arXiv:2603.07974v2 [cs.CR] 13 Mar 2026

ZK-ACE：面向后量子区块链系统的身份中心化零知识授权

Jian Sheng
Wang耶 LLC

2026年2月26日

修订于*2026*年*3*月*13*日：根据参考实现更新约束数量；新增基准测试（第*12*节）。

摘要

直接将后量子签名方案用于区块链交易验证，会产生千字节级的授权凭证。一种被广泛考虑的缓解方法是在零知识电路中验证后量子签名，并仅在链上发布简洁证明。然而，这种方法保留了以签名为中心的授权模型，只是转移了验证成本，并将计算开销高昂的高维格算术嵌入证明者电路中。

我们提出 ZK-ACE（面向**Z**ero-**K**nowledge **A**uthorization for **C**ryptographic **E**ntities，即密码实体的零知识授权），这是一种授权层，可完全以身份绑定的零知识授权语句取代交易中携带的签名对象。证明者无需证明某个特定后量子签名的正确性，而是以零知识方式证明：交易由与链上承诺及绑定的重放状态相一致的身份授权。该构造将确定性身份派生原语（DIDP）视为黑盒，并采用紧凑的身份承诺作为主要的链上身份锚点，辅以逐笔交易的防重放状态。

我们通过明确的基于博弈的安全定义，对 ZK-ACE 的授权可靠性、抗重放性、抗替换性和跨域分离进行了形式化。我们给出了完整的电路约束规范，定义了两种防重放模型（随机数注册表和空值符集合），并在标准假设（知识可靠性、抗碰撞性以及 DIDP 身份根恢复困难性）下提供了基于归约的安全证明。结构化的协议层数据核算表明，与直接部署后量子签名相比，共识可见授权数据减少了一个数量级。该设计支持批量聚合和递归证明组合，并兼容账户抽象及基于 Rollup 的部署架构。

关键词。 后量子密码学、确定性身份、零知识证明、身份承诺、区块链授权、基于格的签名、账户抽象。

1 引言

1.1 后量子签名与区块链可扩展性

NIST 对后量子密码学（PQC）的标准化，是迈向量子对手存在情况下长期安全的必要一步。ML-DSA（原名 Dilithium）等基于格的签名方案 [2] 以及 SLH-DSA 等基于哈希的方案

（原名 SPHINCS$^+$） [3]已成为取代传统公钥签名的领先候选方案。

虽然这些方案非常适用于传统通信场景，但将其直接应用于区块链系统时，后量子安全与可扩展性之间的根本矛盾便会显现。基于格的签名有一个显著特点，即签名大小较大，通常达到数千字节。在区块链环境中，交易有效性必须由所有共识参与者验证，并作为永久账本的一部分存储，因此这类签名大小会直接导致链上数据增长。NIST 安全级别 2 的 ML-DSA 签名约为 2.4 KB，而 Ed25519 签名为 64 字节，secp256k1 ECDSA 签名为 71 字节——每笔交易增加约 30–40$\times$ 倍。

在现代扩展架构中，这种压力会进一步加剧。在基于 Rollup 的系统中，交易数据会以调用数据或 Blob 数据的形式提交到基础层，而基础层的带宽和可用性受到明确限制，且具有相应的经济成本 [17]。因此，较大的后量子签名会降低有效交易吞吐量，并增加每笔交易的边际成本。关键在于，这种可扩展性压力源于授权凭证本身的结构，而非任何特定实现的低效。

1.2 零知识压缩签名验证的局限性

针对数据膨胀问题，一种自然的应对方法是将后量子签名验证移至链下，并以零知识证明取而代之。在这种方法下，证明者生成零知识证明，以证明后量子签名有效，而区块链只需验证简洁证明 [18]。

然而，这种策略存在一个关键局限：验证基于格的签名需要对高维代数结构进行算术运算，而在零知识电路中实现这些运算会产生大量约束。即使在技术上可行，将格验证逻辑嵌入零知识电路也会显著增加证明成本，并且往往只是将可扩展性负担从链上数据转移到链下计算和验证[9, 7]。

