区块链智能合约隐私保护技术研究综述

王有恒[1)] 冯开开[1)] 李汝佳[2)] 段斯斯[2)] 陈庆接[3)]

[1)](山东大学网络空间安全学院 山东 青岛 266237)

[2)](清华大学高等研究院 北京 100084)

[3)](中国人民银行数字货币研究所 北京 100073)

摘 要 区块链智能合约因去中心化、自动执行和数据可追溯等特性,已在多个领域得到应用。然而,其透明性带来的多维度隐私泄露风险,已成为规模化应用的核心瓶颈。尽管已有多种针对智能合约的隐私增强方案,但技术路线碎片化与评估体系缺失,使现有研究缺乏系统性梳理与评估。本文通过定义理想化隐私模型,构建四维评估框架(性能维度、经济维度、易用性维度、通用性维度),系统梳理和分析了四类技术方案(可信执行环境、零知识证明、安全多方计算、同态加密)的技术特征、理论依据及实际应用场景。基于对27项核心文献的实证研究表明:(1)使用纯密码协议构建的方案在数据机密性保障方面具有理论完备性,但面临计算复杂度增长与动态场景适应性不足的双重约束;(2)使用可信执行环境的方案虽能提升交易处理效率,但其硬件依赖导致的攻击面扩大问题亟待解决。研究进一步揭示,现有方案在责任主体界定(系统层-合约层隐私边界模糊)与开发者能力框架层面存在系统性缺陷,亟待培养具备专业能力的隐私合约开发者。本研究可为构建兼具隐私保障、性能均衡、权责明晰的智能合约系统提供技术路线指导。

关键词 区块链;智能合约;隐私保护技术

中图法分类号 TP 309 **DOI号** 10.11897/SP.J.1016.2025.01373

Privacy-Preserving Technologies for Blockchain Smart Contracts:A Survey

WANG You-Heng[1)] FENG Kai-Kai[1)] LI Ru-Jia[2)] DUAN Si-Si[2)] CHEN Qing-Jie[3)]

[1)](School of Cyber Science and Technology, Shandong University, Qingdao, Shandong 266237)

[2)](Institute for Advanced Study, Tsinghua University, Beijing 100084)

[3)](Digital Currency Institute, People's Bank of China, Beijing 100073)

Abstract Blockchain smart contracts are applied in various fields due to their decentralization, automatic execution, and data traceability. However, the transparency of blockchain leads to privacy leakage risks, which have become a core bottleneck for large-scale applications. Although the academic community has extensively researched privacy protection schemes for smart contracts, there is still a lack of a comprehensive review. This paper aims to thoroughly explore the existing privacy protection technologies for smart contracts, systematically organizing and evaluating various schemes. The contributions of this article include the following three aspects: First, We constructed a comprehensive evaluation framework, including an idealized privacy smart contract model and dimensions for assessing privacy protection issues. Second, Through

收稿日期:2024-07-05;在线发布日期:2025-03-28。本课题得到国家重点研发计划项目(2023YFB2705000)、国家自然科学基金(92267203)、北京市自然科学基金(M23015)、中国博士后科学基金(2023M741949)和清华大学水木学者资助计划。 **王有恒**,博士研究生,中国计算机学会(CCF)会员,主要研究领域为区块链、隐私智能合约。E-mail:youheng@mail.sdu.edu.cn。 **冯开开**,博士研究生,主要研究领域为可信执行环境、智能合约及其应用。 **李汝佳**(通信作者),博士,助理研究员,中国计算机学会(CCF)会员,主要研究领域为混合共识算法、可信执行环境。E-mail:rujia@tsinghua.edu.cn。 **段斯斯**(通信作者),博士,研究员,中国计算机学会(CCF)会员,主要研究领域为应用密码学、共识算法。E-mail:duansisi@tsinghua.edu.cn。 **陈庆接**,硕士,主要研究领域为区块链架构设计、隐私保护技术。

exhaustive literature retrieval, we selected and categorized representative studies employing different privacy protection technologies. Finally, using the aforementioned evaluation framework, using the aforementioned evaluation framework, we analyzed the selected literature in-depth, discussing each scheme's differences, theoretical foundations, strengths, weaknesses, and practical applications. Our analysis indicates that: (1) Schemes based on pure cryptographic protocols ensure data confidentiality but face increased computational complexity and limited adaptability to dynamic scenarios. (2) Schemes utilizing Trusted Execution Environments can enhance transaction processing efficiency. Yet, the expansion of the attack surface due to hardware dependencies remains a pressing issue. Further research reveals systemic deficiencies in current schemes. These include unclear delineation of responsibility and gaps in developer competency frameworks. There is an urgent need to cultivate privacy contract developers with specialized skills. This study can provide technical guidance for the construction of a smart contract system with privacy guarantee, balanced performance, and clear rights and responsibilities.

Keywords blockchain; smart contract; privacy-preserving technologies

1 引 言

智能合约由Szabo[1]于20世纪90年代中期首次提出，并于2014年由以太坊在区块链中实现[2]。智能合约是一种自动化程序[3]，它允许在没有可信第三方管理的情况下执行和管理交易。智能合约通过可编程代码，使交易双方可以定义合约规则，并在满足特定条件时自动执行其中的指令。区块链智能合约由分布式节点执行。执行结果经过共识协议达成一致后，存储在公开账本上。因此，区块链智能合约具有去中心化、自动执行、可追溯等优点。它们被广泛应用到去中心化金融[4]、医疗数据保护[5]、物联网[6]等领域。

随着智能合约被广泛应用，其隐私问题成为了一个重要的研究领域。智能合约中的隐私问题泛指合约状态隐私、合约交易参与方的匿名性、合约指令隐私等问题。区块链数据的透明性和智能合约的隐私保护形成了天然的矛盾。具体来说，智能合约的执行过程和结果可被区块链的所有节点观测，导致任何未授权的用户都能获取到相关信息。这种透明性不仅暴露了交易中的身份和数据，还影响了其应用的适用性。例如，在电子投票领域，若投票内容全部在链上可见，选民可能因担心投票信息泄露而避免使用该系统[7-9]；同样，在拍卖和金融交易中，相关组织和用户也因担忧拍卖底价的泄露而回避使用智能合约[10]。

近年来，学术界和工业界提出了多种类型的智能合约隐私保护方案①。从技术上讲，这些方案可以分为两大类：基于硬件辅助的解决方案和基于纯密码协议的解决方案。基于硬件辅助的解决方案[8,10-11]主要依托可信执行环境技术。在这类方案中，智能合约在可信执行环境中执行，合约执行过程以及合约相关数据由可信执行环境提供隐私保护。基于纯密码协议的解决方案主要使用零知识证明[12-16]、安全多方计算[17-19]、同态加密[20-21]等技术。其中，使用零知识证明的方案允许在不泄露任何隐私的情况下，保证数据的正确性和计算结果的正确性；同态加密技术可为智能合约执行者提供直接对密文进行计算的能力；安全多方计算技术允许多个参与方彼此之间进行计算，而不泄露各自的隐私数据。

目前有多项综述工作[22-25]针对智能合约隐私保护方案进行梳理。然而，研究者来自不同学科领域，各自关注的角度、采用的技术方法以及解决的安全问题和预期目标各不相同，导致现有研究成果缺乏系统性。例如，Li等人[22]提出了一系列评估标准，从隐私保护和可用性两方面分析现有隐私保护方案。然而，该研究的侧重点较为宽泛，涉及区块链隐私保护的多个领域，对智能合约隐私保护的研究深度不足。Perez等人[23]关注特定应用领域中的智能合约隐私保护方案。文章分析了在群智感知领域使用智能合约的安全性和隐私问题，并提出了相应的解决方案。然而，该文仅关注特定场景，并未讨论如何构建通用的隐私保护方案。Li等人[24]讨论了如

① 为了让行文更加流畅，本文将具有隐私保护能力的智能合约统称为“隐私智能合约”

何利用可信执行环境保护智能合约的安全性和隐私性。文章介绍了使用可信执行环境的优点和挑战，并讨论了不同可信执行环境架构对安全性的影响。但文章并未涉及对基于纯密码协议的隐私保护方案的讨论。胡甜媛等人[25]将目前智能合约面临的安全问题分为合约安全和隐私安全。但文章重点关注合约安全问题，对隐私安全问题讨论不足。目前学界缺乏对于适用不同应用场景的通用智能合约隐私保护框架的讨论，难以为解决智能合约的隐私问题提供有力指导。

针对以上综述中所面临的诸多问题，本文深入研究智能合约隐私保护技术的评估方法，系统性梳理并评价现有的相关技术，发现当前研究中存在的不足，为智能合约隐私保护领域提供全面且深入的参考。本文对技术间的差异性、理论基础、优劣势以及适用范围进行详细分析。基于深入分析、归纳和总结的信息，本文提出了未来研究方向。具体而言，本文按照如下方法展开。

首先，我们提出一种评估智能合约隐私保护技术的框架，包括理想化隐私智能合约模型（第3节）和智能合约隐私保护方案问题评估维度（第2节中步骤二）等内容。理想化隐私智能合约模型包括以下几个方面：智能合约隐私保护方案角色定义，用于明确隐私智能合约利益相关方；智能合约隐私保护方案协议定义，用于规范隐私智能合约的生命周期；隐私智能合约属性定义，用于明确智能合约隐私保护方案需要实现的具体属性。

其次，我们检索2015年至2024年发表的智能合约隐私方面的论文，并筛选出高质量论文27篇。经过调研发现，智能合约的隐私保护集中于合约计算阶段。基于此，我们将智能合约的隐私保护方案按照技术路线进行分类（智能合约隐私保护方案的发展历程如图1所示）。这一分类与隐私计算的分类相似。具体而言，我们将这些方案按隐私保护技术分为四类：基于可信执行环境、零知识证明、安全多方计算和同态加密的智能合约隐私保护方案。这样的分类旨在展示每种技术的隐私保护能力，性能和适用性，从而明确特性和应用场景。

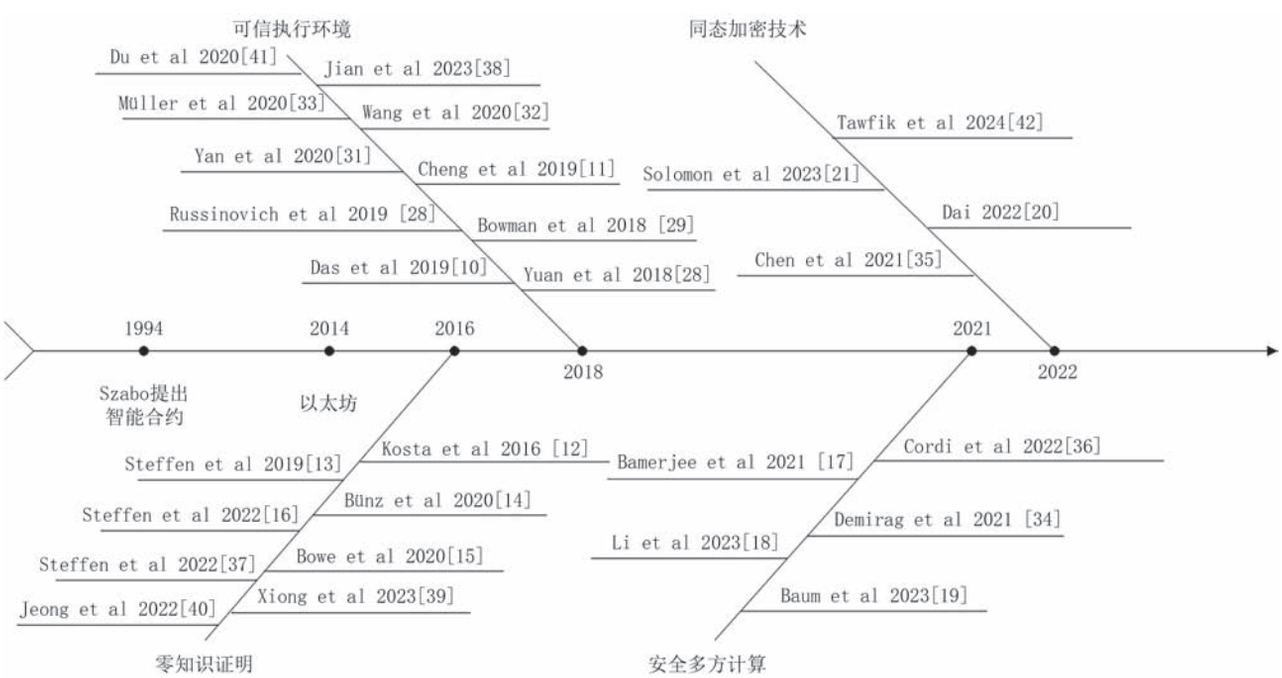


图1　智能合约隐私保护方案的发展

值得强调的是，本文的研究对象是隐私智能合约，而非广泛的隐私计算[26]。两者的应用场景和侧重点不同。具体来说，隐私计算的核心目标是允许多个主体在不直接暴露敏感数据的情况下进行计算，以确保数据隐私和计算结果的正确性。这种技术在需要跨组织协作且不希望共享原始数据的场景中应用广泛，例如医疗数据的联合分析[27]。而智能合约是一个复杂系统，其生命周期中涉及多个步骤和交互，使得隐私保护的复杂性进一步加大。首先，隐私智能合约依赖于区块链等分布式系统。其去中心化架构虽然确保了系统的公开和透明性，但也带来了如增加攻击面等潜在风险。其次，隐私智能合

约不仅关注数据隐私，还强调匿名性和不可链接性等身份隐私。最后，由于智能合约可能需要与外部数据源交互。隐私智能合约必须特别关注数据源的可信性和数据传输的安全性，以防止数据泄露或篡改。这些特点使隐私智能合约面临独特挑战。本文重点分析智能合约特有的隐私保护需求与挑战，关注如何防范复杂合约环境中的隐私风险。我们在第4和第5节中描述的智能合约隐私保护方案对智能合约整体架构、交互流程进行分析。这些方案讨论如何在多层次上实现隐私保护，而不仅限于单一的隐私计算阶段。

进而，我们对筛选出的智能合约隐私保护方案进行描述，并依据定义的评估框架，对四类方案进行评估与对比，评估内容包括隐私属性分析。同时，我们从性能、经济性、易用性和通用性四个维度探讨了各类方案面临的问题与挑战，并对隐私保护技术的安全性进行分析（第4节和第5节），总结了各方案的差异、理论基础、优势与不足。

最后，基于对不同分类智能合约隐私保护方案的分析结果。我们讨论了不同隐私保护技术在智能合约隐私保护领域的应用场景、优点和局限性。根据本文的研究结论，我们提出了三个未来智能合约隐私保护方案的研究方向（第6节）：一是结合智能合约的具体应用场景，根据具体情况定制不同的隐私保护策略；二是通过软硬件结合的方式，实现具有高安全性和高灵活性的隐私保护架构；三是加强对相关从业人员的技术培训，防止因设计失误导致的隐私泄露。

2 本文研究方法

本节介绍研究方法。具体来说，我们沿着文献收集和筛选、评估框架构建、文献评估和结果分析的研究路线开展研究。

步骤一：收集和筛选相关文献

我们将智能合约（Smart Contract）组合以下词组：“隐私的（Private）”、“隐私保护（Privacy-Preserving）”、“机密的（Confidential）”、“可信执行环境（Trust Execution Environment）”、“零知识证明（Zero Knowledge Proof）”、“同态加密（Homomorphic Encryption）”等作为关键字，在IEEE Xplore、ACM Digital Library、Springer Link Online Library和CNKI等在线文献数据库中搜索，重点检索对象包括USENIX Security、CCS、IEEE S&P以及CRYPTO等国际知名期刊与会议。同时，为避免遗漏，我们人工对上述论文所引用的参考文献进行检查，检查和筛选流程在附录A中详细描述。最终，我们检索到与智能合约隐私保护相关的论文771篇。检索结果显示智能合约隐私保护相关的文献数量呈现逐年递增的趋势，表明该研究领域的热度持续上升。本文共计筛选出27篇高质量论文作为分析对象（如表1所示）。

步骤二：构建智能合约隐私保护方案的评估框架

我们从属性评估和问题评估两个维度构建隐私智能合约方案的评估框架。首先，我们对普通智能合约进行概览，总结智能合约生命周期和工作流程；进而，通过对智能合约隐私保护方案的分析，抽象出一个理想的隐私智能合约模型，并阐述其语义定义和属性定义。在智能合约隐私保护方案中，研究人员通过集成多种隐私保护技术来增强隐私性。然而，这些技术和区块链的结合也带来了其他问题。我们从四个维度评估这些问题，分别是：

（1）性能维度：主要关注隐私保护技术对智能合约的效率和执行速度的影响。（2）经济维度：主要关注隐私保护技术在实际应用中是否增加智能合约成本。（3）易用性维度：主要关注增加隐私保护技术后，智能合约开发和使用过程中的便捷程度。（4）通用性维度：主要关注智能合约隐私保护方案针对不同智能合约类型和应用场景的适应能力和灵活性。

步骤三：利用框架评估现有方案

我们将筛选出的27个典型智能合约隐私保护方案按所采用的隐私保护技术分类，归纳为基于可信执行环境、零知识证明、安全多方计算和同态加密四类。我们在表2中对这些方案进行简要总结。在后续中详细描述，并依据智能合约隐私保护方案评估框架，评估对比这四类方案。首先，我们根据理想化隐私智能合约模型对方案属性和语义进行分析。其次，我们从性能、经济、易用性和通用性四个维度探讨了各类隐私保护方案面临的问题与挑战。最后，我们对各类隐私保护技术的安全性简要分析。结合上述分析，我们得出各方案的差异、理论基础、优势与不足。

步骤四：分析和评估结论

我们对现有方案的属性评估结果及存在的问题进行总结。通过这些总结，我们得出了关于隐私技术在智能合约领域应用的相关启示，包括不同种类隐私保护技术适用场景和优劣势。基于这些总结和相关启示，我们对未来研究方向进行展望，旨在帮助该领域研究人员完善智能合约隐私保护方案。

表1　主要文献(按发表时间顺序排列)

作者	名称	发行源	时间
Kosba et al[12]	Hawk	IEEE S&P	2016
Yuan et al[28]	ShadowEth	JCST	2018
Bowman et al[29]	PDO	arxiv	2018
Steffen et al[13]	Zkay	ACM CCS	2019
Das et al[10]	Fastkitten	USENIX Security	2019
Cheng et al[11]	Ekiden	EuroS&P	2019
Russinovich et al[30]	CCF	Microsoft, Redmond	2019
Bünz et al[14]	Zether	FC	2020
Yan et al[31]	CONFIDE	ACM sigmod	2020
Wang et al[32]	Hybridchain	IEEE Access	2020
Müller et al[33]	TZ4Fabric	IEEE SRDS	2020
Bowe et al[15]	Zexe	IEEE S&P	2020
Banerjee et al[17]	zk Hawk	BRAINS	2021
Demirag et al[34]	Absentia	FC	2021
Chen et al[35]	FHE-Blockchain-based	IEEE Syst. J	2021
Steffen et al[16]	ZeeStar	IEEE S&P	2022
Dai[20]	PESCA	ePrint	2022
Cordi et al[36]	GABLE	ePrint	2022
Li et al[18]	Ratel	ePrint	2023
Baum et al[19]	Eagle	FC	2023
Solomon et al[21]	SmartFHE	Euro S&P	2023
Steffen et al[37]	Zapper	ACM CCS	2023
Jian et al[38]	TSC-VEE	IEEE TPDS	2023
Xiong et al[39]	Veri-Zexe	USENIX Security	2023
Jeong et al[40]	Azeroth	IEEE Access	2023
Du et al[41]	EPT	MSN	2023
Tawfik et al[42]	Pricollabanalysis	Cluster Computing	2024

3　隐私智能合约模型定义和属性定义

3.1　智能合约概览

智能合约是部署在区块链网络上的自动化执行程序[3]，它允许在没有可信第三方的管理下执行和管理交易。智能合约的生命周期涉及多个角色，如合约创建者、使用者和区块链节点等，主要经历开发、编译、部署、调用和销毁等五个阶段。首先，开发者使用高级编程语言，如Solidity[43]，来编写智能合约代码。编写完成后，智能合约代码被编译成字节码，同时生成用于调用合约的接口。随后，这些编译后的字节码通过一个特定的交易部署到区块链上，该交易不仅包含合约的字节码，还会创建一个独特的智能合约地址，用于后续的操作。用户可以通过提交交易的方式使用这个地址来调用合约，这包括直接交易调用和合约间的消息传递。最后，智能合约可以被设定为销毁状态，此时合约将无法再被调用。

智能合约调用阶段数据流向如图2所示。在这一阶段，合约调用者将合约相关参数以交易的形式发送到智能合约地址。智能合约在执行过程中，根据预先协定的规则使用“状态转换函数”输出一个新

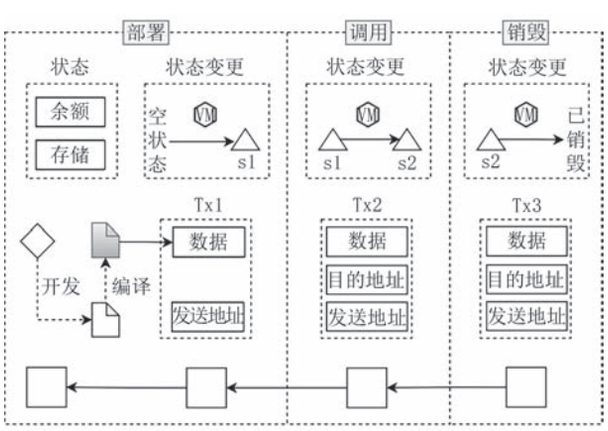


图2　智能合约工作流程

的状态，并将更新的状态存储到区块链平台。智能合约在区块链上被部署后，调用过程中参数通过交易进行传递。交易数据明文存储于区块链上，允许用户随时查询。这对交易数据的隐私安全构成重大威胁。在智能合约执行时，交易内容中携带用户身份相关信息，而区块链的公开透明性可能导致用户身份隐私的泄露。

3.2 隐私智能合约概览

近年来，研究人员提出多种智能合约隐私保护（Privacy-Preserving Smart Contract）方案[10-21]。这些方案为智能合约开发者提供一个框架。这个框架允许他们能够轻松编写隐私智能合约，而无需考虑应用场景的限制。综合分析现有方案，目前的隐私智能合约构建思路大致如下：在开发阶段，平台提供具有指定数据隐私属性的高级语言，使开发人员能够准确描述数据属性（公开或隐私）；在编译阶段，提供编译器，使开发人员无需专业知识也可以开发隐私智能合约应用；在部署阶段，利用隐私保护技术保护合约存储环境，以此保障合约代码安全；在调用阶段，利用相关隐私保护技术保障计算数据和计算过程的隐私安全。

3.3 隐私智能合约语义定义

隐私智能合约方案如图3所示，包含以下几种角色：(1)系统创建者：搭建智能合约隐私保护架构，负责环境变量的配置，并负责后续系统管理以确保智能合约平台的安全性、隐私性；(2)合约创建者：根据实际需求，结合所采用的隐私保护架构，编写和部署智能合约；(3)合约使用者：通过智能合约进行交互，使用其功能来满足自身业务需求的个人或组织；(4)合约执行者：负责实际执行智能合约所定义操作的节点或实体，通常是区块链网络中的参与者和维护者；(5)区块链节点：负责根据智能合约执行结果，更新区块链状态。一个理想的隐私智能合约应包含下列函数：

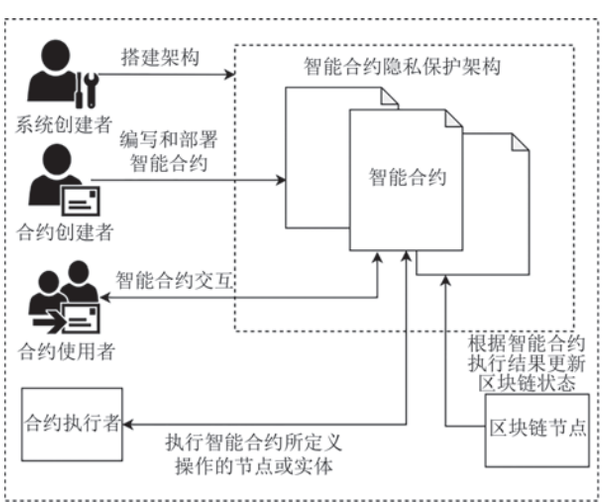


图3 隐私智能合约方案角色概览

(1) $pp \leftarrow \mathrm{Setup}(\lambda)$：系统创建者以安全参数$\lambda$作为输入，输出系统公共参数$pp$，并配置系统环境变量。

(2) $(sk_u, pk_u, addr_u) \leftarrow \mathrm{CreateAccount}(pp)$：以系统公共参数$pp$作为输入，生成并输出密钥对$(sk_u, pk_u)$和地址$addr_u$。本算法由用户u执行。

(3) $(Sig_t, tx) \leftarrow \mathrm{CreateTransaction}(pk_s, sk_s, pk_r, data, tx_{type})$：系统用户（此处的用户在交易部署合约场景下指合约创建者，在交易调用合约场景下指合约使用者）输入发送方公钥pk_s、发送方私钥sk_s、接收方公钥pk_r、相关数据$data$和交易类型标识符$tx_{type} \in \{\mathrm{priv}, \mathrm{public}\}$，输出交易内容$tx$以及对交易的签名$Sig_t$。

