1036>.
- [3] 郎欢, 张蕾, 吴文玲. SM4 的快速软件实现技术[J]. 中国科学院大学学报, 2018, 35 (180-187).
- [4] SIDHU J. Syscoin: A peer-to-peer electronic cash system with blockchain-based services for e-business[C/OL]//26th International Conference on Computer Communication and Networks, ICCCN 2017, Vancouver, BC, Canada, July 31 - Aug. 3, 2017. IEEE, 2017: 1-6. <https://doi.org/10.1109/ICCCN.2017.8038518>.
- [5] BUTERIN V. Ethereum: A next-generation cryptocurrency and decentralized application platform[EB/OL]. 2013[2022-06-06]. <https://ethereum.org/ethereum.html>.
- [6] NUR. Behind the scene on how myetherwallet works (simple illustration)[EB/OL]. 2018. <https://steemit.com/ethereum/@n-nur/behind-the-scene-on-how-myetherwallet-works-simple-illustration>.
- [7] LEE W M. Using the metamask crypto-wallet[M/OL]. Berkeley, CA: Apress, 2023: 111-144. https://doi.org/10.1007/978-1-4842-9271-6_5.
- [8] QI M, XU Z, JIAO T, et al. A comparative study on the security of cryptocurrency wallets in android system[C/OL]//2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom). 2022: 399-406. DOI: 10.1109/TrustCom56396.2022.00062.
- [9] DIEM C. On the discrete logarithm problem in elliptic curves[J/OL]. Compositio Mathematica, 2011, 147(1): 75–104. DOI: 10.1112/S0010437X10005075.

- [10] SHAMIR A. How to share a secret[J/OL]. Commun. ACM, 1979, 22(11): 612-613.
<https://doi.org/10.1145/359168.359176>.
- [11] MAXWELL G, POELSTRA A, SEURIN Y, et al. Simple schnorr multi-signatures with applications to bitcoin[J/OL]. IACR Cryptol. ePrint Arch., 2018: 68. <http://eprint.iacr.org/2018/068>.
- [12] MAXWELL G, POELSTRA A, SEURIN Y, et al. Simple schnorr multi-signatures with applications to bitcoin[J/OL]. Des. Codes Cryptogr., 2019, 87(9): 2139-2164.
<https://doi.org/10.1007/s10623-019-00608-x>.
- [13] NICK J, RUFFING T, SEURIN Y. Musig2: Simple two-round schnorr multi-signatures [C/OL]//MALKIN T, PEIKERT C. Lecture Notes in Computer Science: volume 12825 Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part I. Springer, 2021: 189-221. https://doi.org/10.1007/978-3-030-84242-0_8.
- [14] CHEN Y. Dualms: Efficient lattice-based two-round multi-signature with trapdoor-free simulation[J/OL]. IACR Cryptol. ePrint Arch., 2023: 263. <https://eprint.iacr.org/2023/263>.
- [15] INSIGHTS L. Blockchain technology, bitcoin, and ethereum: A brief overview [EB/OL]. 2018[2022-06-07]. <https://www.ledgerinsights.com/blockchain-technology-bitcoin-ethereum-a-brief-overview/>.
- [16] JENSEN J R, VON WACHTER V, ROSS O. An introduction to decentralized finance (defi)[J]. Complex Systems Informatics and Modeling Quarterly, 2021(26): 46-54.
- [17] PARK D, ZHANG Y, ROSU G. End-to-end formal verification of ethereum 2.0 deposit smart contract[C]//Computer Aided Verification: 32nd International Conference, CAV 2020, Los Angeles, CA, USA, July 21–24, 2020, Proceedings, Part I 32. Springer, 2020: 151-164.
- [18] MICALI S, OHTA K, REYZIN L. Accountable-subgroup multisignatures: extended abstract[C/OL]//REITER M K, SAMARATI P. CCS 2001, Proceedings of the 8th ACM Conference on Computer and Communications Security, Philadelphia, Pennsylvania,

USA, November 6-8, 2001. ACM, 2001: 245-254. <https://doi.org/10.1145/501983.502017>.