因此，后量子签名的零知识验证本身也可能成为性能瓶颈。这种方法并未解决可扩展性问题的根源，而是保留了以签名为中心的授权模式，仅仅转移了其计算成本。这一观察引出了一个更根本的问题：后量子签名验证是否真的必须成为链上授权的组成部分？

1.3 关键观察：授权与签名对象

在共识层，区块链并不天然要求验证特定的密码学签名对象。共识所要求的是，确保给定交易已由正确的实体按照系统规则授权 。

在传统设计中，密码学签名是表达和验证授权的机制。然而，签名归根结底是实现产物，而非授权语义本身。将签名对象视为验证的主要单位，会隐式地将共识安全与特定签名方案的具体属性绑定起来，包括密钥大小、签名大小和验证成本。

本工作采用了不同的视角：授权应被建模为一种语义属性——即某个特定身份已批准某笔特定交易——而不依赖于传统上用于表达该批准的具体密码学制品。从这一视角来看，零知识证明的作用并非验证后量子签名对象的正确性，而是证明底层授权条件成立。

1.4 我们的设定与范围

我们考虑这样一种设定：假定确定性身份派生原语 (DIDP)作为一种密码学构建模块存在。[1] DIDP 提供从 256位根熵值 (REV) 到特定上下文密码密钥的稳定、确定性映射，并且可能适用于多种密码方案，包括后量子构造。其安全属性——包括确定性、上下文隔离以及抵御身份根恢复的能力——均视为既定。在这方面，DIDP 被视为类似于其他通常假定存在的原语，例如可验证随机函数 (VRF)、密钥派生函数 (KDF) 或密码学承诺方案；这些原语常被用作构建模块，而无需在每个应用中重新证明其安全性。

本文并不提出、分析或修改任何特定的 DIDP 构造。我们的贡献始于 DIDP 之上的一层：给定一个确定性身份派生原语，在面向后量子的区块链系统中，交易授权应如何表示、证明与验证？

1.5 贡献

本文做出以下贡献：
1. 我们提出 ZK-ACE，一种构建于确定性身份派生之上的零知识授权层，其中交易授权通过零知识证明而非显式的后量子签名对象来证明。2. 我们提出一种链上数据模型，由紧凑的身份承诺与逐交易的零知识授权证明组成，以恒定大小的验证材料取代千字节级的签名产物。3. 我们形式化规定了 ZK-ACE 所基于的零知识电路陈述，包括其公开输入、私有见证以及五项核心约束。4. 我们定义了基于博弈的安全概念——授权可靠性、抗重放性、抗替换性与跨域分离——并在标准假设（知识可靠性、抗碰撞性、DIDP 身份根恢复困难性）下给出基于归约的安全论证。5. 我们描述了两种重放保护机制（随机数注册表与空值符集合），并对其各自权衡进行形式化分析。6. 我们提出聚合与递归策略，以实现高吞吐量系统的批量授权与可扩展验证。7. 我们提供了对链上授权数据需求的结构性、协议级核算，表明相对于基于签名的后量子授权，数据量降低了一个数量级。8. 我们报告了参考实现与单线程 CPU 基准测试，验证了约束数量，并表明在消费级硬件上可实现亚毫秒级验证以及低于 100 ms 的证明。

1.6 设计目标

授权模型应满足以下属性：

- **安全性**。 在假设的威胁模型下，授权必须具备可靠的密码学安全性，并能抵御经典和量子对手。
- **可验证性**。共识参与者必须能够仅使用公开信息高效验证交易有效性。

[1] ACE-GF [1] 是一种具体实现，满足第 3.2 节定义的 DIDP 接口和安全属性。任何提供相同接口的框架——即将高熵根确定性地映射为上下文专属密钥，并具备确定性、上下文隔离和身份根恢复困难性等属性的框架——都可以替代它。