(4) $(S', \pi_t, Sig_c) \leftarrow \mathrm{Compute}(Sig_t, tx, S)$：输入交易$tx$、交易签名$Sigt$和旧状态$S$。合约执行者根据交易类型和输入数据进行计算，输出（隐私或公开）状态S'、证明π_t以及对输出状态和证明的签名Sig_c。具体执行环境根据所使用的保护方案决定，例如采用可信执行环境的方案，Compute算法在可信执行环境中进行。

(5) $(0/1) \leftarrow \mathrm{UpdateState}(pp, tx, S', Sig_c, \pi_t)$：区块链上的分布式节点以系统公共参数$pp$、交易$tx$、新状态$S'$、证明$\pi_t$、签名$Sig_c$作为输入。如果新状态在链上被接受，则更新状态并输出1，否则输出0。

(6) $S \leftarrow \mathrm{Read}(Id_{user}, Id_{data})$：(6)系统用户通过自己的身份标识符$Id_{user}$和数据标识符$Id_{data}$查询区块链，得到加密状态$S$。然后用户解密$S$得到自己的状态$State$。

3.4 隐私智能合约属性定义

结合前文智能合约方案角色定义，我们总结出智能合约中各角色所关注的隐私特性如下：(1)合约使用者：他们希望保护自己的交易数据和身份信息不被公开，防止隐私泄露。(2)合约创建者：他们需要确保合约中处理的敏感信息不会被未授权的第三方获取，以维护合约的隐私性和用户信任。(3)合约执行者：他们参与交易验证和合约执行，但不应有权访问隐私数据。(4)区块链节点：负责更新和存储区块链状态，未授权的第三方无法获取交易信息。通过总结隐私问题并分析角色隐私利益相关性，我们将智能合约的隐私性分为下列五种性质：

（P1）交易假名性（Pseudonymity）[44]：假名性指的是使用假名作为标识符的过程。发送者假名性和接收者假名性分别指发送者和接收者使用假名作为标识符，而非真实身份。在隐私智能合约中，交易假名性指CreateAccount时，使用生成的假名代指实际用户，而不使用真实身份。

定义1. 交易假名性. 令U表示用户集，Id为真实身份集合，N为假名集合，$U\subseteq Id$为用户真实身份集合，假名映射为$pseud$：$U\rightarrow N$。对任意用户$u\in U$，假名性要求其标识符为$pseud(u)$，而不是u本身。

（P2）交易不可链接性（Unlinkability）[24]：交易的不可链接性要求攻击者无法通过观察交易数据来确定交易之间是否存在关联，以及交易与发送者和接收者之间是否存在关联。

定义2. 交易不可链接性. 令TX表示交易集，交易集容量为k，U表示用户集，A表示攻击者①。对于交易tx_i，tx_j，$\forall i,j\in\{1,k\}$，攻击者A无法确定tx_i，tx_j是否关联，也无法判断交易tx_i与发送方U_a和接收方U_b是否关联。

（P3）交易匿名性（Anonymity）[14]：交易匿名性指的是一个主体在一组可能的主体（匿名集）中无法被识别。具体来说，智能合约匿名性意味着智能合约使用者无法被从潜在的使用者集合中识别。在隐私智能合约合约中，交易匿名性要求在CreateAccount、Compute、UpdateState三个阶段以及执行结果存储到分布式账本后，攻击者无法在匿名集中识别交易的具体参与方。

定义3. 交易匿名性. 令TX表示交易集，U表示用户集，A表示攻击者。对于一组交易$TX=\{tx_1,tx_2,\cdots,tx_n\}$，在交易的传输、执行过程以及发布到分布式账本后，攻击者A无法从U中识别交易tx_i的发送方U_a和接收方U_b。

（P4）状态隐私性：此属性旨在确保交易中的敏感数据不能被未授权实体获取。包括在计算阶段交易数据不能被未授权实体获取（执行时交易数据隐私），以及交易完成后，未授权实体无法获取分布式账本上存储的交易内容（存储时交易数据隐私）。

定义4. 状态隐私性. 令TX表示交易集，U表示用户集，A表示攻击者。对于一组交易$TX=\{tx_1,tx_2,\cdots,tx_n\}$，在交易的传输、执行过程以及发布到分布式账本后，攻击者A无法识别交易tx_i的具体内容。

（P5）功能隐私[15]：在数据执行计算时，攻击者无法获知计算所用函数的具体细节。换言之，当相关状态发生改变时，隐藏所使用的转换函数。

定义5. 功能隐私. 令C表示智能合约，I表示输入集，O表示输出集，S表示初始状态集，S'表示智能合约更新后状态集，A表示攻击者。对于一组输入$I=\{i_1,i_2,\cdots,i_n\}$，一组输出$O=\{o_1,o_2,\cdots,o_n\}$和一组状态变化$S=\{s_1,s_2,\cdots,s_n\}$，$S'=\{s'_1,s'_2,\cdots,s'_n\}$攻击者A观察任意输入、输出和状态变化的组合$(i_1,o_1,s_1,s'_1)$，$(i_2,o_2,s_2,s'_2)$，$\cdots$，$(i_n,o_n,s_n,s'_n)$，无法推断出合约$C$的内部逻辑。

隐私智能合约模型不仅需要具有隐私属性，还需要保留一些合约原有属性以支持模型的实际应用。我们将这些属性总结如下：

（O1）交易的完整性[45]：在交易数据传输、处理以及存储的过程中，确保数据的准确和完整，防止数据被未经授权地篡改或损坏。

定义6. 交易的完整性. 令TX表示交易集，S表示系统状态，T表示时间节点，A表示未授权用户。对于任意的时间节点T，在未授权用户A干预下，TX中的数据及其关联系统状态S保持不变。

（O2）数据的可用性[45]：授权用户能对数据进行及时可靠的访问，主要目标是确保信息系统在需要时能够正常运行，用户可以及时获取到所需的信息或服务。

定义7. 数据的可用性. 令U表示授权用户，T表示时间节点，D表示相关数据。对于任意的时间节点T，授权用户U在任意的时期，总是能够访问相关数据D。

（O3）持久性[46]：当交易成功提交后，其所做的更改应当永久保存在数据库中，即使系统发生故障或重启也不会丢失。

定义8. 持久性. 令T和T'表示数据保存到数据库后的任意两个时间节点，D表示相关数据。对于两个任意的时间节点T和T'，数据D在T和T'应该一致且可以被访问。

（O4）公开可验证性[24]：合约执行的程序和结果是公开可验证的。这意味着任何观察者都可以独立验证合约是否正确执行。包括验证数据的处理和计算过程中所采用的算法、参数和输入数据是否符合预期，并且计算结果是否准确可信。

① 本文提及的攻击者和作恶者均为概率多项式时间敌手。概率多项式时间敌手指多项式时间内运行且其行为具有随机性的敌手。即该敌手的计算能力受限于多项式时间复杂度，并可利用随机性来选择攻击策略。

定义9. 公开可验证性. 令C表示智能合约，V表示验证者，I表示输入，O表示输出，S表示初始状态，S'表示智能合约更新后状态。对于任意输入I、初始状态S以及输出O和新状态S'，存在一个验证过程，验证者V能够独立确认O和S'是由智能合约C根据I和S正确生成的。

(O4)部分可验证性：合约执行的程序和结果在满足特定条件时是可验证的。比如，在给定特定参数或处于规定的执行环境下，观察者可以验证合约是否正确执行。

定义10. 部分可验证性. 令C表示智能合约，V表示验证者，I表示输入，O表示输出，S表示初始状态，S'表示智能合约更新后状态。对于满足特定条件的输入I和初始状态S以及输出O和新状态S'，存在一个验证过程，验证者V能够确认O和S'是由智能合约C根据I和S正确生成的。

(O6)共识可验证性[24]：关于(隐私)状态的共识程序是公开可验证的。这意味着即使状态本身是隐私的，但达成共识的过程可以被公众验证。

定义11. 共识可验证性. 令C表示智能合约，V表示验证者，$\overline{S'}$表示智能合约更新后的隐私状态，P表示共识过程。对于智能合约更新后的隐私状态$\overline{S'}$，存在一个验证过程，验证者V能够确认区块链节点将$\overline{S'}$通过P达成共识并存储到区块链上。

在设计智能合约隐私保护架构时，现有方案主要关注合约状态的隐私性，隐藏合约数据信息。而在合约身份隐私方面，智能合约依靠底层区块链系统实现假名性，但假名性并不能提供强大的隐私保证[47]。方案需要实现不可链接性，以防止攻击者通过观察用户活动来获得用户身份信息。Bünz等人在Zether[14]中提出了匿名性定义，旨在更进一步地保护用户身份隐私。完整性、可用性和持久性这三类属性是针对实际应用中的数据特性。完整性确保数据在传输和存储过程中不被篡改；可用性确保系统随时能可靠访问和使用数据；持久性保证数据长期存储且可访问。根据具体需求和应用场景，架构可以选择性地实现功能隐私、公开可验证性、共识可验证性和部分可验证性这些属性，以增强合约的可验证性和隐私功能，从而提升系统的可靠性。

表2 智能合约隐私保护方案

分类	方案名称	方案描述	主要贡献	时间
可信执行环境	ShadowEth[28]	一个基于公共区块链的隐私智能合约系统。	(1)描述ShadowETH架构和协议，利用可信执行环境保护智能合约的隐私性。(2)提供激励机制，鼓励拥有可信执行环境的参与方加入到由可信执行环境组成的分布式存储系统中。	2018
	PDO[29]	介绍了一种技术，允许不信任各方在保持数据隐私的同时运行智能合约。	(1)提出隐私数据对象技术，允许不信任各方在保持数据隐私的同时运行智能合约。(2)通过无状态的飞地实现了智能合约执行即服务模式，可以在飞地之间移动合约状态。	2018
	FASTKITTEN[10]	一个利用可信执行环境在比特币等加密货币上执行复杂智能合约的框架。	(1)提出了一个名为FASTKITTEN的新系统，它利用可信执行环境在比特币等加密货币上支持任意复杂的智能合约。(2)设计了高效的链下执行协议。	2019
	Ekiden[11]	一个混合可信执行环境区块链的架构。	通过计算与共识分离，提供了一种可信执行环境与区块链(TEE- blockchain)的混合架构。	2019
	CCF[30]	一个构建许可机密区块链的框架。	通过在硬件保护的可信执行环境网络上复制区块链操作，保护应用数据和代码在分类账上的完整性和机密性。	2019
	CONFIDE[31]	针对金融联盟链复杂业务场景的隐私智能合约方案。	提出了一个确保数据机密性和完整性的数据模型和安全协议，作为蚂蚁链平台的插件模块实现，并可通过通用接口插入其他区块链平台。	2020
	Hybridchain[32]	一种新的混合链架构，通过结合区块链和可信执行环境以增强许可区块链的保密性和性能。	(1)提出了一种新的混合链架构。(2)设计了一个将智能合约的执行与链上共识机制解耦的层次网络结构，并只将初始和最终状态上链。(3)设计了一种新的安全键值内存系统，作为可信执行环境内存拓展。	2020

续表

分类	方案名称	方案描述	主要贡献	时间
	TZ4Fabric[33]	基于ARM的TrustZone的隐私智能合约方案。	提出了一个将Hyperledger Fabric[48]与ARM TrustZone[49]结合使用的系统架构，专注于物联网领域。	2020
	TSC-VEE[38]	设计了第一个在TrustZone上执行Solidity[43]智能合约的虚拟机执行环境。	通过设计特定指令集、运行时内存管理机制、混合粒度字节码分析算法和跨隔离环境预取方法，在TrustZone上实现并验证了高效的智能合约执行。	2023
	EPT[41]	一个利用可信执行环境实现区块链上机密智能合约的高效且用户友好的框架。	(1)设计了安全的TEE认证和数据隐私方案，使用户无需执行远程认证即可信任TEE，并确保私密交易的安全传输；(2)提出基于博弈论的激励机制，帮助TEE节点从不可信的区块链节点中收集可靠信息。	2023
零知识证明	Hawk[12]	一个用于构建隐私保护智能合约的框架。	(1)程序分为隐私部分和公开部分；(2)通过编译器将程序编译为区块链与用户之间的加密协议。提供了在分布式加密货币系统中同时实现交易隐私和可编程性的解决方案。	2016
	Zkay[13]	一种使用隐私类型来指定隐私数据所有者的类型化语言。	(1)利用隐私注释标明数据所有权；(2)使用零知识证明确保Zkay合约可转换为等价的可在公共区块链上执行的合约。	2019
	Zether[14]	一个基于账户模型的去中心化、隐私保护支付协议。	(1)Zether协议实现匿名支付;(2)提出一种新的零知识证明机制(Σ-Bullets)，以实现高效范围证明。	2020
	Zexe[15]	一种去中心化隐私计算系统，实现了在分布式账本上执行用户定义的函数。	(1)提出数据结构记录，以及用于执行记录相关的操作的共享执行环境——记录纳米内核;(2)Zexe中零知识证明函数是一个通用函数，隐藏状态转换函数细节实现功能隐私。	2020
	VeriZexe[39]	一个改进的去中心化私密计算框架，旨在提高零知识证明系统的效率和扩展性。	(1)基于Zexe进行改进，通过零知识证明实现交易的隐私性和功能隐私;(2)引入了通用设置，只需一次信任设置即可支持多个应用，解决了Zexe需要为每个应用独立设置的问题;(3)优化验证过程，显著提高交易生成速度和内存使用效率。	2023
	Azeroth[40]	一个基于账户模型的可审计零知识转移框架	(1)通过双账户系统和默克尔树累加器实现隐私保护；(2)采用零知识证明和双重加密机制，在保证隐私的同时，实现了交易的可审计性。	2023
	ZeeStar[16]	Zkay[13]语言的拓展，采用基于Zkay的隐私注释标明隐私数据所有权，并引入同态加密来执行外部数据计算。	(1)利用隐私注释标明数据所有权;(2)使用同态加密技术解决Zkay无法表达涉及外部数据操作的问题，并结合零知识证明实现隐私约束。	2022
	Zapper[37]	一个隐私保护智能合约系统。它允许直观的前端合约开发。	(1)允许开发人员使用直观的前端以开发隐私智能合约；(2)解决Zexe中存在的安全性、耦合性等问题。	2023
安全多方计算	Absentia[34]	基于区块链的安全函数评估方法(Secure Function Evaluation，SFE)。	描述了一个基于区块链的安全函数评估协议，允许多方参与者在不泄露隐私输入的情况下进行计算，并将其在以太坊上实现。	2021
	zkHawk[17]	基于安全多方计算的Hawk[12]拓展实用隐私智能合约。	(1)使用安全多方计算实现Hawk中的管理员角色；(2)在安全多方计算程序中，采用一种相对实用的知识证明来替代之前的零知识证明，以确保输入输出的余额相等。	2021
	GABLE[36]	通过安全多方计算在区块链上进行可审计、可用和弹性的隐私计算方案。	在区块链上实现了包括混淆电路和混淆有限状态机两种安全多方计算方法。	2022
	Ratel[18]	智能合约的安全多方计算扩展。	(1)提出了一个新的编程框架Ratel，它将多方计算原型框架MP-SPDZ[50]与智能合约语言Solidity[43]结合起来;(2)设计了崩溃重置机制，使崩溃的节点能够重新参与新的安全多方计算任务。	2023

续表

分类	方案名称	方案描述	主要贡献	时间
	Eagle[19]	针对在安全多方计算中计算密码原语开销大的问题，实现了一个通用的可组合协议。	(1)提出了Eagle，一种可普遍组合[51]的实现高效隐私保护智能合约的协议；(2)通过结合安全多方计算和零知识证明技术，避免了在安全多方计算实例中计算加密原语。	2023
同态加密	FHE-Blockchain-based[35]	一种基于区块链和完全同态加密的隐私保护计算框架。	(1)在以太坊区块链上集成完全同态加密；(2)设计并实现了基于完全同态加密和区块链的维克里拍卖系统[52]；(3)评估完全同态加密方案在区块链上的可行性和性能。	2021
	PESCA[20]	一种支持门限解密协议的智能合约隐私保护架构。	形式化定义门限解密协议的状态机复制问题，并讨论如何改造现有的拜占庭共识协议以支持门限解密协议。	2022
	SmartFHE[21]	一个由分布式节点执行隐私数据计算的隐私智能合约框架。	(1)形式化定义智能合约隐私保护框架；(2)通过使用完全同态加密，由分布式节点执行隐私数据计算，以满足轻量级用户的隐私保护需求。	2023
	Pricollabanalysis[42]	一种基于区块链的隐私保护医疗协作分析框架。	(1)集成隐私保护技术，实现去中心化医疗数据分析；(2)基于同态加密和安全多方计算，利用智能合约实现医疗数据的安全协同 分析。	2024

4 基于可信执行环境的解决方案

4.1 可信执行环境技术

可信执行环境(Trusted Execution Environment，TEE)提供了受硬件保护的安全执行区域(通常称为飞地)，旨在防止未经授权的外部实体访问敏感数据。现有的可信执行环境方案在安全目标上虽有所不同，但大多数方案旨在实现四个核心安全目标[53]：(1)可验证地启动敏感代码和数据的执行环境，使远程实体能够确保其配置的正确性。(2)运行时隔离，以保护敏感代码以及数据的机密性和完整性。(3)可信输入输出，以实现对外围设备和加速器的安全访问。(4)可信执行环境中数据的安全存储，使得在未来的时间点只有授权实体可以访问这些数据。经过多年的发展，针对不同的体系架构出现了多种技术路线，部分技术路线如表3所示。

表3 各种可信执行环境产品的对比

产品名称	体系结构	数据完整性	数据机密性
Intel SGX[54]	X86	支持	支持
Intel TDX[1]	X86	支持	支持
AMD SEV[2]	X86	支持 (嵌套分页)	支持 (系统隔离分区)
ARM TZ [49]	ARM	支持 (系统隔离分区)	支持
ARM CCA[55]	ARM	支持 (系统隔离分区)	不支持
KeyStone[3]	RISC V	支持 (内存保护)	支持 (内存保护)

注：[1] https://www.intel.com/content/www/us/en/developer/tools/trust-domain-extensions/overview.html
[2] https://www.amd.com/en/developer/sev.html
[3] https://keystone-enclave.org/

4.2 TEE保护智能合约隐私的作用机理

在智能合约隐私保护方面，可信执行环境通过其计算能力和对计算过程的隐私保障，弥补了区块链系统在算力和隐私保护方面的不足。同时，区块链系统可以解决可信执行环境在出错时终止执行所导致的可用性缺失以及无法持久化存储运算结果的问题。

在智能合约的生命周期中，可信执行环境的应用场景主要在初始化(Setup)、合约执行(Compute)、更新状态(UpdateState)以及查询状态(Read)阶段。在初始化阶段，系统完成可信执行环境初始化(如创建可信执行环境飞地，生成密封密钥等操作)，然后把智能合约代码加载到可信执行环境内并完成智能合约的初始化。区块链系统则接收从可信执行环境发送的加密初始状态、密钥和证明等数据，分布式节点达成共识后存储相关数据，完成区块链系统的初始化。在合约执行阶段，可信执行环境建立输入数据的安全通道，接收并解密用户发送的加密输入数据。同时，系统从区块链系统同步并解密智能合约最新的状态，在可信执行环境中根据解密后的数据执行智能合约代码，得到输出结果并加密新状态。在更新状态阶段，区块链系统更新加密后的智能合约状态，同时生

成该状态已存储在区块链系统中的证明。在查询状态阶段，用户从区块链系统查询加密后的智能合约状态，从可信执行环境获取密钥并将它解密。

4.3　具体实现方案

前文已经讨论了可信执行环境在智能合约中应用的意义和场景。下面，我们将重点探讨基于可信执行环境的智能合约隐私保护具体实现方案。这些实现方案（如表4所示）涵盖如何将可信执行环境集成到区块链系统中，以及如何利用可信执行环境保护合约隐私。

表4　基于可信执行环境技术的方案

种类	方案	所用区块链	经济模型	所用TEE
与区块链解耦	ShadowEth[28]	Ethereum	√	SGX
	PDO[29]	Sawtooth	-	SGX
	FASTKITTEN[10]	Bitcoin	√	SGX
	Ekiden[11]	Ethereum	-	SGX
	Hybridchain[32]	Fabric	-	SGX
	TZ4Fabric[33]	Fabric	-	TrustZone
	TSC-VEE[38]	Ethereum	-	TrustZone
	EPT[41]	Ethereum	√	SGX
与区块链耦合	CCF[30]	许可链	-	SGX
	CONFIDE[31]	蚂蚁链	-	SGX

注：√表示方案支持，-表示方案中未提及。

（1）ShadowEth

Yuan等人在2018年提出了针对以太坊的隐私智能合约系统ShadowEth[28]。该系统借助可信执行环境对公开的区块链数据和隐私智能合约进行划分，通过加密通信和验证协议，提供智能合约规范（源码）、智能合约执行功能以及智能合约状态隐私保护。

ShadowEth的主要角色包括用户端、以太坊系统及基于可信执行环境的链下分布式存储系统（TEE-Distributed Storage，TEE-DS）。其中用户端具有和以太坊系统及TEE-DS系统交互的功能；以太坊系统部署了公共的赏金合约（Bounty Contract），用以部署、验证及存储隐私智能合约的元信息，同时提供报酬作为TEE-DS执行智能合约的赏金；TEE-DS在链下分布式存储隐私智能合约的二进制文件及状态，由可信执行环境保护合约的执行和存储，保证了数据隐私及功能隐私。

该系统主要架构如图4所示。首先用户发送调用请求到赏金合约，客户端从赏金合约获得参数并从TEE-DS获取合约二进制。工作客户端把参数及合约二进制发送到自身维护的可信执行环境内执行并得到最新状态。随后，客户端将返回值和签名发送到赏金合约，同时将最新状态发送到TEE-DS，以便TEE-DS的其他节点更新合约状态。赏金合约验证通过后把报酬发给运行可信执行环境的工作客户端。

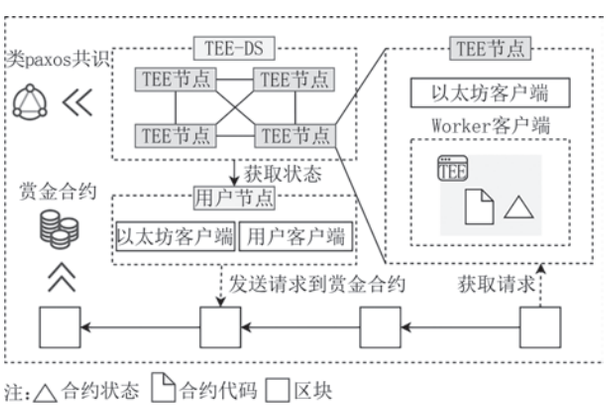


图4　ShadowEth系统架构[28]

ShadowEth的威胁模型假设参与系统的交易各方不互信，任何一方可能会在任何时候崩溃或完全停止响应。区块链系统可信且始终可用，同时信任可信执行环境、硬件制造商以及远程认证服务，并假设私钥的隐私性。ShadowEth中提出激励机制，鼓励拥有可信执行环境的参与方加入到TEE-DS系统中。系统中的工作节点越多，提供可信执行环境服务的稳定性就越高。

（2）Private Data Objects

Bowman等人在2018年提出链下执行隐私智能合约的方案PDO（Private Data Objects）[29]。PDO允许在链下的飞地环境中运行智能合约，解决了隐私数据暴露和节点不信任的问题。方案通过无状态的飞地，实现了合约执行即服务（Smart Contract

Execution as a Service)。任意的飞地可以读取合约上的密文然后进行解密和执行。

PDO的主要角色包括用户、分布式账本、飞地托管(Enclave Hosting)服务、合约所有者、供应(Provisioning)服务以及合约飞地。在该系统中,用户向合约发送调用消息,检查可信执行环境返回的结果并提交包含结果的交易到分布式账本中。分布式账本包括合约注册及飞地注册功能。飞地服务负责执行飞地的注册、更新及撤销,同时验证飞地是否正确运行。合约所有者可以创建PDO实例,注册或撤销合约,添加或移除飞地,并指定供应服务列表。供应服务负责加密合约状态,合约飞地负责执行智能合约。

该方案主要架构如图5所示。用户在调用智能合约方法时,首先从分布式账本中获取最新的合约状态以及飞地信息,然后向合约飞地发送合约调用请求获得执行结果,最终向分布式账本发送包含状态的交易。PDO的主要特点是可信执行环境内部无状态。这使得它可以方便地进行扩展,并且可以更换执行隐私智能合约的可信执行环境设备。

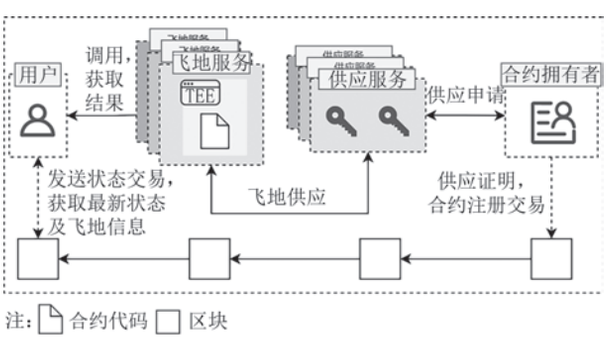


图5 PDO系统架构[29]

PDO的威胁模型假设分布式账本是可用且可靠的,安全性基于其底层协议。方案假设敌手的计算能力有限,依赖密码学原语来保证安全性。方案信任飞地及IAS(Intel Attestation Service),并假设敌手难以对PDO分布式环境中的SGX平台进行可扩展的攻击。