- [19] RISTENPART T, YILEK S. The power of proofs-of-possession: Securing multiparty signatures against rogue-key attacks[C/OL]//NAOR M. Lecture Notes in Computer Science: volume 4515 Advances in Cryptology - EUROCRYPT 2007, 26th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Barcelona, Spain, May 20-24, 2007, Proceedings. Springer, 2007: 228-245. https://doi.org/10.1007/978-3-540-72540-4_13.
- [20] DRIJVERS M, EDALATNEJAD K, FORD B, et al. On the security of two-round multi-signatures[C/OL]//2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19-23, 2019. IEEE, 2019: 1084-1101. <https://doi.org/10.1109/SP.2019.00050>.
- [21] BENHAMOUDA F, LEPOINT T, LOSS J, et al. On the (in)security of ROS[C/OL]// CANTEAUT A, STANDAERT F. Lecture Notes in Computer Science: volume 12696 Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part I. Springer, 2021: 33-53. https://doi.org/10.1007/978-3-030-77870-5_2.
- [22] COCCIA M, ROSHANI S, MOSLEH M. Evolution of quantum computing: Theoretical and innovation management implications for emerging quantum industry[J/OL]. IEEE Transactions on Engineering Management, 2022: 1-11. DOI: 10.1109/TEM.2022.3175633.
- [23] DUCAS L, LYUBASHEVSKY V, PREST T. Efficient identity-based encryption over NTRU lattices[C/OL]//SARKAR P, IWATA T. Lecture Notes in Computer Science: volume 8874 Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaohsiung, Taiwan, R.O.C., December 7-11, 2014, Proceedings, Part II. Springer, 2014: 22-41. https://doi.org/10.1007/978-3-662-45608-8_2.
- [24] ZHANG P, WANG L, WANG W, et al. A blockchain system based on quantum-

- resistant digital signature[J/OL]. Secur. Commun. Networks, 2021, 2021: 6671648:1-6671648:13. <https://doi.org/10.1155/2021/6671648>.
- [25] ZIMMERMANN P, CASAMAYOU A, COHEN N, et al. Computational mathematics with sagemath[M]. SIAM, 2018.
- [26] GOLDSTON J, CHAFFER T J, OSOWSKA J, et al. Digital inheritance in web3: A case study of soulbound tokens and the social recovery pallet within the polkadot and kusama ecosystems[A]. 2023.
- [27] SMEETS I, LENSTRA A, LENSTRA H, et al. The history of the lll-algorithm[J]. The LLL Algorithm: Survey and Applications, 2010: 1-17.
- [28] DEL PINO R. Efficient lattice-based zero-knowledge proofs and applications. (preuves à divulgation nulle de connaissance efficaces à base de réseaux euclidiens et applications)[D/OL]. PSL Research University, Paris, France, 2018. <https://tel.archives-ouvertes.fr/tel-02445482>.
- [29] SM4 分组密码算法[Z]. GM/T 0002-2012. 2012.
- [30] NACHEF V, PATARIN J, VOLTE E. Feistel ciphers[J]. Cham: Springer International Publishing, 2017.
- [31] WANG X, YU H. How to break MD5 and other hash functions[C/OL]//CRAMER R. Lecture Notes in Computer Science: volume 3494 Advances in Cryptology - EU-ROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings. Springer, 2005: 19-35. https://doi.org/10.1007/11426639_2.
- [32] 谢宗晓,甄杰,董坤祥. 国产商用密码算法 SM3 及其相关标准介绍[J]. 中国质量与标准导报, 2021, No.275(14-16).
- [33] TIWARI H, et al. Merkle-damgård construction method and alternatives: a review[J]. Journal of Information and Organizational Sciences, 2017, 41(2): 283-304.
- [34] DUCAS L, KILTZ E, LEPOINT T, et al. Crystals-dilithium algorithm specifications and supporting documentation[C]//2017.

- [35] GROVER L K. A fast quantum mechanical algorithm for database search[C/OL]// MILLER G L. Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996. ACM, 1996: 212-219. <https://doi.org/10.1145/237814.237866>.