- **可扩展性**。链上授权数据和验证成本应保持恒定，不受后量子签名对象大小的影响。

- **可迁移性**。 现有身份应能够迁移到新的授权模型，而不影响既有的恢复机制或身份假设。身份承诺应与证明系统无关，这样升级验证者和证明者就无需进行身份轮换。

- **可聚合性**。 该授权机制应支持批处理和递归组合，以满足高吞吐量系统的需求。

1.7 论文组织结构

第 2 节综述相关工作。第 3 节介绍符号约定、假设的 DIDP 接口和威胁模型。第 4 节描述系统模型和交易流程。第 5 节定义身份承诺和绑定规则。第 6 节阐明授权语义。第 7 节详细说明零知识电路。第 8 节给出形式化安全定义和证明。第 9 节讨论链上验证和部署。第 10 节提供结构化数据核算。第 11 节介绍可扩展性扩展。第 12 节报告参考实现和基准测试。第 13 节讨论局限性，第 14 节总结全文。

2 相关工作

区块链上的后量子签名。已有多项研究探讨将基于格或基于哈希的签名直接集成到区块链交易验证中 [2, 3,4]。以太坊社区已在其长期路线图中探索 PQC 迁移路径。这些研究公认的主要障碍是每笔交易产生的大量数据开销。我们的工作从协议架构层面解决这一障碍，而不是试图优化签名参数。

零知识压缩签名验证。另一条研究路线是在零知识电路中嵌入经典或后量子签名验证。典型示例包括基于 Circom 的 ECDSA 验证 [18]，以及在 SNARK 电路中验证格签名的探索。对于验证过程在代数上较为轻量的经典签名，这种方法在技术上可行；但将其扩展到基于格的方案，会显著增加约束数量和证明者成本。ZK-ACE 完全避免了这一问题，因为它不会在电路中放置任何签名验证逻辑。

透明及基于哈希的证明系统。STARK [9] 和基于 FRI 的交互式预言机证明可在无需可信设置的情况下生成透明且可能具备后量子安全性的证明。Chiesa、Fenzi 和 Weissenberg 的近期研究 [25] 为受约束交织码构造了零知识交互式Oracle可证明系统，其可靠性与证明长度的开销近乎最优（$1 + o(1)$），进一步提升了基于哈希的证明系统的效率。这些进展改进了证明系统层，并与工作于授权模型层的 ZK-ACE 相辅相成。即使采用最优的透明证明系统，将基于格的签名验证嵌入零知识电路，仍需将高维多项式算术运算（数论变换、非原生模约减）表示为算术约束，从而产生规模达数百万个约束的电路。ZK-ACE 完全绕过了这一瓶颈：其电路只需少量零知识友好型哈希计算（参考 Poseidon 参数化下为 $\approx 4{,}000$ 个 R1CS 约束；见第 7.3 节），并兼容任何后端证明系统，包括基于哈希的证明系统和透明构造。

承诺与空值符系统。Zcash Sapling [13]和 Tornado Cash [14] 等注重隐私的协议采用基于承诺的状态模型，并通过空值符集合防止重放。Semaphore [15] 使用身份承诺实现匿名信号传递。ZK-ACE 借鉴了承诺/空值符设计术语，但将其用于一个不同的问题：针对已承诺的确定性身份进行交易授权，而非私密价值转移或匿名成员资格证明。

账户抽象与模块化验证。ERC-4337 [16] 为以太坊引入账户抽象，使账户能够通过验证器模块定义自定义验证逻辑。ZK-ACE可以实例化为此类模块，以零知识证明验证取代签名对象检查。不同于通用的账户抽象验证器，ZK-ACE提供了具有形式化安全保证的完整授权框架。

基于身份的密码学。基于身份的加密 [19]和基于身份的签名 [20]将密码学能力绑定至身份字符串，通常需要可信密钥生成中心。ZK-ACE不假设存在任何可信第三方；身份由用户自主管理，并通过DIDP根据用户控制的根材料确定性地派生。