(3)FASTKITTEN

Das等人在2019年提出一个实用的比特币复杂智能合约执行框架FASTKITTEN[10]。该方案通过把合约执行放在可信执行环境中为比特币提供智能合约的功能,同时实现隐私及高性能。

FASTKITTEN的主要角色包括拥有可信执行环境的操作者,智能合约的参与者以及区块链(比特币)系统。其中,操作者作为核心角色,主要职责包括运行可信执行环境内智能合约,作为其他角色消息的转发枢纽,以及对区块链证据(evidence)的验证。同时,为防止操作者作恶,FASTKITTEN利用比特币的时间锁交易(Time-Locked Transaction)功能设计了一种质押惩罚机制。该机制可以使恶意操作者受到没收质押的惩罚。

该方案主要架构如图6所示。首先是初始化(Set up)阶段,把合约加载到可信执行环境以及发送时间锁交易等初始化操作;接下来是轮次计算(Round Computation)阶段,操作者把上一阶段的输出发送到所有参与者,同时转发参与者的输入,由可信执行环境执行智能合约;最后是最终(Finalization)阶段,此阶段进行最后的结算及支付操作,通过质押惩罚机制防止操作者作恶,通过挑战应答机制防止参与者作恶。

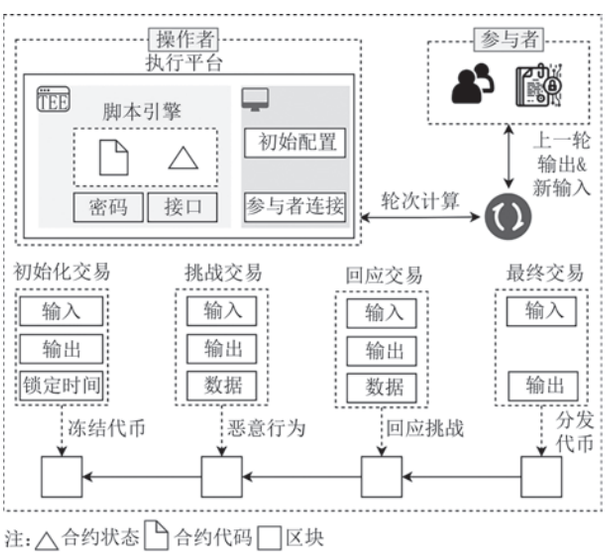


图6 FASTKITTEN系统架构[10]

FASTKITTEN假设系统的参与者可以被攻击者控制,从而表现出恶意行为,但至少有一个参与者是诚实的,即该参与者遵循协议规定的行为。FASTKITTEN假设可信执行环境(包括签名、验证机制以及远程认证机制)是可信的,能够提供安全的执行环境。方案假设底层区块链满足活性、一致性及不可篡改三个安全属性。方案针对敌手方的模型是假设其可以进行拜占庭行为,目标可能是破坏合约的正确执行、窃取参与者的资金或泄露合约的机密信息。

FASTKITTEN的优势是最小化与区块链的交互(只有初始化和结算两次)次数,把智能合约相关的多轮交互流程完全放在链下。这样可以使智能合

约系统拥有高性能及低延迟的特性，理论上可支持多种不同资产类型。所以该方案适用于与链上资产交互较少的智能合约，比如彩票、拍卖、扑克等应用。但是，操作者作为整个系统的核心角色，方案中未设计如何促进操作者长期提供服务的激励机制。同时，方案中也未描述如何根据智能合约的复杂性准确评估时间锁交易中时间设置的方法。

(4)Ekiden

Cheng等人在2019年提出一个具有隐私属性和可信属性的智能合约平台Ekiden[11]。该平台支持千级别的每秒交易(Transactions Per Second，TPS)，通过计算与共识分离，提供了一种可信执行环境与区块链(TEE-blockchain)的混合架构。

Ekiden的主要角色包括拥有可信执行环境的计算节点(包含一个分布式密钥管理委员会)，智能合约的客户端以及共识节点。其中，客户端是智能合约平台的使用者。计算节点的主要职责包括运行可信执行环境内智能合约，校验区块链消息发布的证明并保证计算节点与共识节点和客户端所发送消息的原子性。方案与以太坊的区别是其使用证明代替节点之间的重复执行。计算节点利用可信执行环境执行合约计算。共识节点负责运行区块链系统。

该平台主要流程如图7所示。客户端发送输入到计算节点中的可信执行环境，计算节点从区块链获取加密存储的合约状态并从分布式密钥管理委员会获取用于解密的密钥。然后，基于已有的合约状态和输入，执行可信执行环境内的智能合约，得出输出信息并生成新的合约状态。最后，将新的合约状态加密后发送到区块链进行存储，同时把输出信息返回到客户端。

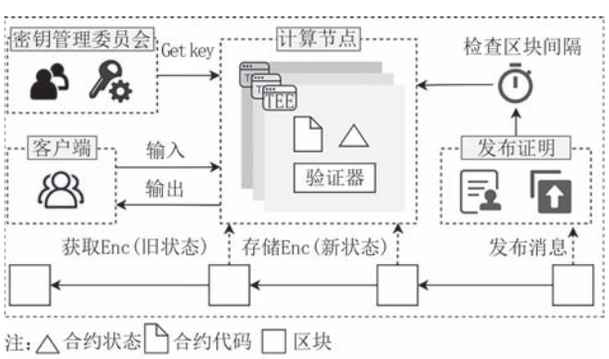


图7　Ekiden系统架构[11]

Ekiden的威胁模型假设可信执行环境(包括远程认证机制和硬件实现)是安全的，但承认其可能受到侧信道攻击的影响，从而泄露机密信息。在区块链系统的安全性假设中，区块链应能够正确执行计算任务并保持高可用性，同时高效地生成数据证明，确保数据已被包含在区块链中。Ekiden假设敌手能够控制操作系统的网络栈，观察全局网络流量，并能够破坏一部分计算节点的机密性。

由于存在一组能够运行可信执行环境的计算节点，Ekiden可以容忍部分计算节点作恶。并且文章对分布式密钥的机密性和可用性属性进行分析，即使部分可信执行环境受到攻击也不会影响整个系统的服务。同时整个架构和协议的设计将可信执行环境和区块链作为两个抽象的通用系统，理论上适用于包括联盟链及公链的不同类型区块链系统。

(5)CCF

CCF[30]由Russinovich等人于2019年提出，是一个基于联盟区块链的新框架。该框架建立在基于硬件保护的可信执行环境网络之上。CCF在硬件保护的可信执行环境中复制区块链操作，旨在解决联盟链机密性和性能问题。

CCF支持运行在抽象分类账上的可编程治理模型。动态服务配置(包括其成员、副本、用户、应用程序代码和治理规则)被表示为键值对进行存储，所有治理操作都被记录在区块链上。框架中包含两类角色，普通用户负责执行服务的应用程序命令。而管理者使用服务提供的接口调用治理模块执行特权命令执行管理服务，管理者通过投票共同负责服务背书和管理配置。程序存储表分为应用表和治理表。应用表作为应用程序所定义相关表的集合(例如记录用户余额的账户表)。治理表是记录当前底层服务协议配置的原始表。

CCF的威胁模型假设攻击者可能监听、篡改或阻断节点通信，破坏一致性或中断服务；或是控制拜占庭节点违反协议并发送恶意交易。CCF采用具备拜占庭容错和崩溃故障容忍性的共识协议，确保所有配置都支持强大的服务完整性。即使某些节点或其密钥被攻破，也可以通过记录在账本中的有关恶意活动的证据来识别被攻破的节点。此外，在治理过程中，攻击者可能通过控制成员或合谋篡改配置，威胁服务的安全性和透明性。因此，服务的安全性依赖于诚实管理者的数量。CCF框架使用硬件保护的可信执行环境复制区块链操作。通过对可信执行环境中的程序代码进行审查和远程认证，将经过验证的代码放入可信执行环境中运行。区块链的运行状态可信且无法被外界获取，从而保证智能合约数据和代码的完整性和安全性。

(6)CONFIDE

Yan等人在2020年提出针对金融联盟链复杂业务场景的隐私智能合约方案CONFIDE[31]。该方案通过一个机密合约引擎以及安全数据传输协议、数据加密协议、密钥管理协议三个关键协议,在其自研的蚂蚁链上实现真实生产系统的业务场景。

CONFIDE的主要角色包括客户端和拥有可信执行环境的区块链节点。其中,客户端是智能合约平台的使用者。区块链节点在原有公共合约引擎功能的基础上,额外通过可信执行环境支持机密合约引擎的功能。隐私智能合约相关的交易及状态通过加密存储在区块链数据库中,同时基于接口定义语言(Interface Definition Language,IDL)概念设计实现了一个机密智能合约语言的扩展。

该方案主要架构如图8所示。客户端在发送交易时分别发送公开交易(Public Tx)和机密交易(Confidential Tx)。其中,机密交易在可信执行环境的机密合约引擎执行,并和公开交易一起通过网络传播并参与共识流程,最后以加密形式存储。

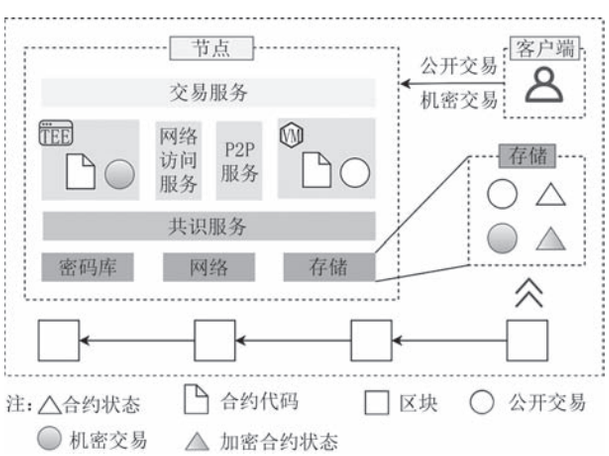


图8 CONFIDE系统架构[31]

CONFIDE假设区块链网络中可能存在恶意节点,这些节点会尝试篡改数据或窃取敏感信息。方案假设可信执行环境能够提供内部代码和数据完整性和机密性的保护,而外部的数据可能被恶意节点篡改或窃取。方案假设只有经过授权的参与者才能访问区块链上的敏感数据,未经授权的访问将被阻止。

CONFIDE基于实际的生产数据对架构性能及扩展性做了评估。同时,在工程实践方面,方案对可信执行环境进行优化:包括优化数据结构、提高内存管理效率以及提升监控系统。

(7)Hybridchain

Wang等人在2020年提出针对许可链(Fabric)机密性的混合架构Hybridchain[32]。该方案将计算与共识解耦,实现计算资源从链上到链下的安全转移。链下节点在可信执行环境内执行交易及相关的智能合约,链上节点验证执行结果并维护分布式账本。

Hybridchain的主要角色包括客户端、计算节点及共识节点。其中,客户端发起相关交易并从系统中接受交易的返回结果;计算节点维护可信执行环境并执行智能合约;共识节点提供包括签名及状态哈希等链上验证,并存储加密后的状态。

Hybridchain的主要架构如图9所示,主要包括三个阶段。首先是密钥生成和注册阶段,发生在客户端及计算节点之间;接下来是计算阶段,每个交易都会生成随机的对称密钥,同时使用签名保障客户端及计算节点之间的通信安全;最后是验证阶段,计算节点把最新状态加密并带上自身签名发送到共识节点,共识节点验证签名并提交交易。

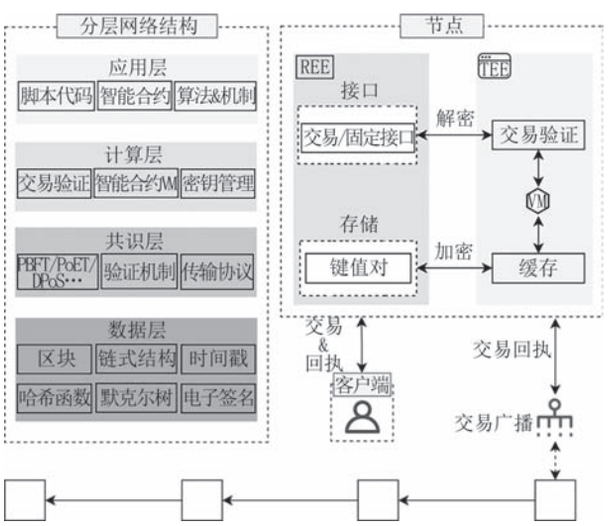


图9 Hybridchain系统架构[32]

Hybridchain在面对各种潜在攻击时的安全性设计包括:在面对数据泄露风险时,方案利用可信执行环境的隐私保护功能,确保交易数据在执行过程中不被泄露;在面对分布式拒绝服务攻击时,方案继承了区块链的异步访问和容错机制;在面对重放攻击和状态分叉攻击时,方案通过在可信执行环境内生成公私钥并建立安全网络通道,防止消息在不同实例之间共享。

(8)TZ4Fabric

Müller等人在2020年提出基于TrustZone的隐私智能合约方案TZ4Fabric[33]。该方案在Fabric上实现了主要针对物联网(Internet of Things,IOT)的隐私智能合约。

TZ4Fabric主要针对内存有限的低功耗嵌入式设备，包括封装器（Wrapper）、链码（Chaincode）及代理（Proxy）三个组件。封装器提供区块链及用户的接口，区块链系统保证数据的可用性和持久存储；链码是具体的智能合约代码，它运行在可信执行环境内的安全世界；代理运行在可信执行环境内的正常世界，负责连接封装器和链码。该方案目前不支持远程认证机制（Remote Attestation），同时在区块链系统上没有验证机制。

该方案的威胁模型假设攻击者拥有管理员权限，对系统具有完全的访问和控制能力，同时攻击者能直接接触所有支持ARM TrustZone的节点，并尝试通过硬件层面进行攻击。在以上攻击者能力的情况下，方案假设操作系统和用户空间不可信，可信执行环境（包括引导程序、固件、OP-TEE以及安全监视）是可信的，攻击者无法篡改或绕过可信执行环境的保护机制。

（9）TSC-VEE

Jian等人在2023年提出一个基于TrustZone的智能合约虚拟机执行环境TSC-VEE[38]。该方案设计了第一个用于在TrustZone中执行Solidity[43]合约的虚拟机执行环境。

TSC-VEE的主要架构如图10所示，主要包括四个部分。一是与TrustZone的隔离和世界切换机制相适应的指令集；二是运行时的内存管理机制，提供一对带有相应处理机制的指令，即分配和释放；三是混合粒度分析算法，避免运行时溢出和无效计算；四是跨隔离环境的预取方法，避免在运行时频繁切换世界状态。

该方案主要流程包含四个阶段。首先是调用阶段，用户对输入数据进行加密，并构造交易发送到可信执行环境中；其次是计算阶段，可信执行环境接收用户发送的交易，对交易中的输入数据进行解密，同时从区块链系统获取加密后的源码和初始状态，使用可信执行环境的密钥进行解密，根据输入数据进行计算得到新的状态并获取用户的返回信息；再次是共识阶段，区块链网络的节点运行共识算法，确认交易的执行结果和新的状态；最后，用户从交易响应中获得状态的密文，并可以使用用户的私钥解密数据获得最终的明文结果。

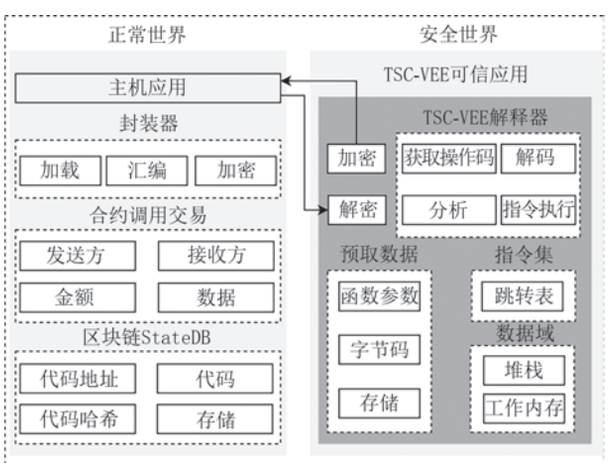


图10　TSC-VEE系统架构[38]

TSC-VEE假设TrustZone的软件组件都是可信的，正常世界中的操作系统和用户空间被认为是不可信的。同时，方案假设用户有安全的方法收集和处理输入数据，且输入数据和输出数据在进入和离开TSC-VEE之前都会被加密。方案不考虑针对TrustZone本身的如侧信道等攻击，也无法保护由于区块链本身状态回滚导致的回滚攻击。

（10）EPT

Du等人在2023年提出一个利用可信执行环境实现区块链上机密智能合约的高效且用户友好的框架EPT[41]。该框架允许用户直接向区块链提交私密交易，与部署在可信执行环境中的机密智能合约交互。其架构如图11所示。

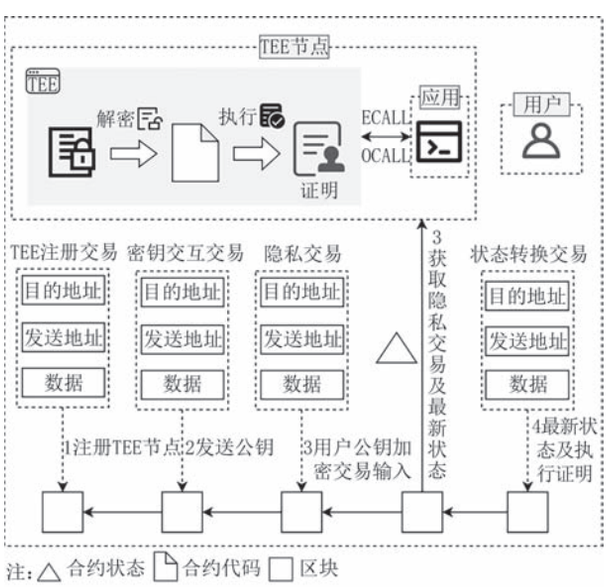


图11　EPT系统架构[41]

EPT的主要角色包括终端用户、区块链、可信执行环境节点、Intel远程认证服务、数据存储智能合约以及区块链节点委员会。其中，终端用户通过标准的区块链客户端发送私密交易，与部署在可信执行环境中的机密智能合约进行交互；区块链作为

分布式账本，维护机密智能合约的状态，负责存储交易记录和合约状态；可信执行环境节点运行机密智能合约，负责执行私密交易并生成执行完整性的证明；Intel远程认证服务负责验证可信执行环境节点，并通过远程认证报告确保仅完整性验证通过的节点能够为用户提供服务；数据存储智能合约是部署在区块链上的公共智能合约，充当用户和可信执行环境节点之间的安全枢纽，负责存储用户的私密交易和机密智能合约的状态，并提供接口供用户与机密智能合约交互；区块链节点委员会由区块链节点随机组成，负责向可信执行环境提供最新的区块和机密合约的状态。

EPT的威胁模型假设区块链本身是可信的，部署在区块链上的智能合约能够正确执行；攻击者可能访问可信执行环境节点的部分主机，并在接收消息时执行恶意行为，但可信执行环境内部的执行过程是安全的；由于智能合约的强制约束确保了系统的正常运行，用户的行为不会影响系统的正常运行。

为了确保委员会成员提供的数据的新鲜性和有效性，EPT采用了基于博弈论的激励方案。这个方案通过奖励和惩罚机制，激励委员会成员提交准确和最新的数据。

4.4 属性分析

可信执行环境在硬件层面实现了智能合约状态隐私性。在此类方案中（如表5所示），在合约执行和更新状态阶段引入可信执行环境。智能合约在可信执行环境中执行并更新状态，确保执行过程中不泄露信息，保证状态隐私性。同时，区块链负责验证和存储执行结果，实现了数据的完整性和持久性。然而由于实现方式的不同，不同方案的属性有所差异。具体讲，可信执行环境与区块链系统结合有两种方式。一是可信执行环境与区块链系统解耦，作为独立的链外系统与区块链系统进行交互；二是可信执行环境与区块链系统耦合，作为区块链系统的一部分，为每个区块链节点配备可信执行环境。解耦模式如ShadowEth[28]和PDO[29]，在链外的可信执行环境中执行合约，增强了执行时的隐私，但会导致系统复杂性增加，进而增加实现不可链接性的难度。而耦合模式如CCF[30]和CONFIDE[31]则将可信执行环境紧密集成到区块链中，增强了数据隐私和完整性，但是计算结果缺乏公开可验证性。

表5 基于可信执行环境解决方案属性分析表

分类	实现方案	隐私属性					其他属性					
		假名性	不可链接性	匿名性	状态隐私性	功能隐私	可用性	持久性	完整性	公开可验证性	部分可验证性	共识可验证性
基于可信执行环境	ShadowEth[28]	√	-	-	√	√	√	-	-	×	√	√[2]
	PDO[29]	√	-	-	√	√	√	√	√	×	√	√
	FASTKITTEN[10]	√	-	-	√	-	-	-	-	×	√	-
	Ekiden [11]	√	-	-	√	√	√	√	√	×	√	√
	CCF[30]	√	-	-	√	-	-	√	√	×	√	√
	CONFIDE[31]	√	-	-	√	√	√	√	√	×	√	√
	Hybridchain[32]	√	-	-	√	-	-	√	√	×	√	√
	Tz4Fabric[33]	√	-	-	√	-	√	√	√	×	√	√
	TSC-VEE[38]	√	-	-	√	√	√	√	√	×	√	√
	EPT[41]	√	-	-	√	√	√	√	√	×	√	√

注：(1) √表示方案具有属性，×表示方案未实现属性，-表示方案中未提；

(2) TEE集群之间的类Paxos算法，非区块链共识算法。

我们的研究结果表明，基于可信执行环境的智能合约隐私保护方案主要用于增强智能合约的状态隐私性和安全性。100%的方案通过在受保护的环境中执行合约，确保了状态隐私性。约有80%方案均可通过结合区块链系统提供数据持久性和完整性。在身份隐私方面，100%的方案支持假名性，但均未考虑不可链接性和匿名性。这些方案在设计上还考虑了计算的可验证性。由于该类方案数据计算过程需要在可信执行环境中执行，无法实现公开可验证性，所有方案具备部分可验证性。

4.5 存在的问题

在智能合约隐私保护方案中，研究人员通过引入不同的隐私保护技术以满足相关属性，但也带来了一系列问题。我们根据第2节的步骤二中所提出的性能、经济、可用性、通用性四个维度对方案进行评估。具体的评估结果如图12所示。

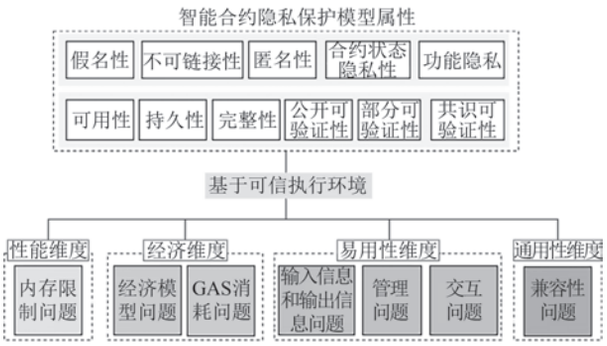


图12　基于可信执行环境的解决方案存在的问题

(1)性能维度

可信执行环境可以为隐私智能合约提供高效的硬件解决方案。然而，当超过内存限制时，系统性能会快速降低。例如，对Intel第二代SGXv2硬件进行的基准测试显示[56]，SGXv1上B-Tree的性能在128 MB之后快速下降，只能支持最大64 GB的数据库大小，同时当数据集大小是飞地页面缓存(Enclave Page Cache，EPC)容量的4倍时，分页使延迟增加了约两个数量级。在SGXv2中将每个插槽的EPC容量提高到最多512 GB(取决于CPU型号)，一个内存中的B-Tree可以扩展到大约120 GB的数据，同时保持高性能。这意味着，当面临复杂的智能合约，超过可信执行环境的内存限制时，性能可能会下降。

(2)经济维度

在经济维度，我们从以下几个方面进行衡量，包括奖励和惩罚机制的经济模型问题以及和区块链系统交互带来的交易燃料问题。在奖励方面，需要考虑如何鼓励可信执行环境操作者与区块链系统交互，即鼓励操作者消耗一定的费用向区块链系统发送交易。惩罚方面，需要考虑如何惩罚恶意的可信执行环境操作者，增加其作恶的成本，如进行一定资金的质押等。交易燃料方面，需要考虑如何减少可信执行环境与区块链系统的交互次数，减少交互过程中因发送交易产生的燃料费用。例如FASTKITTEN[10]最小化与区块链的交互(只有初始化和结算两次)，把智能合约相关的多轮交互流程完全放在链下，减少燃料费用。但是对于系统中单一的操作者，FASTKITTEN并未提供奖励机制来促使该操作者持续提供可信执行环境相关的服务。Ekiden[11]也未针对可信执行环境集群提供奖励机制，以吸引更多可信执行环境所有者的加入。

(3)易用性维度

可信执行环境虽然提供了安全的硬件环境，但对输入信息和输出信息并未严格控制。方案中一般需要通过额外建立安全通道对输入信息和输出信息进行加密。对于可信执行环境操作者的管理问题有两个方面需要考虑：一是针对单一操作者，需要处理单一可信执行环境意外宕机或恶意关闭等情况，例如FASTKITTEN[10]依赖单一的可信执行环境操作者，虽然设计了惩罚机制，整个系统依然会处于无法对外提供服务的状态。二是针对多操作者，则需要解决操作者之间的分布式密钥管理情况。例如Ekiden[11]虽然有一个可信执行环境的集群，但对于分布式密钥管理问题只有初步设计，并未提供成熟的方案。此外，还存在合约涉及资产时需要与区块链进行交互的问题。例如Ekiden[11]在可信执行环境内执行智能合约不涉及区块链上的原生资产。而TZ4Fabric[33]在与区块链系统的交互过程中，区块链系统没有验证机制。