3 预备知识与假设

3.1 符号约定

本文始终使用以下符号约定。

- λ：安全参数。
- $H(\cdot)$：一种密码学哈希函数。在零知识电路内部使用时， H 采用零知识友好型哈希函数（例如 Poseidon[6]）进行实例化；在电路外部，它可以是传统哈希函数。
- $\|$：可无歧义解析的拼接（例如，长度前缀编码）。
- REV：作为身份根的256位根熵值，通过 DIDP 派生。临时的；从不持久存储。
- 上下文：派生上下文元组（算法标识符,域, 索引），由 DIDP 用于从 REV 派生特定于上下文的密钥。
- $\mathrm{ID}_{\mathrm{com}}$：链上身份承诺。
- 交易哈希：所有与授权相关的交易字段的哈希。
- 域：区块链或应用域分隔符。
- 目标：目标绑定摘要（地址、调用数据哈希或账户上下文）。
- rp_com：防重放承诺（随机数承诺或空化符）。
- PPT：概率多项式时间。
- negl(λ)：关于 λ可忽略的函数。
- $x \xleftarrow{\$} S$： x 从集合 S中均匀随机采样。

3.2 假设的 DIDP 接口（黑盒）

我们假设存在一种确定性身份派生原语 （DIDP），并将其视为黑盒构建模块。我们不限定具体构造，而仅对零知识授权层所需的接口和属性进行形式化。任何满足该接口的具体框架都可以使用；ACE-GF [1] 便是其中一种实例。

DIDP 的核心是一个高熵身份根，称为根熵值 (REV)。这是一个 256 位的值，用于唯一标识实体的密码学身份，并作为所有派生密钥的唯一熵源。REV 仅在执行期间临时存在于内存中，绝不会被持久化存储或导出。

DIDP 提供两种主要操作：

- **身份重建**：REV $\leftarrow$ Unseal(params,SA, Cred)，其中 SA 是密封制品（REV 的加密编码），Cred 是授权凭证。若身份验证成功，则返回 REV；$\perp$ 否则不返回。该凭证控制对身份根的访问，但不会影响密钥派生。
- **上下文特定密钥派生**：Key $\leftarrow$ Derive(REV,Ctx)，其中 Ctx $=$ (AlgID,Domain, Index) 是一个上下文描述符，用于编码目标密码学算法（例如 Ed25519、Secp256k1、ML-DSA）、应用域（例如签名、加密）以及派生索引。该上下文元组可确保密码隔离：在不同上下文元组下派生的密钥在计算上相互独立。

这两个操作构成的接口清晰地将授权流程（通过密封制品和凭证控制对 REV 的访问）与身份流程（使用上下文隔离的派生从 REV 确定性地派生密钥）分离开来。零知识授权层仅与身份流程交互：它假设证明者已获取 REV（通过 Unseal），并使用 REV 和上下文元组 Ctx 派生相关密码学材料。

3.3 从 DIDP 假定的安全属性

零知识授权层依赖于所假定的 DIDP 接口的以下安全属性：

- **确定性**：对于固定输入，Derive 产生相同的输出。特别地，相同的 (REV,Ctx) 对始终产生相同的派生密钥。
- **上下文隔离**：不同的上下文元组 Ctx $\neq$ Ctx′ 产生计算独立的派生密钥。这确保在某一上下文下派生的密钥一旦被攻破，既不能据此恢复其他上下文下派生的密钥，也不会泄露底层 REV 的信息。
- **身份根恢复困难性**：我们通过以下博弈 $\mathsf{Game}_{\text{rec-didp}}$ $\mathcal{A}$ (λ) 来形式化该属性：

  1. 挑战者采样 REV $\$ \leftarrow \{0,1\}^{256}$。2. 对手 $\mathcal{A}$ 获得对 $\mathcal{O}_{\mathsf{pub}}$ 的预言机访问；该预言机在输入 (salt,domain) 时返回H(REV‖salt‖domain)，并获得对 $\mathcal{O}_{\mathsf{derive}}$ 的预言机访问；该预言机在输入 Ctx 时返回 H(Derive(REV,Ctx))。这些预言机刻画了在不获知 REV 本身的情况下观察身份承诺和派生公有标识符的能力。3. $\mathcal{A}$ 输出一个候选 REV*。4. 若 REV* $=$ REV，则 $\mathcal{A}$ 获胜。对于任意概率多项式时间对手 $\mathcal{A}$：

$$\mathsf{Adv}_{\mathcal{A}}^{\text{rec-didp}}(\lambda) = \Pr[\mathsf{REV}^* = \mathsf{REV}] \leq \mathsf{negl}(\lambda).$$

  非 效率有限的对手都无法仅凭公开输出恢复身份根。
  我们注意到，身份根恢复困难性是授权可靠性证明（定理 8.1）所需的最小 假设。证明中的案例分析（第 8.1 节）表明，任何成功的伪造都必须要么产生哈希碰撞（由 H 的抗碰撞性处理），要么从提取出的见证中得到正确的 REV（由恢复困难性处理）。更强的概念——例如，在不知道 REV 的情况下，无法为新上下文 Ctx 伪造 H(Derive(REV，Ctx))——可由恢复困难性与 H 的抗碰撞性结合推出，但无需将其作为单独的假设。

- **多算法隔离**：对于同一 REV，通过 Ctx 中不同的 AlgID 值派生出的不同密码学算法密钥，在计算上彼此隔离，因此某个派生密钥遭到泄露，并不会使其他密钥也能被派生。

对于 DIDP 的内部结构，不作任何额外假设。特别是，本研究不依赖任何特定的密钥派生函数（例如 HKDF [21]），

内存困难型密码哈希方案（例如 Argon2 [5]）或实例化过程中使用的助记词编码。

3.4 零知识证明系统模型

本文中的构造假设存在一个通用零知识证明系统，支持简洁非交互知识论证，并建立在交互式证明的基础框架之上 [22]。具体的证明系统并未固定，可采用 SNARK- [8], PLONK- [7], STARK- [9], 或基于内积论证的构造 [10]来实例化。我们假设该系统满足以下标准性质：

- **完备性**：拥有有效见证的诚实证明者总能生成可被接受的证明。
- **知识可靠性**：对于任何以不可忽略的概率生成可接受证明的概率多项式时间证明者 $\mathcal{P}^*$ ，都存在一个概率多项式时间提取器 $\mathcal{E}$ ，该提取器在能够访问 $\mathcal{P}^*$ 的情况下，以压倒性概率输出有效见证。形式化地，

$$\Pr[\mathsf{Verify}(\mathsf{vk}, \pi, x) = 1 \ \wedge \ \mathcal{E}^{\mathcal{P}^*}(x) \text{ fails}] \leq \mathsf{negl}(\lambda).$$

- **零知识性**：除陈述的有效性外，证明不会泄露有关私有见证的任何信息。
- **公开输入绑定**：证明在密码学意义上与声明的公开输入绑定。对于任何概率多项式时间对手，

$$\Pr[\mathsf{Verify}(\mathsf{vk}, \pi, x) = 1 \ \wedge \ \mathsf{Verify}(\mathsf{vk}, \pi, x') = 1 \ \wedge \ x \neq x'] \leq \mathsf{negl}(\lambda).$$

- **简洁性**：证明大小和验证复杂度均不取决于见证的大小。

具体的提取模型取决于所选的证明系统：基于配对的简洁非交互知识论证（例如 Groth16）通常在代数群模型中或通过非黑盒提取实现知识可靠性；而基于交互式预言机证明的系统（例如 STARK、PLONK）则依赖于从底层交互式预言机证明中进行点查询提取。我们的安全性归约以知识可靠性优势 $\mathsf{Adv}^{\mathsf{ks}}$ 表述，并且兼容任何在各自提取模型下满足上述属性的证明系统。