(4)通用性维度

通用性维度与区块链结合方式不同，会存在不同通用性的情况。对于可信执行环境作为区块链系统一部分的方案，需要针对不同的区块链系统进行适配，存在一定问题。对于可信执行环境作为独立的系统的方案，则具有较好的通用性。目前，针对不同的体系架构，存在多种可信执行环境的技术路线。尽管这些技术路线具备一定的共性，但在使用可信执行环境来实现智能合约隐私保护的系统中，可能会面临需要适配多种技术路线的挑战。例如TZ4Fabric[33]使用ARM TrustZone作为可信执行环境，Hybridchain[32]基于Intel SGX作为可信执行环境。

4.6　解决途径讨论

针对前文中基于可信执行环境的隐私保护方案所出现的问题，当前学术界提出了一些解决方案。

在性能维度，Hybridchain[32]扩展内存。它设计了安全的键值存储机制，以解决内存限制。针对经济模型问题，ShadowEth[28]提供了激励机制，鼓励拥有可信执行环境的参与方加入到TEE-DS系统中，以提升系统的稳定性。FASTKITTEN[10]设计了一种质押惩罚机制，在操作者作恶时，能够通过没收质押来进行惩罚。针对燃料消耗问题，FASTKITTEN[10]最小化与区块链的交互(只有初始化和结算两次)，把智能合约相关的多轮交互流程完全放在链下，减少燃料费用。在易用性维度，针对管理问题，PDO[29]设计了智能合约执行即服务模式，使可信执行环境中的飞地无状态，支持在可信执

行环境之间移动合约。这种设计允许在当前可信执行环境出现问题时能够及时切换。针对交互问题，Ekiden[11]设计了证明发布(Proof of Publication)协议，当区块链消息发布时，计算节点能够通过校验证明验证消息。针对输入信息和输出信息，ShadowEth[28]通过加密通信保证了可信执行环境中输入信息和输出信息的数据隐私。在通用性维度，Ekiden[11]把可信执行环境和区块链作为两个抽象的通用系统，以松耦合的方式，一定程度上兼容不同区块链系统以及不同技术路线的可信执行环境。

4.7 可信执行环境安全性讨论

在基于可信执行环境技术的智能合约隐私保护方案的讨论中，安全性是一个关键问题。以下将重点探讨可信执行环境技术在实际隐私智能合约应用中的安全挑战和应对策略。我们将从数据传输、可信执行环境以及可信执行环境与区块链结合等三个部分进行讨论。

首先是数据传输部分。可信执行环境提供了一个隔离的安全执行环境，但是与可信执行环境交互过程是在安全执行环境之外，无法依赖硬件提供的安全。在可信执行环境之外的数据传输中，敏感信息可能会被泄露。数据传输主要有三种情况，一是数据输入和数据输出，输入数据包括用户调用基于可信执行环境的隐私智能合约的参数等信息，数据输出包括可信执行环境在执行用户调用请求后返回的结果以及合约最新状态；二是对于需要外部预言机[57]支持的隐私智能合约，其与外部预言机的交互过程中可能存在泄露隐私的风险；三是在可信执行环境集群中，不同处理器厂商的可信执行环境实现可能不完全兼容，这可能导致数据在集群节点间传输时出现数据无法解密的安全问题。为了解决数据传输部分的安全挑战，研究人员主要以三个策略进行应对。一是在数据输入和输出的过程中，使用加密技术来保护用户和可信执行环境之间通信数据的机密性；二是使用密码学应用协议，如TLS(Transport Layer Security)协议，来验证预言机数据的来源，并确保数据在传输中未被篡改；三是设计完善的针对可信执行环境集群的分布式密钥管理机制，同时推动行业标准的制定和遵循，以确保不同可信执行环境技术之间的兼容性和互操作性。

其次是可信执行环境部分。可信执行环境的安全挑战主要包括两部分，一是验证可信执行环境本身的真实性和完整性，尤其是针对远程的可信执行环境；二是执行环境本身存在的如侧信道攻击等安全挑战，在我们调研的基于可信执行环境的隐私保护方案中，针对侧信道攻击并未提供成熟的解决机制。为了解决可信执行环境部分的安全挑战，研究人员主要以三个策略进行应对。一是支持远程认证，允许用户通过安全通道验证可信执行环境的真实性和完整性，如Intel SGX[54]技术等；二是通过数字签名和哈希算法来确保代码和数据在执行过程中没有被篡改，保证数据的完整性；三是采用多因素认证机制，增加对可信执行环境的保护，确保只有经过验证的实体才能进行敏感操作。

最后是可信执行环境与区块链结合部分。对于使用区块链作为持久性存储来保存隐私智能合约状态的系统来说，与区块链上的状态保持一致性是一个挑战，尤其是对于可信执行环境集群。如果基于可信执行环境的隐私智能合约系统允许在没有区块链共识的情况下执行交易，那么可能会产生不一致的状态，因为这些交易可能不会被区块链网络的其他部分认可。为了解决可信执行环境与区块链结合部分的安全挑战，通过实施证明发布机制来确保交易在执行前已经被共识层提交，从而提供了一定程度的状态一致性保护。

5 基于密码学技术的解决方案

5.1 基于零知识证明的解决方案

5.1.1 零知识证明技术

在1985年Goldwasser等人[58]首次提出零知识证明(Zero Knowledge Proof，ZKP)的概念。它是运行在证明者和验证者之间的一种两方密码协议，可用于进行成员归属命题证明或知识证明。零知识证明允许一方(证明方)向另一方(验证方)证明关于公共输入x和隐私输入w的陈述的有效性，即$f(x;w)=$true，且不透露任何关于隐私输入w的信息。一个零知识证明应具有如下三个性质：(1)完整性(Completeness)：如果声明是真实的，那么在零知识证明系统中，诚实的证明者(Prover)可以成功地说服验证者(Verifier)这一点。通俗来说，如果声明是真实的，那么诚实的证明者总能够生成一个可接受的证明，使得验证者相信声明的真实性。(2)合法性(Soundness)：如果声明是假的，那么任何恶意的证明者在零知识证明系统中都无法成功欺骗验证者。通俗来说，如果声明是假的，那么在任何情况下，恶意的证明者都不可能生成一个可接受的证明，使得验证者相信声明的真实性。(3)零知识性(Zero-

knowledge)：指的是在证明过程中，证明者能够向验证者证明某个陈述的真实性，而不需要透露除该陈述的真实性以外的任何其他信息。通俗来说，验证者只会获得有关声明真实性的信息，而无法获得有关如何证明这一点的任何信息。这确保了证明的过程不会泄露证明者希望保持隐私的信息。

在过去的几十年中，零知识证明相关技术被人广泛研究。Blum等人开发了非交互式零知识证明(Non-Interactive Zero-Knowledge)[59]，使用共享随机数来创建公共参考串(Common Reference String)。在非交互式零知识证明中证明者和验证者在协议过程中不需要进行多轮通信，证明者可以在不与验证者通信的情况下构建证明。简洁非交互式零知识证明(Zero-Knowledge Succinct Non-Interactive Argument of Knowledge)[60-64]不但具有非交互性，而且非交互式零知识证明的长度通常与证明的计算复杂度相关，而与输入的大小无关。Groth-16[65]在Pinocchio[60]的基础上改进了证明大小和验证器的复杂性。

5.1.2　ZKP保护智能合约隐私的作用机理

得益于零知识证明的零知识性，它被广泛应用于加密货币[66-67]和身份认证[68-69]等领域。它通过提供一种在证明陈述真实性的同时不泄露其他信息的方式，为信息交换和身份认证提供更高的安全性和隐私性。

在智能合约的生命周期中，零知识证明技术的应用场景如图13所示，主要应用于初始化(Setup)、创建交易(CreateTranscation)、合约执行(Compute)以及更新状态(UpdateState)阶段。在初始化(Setup)阶段，执行零知识证明方案初始化过程，并确保该过程正确且被诚实地执行。该过程通常涉及生成密钥和参数，这些密钥以及参数用于后续密码学相关操作。存在部分方案(例如Bulletproof[70])不依赖可信设置阶段。当用户创建交易(CreateTranscation)时，根据合约要求，用户对交易相关数据进行加密，确保数据在传输过程中的隐私性。同时，生成零知识证明π并将其附加到交易中，以证明交易的格式正确且用户状态符合智能合约的要求，而不会泄露任何额外的信息。合约执行计算(Compute)时，合约执行者验证交易中携带的零知识证明π，验证成功后执行智能合约的相关计算，验证失败将拒绝执行计算。计算完成后，合约执行者生成零知识证明π'，保证相关计算按合约规则正确执行。更新状态(UpdateState)时，分布式节点验证新生成的状态证明π'，验证π'正确后，将新状态更新到区块链上。在数据传输和存储过程中，数据以加密格式处理，利用零知识证明保证加密数据的正确性。

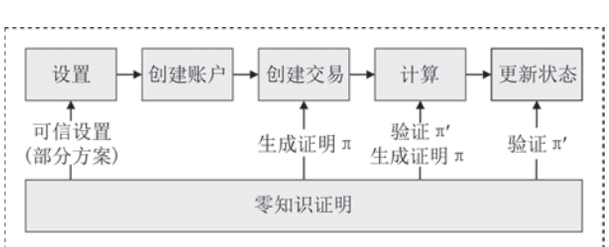


图13　零知识证明应用场景

在智能合约中使用零知识证明的意义是在不泄露数据隐私的情况下，证明相关计算输入数据格式的良好性并保证计算按合约规定的逻辑执行。

5.1.3　基于零知识证明的隐私货币协议Zerocash

Ben-Sasson等人于2014年提出了一种基于简洁非交互式零知识证明的匿名数字货币Zerocash[66]，通过零知识证明实现身份匿名和交易金额保密。Zerocash引入了一种新的数据结构“硬币(coin)”，主要包含硬币值、硬币地址、硬币承诺和硬币序列号四个部分。硬币值v表示面额，硬币地址$addr$是持有者的公钥地址，硬币承诺cm是硬币生成时记录在分类账中的承诺，序列号sn用于防止双重支出。

硬币只能通过消耗旧硬币并生成新硬币的方式使用。Zerocash中包含两种交易：铸币交易(Mint)用于生成新硬币，注入交易(Pour)用于转移输入硬币的值至新的输出硬币，并标记输入硬币为消耗。通过这两种交易，Zerocash实现了匿名用户之间的安全转账。

在铸币交易中，硬币的生成过程如图14所示。硬币的生成依赖硬币值v、地址$addr$以及用户公钥和私钥，还需生成随机数r、s、ρ以计算生成承诺cm和序列号sn。生成的承诺cm被存储在默克尔树中。

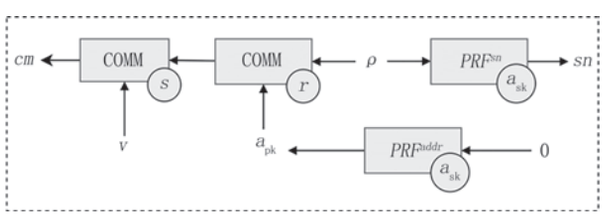


图14　Zerocash硬币组成[66]

在注入交易中，用户将旧硬币c_{old1}、c_{old2}注入新硬币c_{new1}，c_{new2}。注入交易构建两个向量$\vec{x}$和$\vec{a}$，其中$\vec{x}$作为零知识证明的公共输入包含默克尔树根rt、旧

硬币序列号sn_{old}、新硬币承诺cm_{new}等信息，$\vec{a}$作为零知识证明的隐私输入包含路径信息和硬币数据。通过零知识证明π_{POUR}，保证用户拥有旧硬币，旧硬币的承诺出现在分类账上，新旧硬币和公共值v_{pub}的金额达到平衡。Zerocash通过铸币和注入交易，数据以硬币为载体加密传输，利用简洁非交互式零知识证明保证交易的数据隐私和身份隐私。

5.1.4 具体实现方案

前文已经讨论过零知识证明在智能合约中应用的意义和场景。下面，我们将重点探讨基于零知识证明的智能合约隐私保护方案。这些方案将涵盖如何将零知识证明有效地集成到智能合约中，以及如何利用零知识证明保护合约隐私。我们总结相关文献，具体如表6所示。

表6 基于零知识证明的方案

种类	方案	支持EVM	可信设置	ZKP原语	密码学假设
混币拓展类	Hawk	√	管理者	BCTV14[71]	q-SDH[72]
	Zether	√	无需可信设置	Σ-Bullets[14]	Dlog[72]
	Zexe	×	每个应用程序需单独进行信任设置。	GM17[73]	XPKE[73]
	Veri Zexe	×	零知识证明设置	Plonk[74]	q-Dlog[74]
	Azeroth	√	零知识证明设置	Groth16[65]	q-SDH[75]
	Zapper	×	零知识证明设置	GM17[73]	XPKE[73]
隐私注释语言类	Zkay	√	-	GM17[73]	XPKE[73]
	ZeeStar	√	零知识证明设置	Groth16[65]	q-SDH[75]

注：√表示方案支持，×表示方案不支持，-表示方案未提及。

(1)Hawk

2016年，Kosba等人提出一个分布式的智能合约系统Hawk[12]。方案引入管理员角色执行隐私数据计算。在保证交易隐私性的前提下，Hawk可以确保用户编写隐私智能合约时，无需考虑其具体的密码学实现，提供了可编程性。

Hawk方案的总体架构如图15所示。其主要角色包括用户、管理员和分布式账本。分布式账本为应用提供时间服务、账本、假名和对抗消息重排机制以保证数据的完整性和可用性。管理员角色被允许查看用户数据，负责执行数据计算等相关操作。在Hawk的威胁模型中，假设参与者可能通过违反协议或提前终止交易来获取最大利益。管理员可能偏离协议，甚至与其他参与者合谋，或在协议执行过程中恶意终止，但不会影响程序正确执行。为防范此类风险，Hawk中设置惩罚机制，当管理员恶意中止协议时，用户可以收到相应补偿。

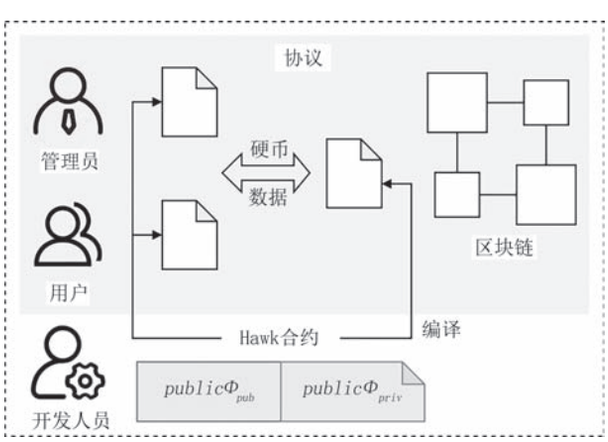


图15 Hawk系统架构[12]

Hawk程序被分为Public和Private两部分(下文使用φ_{priv}和φ_{pub}代替)。φ_{priv}负责接收隐私数据并执行后续计算，φ_{pub}不涉及隐私数据和货币，负责检查输入格式是否正确，以及当有参与方终止合约时，重新分配公共存款。

Hawk结合Zerocash[66]的思想，实现了冻结(Freeze)、计算(Compute)、完成(Finalize)三个基本密码学原语。以投标交易为例，Hawk解释了其协议过程。在冻结阶段，各方不仅提交数据，还可以提交硬币。用户通过承诺和零知识证明，将资金和私有输入冻结在区块链上，保证隐私和正确性。零知识证明确保提交的资金金额和私有输入有效且唯一，避免重复使用。冻结的硬币将由φ_{priv}决定如何支付和分发。在计算阶段，各方将提交的数据和资金使用管理员的公钥加密后发送给管理员。管理员解密这些数据，并根据φ_{priv}执行相关计算。计算完成后，管理员生成零知识证明，确保计算结果基于正确的用户输入并符合合约规则，例如资金总和在输入和输出之间是平衡的。在完成阶段，管理员根据φ_{priv}的计算结果为用户生成新的私有硬币，并提交这些硬币及其正确性的零知识证明给区块链。此时，冻结的硬币通过φ_{pub}被重新分配给用户。

(2)Zether

Bünz等人在2020年提出一种基于账户模型的去中心化隐私支付方案Zether[14]，利用Elgamal加密原语[76]隐藏交易金额。Zether协议以Zether智能合约的形式部署在以太坊上，并且使用Zether令牌(Zether Token)作为Zether账户之间传输的载体。Zether方案的威胁模型假设对手具备观察交易、操纵交易顺序、与智能合约交互以及查询公共状态的能力。主要攻击目标包括超额提款、隐私泄露、前置攻击。方案通过零知识证明确保提款金额合法，采用加密账户余额和隐私机制防止交易信息泄露，并通过延迟确认和批量处理降低前置攻击风险。方案的安全性依赖于密码学原语的安全性。

Zether协议的执行流程如下文所述。首先，创建一个使用Elgamal公钥y进行标识的Zether账户，使用资助交易(Fund)将以太坊账户的加密货币转换成Zether令牌，将以太币存储在智能合约中，并将对应数量的Zether令牌利用同态加密技术添加到账户y的余额。然后，使用传输交易(Transfer)在不同Zether账户传输Zether令牌以实现隐私交易。最后，使用燃烧交易(Burn)将Zether令牌转换为对应以太坊地址的以太币。Zether智能合约支持与任意智能合约进行互操作。由于普通合约没有任何秘密状态，无法生成零知识证明，也无法启动Zether令牌转移。Zether引入锁定(Lock)/解锁(Unlock)功能，允许用户将Zether账户与任意智能合约地址相锁定，以实现Zether与其他智能合约的互操作。

Zether实现了交易金额的隐私，并在其拓展方案中支持匿名传输。当Alice向Bob转账时，选取包含Alice和Bob的共k个账户的匿名集，Alice使用匿名集中所有人的公钥分别对一个基础量(即转账金额)进行加密(只有交易发送方和接收方的公钥加密的两个基础量是非零的，它们的值仅在符号正负上不同)，并使用零知识证明确保构造格式良好。但Zether对加性同态加密的依赖将其功能限制在隐私货币转移和有限的隐私智能合约类别。

Zether中使用优化的零知识证明方案Σ-Bullets保证各项交易中加密的传输余额和发送者余额的正确性。Σ-Bullets通过允许将不同的专用Σ协议(例如一对多证明[77]或累加器证明[78])与通用电路基证明系统BulletProof[70]相结合，以提高零知识证明的性能。在传输交易中，假设用户希望将数量b^*从账户y转移到账户$\bar{y}$。智能合约需要使y的余额减去b^*并使$\bar{y}$的余额加上相同的数量。为在交易过程中隐藏数量b^*，需要分别使用y和$\bar{y}$加密传输金额b^*得到(C,D)和$(\bar{C},\bar{D})$。使用公钥y对发送者余额b'的加密得到(C_L,C_R)。并在相关数据加密的情况下，证明发送方账户余额充足以及发送金额与接收金额一致。使用零知识证明内容如下：

$$st_{Transfer}:\{(y,\bar{y},C_L,C_R,C,D,\bar{C},g;\ sk,b^*,b',r):$$
$$C=g^{b^*}y^r\wedge\bar{C}=g^{b^*}\bar{y}^r\wedge D=g^r\wedge$$
$$C_L/C=g^{b'}(C_R/D)^s k\wedge y=g^{sk}\wedge$$
$$b^*\in[0,MAX]\wedge b'\in[0,MAX]\}$$

以此证明(C,D)和$(\bar{C},\bar{D})$组成良好且都加密相同的数量b^*，交易金额b^*和发送方剩余余额b'为正，g是加密算法中循环群G的生成元。在燃烧交易中用户使用零知识证明证明的内容如下：

$$st_{burn}:\{(y,C_L,C_R,u,b,g,g_{epoch};\ sk):$$
$$y=g^{sk}\wedge C_L=g^bC_R^{sk}\}$$

以此证明用户知道公钥y所对应的私钥sk，以及账户中的加密余额与用户想要换回的金额相等，也就是(C_L,C_R)是使用公钥y对金额b的有效加密，同时又不会泄露金额。

(3)Zkay

Steffen等人在2019年提出了Zkay[13]，其对手模型假设攻击者具备监听、记录、篡改或重放交易的能力，从而窃取用户隐私数据或破坏智能合约的正确性。此外，攻击者还可能通过分析状态变更和数据访问模式的漏洞，推测合约中的私有变量值。Zkay通过引入隐私注释来标识隐私数据的所有者，并将智能合约中的隐私保护机制转换为等效的、可在公共区块链上执行的智能合约，确保隐私数据在去中心化环境中的安全性。

Zkay中将数据类型τ(例如整数和布尔值)扩展到形如$\tau@\alpha$类型，如算法1所示。其中α确定表达式的所有者。表达式的值只能由其所有者看到。所有者α可以是全部(表示该值是公开的)，也可以是类型地址的表达式。所有者为me(msg. sender)的表达式称为自拥有。为防止隐式信息泄露，所有者α的私有表达式不能直接分配给具有不同所有者$(\alpha'\neq\alpha)$的变量。开发人员可以使用reveal方法将表达式赋予其他所有者。

算法1.　zkay中的规范合约Medstats(隐私注释和重新分类以蓝色表示)[13]

```
1: contract MedStats {
2:     final address hospital;
```

```
3:    uint @hospital count;
4:    mapping(address !x → bool@x) risk;
5:
6: constructor(){
7:    hospital = me;
8:    count = 0;
9: }
10:
11: function record(address don, bool@me r){
12:    require(hospital == me);
13:    risk[don] = reveal(r, don);
14:    count = count + (r ?  1 :  0);
15: }
16:
17: function check(bool@me r){
18:    require(reveal(r == risk[me], all));
19: }}
```

为扩展Zkay合约的应用领域，文中提出一种完全自动化的Zkay合约转换机制，以生成可部署在以太坊等公共区块链上的等价公开智能合约。首先，使用调用者提供的加密参数替换私有表达式；然后，使用调用者提供的明文参数替换解密的表达式；最后，要求调用者提供证明这些参数正确性的非交互式零知识证明。生成的公开智能合约如算法2所示。文中使用Zkay赋值、变量声明和布尔表达式（例如等式）约束对证明电路ϕ进行编码。通过非对称加密和使用标准语义解密来增强表达式：表达式$v==enc(x,R,k)$表示使用随机数R和公钥k加密x以产生密文数据类型bin的值v，而$dec_{\tau}(x,k)$表示使用密钥k解密数据类型bin（即密文数据类型）的x，并返回数据类型τ的值。算法3展示了一个证明电路编码的实例。

算法2. 转换的完全公开合约MedStats（转换后新增变量、数据类型和函数以蓝色表示）[13]

```
1: contract MedStats {
2:    final address hospital;
3:    bin count;
4:    mapping(address → bin) risk;
5:
6:    constructor(bin v0, bin proof) {
7:       hospital = me;
8:       count = v0;
9:       verifyχ (proof, v0, pk(me));
10:   }
11:
12:   function record(address don,  bin r,  bin v0,
      bin v1,  bin proof) {
13:      require(hospital == me);
14:      risk[don]= v0;
15:      v2 = count;
16:      count = v1;
17:      verify_Φ (proof, r, v0, v1, v2, pk(don), pk
      (me));
18:   }
19:
20:   function check(bin r, bool v0, bin proof) {
21:      require(v0); verifyψ (proof, r, risk[me],
      v0);
22: } }
```

算法3. 函数record的证明电路φ[13]

```
1: φ(r, v0, v1, v2, pk_don , pk_me ,  sk, R0, R1 ) {
2:    r dec = dec_unit(r, sk);
3:    v0 == enc (r_dec , R0, pk_don);
4:    countdec  = de_cunit (v2, sk);
5:    v1_dec = countdec  + (r_dec ?  1 :  0);
6:    v1 == enc (v1_dec, R1, pk_me );
7: }
```

（4）ZeeStar

2022年，Steffen等人提出ZeeStar[16]，沿用了Zkay[13]的威胁模型。方案包含一门语言和编译器，旨在解决Zkay[13]无法表达涉及外部数据（Foreign Data）操作的问题。Zeestar采用基于Zkay的隐私注释标明隐私数据所有权。方案使用零知识证明验证相关计算的正确性，并使用同态加密执行外部数据计算，以实现隐私约束。

ZeeStar编译分为三个阶段，如图16所示。首先分析输入中的隐私注释，明确数据所有权，自拥有数据相关计算使用私钥解密执行计算后再加密，外部数据相关计算采用同态加密计算；如果输入合约类型良好，ZeeStar将其转换为公共区块链可执行的合约，并收集构造零知识证明电路所需的信息，包括公共输入，隐私输入和相关约束；最后，根据相关输入，构建证明电路。

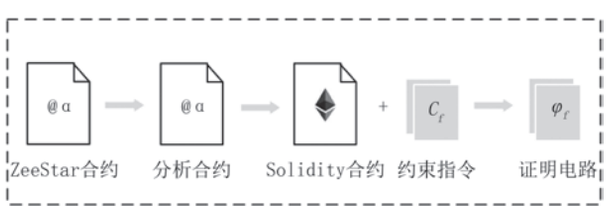


图16 ZeeStar编译过程[16]

上面介绍的工作都仅限于隐藏状态转换的输入和输出，而没有关注转换函数。转换函数的泄漏也

会威胁交易隐私。例如，以太坊目前支持数千个独立的ERC-20“令牌”合约①，每个合约代表以太坊账本上不同的货币。即使这些合约各自采用Zerocash[66]等协议来隐藏令牌支付的细节，相应的交易仍然会揭示正在交换哪个令牌[15]。

(5)Zexe

Bowe等人在2020年提出Zexe[15]。文中提出一种基于UTXO模型的最低程度共享执行环境——记录纳米内核(Record nano-kernel)，以支持引入的新数据结构——记录(Record)其相关操作。

在此基础上，Zexe实现了一种分布式隐私计算方案(Decentralized Private Computation)，以实现区块链对记录纳米内核中隐私保护计算的支持。在Zexe协议中，除了针对交易信息的攻击，攻击者可能试图推测或推断转换函数(即智能合约的执行逻辑)。

记录的组成如图17所示，以Zerocash[66]中的硬币为基础，并扩展两个分别名为出生断言(birth predicate)和死亡断言(death predicate)的脚本(或任意程序)。出生断言确定在什么条件下可以创建记录，死亡断言确定在什么条件下可以消费该记录。记录中还包含公开的一段共享内存——交易备忘录(Transaction Memorandum)，以及保留隐私的一段共享内存——辅助输入(Common Aux Input)。

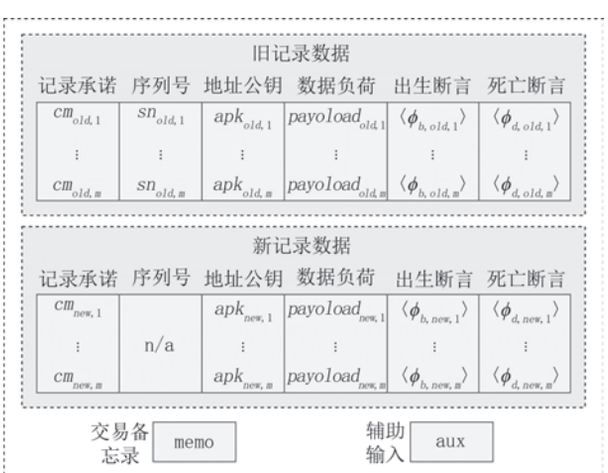


图17　Zexe记录的数据格式[15]

Zexe中提出的分布式隐私计算方案算法如算法4所示。首先，可信方调用DPC. Setup生成公共参数。然后，用户调用DPC. GenAddress创建地址密钥对。用户首先选择一些旧记录消费，并创建一些新记录。新创建的记录包括它们的有效负载和断言以及本地数据等内容，例如交易备忘录(所有断言看到的共享内存并在分类帐上发布)和辅助输入(所有断言看到的共享内存，但不在分类帐上发布)。当用户知道要消费的记录所对应的密钥，且本地数据满足所有相关断言(旧记录的死亡断言和新记录的出生断言)时，用户调用DPC. Execute生成包含零知识证明的交易。

算法4.　ZEXE分布式隐私计算框架[15]

DPC. Setup

Input: security parameter 1^{λ}

Output: public parameters pp

DPC. GenAddress

Input: public parameters pp

Output: addr. key pair$(\mathrm{apk}, \mathrm{ask})$

DPC. Execute^{L}

Input:

1: public parameters pp

2: old $\begin{cases}\text{records}\left[\mathrm{r}_i\right]_1^m \\ \text{address secret keys}\left[\mathrm{ask}_i\right]_1^m\end{cases}$

3: new $\begin{cases}\text{address public keys}\left[\mathrm{apk}_j\right]_1^n \\ \text{record payloads}\left[\mathrm{payload}\right]_j^n \\ \text{record birth predicates}\left[\Phi_{b,j}\right]_1^n \\ \text{record death predicates}\left[\Phi_{d,j}\right]_1^n\end{cases}$

4: auxiliary predicate input aux

5: transaction memorandum memo

Output: new records$\left[\mathrm{r}_j\right]_1^n$ and transaction tx

DPC. Verify^{L}

Input: | public parameters pp and transaction tx

Output: decision bit b

Zexe使用零知识证明保证交易组成格式的正确性和状态符合断言要求。证明内容包括：旧的记录真实存在并且本地数据符合旧记录死亡断言，新的记录正确生成并且本地数据符合新记录出生断言。在Zexe中，零知识证明函数是一个通用函数，用来解释转换函数并根据证明者的隐私输入所导出的指令。这确保隐私数据在计算过程中，对手无法知道函数的具体细节，实现了对于转换函数的隐藏(即功能隐私)。但是Zexe仅实现了对基于UTXO模型中的编程应用程序的支持，当用户可以随时自由加入和离开系统时，Zexe无法在基于帐户的模型中有效地实现数据隐私和功能隐私。

① Vogelsteller et al. "ERC-20 Token Standard". https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20.md.

(6)VeriZexe

Xiong等人[39]提出了一种去中心化隐私计算方案VeriZexe。该方案通过引入通用设置,旨在解决原始ZEXE[15]方案中因依赖非通用的非交互式零知识证明而带来的问题。这种问题导致每个应用都需要生成专门的可信设置,增加了应用部署的复杂性和信任假设。

与ZEXE相比,VeriZexe采用了通用零知识证明系统。只需一次通用的系统设置就可以支持任意数量的应用,而不需要为每个应用分别进行可信设置。VeriZexe通过通用设置和优化的验证电路,有效解决ZEXE在交易生成效率和内存使用方面的瓶颈,同时保持了对交易数据和功能逻辑的隐私保护。

(7)Zapper

2022年,Steffen等人提出一个隐私智能合约系统Zapper[37],允许开发人员使用直观的Python嵌入式前端在类似以太坊的标准编程模型上实现智能合约。在Zapper的威胁模型中,假设攻击者能够拦截诚实用户的交易,进而监听、记录、篡改、伪造或重放交易,从而破坏隐私性或影响账本状态。该方案旨在解决Zexe[15]中存在的诸如应用漏洞、缺乏模块化、需要为每个应用程序使用单独可信设置以及编程模型非标准化等问题。

Zapper包含两个主要组成部分,客户端和执行器。Zapper客户端允许用户创建类和交易,并将用户创建的类转换成Zapper汇编语言Zasm格式提交给Zapper执行器。Zapper执行器存储类和对象,并负责执行交易。当客户端提交给执行器一个类时,后者强制执行多种访问策略以防止恶意代码的威胁。执行器对接收到的Zasm代码执行静态类型分析并对动态指针参数进行动态类型检查。检查 通过后,对相关代码设置内联调用并分配寄存器。

Zapper的总体概览如图18所示,它使用交易表示和更新系统状态。Zapper通过改进基于Zexe[15]

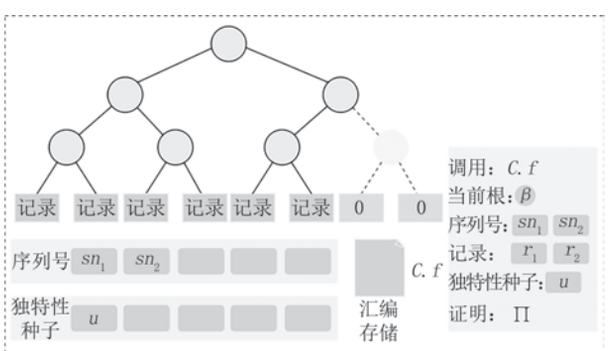


图18 Zapper概览[37]

和Zerocash[66]的技术,使用默克尔树隐藏访问的对象。交易在树中插入新的记录以更新所涉及对象的状态。客户端从对象树的本地副本中获取访问对象的最新记录,在本地模拟Zasm的执行,然后创建交易$tx=(C.f,\beta,[sn],[\hat{r}_{\text{out}}],u,\pi)$。交易由调用函数$C.f$、默克尔树的当前根哈希$\beta$、访问记录的序列号$[sn]$、新生成记录的列表$[\hat{r}_{\text{out}}]$、唯一种子$u$和零知识证明$\pi$组成。零知识证明保证$C.f$正确执行、发送方未被未经授权的用户冒充、$C.f$未被修改。新记录被附加到对象树中,而不替换它们的原始版本,使用序列号隐私识别无效记录。执行器收到交易后,首先进行有效性检查:(1)检查证明π的有效性。(2)检查β是否是对象树的有效根散列。(3)检查$[sn]$中的序列号是否是唯一的,并且尚未出现在序列号列表中。(4)检查u是否尚未出现在唯一种子列表中。检查通过后,将交易中所携带的序列号$[sn]$、种子u、新的加密记录$[\hat{r}_{\text{out}}]$插入相应位置,并将被使用的相应记录的oid字段设置为保留值0,表明该记录已无效。

Zapper利用对象所有权特征和访问控制策略允许独立和模块化地开发类,解决了Zexe[15]系统过于耦合缺乏模块化的问题。并且Zapper只需要为其与应用程序无关的证明电路进行一次可信设置。但是Zapper也存在一些问题,比如:(1)相对于Zexe,Zapper并未实现功能隐私。(2)Zapper只允许用户创建已知所有访问对象的数据的交易。(3)Zapper目前只支持Zasm程序,存在一些功能上的缺失,例如不支持指针算术、自修改代码以及递归等。

(8)Azeroth

Jeong等人提出了Azeroth[40],这是一个基于账户模型区块链的可审计零知识转移框架。Azeroth通过结合零知识证明与加密交易,实现了隐私保护与合法性验证的平衡。它在满足隐私要求的同时,支持对交易的合法性进行检查,保证了交易的透明度和合规性。

Azeroth的设计旨在解决区块链交易中的隐私问题。Azeroth引入了双重账户系统,即普通账户和加密账户。用户可以在这两个账户之间执行存款、取款和转账操作,并确保敏感信息在传输过程中保持机密。此外,Azeroth使用默克尔树累加器来管理加密值的传输和接收。通过默克尔树的不可篡改性,框架确保了加密账户中的值是安全的,并隐藏交易的具体流向和功能类型。Azeroth引入审计机制,攻击者可能获取审计员权限或与其合谋,滥用审计

功能。他们可能伪造交易证明，使非法交易合法化或泄露用户交易信息。恶意审计员还可能选择性拒绝交易，干扰正常交易流程。

Azeroth通过引入双重加密机制和简洁非交互式零知识证明技术，在隐私保护的同时实现可审计性。每笔交易的信息不仅加密给接收方，还同时加密给授权的审计员。具体来说，Azeroth为每笔交易生成一个加密的证明（简洁非交互式零知识证明），其中包括交易的合法性验证、金额一致性和参与方的身份验证等信息。审计员能够解密这些信息并进行独立验证。通过这种设计，审计员可以对交易进行透明的审查，但无法干预或操纵交易内容。此外，Azeroth通过阈值加密技术增强了审计的灵活性和安全性。审计权限可以由多个授权方共同管理，只有当足够多的审计员同意时，才能解密和审查交易。这样设计不仅保证了审计过程的透明性，也确保了隐私保护与监管合规之间的平衡。

5.1.5　属性分析

零知识证明协议保证了智能合约的状态隐私性。在此类方案中（如表7所示），交易的创建、执行、更新状态过程引入了零知识证明生成和验证算法。在智能合约生命周期中，基于所选方案的密码学假设，零知识证明在不泄露其他信息的情况下，确保加密数据输入的正确性，并证明计算过程按预定执行。这保护了智能合约的状态隐私性，且实现了公开可验证性。

表7　基于密码学的隐私保护技术方案属性分析表

分类	实现方案	隐私属性					其他属性					
		假名性	不可链接性	匿名性	状态隐私性	功能隐私	可用性	持久性	完整性	公开可验证性	部分可验证性	共识可验证性
基于零知识证明	Zether[14]2	√	√	√3	√	×	√	√	√	√	√	√
	Zkay[13]	-	-	-	√	×	-	-	-	√	√	-
	Zeestar[16]4	-	-	-	√	×	-	-	-	√	√	-
	Zexe[15]	√	√	√	√	√	√	√	√	√	√	√
	VeriZexe[39]	√	√	√	√	√	√	√	√	√	√	√
	Hawk[12]	√	√	√5	√	×	×	√	×	√	√	√
	Zapper[37]	√	√	√	√	×	√	√	√	√	√	√
	Azeroth[40]	√	√	√	√	√	√	√	√	√	√	√
基于安全多方计算	Absentia[34]	√	-	-	√	-	√	√	√	√6	√	√
	ZkHawk[17]	√	-	-	√	-	√	√	√	×7	×	√
	Gable[36]	√	-	-	√	-	√	√	√	×	×	√
	Ratel[18]	√	-	-	√	-	√	√	√	√8	√	√
	Eagle[19]	√	-	-	√	-	√	√	√	×	×	√
基于同态加密	FHE-Blockchain-based[35]	√	-	-	√	×	√	√	√	×	×	√
	SmartFHE[21]	√	-	-	√	×	√	√	√	×	×	√
	PESCA[20]	√	-	-	√	×	√	√	√	×	×	√
	PriCollabAnalysis[42]	√	-	-	√	×	√	√	√	×	×	√

注：1√表示方案具有属性，×表示方案未实现属性，-表示方案中未提及；

2，4方案中使用同态加密技术，但依据作者描述将其分类为基于零知识证明类型；

3拓展方案中提供匿名性实现方法；

5匿名性作为可选参数；

7能够证明输入和输出余额相等；

6，8方案中借助零知识证明技术实现公共可验证性。

与传统区块链相比，调研文献中多数基于混币思想拓展的方案并未影响数据的完整性、持久性。方案中的相关密码学操作也并未影响授权用户访问数据，保护了数据可用性。唯一的例外是Hawk[12]。它引入了管理员角色，管理员可以访问交易数据，而其他用户无法查看。不诚实的管理员可以在用户不知情的情况下篡改数据，破坏数据的完整性和可用性。此外，不同智能合约隐私保护方案的实现方式导致了相关属性的差异。例如，Zether[14]的拓展方案将发送者和接收者隐藏在匿名集中，确保交易双方信息无法被观察，实现了匿名性。Zexe[15]和VeriZexe[39]方案中交易信息只包括记录消耗和记录创建，隐藏了交易双方身份信息，实现了匿名性。在隐私智能合约中，身份隐私的保护至关重要。它确保参与者在执行合约或交易时不暴露真实身份。这是合约隐私保护的核心之一，

也是隐私智能合约与隐私计算的关键区别。Zexe[15]和Azeroth[40]中零知识证明函数是一个通用函数，隐藏了状态转换函数细节，实现了功能隐私。Zkay[13]和ZeeStar[16]利用具有隐私注释的编程语言实现了公开可验证性。

我们的研究结果表明，基于零知识证明的智能合约隐私保护方案聚焦智能合约的状态隐私性、身份隐私以及可验证性。100%的方案通过在合约的执行过程中引入零知识证明技术，确保了状态隐私性和计算的可验证性。约有60%方案通过结合区块链系统提供数据持久性和完整性。在身份隐私方面，除Zkay[13]和ZeeStar[16]外，剩余方案均实现了假名性、不可链接性和匿名性。尽管各方案在隐私属性和具体实现上存在差异，它们都通过结合零知识证明的优势，有效维护了合约输入、计算和输出的隐私性。

5.1.6 存在的问题

在智能合约隐私保护方案中，研究人员通过引入不同的隐私保护技术以满足相关属性，但也带来了一系列问题。我们根据第2节的步骤二中所提出的性能、经济、可用性、通用性四个维度对方案进行评估。具体的评估结果如图21所示。

(1)性能维度

从性能维度考虑，零知识证明系统的实现效率低下，不仅增加了链上链下计算的成本，也导致隐私交易执行时间的增加。例如，Zkay[13]中Zkay合约转换成等价的公开合约时，示例场景中每笔交易的转换所需时间，其中超过99%的时间用于ZoKrates[79]中的验证者或证明生成；Hawk[12]中对协议进行测试，用户在注入、冻结和计算三个阶段所需的零知识证明生成时间在一分钟左右；使用非通用证明系统[73]的Zexe[15]，用户需要超过40秒才能生成链下计算正确性的证明。在Zapper[37]中，客户端在本地创建交易时，运行时间主要由零知识证明生成阶段主导。该阶段平均占据了99.7%的总运行时间；ZeeStar[16]中提到，为具有2048位密钥的Paillier加密[80]生成基于Groth16[65]的简洁零知识证明，需要超过256 GB的内存空间。

(2)经济维度

零知识证明的生成和验证涉及大量的计算和加密操作，这些操作需要大量的计算资源。在以太坊上，每笔交易都需要为所花费的计算资源支付燃料才能执行。因此，基于零知识证明的隐私保护方案所需要的燃料成本高昂。Zether测试了方案中所提出五种交易的燃料消耗量，包括燃烧交易、资助交易、传输交易、锁定和解锁交易。其中，传输交易的燃气成本为7.188×10^6，远高于其他四类交易。通过分离具体操作，研究人员认为大部分燃料成本是由椭圆曲线操作产生的。传输交易中椭圆曲线操作占总操作的90%，这导致其燃料消耗远高于其他交易。

(3)易用性维度

基于零知识证明的智能合约隐私保护方案要求合约开发人员具有相关密码学知识。例如，Zether协议的主要实现目标是支付领域，它所提出的实施方案主要应用于隐私货币转移和有限的隐私智能合约领域。当应用程序或系统发生更改时，开发人员需要手动调整所涉及的密码原语；Zexe使用基于记录的非标准模型，开发人员需要手动将合约逻辑编码为简洁零知识证明的多个等级1约束系统(Rank-1 Constraint System)。

(4)通用性维度

在基于零知识证明的隐私保护方案中，所生成的证明仅对当前状态有效，一旦状态发生改变，之前生成的证明作废。Zether[14]中提出了一个关于非正常预先交易的问题(Front-running)。例如，Zether传输交易中需要使用零知识证明保证发送方余额为正。用户Alice会根据其当前账户余额生成证明。如果另一个用户Bob将一些Zether代币传输到Alice，Bob的交易首先被处理，Alice余额将发生改变。先前生成的证明不再有效，这导致Alice的交易被拒绝。虽然Bob是非恶意的，Alice仍会失去为交易支付的费用。

零知识证明技术在隐藏交易数据和身份信息方面具有优势，有利于实现智能合约的隐私属性。然而，零知识证明通常适用于特定类型的数学问题或证明。当其与智能合约结合时，由于现实场景中应用领域的需求各异，合约开发人员需要为不同的需求构建不同的证明电路。同时，零知识证明难以实现复杂的功能。例如，Zapper[15]中所使用的零知识证明技术不支持指针算术、自修改代码和递归等功能。零知识证明的生成和验证存在复杂计算开销高的问题，且所需的时间和资源都非常昂贵。这些问题都限制了基于零知识证明的智能合约隐私保护方案的通用性。

5.1.7 解决途径讨论

上文所提出的问题中，性能问题是零知识证明技术固有的问题；而零知识证明技术与智能合约相结合带来了燃料消耗、开发人员门槛要求高、非正常

预先交易等问题。当前学术界针对前文中基于零知识证明的隐私保护方案所出现的问题提出了一些解决方案。

针对零知识证明的性能和所带来的计算成本的问题。Zether中Σ-Bullets协议选择使用支持预编译合约的BN-128椭圆曲线，并根据文献[70]中描述对加密操作进行优化；Hawk[12]通过使用对零知识证明友好的密码原语并构建特定的电路生成器以优化性能。VeriZexe[39]通过引入累积证明方案和多重标量乘法优化等优化技术，降低了交易生成时间和内存使用。尽管不同方案都对零知识证明系统进行了优化，生成证明和验证计算所要求的计算资源仍然相对较高。

为了使不具备专业密码学技术的开发人员也可以开发隐私智能合约应用程序，Hawk提出了一个编译器模型。编译器将程序编译成区块链和用户之间的加密协议，使隐私智能合约具有可编程性；Zkay和其拓展ZeeStar（通过使用加性同态加密支持对外部数据的操作）提出了基于隐私注释的高级编程语言及对应编译器，旨在使没有专业密码学技术的开发人员也可以实现数据隐私。

关于非正常预先交易的问题，Zether[14]方案将所有的资金传输保留在一个挂起（Pending）状态中，这些转账会不时地转入账户，最终完成资金的转移并改变用户状态。为了规定用户状态何时允许改变，Zether将时间分为纪元（Epoch）时期，其中一个纪元由k个连续区块组成，用户需要在纪元开始时发布他们的交易，在每一个纪元结束时，待处理的转账将被转入相应的账户。但是，智能合约的刷新需要用户向其发送交易。每个用户不可能在每个纪元都发送刷新消息，而且他们也无法确定合适的时间发送信息。为了实现对智能合约的刷新，Zether定义了一个滚入（Rock over）方法，用户执行其他方法时需要首先调用滚入方法以刷新自身状态。这在一定程度上会导致用户状态更新的滞后性。而在Zapper[37]中访问不同的并发交易不会相互影响。第三方无权访问交易中涉及的任何对象，因此无法通过尝试消耗相同的对象来执行非正常预先交易攻击或阻止交易。如果两个并发交易访问相同的记录（即读取或写入同一个对象），则分类账拒绝它最后收到的交易。只要所有断言仍然成立且所涉及的对象仍然存在，用户可以重新创建交易并请求执行。

5.1.8 零知识证明安全性讨论

在零知识证明技术的讨论中，安全性是一个关键问题。以下将重点探讨零知识证明系统在实际应用中的安全挑战和应对策略。我们将从数据传输、零知识证明技术自身安全性部分进行讨论。

在基于零知识证明的智能合约隐私保护方案中，数据传输的安全性至关重要。尽管零知识证明技术基于相关的密码学假设，能在不泄露敏感信息的前提下验证数据的正确性，但与外部环境的交互仍存在潜在风险。数据传输中的安全问题主要体现在输入数据（包括用户提交的参数和待验证数据）以及与外部预言机[57]的交互阶段所面临的信息泄露、数据篡改及外部依赖风险。数据传输过程需要通过加密通信来保护用户和零知识证明之间的数据机密性，应用如TLS的密码学应用协议以确保数据在传输中的完整性。此外，选择经过验证的可信预言机可降低隐私泄露的风险，并实施审计机制以确保预言机行为符合预期。通过这些策略，可以有效提升数据传输的安全性，进一步保护用户隐私。

零知识证明技术自身的安全性同样面临重要挑战。首先，一些零知识证明系统需要可信设置阶段，以生成公共参数。这一阶段的安全性依赖于设置者的完全可信。设置者必须确保不会恶意生成参数或泄露秘密信息。系统的安全依赖于设置者的诚信，这增加了信任成本。为了解决这一问题，现代系统引入了安全多方计算来生成公共参数。只要参与者中至少有一方是诚实的，整个系统的安全性即可得到保障。此外，近年来也发展了无需可信设置的方案。例如，Bulletproofs[70]是一种不依赖于可信设置的零知识证明方案，以避免可信设置的信任风险。为了增强可信设置过程的安全性，系统还可以采用可验证的设置过程，保护可信设置过程的公正性和透明性。

其次，针对不同攻击模型的抵抗能力也是零知识证明系统安全性的重要考量。例如，拒绝服务（DoS）攻击是一个常见问题。在这种攻击中，攻击者通过发送大量虚假或无效的证明请求来占用验证者的计算资源。这可能导致验证者无法及时处理合法请求。这种攻击在去中心化系统（如区块链）中尤为危险。因为系统资源有限，攻击者的恶意请求可能严重影响系统性能，甚至导致崩溃。为了解决这一问题，可以采用限制请求频率或引入支付验证机制等缓解策略。在区块链中，验证请求通常需要消耗一定的费用（如燃料费用）。通过这种方式，可以增加攻击者发起恶意请求的成本，从而降低系统遭受拒绝服务攻击的风险。

最后，随着量子计算技术的快速发展，抗量子计算的安全性成为零知识证明领域的重要课题。近年来，抗量子计算的零知识证明方案得到了研究和发展。例如，Ben-Sasson等人提出了基于哈希函数和多项式承诺的零知识可扩展透明知识论证[81]（Zero-Knowledge Scalable Transparent Arguiament of Knowledge，ZK-STARK）。ZK-STARK的安全性依赖于量子计算难以高效破解的哈希函数。因此，ZK-STARK被认为是量子安全的。此外，ZK-STARK具备透明性，不需要可信设置。

5.2 基于安全多方计算的解决方案

5.2.1 安全多方计算技术

1986年，Yao[82]提出了安全多方计算（Secure Multi-Party Computation，MPC）。它是一种隐私保护的计算模型，允许多个参与方彼此之间进行计算，而不暴露各自的隐私数据。

Yao在1986年提出了使用混淆电路实现安全两方计算的协议[82]，这被视为基于混淆电路的安全多方计算的雏形。这种方法的基础是将计算任务转化为一个布尔电路的表示。发送方对该电路进行“混淆”加密，生成一个混淆电路。混淆电路隐藏了原电路的具体结构和参数，但可以根据输入正确计算出结果。发送方将混淆电路传输给其他参与方。参与方利用自己的私有输入，通过一些密码学协议与混淆电路进行交互，最终能够获得正确的计算结果，但无法获知其他参与方的私有输入。另外一种安全多方计算主要实现方式是基于秘密共享，核心思想是将每个参与方的私有输入数据分割并分享给其他参与方。每个参与方只能获得其他参与方输入的一部分信息，而无法单独重构完整的输入。主要涉及三个步骤，一是输入分享阶段，每个参与方将自己的私有输入通过一种特殊的方式分割成多份秘密分享，并分发给其他所有参与方。二是计算阶段，参与方利用获得的秘密分享，按照事先商定的安全协议进行联合计算，但无法查看其他参与方完整的输入。三是重构阶段，通过组合所有参与方的计算结果，重构出最终的输出数据。