3.5 威胁模型

我们考虑具有以下能力的对手 $\mathcal{A}$ ：

- 完全查看链上状态、交易历史以及内存池中可见的提交。
- 在共识协议允许的范围内，能够提交、重新排序、重放和替换交易载荷，并尝试在不同上下文之间重用证明。
- 针对传统公钥原语实施离线密码攻击的能力，包括利用未来量子计算发起的攻击。

对手可能通过重复使用授权凭证发起重放攻击，通过将有效授权材料绑定到非预期交易来发起替换攻击，或通过关联不同交易或上下文中的授权数据来发起关联攻击。

我们不 假设存在可信硬件、可信执行环境或私有内存池。所有授权凭证均假设为公开可见。身份对应的底层 DIDP 根材料（REV）遭到泄露不在讨论范围内，并视为基础原语发生密钥泄露。

4 系统模型

4.1 角色

ZK-ACE 涉及以下角色：

- **证明者**。证明者是通过 DIDP 派生出的确定性身份的控制实体，通常是用户或钱包应用程序。证明者会生成零知识证明，以证明给定交易已获得相应身份的授权。

- **验证者**。 验证者是区块链系统中面向共识的组件，例如链上智能合约或原生验证规则。验证者会检查已提交证明的有效性，并根据协议规则实施重放保护。

- **可选聚合器或中继器**。 在可扩展的部署中，可选聚合器或中继器可以收集多个证明者的授权请求、聚合证明，或向区块链提交批量授权数据。该角色不受信任，也不会获知私有身份材料。

ZK-ACE 的安全性不依赖聚合器或中继器是否诚实；它们仅用于提升性能和可用性。

4.2 链上状态

ZK-ACE 通过仅存储授权验证和重放防护所严格必需的数据，最大限度地减少链上状态：

- **身份承诺**。 对于每个参与 ZK-ACE 的身份，区块链都会存储一个指向底层确定性身份的紧凑承诺 $\mathsf{ID}_{\mathsf{com}}$ 。该承诺是主要的链上身份锚点，供此身份关联的所有后续授权证明使用。

- **重放保护状态**。 根据所选的重放防护模型（第 7.4 节），区块链还会维护：– 一个随机数注册表，用于对授权随机数实施单调性或唯一性约束；或– 一个无效标识符集合，用于标记已使用的授权令牌；或– 一个与身份承诺相关联的授权索引 。

- **验证者参数**。 证明系统的验证密钥 vk。

不会存储私钥 、后量子签名或身份根材料（REV）。 n.

4.3 交易授权流程

ZK- 单笔交易的 ACE 授权流程按算法中的规定执行。 1.

在整个过程中，不会在区块链上公开或验证任何后量子签名对象。授权有效性完全通过零知识证明的正确性，以及该证明与已承诺身份和交易哈希之间的绑定关系来确立。

5 身份承诺与绑定模型

5.1 承诺构造

对于参与 ZK-ACE 的每个确定性身份，区块链都会存储一个紧凑的身份承诺，其定义如下：

$$\mathsf{ID}_{\mathsf{com}} = H(\mathsf{REV} \| \mathsf{salt} \| \mathsf{domain}), \tag{1}$$

其中：
- REV 是 256位根熵值，充当身份根，并通过 DIDP 解封操作临时重构；

- 盐值 $\$ \leftarrow \{0,1\}^{\lambda}$ 是特定于身份的随机值或伪随机值；
- 域编码域分离参数，例如目标区块链或应用命名空间。

算法1 ZK-ACE 交易授权

证明者端：

1: 构造交易负载 tx，并计算其交易哈希 $\leftarrow H$(tx)。2:使用通过 DIDP 派生的身份材料，生成见证 w =(REV,盐值, 上下文, 随机数,aux)。3: 计算公开输入 π_{pub} =(ID$_{\text{com}}$,交易哈希, 域,目标,rp_com)。4: 生成零知识证明 $\pi \leftarrow$ 证明(pk w, π_{pub})。5: 将 (tx π, π_{pub}) 提交到区块链（直接提交或通过中继者提交）。