2008年Ben-David等人提出FairplayMP[83]，除基于Yao提出的两方计算，还可以执行多方协议。Pinkas等人[84]在2009年首次对高级加密标准（AES）电路进行了主动安全的两方评估。Damgård等人[85]在2012年提出一种基于同态加密及秘密共享等技术实现的安全多方计算的协议框架SPDZ[85]。

5.2.2 MPC保护智能合约隐私的作用机理

在智能合约的生命周期中，合约执行计算（Compute）时，智能合约可以在不泄露任何参与方隐私数据的情况下执行计算。以基于秘密共享的实现方式为例。首先，输入数据经过加密并分割成多个秘密份额，然后在参与方之间分发，部署在区块链上的智能合约也可接收这些加密的秘密份额作为输入。如果输入数据涉及链上资产等情况，需要冻结各方在区块链系统输入数据相关资产。其次，各方执行安全多方计算协议，参与方利用获得的秘密分享，按照事先商定的安全协议进行联合计算，但无法看到其他参与方完整地输入，各参与方通过组合各自的计算结果得到最终输出数据。智能合约负责协调和控制计算步骤，并组合最终输出数据。最后，该输出信息由合约参与方发送到区块链系统，区块链系统对输出信息进行相关的检查（如输入信息与输出信息的匹配等）并处理可能存在的奖励及结算过程。

5.2.3 具体实现方案

前文已经讨论了安全多方计算在智能合约中应用的意义和场景。下面，我们将重点探讨基于安全多方计算的智能合约隐私保护具体实现方案。这些实现方案将涵盖如何将安全多方计算有效地集成到区块链系统中，以及如何利用安全多方计算保护合约隐私。我们总结自2021年到2023年的相关文献，具体如下。

（1）Absentia

Demirag等人在2021年提出一种基于区块链的安全函数计算协议（Secure Function Evaluation，SFE）Absentia[34]。SFE是安全多方计算的一个子集，其中多个参与者具有其他参与者不能获知的隐私输入。该方案是第一个对以太坊实现SFE的协议。

Absentia的主要角色包括用户、Absentia去中心化应用（DApp）以及受托者（Trustee）。受托者是系统的操作者，在链下执行安全多方计算协议，提供数据隐私。用户是系统的使用者，提供加密输入以及即将支付给受托者完成协议的费用。Absentia去中心化应用部署在区块链（以太坊）上，记录并验证每个步骤的输出和相应的零知识证明（证明该步骤正确执行）。区块链为Absentia提供了一个安全的公告板的作用，SFE协议可由区块链系统本身进行公开验证，同时提供一个内置的机制，为参与者所作的工作提供经济补偿。

该协议假设受托者之间是非串通的，不会相互勾结来泄露参与者的隐私，只要有一个诚实的受托者存在，参与者的隐私就能得到保护。协议假设参与者的输入数据是私有的，不会被其他参与者或受托者获知。协议假设区块链具有不可篡改性和抗抵赖性。

(2)zkHawk

zkHawk[17]由Banerjee等人在2021年提出，方案使用安全多方计算协议对Hawk[12]进行拓展。方案中使用安全多方计算协议实现Hawk中管理员角色。针对在安全多方计算程序中执行零知识证明会产生昂贵费用的问题，zkHawk采用一种相对实用的知识证明代替之前的零知识证明，以确保输入输出的余额相等。

该方案针对隐私智能合约的设计提出了两种分类：一是非交互式，由区块链执行智能合约，各方之间无交互；二是交互式的，智能合约在链下执行，各方相互执行安全多方计算协议。其中，zkHawk属于交互式的隐私智能合约协议，合约在链下执行，执行最后的结果再发送到区块链，由区块链进行相关的检查，结算之后关闭合约。

zkHawk主要包括冻结，计算和最终三个阶段。在冻结阶段，每个参与方发送输入代币到区块链系统中，区块链系统对输入代币进行冻结使参与方无法花费；在计算阶段，参与方执行安全多方计算协议，并输出所有各参与方都可以访问的最终结果；在最终阶段，由至少一个合约参与方把计算阶段的结果(包括知识证明)发送到区块链系统，区块链系统检查输入与输出代币之和相等，并将输出代币奖励给各参与方。计算的最终结果在区块链上持久存储，同时通过知识证明实现了一定的可验证性。

方案假设攻击者能够在协议开始前控制一部分智能合约的参与方，这些参与方可能会试图通过恶意行为来破坏协议的隐私性或正确性，攻击者可以试图从参与方获取信息，或者通过参与方发送错误的信息干扰协议的执行。方案假设区块链是一个可信的实体，用于确保协议的可用性和正确性。

(3)GABLE

Cordi等人在2022年提出通过安全多方计算在区块链上进行可审计、可用和弹性的隐私计算方案GABLE[36]。该方案包括一个区块链安全多方计算架构和系统，在区块链上实现了包括混淆电路和混淆有限状态机两种安全多方计算方法，同时规定了参与者的能力和角色。

GABLE的主要角色包括Garbler、输入提供方、输入解锁方、输出接收方、评估方。其中Garbler生成加密函数和输入编码，并将加密函数发布到区块链(以太坊)上，同时将输入编码分发到其他角色。输入提供方为计算提供加密输入。输入解锁方管理加密的输入，防止输入提供者获得有关计算的未经授权的信息。输入接收方从计算中获得指定输出。区块链作为评估方执行智能合约，并接收输入提供方提供的混淆输入。同时，区块链作为一个关于合约执行的不可篡改的公共账本，对混淆输入和输出提供持久性存储。

方案的威胁模型假设Garbler是诚实的，会正确执行任务，并在任务完成后安全地删除其所有状态信息。方案假设输入提供方和输入解锁方不会共谋，对于任何输入，输入提供方和输入解锁方不会同时被恶意参与者控制。

(4)Ratel

Li等人在2023年提出智能合约的安全多方计算扩展Ratel[18]，该方案核心包括一个安全多方计算的原型框架MP-SPDZ[50]以及一个轻量崩溃重置机制。在方案中，机密数据以编码的方式共享给安全多方计算委员会。安全多方计算委员会在不泄露机密数据的前提下，集体对秘密共享数据进行计算，同时与现有区块链系统一起维护机密数据的存储。方案的主要架构如图19所示。

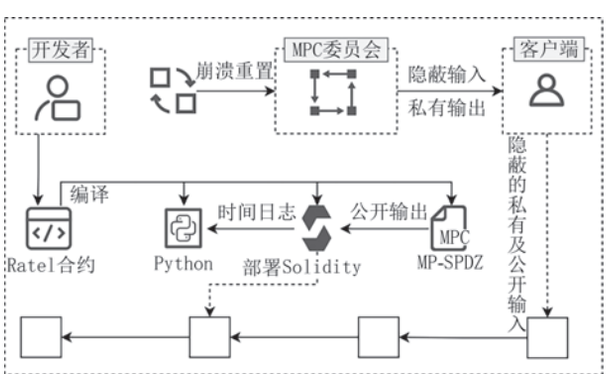


图19　Ratel系统架构[18]

Ratel主要角色包括开发者、应用合约、安全多方计算委员会以及客户端。其中，开发者负责编写Ratel源码并将其编译成目标程序。应用合约部署在区块链系统，充当协调安全多方计算委员会的协调器，同时也作为接收和序列化客户端请求的网关。安全多方计算委员会由安全多方计算节点组成，负责维护秘密共享状态，并根据智能合约指令对

秘密共享数据调用MP-SPDZ程序执行。客户端可以发送安全多方计算请求，提供公共和私有的输入，并从合约中查询公共和授权的私有状态。

该方案使用零知识证明技术，使客户端能够证明他们的私有输入和已知的秘密共享状态满足应用定义的条件，而不泄露实际值。同时，该方案为安全多方计算委员会设计了崩溃重置机制，用于恢复丢失的私有状态更新，使崩溃的节点能够重新参与新的安全多方计算任务。

Ratel的敌手模型假设区块链具有可用性、最终性和完整性。敌手可以完全控制任意数量的客户端。对于MPC委员会(大小为n)，假设敌手可以主动破坏最多t个节点，可以导致最多f个节点出现故障，在满足$t+2f<n$的前提下，系统在同步网络中是安全的。

(5)Eagle

Baum等人在2023年提出高效的智能合约隐私保护方案Eagle[19]。该方案主要针对在安全多方计算中计算密码原语开销大的问题，实现了一个通用的可组合协议。方案包括一种输入协议，可以预处理服务端生成隐私输出(如代币数量)所需的数据，并直接发布到区块链系统中。

Eagle流程主要包括以下几个阶段：初始阶段，服务端初始化系统，包括采样门限签名密钥对，为安全多方计算执行提供足够的担保，初始化合约等；注册阶段，每个客户端将其希望作为隐私智能合约输入的代币转移到区块链系统中；验证输入阶段，服务端对输入进行验证；评估阶段，服务端验证输入后，计算并得到输出，包括每个客户端新承诺的信息份额和随机性；开启阶段，客户端发布上阶段的输出到锁定合约中，锁定合约以承诺的形式生成输出代币，并把机密代币转回客户端的地址；提取阶段，接收到输出的客户端计算出代币承诺；中止阶段，在服务器停止响应或恶意行为的情况下，诚实的服务器可以请求进入中止阶段。

方案假设敌手是静态的，即在协议开始之前，敌手已经确定了要攻击的目标和策略。敌手可以控制MPC服务器和客户端，并可以主动发起攻击，包括篡改输入、拒绝服务及试图泄露隐私信息等。方案假设敌手有如下能力：可以利用网络延迟、丢包等手段干扰协议的正常执行；敌手具有强大的计算能力，可以尝试破解加密算法或进行复杂的计算攻击；敌手可能试图通过各种手段获取用户输入、内部状态或输出信息。

5.2.4 属性分析

安全多方计算实现了智能合约的状态隐私性。在智能合约的计算过程中，各方执行安全多方计算协议，并得到最终输出信息。最后该输出信息由合约参与方发送到区块链系统，区块链系统对输出进行相关的检查，进行合约最终的结束流程。该类方案通过与区块链系统的结合，实现了数据的持久性和完整性。而且方案不会限制用户访问加密后的合约状态，实现了数据可用性。根据智能合约隐私保护方案的评估框架对基于安全多方计算的解决方案的属性分析结果如表7所示。

不同基于安全多方计算的隐私保护方案的实现方式导致了相关属性的差异。Absentia[34]通过区块链记录并验证零知识证明(证明该步骤正确执行)，提供了公开可验证性。Ratel[18]把隐私数据共享给安全多方计算委员会。该方案使用零知识证明技术，使客户端能够证明私有输入和已知的秘密共享状态满足应用定义的条件，实现了公开可验证性。

我们的研究结果表明，基于安全多方计算的智能合约隐私保护方案重点关注智能合约的状态隐私性。所有方案中各方执行安全多方计算协议，隐藏交易数据，确保了状态隐私性。但只有两项方案借助零知识证明技术实现了公开可验证性。

5.2.5 存在的问题

在智能合约隐私保护方案中，研究人员通过引入不同的隐私保护技术以满足相关属性，但也带来了一系列问题。我们根据第2节的步骤二中所提出的性能、经济、可用性、通用性四个维度对方案进行评估。具体的评估结果如图21所示。

(1)性能维度

从性能角度考虑，基于安全多方计算的解决方案主要受到两个因素的影响：一是参与方之间的通信成本，这会影响安全多方计算协议的整体性能。Ratel[18]对性能瓶颈进行分析，发现通信延迟主要来源于安全多方计算程序的在线阶段，其中包含48轮通信，耗时5.12秒，而委员会上的交易纯计算时间约为0.1秒；二是在安全多方计算中计算密码原语的高开销(例如Absentia[34]中引入零知识证明确保计算的正确性)，这也会显著影响方案性能。

(2)经济维度

将安全多方计算与区块链系统结合有两种方式，一是每个节点都参与安全多方计算，二是一组独立的参与方运行安全多方计算，并与区块链系统进行交互。在经济维度，我们从以下几个方面进行衡

量，包括奖励机制的经济模型问题以及和区块链系统交互带来的交易燃料问题。在奖励方面，需要考虑如何鼓励参与方加入系统提供计算资源，以及鼓励参与方消耗一定的费用向区块链系统发送交易；交易燃料方面，如何减少安全多方计算与区块链系统的交互次数，减少交互过程中因发送交易产生的燃料费用。在我们调研的方案中，大部分方案缺乏对交互问题的讨论。例如zkHawk[17]在最终阶段需要由至少一个合约参与方与区块链进行交互，但其未考虑如何激励合约参与方参与该交互。

(3)易用性维度

在易用性维度，与区块链结合方式的不同，会有不同的易用性情况。对于安全多方计算与区块链系统独立的方式，由于两者是异步的，即安全多方计算可在跨越多个区块的间隔期间执行，存在两者之间数据的协调问题。而对于每个节点都参与安全多方计算的方式，则不需要考虑这个问题。一个稳定的安全多方计算系统，要确保能够不间断的计算。当节点出现故障时，因为安全多方计算节点的内部状态是私有的，不能直接被其他节点复制，所以系统需要考虑节点故障时的恢复方案。例如MP-SPDZ在发生任何故障(即使是良性的停止故障)时都会中止计算，导致在节点发生故障时整个系统可能需要重启，影响系统的可用性和持续性。在适用于隐私保护去中心化应用开发的编程模型问题中，针对目前主流的智能合约编程语言，需要设计用于隐私保护的专用语言扩展。在与客户端交互的过程中，方案需要考虑客户端的访问控制问题，包括验证从客户端接收的输入信息，以及只允许授权客户端查询私有状态。

(4)通用性维度

在通用性维度，与区块链结合方式的不同，会有不同的通用性情况。对于每个节点都参与安全多方计算的方式，需要针对不同的区块链系统进行适配，存在一定的兼容性问题。首先是不同区块链平台系统在架构、共识机制、智能合约语言和虚拟机等方面存在差异；其次不同区块链系统的节点性能和资源限制不同，可能影响安全多方计算协议的可行性；最后不同区块链系统对隐私保护的要求和定义可能不同。对于安全多方计算与区块链系统独立的方式，则具有较好的通用性，例如Ratel[18]有一个与区块链系统独立运行的安全多方计算委员会。

5.2.6 解决途径讨论

针对前文中基于安全多方计算的隐私保护方案所出现的问题，当前学术界提出了一些解决方案。

在性能维度，Eagle[19]针对在安全多方计算中计算密码原语开销大的问题，对服务端生成隐私输出所需的数据进行预处理；zkHawk[17]设计了一个相对实用的知识证明来代替零知识证明。在经济维度，Absentia[34]提供了一个内置的机制，为参与者所作的计算工作提供经济补偿。在易用性维度，针对协调问题，区块链系统可提供隐私数据的持久存储。例如Ratel[18]通过安全多方计算委员会对秘密共享数据进行计算，同时与区块链系统共同维护隐私数据的存储。针对节点故障问题，Ratel[18]设计了崩溃重置机制，用于恢复丢失的私有状态，使故障节点能够重新参与到新的安全多方计算任务。针对编程模型问题，对Solidity[43]语言设计一些隐私相关的扩展。例如Ratel[18]可以指定Solidity函数的哪个部分由安全多方计算执行，并使用Python语言连接Solidity和安全多方计算。针对访问控制问题，GABLE[36]制定了参与者的能力和角色，对角色进行访问控制，其中输出接收方角色可以从计算中获得指定的输出信息。在通用性维度，Eagle[19]实现了一个通用的可组合协议，独立的参与方运行安全多方计算有助于兼容不同的安全多方计算方案和区块链系统。

5.2.7 安全多方计算安全性讨论

在基于安全多方计算技术的隐私智能合约的讨论中，安全性是一个关键问题。以下将重点探讨安全多方计算技术在实际应用中的安全挑战和应对策略。我们将从数据输入、合约代码以及安全假设等三个部分进行讨论。首先是数据输入部分。安全多方计算技术在隐私智能合约中的应用主要为了解决互不信任的参与方之间在保护隐私的同时进行协同计算的问题，其首要目的是在计算过程中保护各参与方的隐私输入，确保计算时不泄露任何本地数据。安全多方计算通常涉及密钥的生成、分发和管理，这些过程中的任何疏忽都可能导致安全漏洞，从而导致输入数据的泄露。为解决数据输入部分的安全挑战，我们可以通过将密钥分片分布存储，提高系统的安全性。即使某个参与方的密钥分片被泄露，攻击者也无法恢复出完整的密钥，从而增强了密钥管理的安全性。

其次是合约代码部分。安全多方计算允许多个参与方在不泄露各自输入的情况下共同计算一个函数，但一般智能合约是图灵完备的，支持复杂的函数逻辑。随着函数逻辑的复杂度增加，安全多方计算

协议的设计和实现变得更加困难，容易引入设计上的缺陷。为解决合约代码部分的安全挑战，设计者从编程模型上进行应对，对智能合约语言设计一些隐私相关的扩展，并指定合约中具体的函数或者函数内部具体的逻辑由安全多方计算执行，降低安全多方计算的计算复杂性。

最后是安全假设部分。基于安全多方计算技术的隐私智能合约系统需要在没有可信第三方的情况下安全地进行计算，并能够抵御不同类型的攻击，包括半诚实模型和恶意模型。其中半诚实模型（也称为被动攻击模型）假设参与方会遵守协议的规则，但会尝试从执行过程中获取额外的信息。该模型的安全性通常通过确保没有一方能够从观察协议的执行中获得除了预期输出之外的任何信息来实现。而恶意模型（也称为主动攻击模型）则假设参与方可能会采取任何行动来破坏协议的安全性，包括发送错误信息、不按照协议规则行事、尝试猜测或篡改其他方的输入等。该模型下的安全性要求协议能够抵御各种形式的主动攻击，确保即使在存在恶意行为的情况下，协议的安全性和数据的隐私性仍然得到保障。为解决安全假设部分的安全挑战，在实际应用中，选择哪种模型取决于对参与方行为的预期和信任程度。半诚实模型提供了一种较为简单的安全假设，适用于参与方遵循协议规则但可能试图获取额外信息的场景。而恶意模型则为更为敌对的环境提供了安全保障，适用于需要防御各种潜在攻击的场景。

5.3 基于同态加密的解决方案

5.3.1 同态加密技术

同态加密（Homomorphic Encryption，HE）是一种允许在加密数据上直接进行计算的方案。同态概念由Rivest等人[86]在1978年首次提出，以支持第三方（如云服务提供商）对加密数据执行某些计算功能，并保留加密数据的功能和格式特征。研究人员将后续同态加密算法的实现方案分为三类[87]：(1)部分同态加密（Partially Homomorphic Encryption）只允许一种类型的操作，但没有对用法数量的限制。(2)近似同态加密（SomeWhat Homomorphic Encryption）允许有限数量的多种类型操作。(3)完全同态加密（Fully Homomorphic Encryption）允许无限种类无限数量的操作。

部分同态加密方案所提出的加密算法只具有一种同态特性，加性同态或乘性同态。加性同态方案（如Paillier算法[80]）具有加性同态性质，即对于加密密钥k，明文m_1、m_2，其对应的密文$\mathrm{Enc}_k(m_1)$，$\mathrm{Enc}_k(m_2)$。满足以下性质$\mathrm{Enc}_k(m_1)+\mathrm{Enc}_k(m_2)=\mathrm{Enc}_k(m_1+m_2)$。乘性同态算法（如Elgamal算法[76]和RSA算法[88]）具有乘性同态性质，即满足以下性质$\mathrm{Enc}_k(m_1)\times\mathrm{Enc}_k(m_2)=\mathrm{Enc}_k(m_1\times m_2)$。近似同态加密方案，允许两种类型的操作但限制操作数量。如Boneah等人[89]提出Boneh-Goh-Nissim方案，允许无限次数加法同态运算但只允许一次乘法同态运算。2009年，Gentry[90]提出了第一个完全同态加密解决方案，该方案基于理想格假设。完全同态加密算法随着后续发展，主要包括以Gentry方案（2009）[90-91]为代表的第一代方案，以BFV（Brakerski-Fan-Vercauteren）[92]方案和BGV（Brakerski-Gentry-Vaikuntanathan）[93]方案为代表的第二代方案，以GSW（Gentry-Sahai-Waters）[94]方案为代表的第三代方案以及支持浮点数近似计算的CKKS（Cheon-Kim-Kim-Song）[95]方案等。

5.3.2 HE保护智能合约隐私的作用机理

基于同态加密技术在数据处理方面的隐私特性，其被广泛应用于云计算[96-98]、密文检索[89,99]、电子选举[100-101]等领域。它通过支持数据以加密的格式进行计算，帮助确保计算过程的安全性、隐私性和可信度。隐私智能合约的生命周期中，同态加密技术主要应用于计算（Compute）阶段，数据以加密格式执行同态计算，并生成加密结果。在智能合约中使用同态加密是为了实现数据加密传输，且在不解密数据的情况下完成数据计算，保护数据的隐私性。

5.3.3 具体实现方案

前文已经讨论过同态加密技术在智能合约中应用的意义和场景。下面，我们将重点探讨基于同态加密技术的智能合约隐私保护方案。这些方案将涵盖如何将同态加密技术有效地集成到智能合约中，以及如何利用同态加密技术保护合约隐私。我们总结相关文献，具体如下。

(1)FHE-Blockchain-Based

Chen等人[35]在2021年探讨了将基于理想格的完全同态加密[90]集成到以太坊区块链中的可行性。论文中提出了一个基于完全同态加密的区块链隐私计算框架，旨在实现数据的全程加密计算，提供去中心化的数据隐私保护。

该框架采用链上-链下协作的方式：在链下处理阶段，用户使用完全同态加密的公钥对隐私数据进行加密，并将密文存储在以太坊区块链上，确保数据的不可篡改性。在链上计算阶段，智能合约调用完

全同态加密的运算函数处理密文数据。以太坊虚拟机执行完全同态加密的计算。计算结果仍然是密文，只有拥有私钥的用户才能解密。框架的安全性依赖于链上系统安全性和链上参与者安全性的假设。链上系统安全性依赖完全同态加密方案数学难题的计算复杂性。链上参与者安全性假设所有参与者都是理性的，并且仅追求自身利益最大化。

为验证该框架的可行性，文章通过实现维克里拍卖系统[52]进行了实验。在该系统中，投标者使用完全同态加密的公钥对出价进行加密，并将密文提交至区块链。智能合约在链上对密文进行同态计算，确定最高和次高出价者。最终，只有拥有私钥的用户（如拍卖智能合约的管理者）才能解密结果，确保了投标过程的隐私性和数据的安全性。通过实验，研究量化了在以太坊上执行完全同态加密计算的燃料消耗和计算开销，分析了完全同态加密计算方案在区块链上的实际适用性。

（2）PESCA

Dai在2022年提出PESCA[20]。文中给出一个支持门限加密的拜占庭容错协议的形式化定义，并基于此提出了一个智能合约架构，以实现隐私保护应用程序。该方案的安全模型基于拜占庭假设，攻击者可能控制部分拜占庭节点，干扰共识流程，因此计算的正确性依赖于诚实超多数的共识节点。此外，攻击者还可能通过操纵密钥管理机制，破坏加密数据的安全性。

在PESCA中，分布式节点使用支持Shamir密钥[102]的门限完全同态加密方案[103]计算加密程序状态和输入，计算的正确性依赖于诚实超多数的拜占庭共识节点。所有用户的隐私数据通过全局公钥加密，动态共识节点集使用分布式密钥生成（Distributed Key Generation）方案[104-105]和动态主动秘密共享（Dynamic Proactive Secrete Sharing）方案[105-107]生成和维护全局私钥的Shamir秘密共享。研究人员在文中给出了如何修改共识协议以支持门限加密的办法，并对改进后协议的活性和安全性给出证明。

PESCA所提出的智能合约隐私保护架构将计算分为四类：透明链上计算执行对公开数据的计算，用户链下计算为链上智能合约提供结构化输入，零知识证明计算提供数据有效性检查，隐私链上计算将隐私状态的任何计算表示为完全同态加密电路。用户可以以此架构为基础开发隐私保护应用程序。

（3）SmartFHE

SmartFHE[21]由Solomn等人于2023年提出。它是一个使用完全同态加密支持隐私智能合约的框架，旨在支持轻量级用户使用隐私智能合约。SmartFHE框架使用完全同态加密方案保证输入输出值的隐私性，由分布式节点执行隐私计算，更改区块链状态。用户需要负责加密输入，并使用零知识证明确保输入的正确构造。

SmartFHE的威胁模型假设攻击者可能监听交易元数据、分析状态变化或利用区块链上的公开信息，推测用户隐私数据。攻击者可能篡改、重放或阻断交易，影响智能合约的正确执行或账本状态。此外，完全同态加密计算中的并发问题或密文噪声增长，可能导致隐私泄露或解密失败。

SmartFHE中，研究人员首先提出一个基于公共加密货币（如以太币）的智能合约隐私保护方案。它被视为支持智能合约的隐私保护执行和基于账户模型的分布式账本支付所需的扩展。方案定义包括安全设置、创建账户、创建交易、交易验证、计算、更新状态六个阶段。研究人员在文章中对其所提出框架的正确性和安全性给出了证明。SmartFHE是基于文中提出的智能合约隐私保护框架的完全同态加密技术实现方案。对于公开账户之间的操作（支付和智能合约），SmartFHE采取与以太坊相同的方式处理。对于涉及隐私账户的隐私支付，SmartFHE允许账户所有者发起隐藏传输值或用户余额的交易，并使用零知识证明保证交易输入格式良好。交易中公开账户的余额改变，直接进行明文计算。交易中隐私账户的余额改变，通过同态加密方法Priv.HomAdd(pk, $\vec{b}$, $\vec{c}$)计算。其中pk为账户公钥，$\vec{b}$代表账户加密余额，$\vec{c}$代表传输金额。