验证者端（给定 tx, π,π_{pub}）**:**6: **上下文绑定检查：** 根据提交的交易负载重新计算交易哈希 $' \leftarrow H$(tx)，并验证 TxHash$' = \pi_{\text{pub}}$.TxHash。若不匹配则拒绝。7: **公共输入验证：** 验证 π_{pub}·ID$_{\text{com}}$ 是否为已注册的链上身份承诺，并确认 π_{pub}·域和 π_{pub}·目标是否与该交易上下文的预期值相符。若有任何不匹配则拒绝。8: **证明验证：** 验证 ZKVerify(vk π, π_{pub}) $? = 1$。若验证失败则拒绝。9: **重放检查：** 验证重放谓词（随机数新鲜性或空化符新颖性）。若不满足则拒绝。10: 若所有检查均通过，则接受该交易并更新重放状态。

该承诺充当链上的主要身份锚点。授权还需要每笔交易的公开输入（交易哈希、域、目标、rp_com）以及防重放状态，但 ID$_{\text{com}}$ 是持久的身份绑定组件。REV 和任何派生密钥都不会在链上公开。

引入盐值具有多重作用。首先，它支持< стиль id='1'>承诺重新随机化，使同一底层身份能够随着时间推移关联到多个不可链接承诺。其次，它提供了< стиль id='3'>跨链或跨域隔离机制，确保在不同执行环境中使用的身份承诺在密码学上彼此不同。最后，基于盐值的随机化通过防止轻易关联从同一身份派生的承诺，支持基本的隐私控制。

5.2 公共输入绑定规则

ZK-ACE 中的授权证明会针对一组公开输入进行验证，这些输入将证明绑定到特定交易和执行上下文。这些公开输入对共识可见，并且必须向验证者明确声明。以下值是必需的：

1. **交易哈希**（TxHash）。将授权证明绑定到特定的交易负载，从而防止替换攻击。2. **链或域标识符**（domain）。确保授权证明仅在预期的执行环境中有效，防止跨链重放。3. **派生目标哈希**（target）。这是对从上下文派生的授权密钥材料的哈希承诺，即 target $= H$（Derive(REV,Ctx)），确保该证明绑定到特定的身份派生上下文。4. **防重放承诺**（rp_com）。用于强制执行单次使用或单调授权语义的随机数值、空化符或授权索引承诺。

这些公开输入共同确保每个授权证明都绑定到授权者身份（通过 ID$_{\text{com}}$）、获授权的交易（通过交易哈希）、所使用的派生上下文（通过目标），以及授权有效的时间和地点（通过域和 rp_com）。

5.3 可链接性考量

ZK-ACE 允许将多个身份承诺关联到同一个底层确定性身份。这种灵活性能够在隐私、可用性和可审计性之间实现不同的权衡。

一种做法是为每个身份维护一个长期有效的承诺，以尽可能简化流程并尽量减少链上状态。另一种做法是，证明者可以定期更换盐值并注册新的承诺，从而降低授权事件之间的可链接性，但代价是需要进行更多链上更新。

重要的是，可链接性完全由承诺层控制。授权证明本身不会泄露关于底层身份的任何额外信息，超出所选承诺策略所隐含的信息。

6 授权语义

6.1 我们证明什么：授权

从较高层面来看，ZK-ACE 证明交易是在确定性派生框架下由已承诺身份授权的。可以将以零知识方式证明的授权陈述表示为：

> 存在一个根熵值 REV 及其关联的派生上下文，使得：*(i)* 从 REV 派生的身份承诺与链上承诺$\mathsf{ID}_{\mathsf{com}}$相匹配；*(ii)* 由 REV 锚定的身份，依据规定的派生上下文和防重放规则，已授权哈希为 TxHash 的交易。