SmartFHE的总体概览如图20所示，允许用户编写智能合约以执行对私有数据和私有账户余额的操作。智能合约提供给用户可以调用的函数以操作其输入数据。当输入为隐私类型时，函数内的操作将转换成对应的同态计算。相关代码将根据使用的完全同态加密方案的类型转换为算术或布尔电路。由于隐私智能合约直接对加密数据进行操作，用户需要提供零知识证明确保用户输入数据格式良好。分布式节点检查这些零知识证明，并直接在密文上执行请求的同态计算，并相应地更新区块链状态。

在SmartFHE中，数据以加密格式由分布式节点执行计算。与使用链下计算的方案[13,15-16]相比，

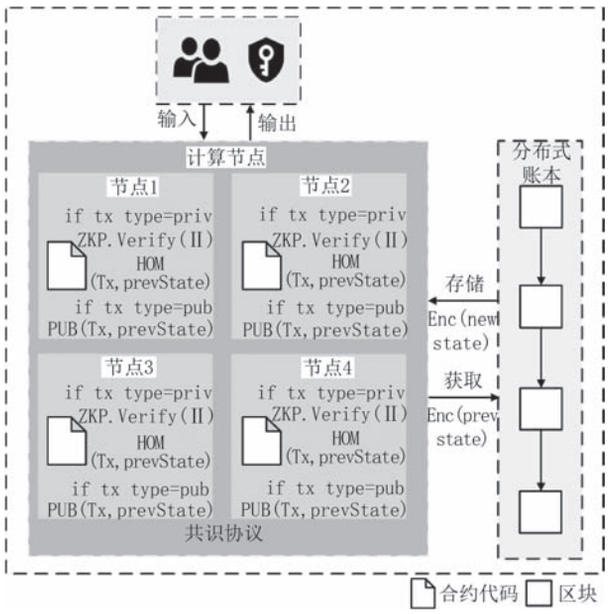


图20 SmartFHE概览[21]

SmartFHE的一个优势是，每个用户只关心自己的输入，用户数量的增加不会影响每个用户的成本。因此，SmartFHE支持轻量级用户使用，用户只需确保数据格式符合要求，而无需提供复杂的零知识证明来确保整个计算的正确性。然而，随着用户数量的增加，分布式节点需要处理的信息也会增多。

(4)PriCollabAnalysis

Tawfik等人在2024年提出PriCollabAnalysis[42]，一种基于区块链的隐私保护医疗协作分析框架。该框架核心采用秘密共享[102]、同态加密[80]和安全多方计算[108]技术，使医疗数据在全程加密状态下完成计算，确保数据隐私，同时支持医疗机构间的协作分析。

医疗数据首先通过对称加密存储于分布式文件系统。加密密钥采用公钥加密保护，确保数据存储安全。多个医疗机构使用秘密共享方案对计算所需的数据进行分片，使得单个机构无法获取完整数据，从而增强隐私保护。智能合约负责协调安全多方计算过程，在计算过程中利用加法同态加密进行数据聚合，确保计算结果在链上存储和验证的同时仍保持加密状态。

该框架包含三个核心部分：链上隐私计算部分通过智能合约使用加法同态加密进行数据聚合计算(如求和、均值计算)，确保计算全程在密文域中进行，避免数据泄露。链下数据预处理部分负责在分布式文件系统中加密存储数据，并管理秘密共享密钥，确保医疗数据在链下安全存储并受到访问控制。基于同态加密的安全多方计算部分允许多个医疗机构在无需彼此信任的情况下联合执行隐私保护的数据分析，利用秘密共享和同态加密技术安全地计算全局统计信息，从而实现高效、安全的数据协作。

5.3.4 属性分析

同态加密技术确保了智能合约的状态隐私性(分析结果如表7所示)。在智能合约的计算过程中，使用完全同态加密技术以加密格式处理数据，从而保护合约状态的隐私性。然而，在完全同态加密方案中，评估方与加密输入的所有者不同，输入数据所有者无法直接验证加密计算的正确性。

我们的研究结果表明，基于同态加密的智能合约隐私保护方案重点关注智能合约的状态隐私性。所有方案中数据以加密的格式进行计算，确保了状态隐私性。但由于同态加密技术的固有特性，方案均无法实现公开可验证性。

5.3.5 存在的问题

在智能合约隐私保护方案中，研究人员通过引入不同的隐私保护技术以满足相关属性，但也带来了一系列问题。我们根据第2节的步骤二中所提出的性能、经济、可用性、通用性四个维度对方案进行评估。具体的评估结果如图21所示。

性能维度

在当前的完全同态加密方案中，存在多种影响性能的因素。例如，现阶段多数完全同态加密方案[92-94]是以格密码为基础进行构造的。格密码在密文中引入一个噪声分量以增强安全性。当密文经过一系列同态操作后，噪声分量会累计，导致密文解密失败或解密结果不可信。Gentry[90]提出了引导(Bootstrapping)技术以处理噪声问题。但是，基于算术的完全同态加密方案[92-94]引导过程往往非常慢，通常需要几分钟[109]；基于布尔的完全同态加密方案(TFHE[110])的引导过程相对迅速，但其支持精度低。TFHE及其变体通常难以支持超过20位的精度[111]，这导致其在区块链中的实际应用效率低下。此外，在同态操作过程中，密文、密钥等维度急剧膨胀也会导致方案出现性能问题[112]。Doan等人[113]对完全同态加密方案BFV[92]、BGV[93]、CKKS[95]的多种实现库Seal、PALISADE和Helib进行了测试。测试结果显示，现有完全同态加密方案在同态加法和同态乘法的执行速度明显慢于部分同态加密方案。

同态加密适用于加密数据的计算，但计算执行

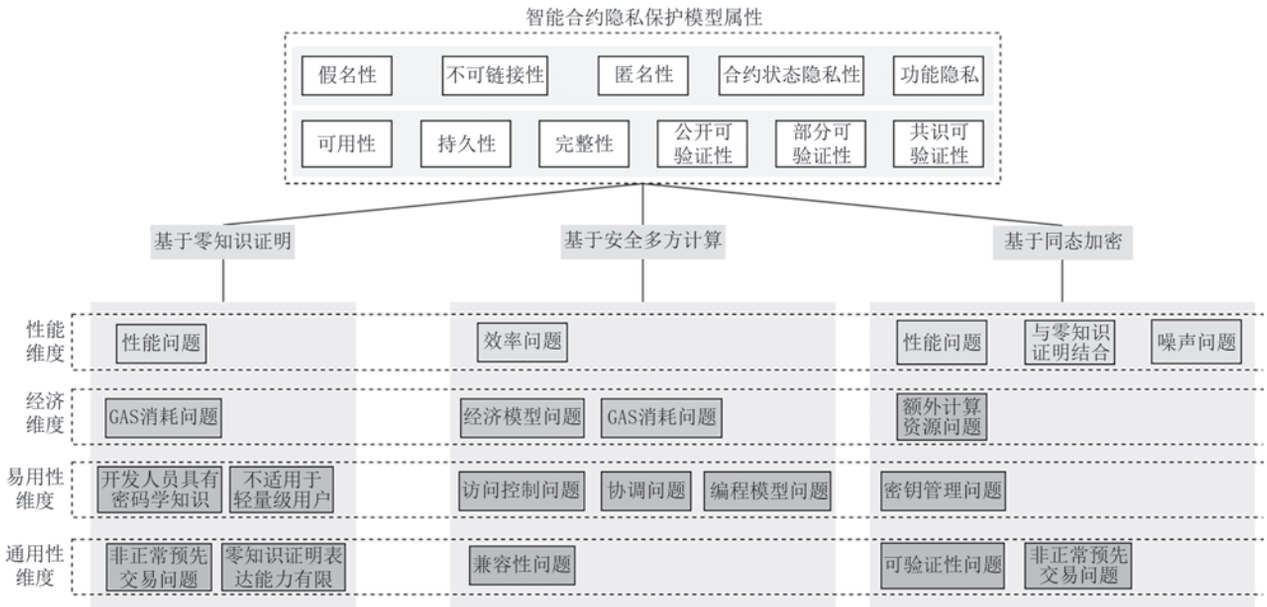


图21　基于密码学技术的解决方案存在的问题

者无法判断加密输入是否符合合约要求，需要引入零知识证明保证输入格式的正确性。在智能合约中，同态加密与零知识证明相结合也带来了相应的性能问题。使用基于格的零知识证明系统结合同态加密技术构建智能合约隐私保护方案，虽能提供后量子安全性，但面临昂贵的存储代价。尽管基于格的零知识证明生成速度较快，但其证明大小通常在数百千字节到数兆字节之间[114-115]，这使得它们难以在区块链中工作。相比之下，基于椭圆曲线或哈希的零知识证明系统可以节省存储空间，但代价是更长的证明生成时间。

经济维度

在基于同态加密的方案中，区块链上节点需要执行额外的密码学协议以实现同态计算。相对于传统智能合约架构，这需要额外的计算资源。Chen等人[35]指出，在完全同态加密方案中，比特级处理时，单个布尔运算和算术运算的燃料消耗与明文计算相近，但重加密操作显著增加了燃料费用。例如，比特级交换操作涉及4次XOR、4次AND、2次加密和5次重加密，每次执行燃料消耗超过1×10^6。在进行数字级处理时，操作数量（除固定操作外）均随比特数增加而成比例增长。以完全同态加密方案下的维克里拍卖为例，其排序算法需要调用(n−1)+(n−2)次交换操作，这使得燃料消耗随竞标人数增长而大幅增加。实验表明，即使仅有5名竞标者，拍卖的燃料消耗约为8×10^7。

易用性维度

同态加密技术应用于智能合约中，不同角色间如何安全地管理和共享密钥是系统创建者需要面临的关键问题。例如，在PESCA中，隐私数据使用同一个密钥加密，密钥管理不当所导致的隐私泄露问题会更加严重。

通用性维度

在完全同态加密方案中，评估方与加密输入的所有者不同，输入数据所有者无法在不重复计算的情况下验证加密数据计算的正确性。虽然Fiore等人[116]提出了一种加密数据的可验证计算方案，但其实现的成本昂贵。此外，同态加密技术应用于加密数据计算，通常需要引入零知识证明保证加密输入格式的正确性。对隐私状态（例如加密账户余额）进行操作会导致并发问题，即5.1.6中提到的非正常预先交易问题。账户状态的变化会导致与该账户相关的所有挂起交易的零知识证明无效。

5.3.6　解决途径讨论

当前学术界针对前文中基于同态加密的隐私保护方案所出现的问题提出了一些解决方案。

针对噪声问题，SmartFHE[21]提出了一种结合零知识证明的替代技术以提供比引导更好的性能。以用户的加密余额b为例，调用方法$b',\pi\leftarrow$Priv. RefreshBal(sk_{priv}, b)产生新的加密账户余额b'。用户使用密钥sk_{priv}解密其私人账户余额b以获得明文值bal，然后重新加密bal产生b'。用户将需要提供一个零知识证明π，以表明b'加密的明文与b中加密的明文相等。

对于零知识证明与同态加密相结合产生的存储和性能问题，SmartFHE采用分级的BFV方案[117]作为同态加密方案，BulletProof方案[70]作为零知识证明方案，使用Bulletproof来证明明文的属性（例如秘密值在特定范围内）。BFV方案基于Ring-LWE[118]难题构造，因此可采用短离散对数证明[119]。其中的Pedersen承诺可用于构造BulletProof。SmartFHE凭借这一措施在保证量子安全性的同时，降低证明生成和存储的开销。

针对密钥管理的问题。在PESCA[20]中，区块链分布式节点使用支持Shamir密钥[102]的门限完全同态加密方案。Dai根据区块链网络中节点动态加入和退出的特点，提出使用分布式密钥生成方案[104-105]和动态主动秘密共享[105-107]方案在动态共识集合中生成和维护全局私钥的Shamir秘密共享。但这会增加区块链节点的通信复杂度，从而导致成本增加。

针对同态计算的可验证性问题，区块链通过共识提供了一种简单的解决方案。大多数计算节点诚实的假设为同态计算的正确性提供了保证[12]。针对非正常预先交易问题，SmartFHE[21]引入隐私交易的自动余额滚动和智能合约的隐私帐户锁定机制两种技术。所有隐私帐户的资金传输都被保存为挂起状态直到纪元结束。SmartFHE将在纪元结束时自动将这些挂起的资金转入隐私账户余额（Zether[14]要求用户触发滚动）。为了解决多纪元中的并发问题，SmartFHE允许将隐私帐户锁定到任意类型的其他帐户（通过tx_{lock}交易）。锁定机制允许用户根据需要将其帐户挂起，以防止在其隐私交易仍处于挂起状态时对其余额进行任何更改。被锁定者可以发出tx_{lock}交易以接受新的状态更新，从而将锁定帐户的完全控制权返回给锁定者。

5.3.7 同态加密技术安全性讨论

在讨论同态加密技术在隐私智能合约中的应用时，安全性是关键问题。以下将探讨同态加密技术在实际应用中的安全挑战和应对策略，主要聚焦于同态加密系统自身的安全性问题。

首先，噪声管理是同态加密系统中的一个安全挑战。在完全同态加密中，每次加密运算都会累积一定的噪声，尤其是乘法运算会导致噪声急剧增加。如果噪声增长得过快，最终会导致密文无法被正确解密。为了解决这一问题，经典的解决方案之一是引导，它通过重新加密已经计算过的密文来消除噪声。虽然这个过程非常耗费计算资源，但它是确保同态加密可以执行无限次运算的关键技术。此外，近年来也出现了不需要频繁重加密的优化方案。例如，Brakerski-Gentry-Vaikuntanathan（BGV）[93]和Cheon-Kim-Kim-Song（CKKS）[95]等方案，通过调整加密过程中的噪声控制，减少了重加密的频率和复杂度，从而提高了同态加密的效率。

其次，计算效率与安全性平衡也是同态加密系统安全性的重要考量。全同态加密提供了非常高的安全性，但这通常伴随着极高的计算成本，尤其是在处理大规模计算时，运算时间和所需资源急剧增加。为此，研究者提出了多种优化方案，例如在特定应用场景中使用部分同态加密方案以提高计算效率。以Paillier[80]加密和ElGamal[76]加密为代表的部分同态加密系统被广泛用于电子投票和隐私保护的货币交易系统中，它们在保证安全性的同时，显著减少了计算开销。

另外，密钥管理是同态加密系统中的一个安全性问题。在多数同态加密方案中，密钥的安全性直接影响系统的整体安全性。然而，管理和保护这些密钥，尤其是在分布式系统中的密钥，是一个重要的研究问题。为增强密钥管理的安全性，系统可以采用分布式密钥生成和管理技术[104-105]。这可以确保密钥不会存储在单个节点上，从而降低了密钥泄露的风险。虽然这种技术提高了同态加密系统的安全性，但也增加了系统的复杂性。

最后，随着量子计算技术的快速发展，抗量子计算的安全性成为同态加密领域的重要课题。以Brakerski-Gentry-Vaikuntanathan[93]方案为代表的基于格密码的同态加密方案，通过依赖难以被量子计算破解的数学问题（如容错学习问题）提供了有效的防御。因此，这些方案被认为具有量子安全性，能够在未来量子计算机出现时继续保障同态加密系统的安全性。

6 结论与建议

结合第4节和第5节的分析，本节探讨现有隐私保护技术的启示及未来发展。

6.1 本文启示

启示一：依赖可信执行环境技术的方案存在诸多安全挑战。解决这些问题不仅能促进隐私智能合约的广泛应用，也有助于推动可信执行环境技术的进一步发展。我们将部分问题整理如下：

（1）侧信道攻击风险

我们的调研表明，当前基于可信执行环境的隐

私保护方案在侧信道攻击防护方面仍显不足，合约隐私信息仍面临泄露风险。即使TEE完全安全，攻击者依然可以从链上合约的调用过程中推测出有效的隐私信息。例如Jean-Louis等人[120]在不破坏可信执行环境的情况下发现的基于可信执行环境智能合约平台的隐私漏洞，包括处理持久化合约存储时访问模式泄漏以及状态一致性相关的问题。

(2)权衡可用性和安全性

单个可信执行环境无法保证高可用性。一旦TEE的宿主机出现宕机、掉线、被攻击等情况，TEE的隐私状态将无法被访问。为了提升系统的高可用性和抗攻击能力，基于可信执行环境的智能合约隐私保护方案通常要求建立TEE集群。但此类集群增大了攻击面，潜在安全风险变大。在我们调研的基于可信执行环境的隐私保护方案中，尚未发现能同时平衡高可用性和高安全性的方案。

(3)异构可信执行环境管理

如节4.1所述，可信执行环境有多种异构技术路线。在我们调研的基于可信执行环境的隐私保护方案中，部分方案单独使用Intel SGX[28-29,31-32]或ARM TrustZone[33,38]组成可信执行环境集群。兼容异构、多样化的可信执行环境技术能够进一步提升集群稳定性，但如何管理异构可信执行环境需进一步研究。具体来说，一是使用可信执行环境来实现隐私保护的智能合约需针对每种技术进行适配；二是异构可信执行环境有不同的远程认证方式，兼容或是设计统一的认证方式是一个需要权衡的考量；三是可信执行环境集群进一步增加了分布式密钥管理的复杂程度。这些因素共同影响异构可信执行环境的管理和操作，需要综合考虑以实现高效且安全的系统运行。

启示二：基于单一密码协议的智能合约隐私保护方案有局限性，密码协议和可信执行环境技术的结合是具有前景的方向。我们从三个角度总结局限性：

(1)零知识证明技术在隐藏交易数据和身份信息方面具有优势，有利于实现智能合约的隐私相关属性。但是，零知识证明通常适用于特定类型的数学问题或证明，当与智能合约相结合，现有方案可能难以直接适用。比如Zapper[37]中所使用的零知识证明技术不支持指针算术、自修改代码和递归等功能。此外，证明电路通常仅适用于特定应用程序，应用场景发生改变后，需要重新构造证明电路。而且，这项技术还存在证明过程繁琐、构造复杂、表达能力有限等问题。这些问题限制了基于零知识证明的智能合约隐私保护方案的适用范围。

(2)安全多方计算系统由多个节点组成，每个节点都持有计算过程中的私有状态信息。节点之间进行大量的交互和通信，这会带来额外的计算和带宽开销。此外，节点内部状态具有私有性，无法直接被其他节点复制。因此，安全多方计算系统一般通过节点故障时的恢复机制，以确保计算过程不会中断。例如，Ratel[18]改进了MP-SPDZ，在检测到偏差时会重新启动停止的节点。同时，它为安全多方计算委员会设计了崩溃重置机制，用于恢复丢失的私有状态更新，使崩溃的节点能够重新参与新的安全多方计算任务。但是，该机制增加了用于协助崩溃节点恢复的其他节点的带宽开销及额外资源消耗。

(3)基于完全同态加密的方案在适用范围方面存在诸多限制。在完全同态加密方案中，评估方与加密输入的所有者不同，输入数据所有者无法在不重复整个计算的情况下验证计算的正确性。这一因素使其在需要高可信度和可验证计算结果的场景中不适用。例如，由于交易双方需要对交易过程和结果进行验证，以确保交易的准确性和公正性，基于完全同态加密的方案并不适用于金融交易场景。

启示三：探索如何通过经济模型提升基于可信执行环境的隐私智能合约的可用性，是未来研究的关键方向之一。单个可信执行环境的可用性高度依赖于其操作者。若操作者进行恶意行为如宕机或断网，可信执行环境将无法提供服务。为了解决这一问题，引入经济模型成为一种有效的策略。例如，FASTKITTEN[10]实施了一种质押惩罚机制，通过没收作恶操作者的质押资金来增加其违规成本。该经济模型有效激励操作者确保服务持续稳定，从而减少恶意行为。然而，这种方法未从根本上解决可用性问题。未来的研究需关注以下几点：(1)深入研究如何精确设定经济模型中的质押金额，确保在激励和惩罚操作者的同时，不影响资金提供方的参与意愿。(2)探索在操作者作恶时如何迅速切换至新的可信执行环境，以确保系统的连续可用性。

6.2 建议和未来工作

通过总结上述启示，我们得出当前智能合约中常用的隐私保护技术对比，如表8所示。结合前文，我们对隐私智能合约的未来研究方向有以下几条建议。

建议一：具体应用场景中需要结合多种方案来实现隐私保护。不同隐私保护技术各有其适合的应

表8 智能合约隐私保护技术总结

隐私技术	应用场景	优点	局限
可信执行环境	保障执行环境安全	执行环境相对独立，抵抗硬件攻击，通用性强，性能良好	依赖硬件安全假设，生态转移能力差
零知识证明	交易隐私证明	维护安全性和隐私性，证明信息未被篡改，真实有效防泄漏	需要开发人员具有密码学相关知识，计算性能差，表达能力有限
同态加密	加密数据计算	交易加密和计算分开，交易数据加密不可见，节省链上成本	计算速度慢，无法满足可验证性
安全多方计算	去中心化计算	在保护信息隐私和安全的情况下，实现去中心化计算	计算开销大，效率低，多方协作难度大

用场景、优势和局限性。因此，未来研究的一个重要方向是根据智能合约的具体应用场景，结合多种方案定制不同的隐私保护策略。例如，在基于区块链的医疗数据共享平台中，结合具体隐私需求，可以采取如下措施：在需要进行数据分析时，使用同态加密技术直接对密文进行计算；通过零知识证明确保上传到链上的加密数据的有效性；并在可信执行环境中处理敏感数据和计算任务，在多方联合数据分析的场景下，安全多方计算技术可实现不同机构间的隐私保护数据共享。

建议二：利用软硬件一体架构实现智能合约隐私保护架构。单一类型隐私保护技术存在安全上限。例如，可信执行环境安全假设过于严格，传统密码协议缺乏灵活性。因此，如何通过软硬件结合的方式，实现具有高安全性和高灵活性的隐私保护架构也是该领域未来的一个重要研究方向。我们将两个潜在的结合方向总结如下：

(1)使用可信执行环境提升零知识证明在隐私智能合约中的适用场景。具体讲，为了缓解零知识证明高计算资源需求对轻量级用户的影响，可以将计算任务外包给服务器。然而，这种做法面临外包服务器的不受信任问题。而通过采用具备可信执行环境的服务器，能够确保零知识证明的生成过程安全可信。在隐私智能合约中，可信执行环境服务器可用于零知识证明生成，用户可将该计算委托给这些服务器。它们提供隔离和受保护的计算环境，确保证明生成过程的安全和隐私。由于可信执行环境内部的计算和数据受到保护，用户无需担心数据泄露或被篡改。这减轻了本地设备的计算负担并能够利用强大的外部资源进行复杂计算。这种结合可信执行环境和零知识证明的方式，为隐私智能合约提供了更高效且安全的计算支持。

(2)使用证明协议优化可信执行环境的安全性。在隐私智能合约中，私密状态数据在可信执行环境中存储和计算。为应对可信执行环境因退出或离线导致的数据泄露风险，可利用密码学技术生成数据销毁证明，从而确保敏感信息在离开可信执行环境时已被安全删除。此过程包括在可信执行环境内部生成一个证明，验证所有敏感数据已被销毁且无法恢复。这种证明可以包括操作日志、加密销毁标记或零知识证明，确保数据销毁操作的正确性和完整性。这些密码学证明为用户提供可验证的证据，证明数据销毁的操作执行完毕。

(3)引入可信执行环境作为中立且可信的第三方，以突破基于安全多方计算的智能合约隐私保护方案性能瓶颈。这样能够降低由于执行智能合约的安全多方计算的参与方需要相互之间建立信任带来的通信成本，并借助可信执行环境提供的安全硬件隔离环境，提高运算效率，支持复杂的算法逻辑。

建议三：针对智能合约开发人员进行培训，该领域需要更具有专业能力的隐私合约开发人员。在智能合约隐私保护方案中，对于隐私性保障的责任，系统创建者和合约开发者之间并无明确定义。在我们调研的结果中，未发现能够自动化实现对具有隐私漏洞的智能合约进行检测和修复的方案。因此，为了确保智能合约中隐私得到有效保护，建议针对开发人员进行系统化的培训。内容应涵盖以下几方面：首先，开发人员需要掌握隐私技术相关基础知识，包括可信执行环境、同态加密、零知识证明和安全多方计算等技术原理。其次，培训应包括隐私智能合约的编写最佳实践案例，例如如何高效实现隐私保护功能，如何避免常见的隐私泄露风险等。最后，开发者还应了解常用隐私保护工具和框架的使用方法，以便在开发中选择合适的技术方案。

参 考 文 献

[1] Nick S. Formalizing and securing relationships on public

networks. First Monday, 1997, 2(9): 1-33

[2] Buterin V. A next-generation smart contract and decentralized application platform. White Paper, 2014, 3(37): 2-1

[3] Tolmach P, Li Y, Lin S W, et al. A survey of smart contract formal specification and verification. ACM Computing Surveys (CSUR), 2021, 54(7): 1-38

[4] Tapscott D, Tapscott A. Blockchain revolution: How the technology behind bitcoin is changing money, business, and the world. Penguin Group, 2016

[5] Griggs K N, Ossipova O, Kohlios C P, et al. Healthcare blockchain system using smart contracts for secure automated remote patient monitoring. Journal of Medical Systems, 2018, 42: 1-7

[6] Christidis K, Devetsikiotis M. Blockchains and smart contracts for the internet of things. IEEE Access, 2016, 4: 2292-2303

[7] Tso R, Liu Z Y, Hsiao J H. Distributed E-voting and E-bidding systems based on smart contract. Electronics, 2019, 8(4): 422

[8] Li R, Galindo D, Wang Q. Auditable credential anonymity revocation based on privacy-preserving smart contracts// International Workshop on Data Privacy Management. Cham: Springer International Publishing, 2019: 355-371

[9] Li R, Wang Q, Liu F, et al. An accountable decryption system based on privacy-preserving smart contracts//Proceedings of the Information Security: 23rd International Conference, ISC 2020, Bali, Indonesia, 2020: 372-390

[10] Das P, Eckey L, Frassetto T, et al. {FastKitten}: Practical smart contracts on bitcoin//Proceedings of the 28th USENIX Security Symposium (USENIX Security 19. ClaraSanta, USA, 2019: 801-818

[11] Cheng R, Zhang F, Kos J, He W, et al. Ekiden: A platform for confidentiality-preserving, trustworthy, and performant smart contracts//Proceedings of the 2019 IEEE European Symposium on Security and Privacy (EuroS&P). Stockholm, Sweden, 2019: 185-200

[12] Kosba A, Miller A, Shi E, Wen Z, et al. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts//Proceedings of the 2016 IEEE symposium on security and privacy (SP). San Jose, USA, 2016: 839-858

[13] Steffen S, Bichsel B, Gersbach M, et al. zkay: Specifying and enforcing data privacy in smart contracts//Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. London, UK, 2019: 1759-1776

[14] Bünz B, Agrawal S, Zamani M, et al. Zether: Towards privacy in a smart contract world//Proceedings of the International Conference on Financial Cryptography and Data Security. Cham: Springer International Publishing, 2020: 423-443

[15] Bowe S, Chiesa A, Green M, et al. Zexe: Enabling decentralized private computation//Proceedings of the 2020 IEEE Symposium on Security and Privacy (SP). San Francisco, USA, 2020: 947-964

[16] Steffen S, Bichsel B, Baumgartner R, et al. Zeestar: Private smart contracts by homomorphic encryption and zero-knowledge proofs//Proceedings of the 2022 IEEE Symposium on Security and Privacy (SP). San Francisco, USA, 2022: 179-197

[17] Banerjee A, Clear M, Tewari H. zkhawk: Practical private smart contracts from mpc-based hawk//Proceedings of the 2021 3rd Conference on Blockchain Research & Applications for Innovative Networks and Services (BRAINS). Paris, France, 2021: 245-248

[18] Li Y, Soska K, Huang Z, et al. Ratel: Mpc-extensions for smart contracts//Proceedings of the 19th ACM Asia Conference on Computer and Communications Security. Singapore, 2024: 336-352

[19] Baum C, Chiang J H, David B, et al. Eagle: Efficient privacy preserving smart contracts//International Conference on Financial Cryptography and Data Security. Cham: Springer Natrue Switzerland, 2023: 270-288

[20] Wei D. Pesca: A privacy-enhancing smart-contract architecture. https://eprint.iacr.org/2022/1119

[21] Solomon R, Weber R, Almashaqbeh G. smartfhe: Privacy-preserving smart contracts from fully homomorphic encryption// Proceedings of the 2023 IEEE 8th European Symposium on Security and Privacy (euroS&p). Delft, The Netherlands, 2023: 309-331

[22] Peng L, Feng W, Yan Z, et al. Privacy preservation in permissionless blockchain: A survey. Digital Communications and Networks, 2021, 7(3): 295-307

[23] Perez A J, Zeadally S. Secure and privacy-preserving crowdsensing using smart contracts: Issues and solutions. Computer Science Review, 2022, 43: 100450

[24] Li R, Wang Q, Wang Q, et al. SoK: TEE-assisted confidential smart contract. Proceedings on Privacy Enhancing Technologies, 2022, 3: 711-731

[25] Hu Tian-Yuan, Li Ze-Cheng, Li Bi-Xin, et al. Review of contractual security and privacy security of smart contracts. Chinese Journal of Computers, 2021, 44(12): 2485-2514 (in Chinese)
(胡甜媛，李泽成，李必信，等. 智能合约的合约安全和隐私安全研究综述. 计算机学报，2021, 44(12): 2485-2514)

[26] Li F, Li H, Niu B, et al. Privacy computing: Concept, computing framework, and future development trends. Engineering, 2019, 5(6): 1179-1192

[27] Antunes R S, André da Costa C, Küderle A, et al. Federated learning for healthcare: Systematic review and architecture proposal. ACM Transactions on Intelligent Systems and Technology (TIST), 2022, 13(4): 1-23

[28] Yuan R, Xia Y B, Chen H B, et al. Shadoweth: Private smart contract on public blockchain. Journal of Computer Science and Technology, 2018, 33: 542-556

[29] Bowman M, Miele A, Steiner M, et al. Private data objects: an overview. arXiv preprint arXiv:1807.05686, 2018

[30] Russinovich M, Ashton E, Avanessians C, et al. A framework for building confidential verifiable replicated services. Microsoft, Redmond, USA, Technical Report MSR-TR-2019-16, 2019

[31] Yan Y, Wei C, Guo X, et al. Confidentiality support over financial grade consortium blockchain//Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data. Portland, USA, 2020: 2227-2240

[32] Wang Y, Li J, Zhao S, et al.Hybridchain: A novel architecture for confidentiality-preserving and performant permissioned blockchain using trusted execution environment. IEEE Access, 2020, 8: 190652-190662

[33] Müller C, Brandenburger M, Cachin C, et al. TZ4Fabric: Executing smart contracts with ARM TrustZone: (Practical experience report)//Proceedings of the 2020 International Symposium on Reliable Distributed Systems (SRDS). Shanghai, China, 2020: 31-40

[34] Demirag D, Clark J. Absentia: Secure multiparty computation on ethereum//Proceedings of the Financial Cryptography and Data Security. FC 2021 International Workshops: CoDecFin, DeFi, VOTING, and WTSC, Virtual, 2021: 381-396

[35] Chen P C, Kuo T H, Wu J L. A study of the applicability of ideal lattice-based fully homomorphic encryption scheme to ethereum blockchain. IEEE Systems Journal, 2021, 15(2): 1528-1539

[36] Cordi C, Frank M P, Gabert K, et al. Auditable, available and resilient private computation on the blockchain via MPC// International Symposium on Cyber Security, Cryptology, and Machine Learning. Cham: Springer Internaational Publishing, 2022: 281-299

[37] Steffen S, Bichsel B, Vechev M. Zapper: Smart contracts with data and identity privacy//Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security. New York,USA, 2022: 2735-2749

[38] Jian Z, Lu Y, Qiao Y, et al. TSC-VEE: A trustzone-based smart contract virtual execution environment. IEEE Transactions on Parallel and Distributed Systems, 2023, 34(6): 1773-1788

[39] Xiong A L, Chen B, Zhang Z, et al. {VeriZexe}: Decentralized private computation with universal setup//Proceedings of the 32nd USENIX Security Symposium (USENIX Security 23. Anaheim, USA, 2023: 4445-4462

[40] Jeong G, Lee N, Kim J, et al. Azeroth: Auditable zero-knowledge transactions in smart contracts. IEEE Access, 2023, 11: 56463-56480

[41] Du J, Xu Z, Liu D, et al. EPT: Enhancing user transparency for confidential smart contract//Proceedings of the 2023 19th International Conference on Mobility, Sensing and Networking (MSN). Nanjing, China, 2023: 431-438

[42] Tawfik A M, Al-Ahwal A, Eldien A S, et al. PriCollabAnalysis: Privacy-preserving healthcare collaborative analysis on blockchain using homomorphic encryption and secure multiparty computation. Cluster Computing, 2025, 28(3): 191

[43] Dannen C. Introducing Ethereum and solidity. Berkeley, USA: Apress, 2017

[44] Pfitzmann A, Köhntopp M. Anonymity, unobservability, and pseudonymity—a proposal for terminology//Designing Privacy Enhancing Technologies: International Workshop on Design Issues in Anonymity and Unobservability Berkeley, USA, 2001: 1-9

[45] Lundgren B, Möller N. Defining information security. Science and Engineering Ethics, 2019, 25: 419-441

[46] Gray J, Reuter A. Transaction processing: Concepts and techniques. Elsevier, 1992

[47] Androulaki E, Karame G O, Roeschlin M, et al. Evaluating user privacy in bitcoin//Financial Cryptography and Data Security: 17th International Conference, FC 2013, Okinawa, Japan, 2013: 34-51

[48] Androulaki E, Barger A, Bortnikov V, et al. Hyperledger fabric: A distributed operating system for permissioned blockchains//Proceedings of the 13th EuroSys Conference. Porto, Portugal, 2018: 1-15

[49] Ngabonziza B, Martin D, Bailey A, et al. Trustzone explained: Architectural features and use cases//Proceedings of the 2016 IEEE 2nd International Conference on Collaboration and Internet Computing (CIC). Pittsburgh, USA, 2016: 445-451

[50] Keller M. MP-SPDZ: A versatile framework for multi-party computation//Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security. Virtual Event, USA, 2020: 1575-1590

[51] Canetti R. Universally composable security: A new paradigm for cryptographic protocols//Proceedings 42nd IEEE Symposium on Foundations of Computer Science. Newport Beach, USA, 2001: 136-145

[52] Vickrey W. Counterspeculation, auctions, and competitive sealed tenders. The Journal of Finance, 1961, 16(1): 8-37

[53] Schneider M, Masti R J, Shinde S, et al. Sok: Hardware-supported trusted execution environments. arXiv preprint arXiv: 2205.12742, 2022

[54] Costan V , Devadas S .Intel SGX Explained.IACR Cryptology ePrint Archive,2016,201686

[55] Li X, Li X, Dall C, et al. Design and verification of the arm confidential compute architecture//Proceedings of the 16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22. Carlsbad, USA, 2022: 465-484

[56] El-Hindi M, Ziegler T, Heinrich M, et al. Benchmarking the second generation of intel sgx hardware//Proceedings of the 18th International Workshop on Data Management on New Hardware. Philadelphia, USA, 2022: 1-8

[57] Turing A M. On computable numbers, with an application to the Entscheidungsproblem. J. of Math, 1936, 58(345-363): 5

[58] Goldwasser S, Micali S, Rackoff C. The knowledge complexity of interactive proof-systems//Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali. New York,USA, 2019: 203-225

[59] Blum M, De Santis A, Micali S, et al. Noninteractive zero-knowledge. SIAM Journal on Computing, 1991, 20(6): 1084-1118

[60] Parno B, Howell J, Gentry C, et al. Pinocchio: Nearly practical verifiable computation. Communications of the ACM, 2016, 59(2): 103-112

[61] Groth J. Short pairing-based non-interactive zero-knowledge arguments//Abe M. eds. Advances in Cryptology-ASIACRYPT 2010: 16th International Conference on the Theory and Application of Cryptology and Information Security. Berlin, Germany:Springer, 2010: 321-340

[62] Lipmaa H. Progression-free sets and sublinear pairing-based non-interactive zero-knowledge arguments//Theory of Cryptography: 9th Theory of Cryptography Conference, TCC 2012, Taormina, Italy, 2012: 169-189

[63] Gennaro R, Gentry C, Parno B, et al. Quadratic span programs and succinct NIZKs without PCPs//Advances in Cryptology-EUROCRYPT 2013: 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques. Athens, Greece, 2013: 626-645

[64] Ben-Sasson E, Chiesa A, Tromer E, et al. Succinct non-interactive arguments for a von neumann architecture. IACR Cryptol. ePrint Arch., 2013, 2013: 879

[65] Groth J. On the size of pairing-based non-interactive arguments//Advances in Cryptology - EUROCRYPT 2016: 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, 2016: 305-326

[66] Sasson E B, Chiesa A, Garman C, et al. Zerocash: Decentralized anonymous payments from bitcoin//Proceedings of the 2014 IEEE Symposium on Security and Privacy. Berkeley, USA, 2014: 459-474

[67] Miers I, Garman C, Green M, et al. Zerocoin: Anonymous distributed e-cash from bitcoin//Proceedings of the 2013 IEEE Symposium on Security and Privacy. Berkeley, USA, 2013: 397-411

[68] Yang X, Li W. A zero-knowledge-proof-based digital identity management scheme in blockchain. Computers & Security, 2020, 99: 102050

[69] Camenisch J, Lysyanskaya A. An efficient system for non-transferable anonymous credentials with optional anonymity revocation//Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques. Berlin, Germany, 2001: 93-118

[70] Bünz B, Bootle J, Boneh D, et al. Bulletproofs: Short proofs for confidential transactions and more//2018 IEEE Symposium on Security and Privacy (SP). San Francisco, USA, 2018: 315-334

[71] Ben-Sasson E, Chiesa A, Tromer E, et al. Succinct {Non-Interactive} zero knowledge for a von neumann architecture//Proceedings of the 23rd USENIX Security Symposium (USENIX Security 14). 2014: 781-796

[72] McCurley K S. The discrete logarithm problem//Proceedings of the Symposium in Applied Mathematics. 1990, 42: 49-74

[73] Groth J, Maller M. Snarky signatures: Minimal signatures of knowledge from simulation-extractable SNARKs//Annual International Cryptology Conference. Cham: Springer International Publishing, 2017: 581-612

[74] Chiesa A, Hu Y, Maller M, et al. Marlin: Preprocessing zkSNARKs with universal and updatable SRS//Advances in Cryptology-EUROCRYPT 2020: 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, 2020: 738-768

[75] Tanaka N , Saito T .On the q-Strong Diffie-Hellman Problem. IACR Cryptology ePrint Archive, 2010, 2010215

[76] ElGamal T. A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory, 1985, 31(4): 469-472

[77] Bootle J, Cerulli A, Chaidos P, et al. Short accountable ring signatures based on DDH//European Symposium on Research in Computer Security. Cham: Springer International Publishing, 2015: 243-265

[78] Camenisch J, Lysyanskaya A. Dynamic accumulators and application to efficient revocation of anonymous credentials//Advances in Cryptology—CRYPTO 2002: 22nd Annual International Cryptology Conference Santa Barbara, California, USA, 2002: 61-76

[79] Eberhardt J, Tai S. Zokrates-scalable privacy-preserving off-chain computations//2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData). Halifax, Canada, 2018: 1084-1091

[80] Paillier P. Public-key cryptosystems based on composite degree residuosity classes//Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques. Berlin, Germany, 1999: 223-238

[81] Ben-Sasson E, Bentov I, Horesh Y, et al. Scalable, transparent, and post-quantum secure computational integrity. Cryptology ePrint Archive, 2018.2018046

[82] Yao A C C. How to generate and exchange secrets//Proceedings of the 27th Annual Symposium on Foundations of Computer Science (Sfcs 1986). Toronto, Canada, 1986: 162-167

[83] Ben-David A, Nisan N, Pinkas B. FairplayMP: A system for secure multi-party computation//Proceedings of the 15th ACM Conference on Computer and Communications Security. Alexandria, USA, 2008: 257-266

[84] Pinkas B, Schneider T, Smart N P, et al. Secure two-party computation is practical//Advances in Cryptology-ASIACRYPT 2009: 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, 2009: 250-267

[85] Damgård I, Pastro V, Smart N, et al. Multiparty computation from somewhat homomorphic encryption//Annual Cryptology Conference. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012: 643-662

[86] Rivest R L, Adleman L, Dertouzos M L. On data banks and privacy homomorphisms. Foundations of Secure Computation, 1978, 4(11): 169-180

[87] Acar A, Aksu H, Uluagac A S, et al. A survey on homomorphic encryption schemes: Theory and implementation. ACM Computing Surveys (Csur), 2018, 51(4): 1-35

[88] Rivest R L, Shamir A, Adleman L. A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM, 1978, 21(2): 120-126

[89] Boneh D, Goh E J, Nissim K. Evaluating 2-DNF formulas on ciphertexts//Theory of Cryptography: Second Theory of Cryptography Conference, TCC 2005, Cambridge, USA, 2005: 325-341

[90] Gentry C. A fully homomorphic encryption scheme. Stanford university, 2009

[91] Gentry C, Halevi S. Implementing gentry's fully-homomorphic encryption scheme//Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques. Berlin, Germany, 2011: 129-148

[92] Brakerski Z. Fully homomorphic encryption without modulus switching from classical GapSVP//Proceedings of the Annual Cryptology Conference. Berlin, Germany, 2012: 868-886

[93] Brakerski Z, Gentry C, Vaikuntanathan V. (Leveled) fully homomorphic encryption without bootstrapping. ACM Transactions on Computation Theory (TOCT), 2014, 6(3): 1-36

[94] Gentry C, Sahai A, Waters B. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based//Advances in Cryptology-CRYPTO 2013: 33rd Annual Cryptology Conference, Santa Barbara, USA, 2013: 75-92

[95] Cheon J H, Kim A, Kim M, et al. Homomorphic encryption for arithmetic of approximate numbers//Advances in cryptology-ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong kong, China, 2017: 409-437

[96] Tebaa M, El Hajji S, El Ghazi A. Homomorphic encryption method applied to Cloud Computing//Proceedings of the 2012 National Days of Network Security and Systems. Marrakech, Morocco, 2012: 86-89

[97] Zhao F, Li C, Liu C F. A cloud computing security solution based on fully homomorphic encryption//16th International Conference on Advanced Communication Technology. Pyeongchang, Republic of Korea, 2014: 485-488

[98] Mr M M P, Dhote C A, Mr D H S. Homomorphic encryption for security of cloud data. Procedia Computer Science, 2016, 79: 175-181

[99] Hu H, Xu J, Ren C, et al. Processing private queries over untrusted data cloud through privacy homomorphism//Proceedings of the 2011 IEEE 27th International Conference on Data Engineering. Hannover, Germany, 2011: 601-612

[100] Anggriane S M, Nasution S M, Azmi F. Advanced e-voting system using Paillier homomorphic encryption algorithm//Proceedings of the 2016 International Conference on Informatics and Computing (ICIC). Mataram, Indonesia, 2016: 338-342

[101] Sheela A C S, Franklin R G. E-voting system using homomorphic encryption technique. Journal of Physics: Conference Series, 2021, 1770(1): 012011

[102] Shamir A. How to share a secret. Communications of the ACM, 1979, 22(11): 612-613

[103] Boneh D, Gennaro R, Goldfeder S, et al. Threshold cryptosystems from threshold fully homomorphic encryption//Advances in Cryptology-CRYPTO 2018: 38th Annual International Cryptology Conference, Santa Barbara, USA, , 2018: 565-596

[104] Das S, Yurek T, Xiang Z, et al. Practical asynchronous distributed key generation//2022 IEEE Symposium on Security and Privacy (SP). San Francisco, USA, 2022: 2518-2534

[105] Groth J. Non-interactive distributed key generation and key resharing. Cryptology ePrint Archive, 2021.2021339

[106] Goyal V, Kothapalli A, Masserova E, et al. Storing and retrieving secrets on a blockchain//IACR International Conference on Public-Key Cryptography. Cham: Springer International Publishing, 2022: 252-282

[107] Maram S K D, Zhang F, Wang L, et al. CHURP: Dynamic-committee proactive secret sharing//Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. London, UK, 2019: 2369-2386

[108] Damagard I, Cramer R, Nielsen J B. Secure multiparty computation and secret sharing. USA: Cambridge University Press, 2015

[109] Viand A, Jattke P, Hithnawi A. SoK: Fully homomorphic encryption compilers//Proceedings of the 2021 IEEE Symposium on Security and Privacy (SP). San Francisco, USA, 2021: 1092-1108

[110] Chillotti I, Gama N, Georgieva M, et al. TFHE: Fast fully homomorphic encryption over the torus. Journal of Cryptology, 2020, 33(1): 34-91

[111] Liu Z, Micciancio D, Polyakov Y. Large-precision homomorphic sign evaluation using FHEW/TFHE bootstrapping//Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security. Cham: Springer Nature Switzerland, 2022: 130-160

[112] Li Zeng-Peng, Ma Chun-Guang, Zhou Hong-Sheng. Research on fully homomorphic encryption. Journal of Cryptologic Research, 2017, 4(06): 561-578 (in Chinese)
(李增鹏，马春光，周红生. 全同态加密研究. 密码学报，2017, 4(06): 561-578)

[113] Doan T V, Messai M L, Gavin G, et al. A survey on implementations of homomorphic encryption schemes. The Journal of Supercomputing, 2023, 79(13): 15098-15139

[114] Lyubashevsky V, Nguyen N K, Plançon M. Lattice-based zero-knowledge proofs and applications: shorter, simpler, and more general//Annual International Cryptology Conference. Cham: Springer Nature Switzerland, 2022: 71-101

[115] Nguyen N K, Seiler G. Practical sublinear proofs for R1CS from lattices//Annual International Cryptology Conference. Cham: Springer Nature Switzerland, 2022: 133-162

[116] Fiore D, Gennaro R, Pastro V. Efficiently verifiable computation on encrypted data//Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security. Scottsdale , USA, 2014: 844-855

[171] Fan J , Vercauteren F . Somewhat practical fully homomorphic encryption. IACR Cryptology ePrint Archive, 2012, 2012144

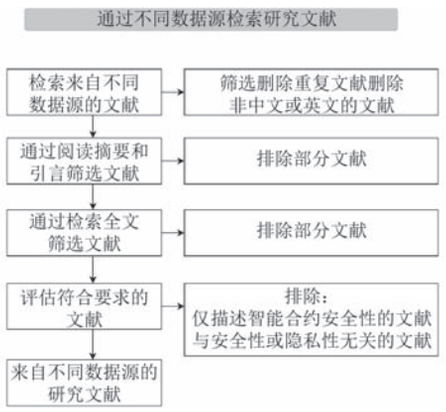
[118] Lyubashevsky V, Peikert C, Regev O. On ideal lattices and learning with errors over rings//Advances in Cryptology-EUROCRYPT 2010: 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, France, 2010: 1-23

[119] Del Pino R, Lyubashevsky V, Seiler G. Short discrete log proofs for FHE and ring-LWE ciphertexts//IACR International Workshop on Public Key Cryptography. Cham: Springer International Publishing, 2019: 344-373

[120] Jean-Louis N, Li Y, Ji Y, et al. Sgxonerated: Finding (and partially fixing) privacy flaws in tee-based smart contract platforms without breaking the tee. Proceedings on Privacy Enhancing Technologies, 2024, 1:617-634

附录 A.

在本部分，我们进一步介绍文中的文献筛选方法。如附图1所示，首先从不同数据源检索相关文献，并初步进行筛选；然后，通过阅读摘要和引言部分进一步筛选文献；再后，简略阅读全文对文献进行更深入的筛选；最后对文献质量及主题相关性进行评估，以确定最终筛选结果。



附录 图1 文献筛选方法

WANG You-Heng, Ph. D. candidate. His research interests include blockchain and privacy-preserving smart contracts.

FENG Kai-Kai, Ph. D. candidate. His research interests include blockchain-based applications and TEE-based applications.

LI Ru-Jia, Ph. D., assistant researcher. His research interests include hybrid consensus algorithms and trusted execution environments.

DUAN Si-Si, Ph. D., assistant researcher. Her research interests include applied cryptography and consensus algorithms.

CHEN Qing-Jie, master. His research interests include blockchain architecture design and privacy-preserving technologies.

Background

The concept of a smart contract was first introduced by Szabo in the mid-1990s and was implemented in a distributed system by Ethereum in 2014. A smart contract is an automated contract expressed in computer code that allows transactions to be executed and managed without the need for a trusted third party. By utilizing programmable code, smart contracts enable the parties involved to define the contract rules and automatically execute instructions when certain conditions are met.

These contracts are deployed on a blockchain network and executed by distributed nodes. The results of executing a smart contract are stored on a distributed ledger after reaching a consensus. Consequently, smart contracts offer advantages such as decentralization, automatic execution, traceability, and programmability. They are widely used in areas like decentralized finance, healthcare data protection, and the Internet of Things (IoT).

Despite their widespread use, privacy issues hinder the further development of smart contracts. The execution process and results of a smart contract can be observed by all nodes in the blockchain, allowing unauthorized users to access related information. This transparency exposes the identities and data involved in transactions, affecting the applicability of smart contracts. To address these privacy issues, various privacy-preserving smart contract solutions have been proposed in recent years. These solutions can be divided into two main categories: hardware-based and cryptographic-based. Hardware-based solutions use Trusted Execution Environment (TEE) technology to execute smart contracts within a trusted environment, protecting the execution process and related data. Cryptographic-based solutions use cryptographic techniques such as Zero-Knowledge Proofs, Secure Multi-Party Computation, and Homomorphic Encryption.

Although significant research has been conducted on privacy-preserving smart contracts, these solutions still have some issues.

Several reviews have been conducted on the topic of smart contract privacy. However, researchers from different fields tend to focus on varying aspects, use different technical methods, and address different security issues and goals, leading to a lack of systematic research results. This paper discusses the privacy and security issues of smart contracts, aiming to deeply study privacy protection methods for smart contracts and establish a comprehensive framework suitable for various application scenarios. It systematically reviews and evaluates existing technologies to identify current research gaps. This work seeks to provide researchers with insights into the latest research trends and future directions in this field. It also promotes further research in smart contract privacy protection and advances the application of smart contracts in the digital economy. This work was supported in part by the National Key R\&D Program of China (Grant No. 2023YFB2705000), the National Natural Science Foundation of China (Grant No. 92267203), Beijing Natural Science Foundation (Grant No. M23015), China Postdoctoral Science Foundation (Grant No. 2023M741949), and Tsinghua Shuimu Scholar Program.