该陈述本质上是存在性陈述。零知识证明证明存在一个有效的授权身份，且该身份与链上承诺一致，同时不泄露 REV 或任何派生密钥。在此模型中，授权是一种语义属性：如果能够证明某笔交易已由链上锚定的身份按照系统的确定性授权规则批准，则该交易有效。

6.2 我们不证明什么

为避免歧义，我们明确说明 ZK-ACE不 试图证明的内容：

- **不验证后量子签名对象**。 零知识电路不验证任何 ML-DSA 或其他后量子签名的正确性。签名对象不会作为输入提供给电路，也不会出现在链上。
- **电路内部不进行格算术**。该电路既不实现基于格的密码学运算，也不编码任何后量子签名方案的验证逻辑。
- **不对后量子签名进行压缩**。ZK-ACE 并不声称压缩后量子签名或减小其大小。相反，它彻底消除了将后量子签名材料置于链上的需要。

因此，该构造不应被解释为对后量子签名的零知识优化。它所代表的是一种授权语义的转变，即以确定性授权的零知识证明取代显式签名验证。

6.3 授权令牌定义

为便于具体说明，我们定义一种统一的授权令牌，用于捕捉交易批准的语义意图。授权令牌定义如下：

$$\mathsf{Auth} = H(\mathsf{REV} \| \mathsf{Ctx} \| \mathsf{TxHash} \| \mathsf{domain} \| \mathsf{nonce}), \tag{2}$$

其中，每个组件都将授权绑定到特定的身份根（REV）、派生上下文（Ctx）、交易（TxHash）、执行域（domain）和防重放状态（nonce）。

授权令牌 Auth 不是密码学签名，也无意在零知识证明之外公开验证。它是电路所强制执行的授权语义的一种概念性表示。零知识证明证实，可以从承诺的身份根正确派生出这样的令牌，并且该令牌将身份与所声明的公开输入绑定在一起。

7 ZK 电路规范

本节说明 ZK-ACE 所证明的零知识陈述及相应的电路结构。该电路旨在兼容标准约束系统（例如 R1CS 或类 PLONK 算术化），并避免嵌入后量子签名验证逻辑。

7.1 电路陈述

设 $\mathrm{ID}_{\mathrm{com}}$ 为链上身份承诺，TxHash 为交易哈希，domain 为链/应用域标签，target 为派生目标哈希，rp_com 为防重放承诺。ZK-ACE 电路证明以下陈述：

> 证明知道 (REV、盐值、上下文、随机数、aux)，使得：
> 1. 由(REV、盐值、域) 重新计算出的身份承诺等于公开承诺$\mathrm{ID}_{\mathrm{com}}$；2. 目标绑定与(REV、上下文)下的确定性密钥派生一致；3. 交易哈希交易哈希 在派生上下文和防重放规则下获得授权。

此声明是存在性的：该证明确立了某个与 $\mathrm{ID}_{\mathrm{com}}$ 一致的身份根授权了特定的交易上下文，同时不泄露 REV 或任何派生密钥。

7.2 见证与公开输入

私有见证（秘密输入）。见证包括：

- REV：256位根熵值（身份根，通过 DIDP 解封临时重构）；
- 盐值：与身份关联的承诺盐；
- 上下文：派生上下文元组 (算法标识符,域, 索引)，用于指定目标密码算法、应用域和派生索引；
- 随机数：防重放值（或用于派生空化符的材料）；
- 辅助数据：特定电路实例化可能需要的辅助秘密材料（例如，多轮 Poseidon 调用的中间吸收状态，或可变长度输入编码的填充值）。核心约束 (C1)–(C5) 的表述不涉及辅助数据；在见证定义中包含该项，是将其作为实例化预留槽位，以便具体部署能够携带特定于实现的中间值，而无需修改抽象见证结构。

公开输入。公开输入包括：

- $\mathrm{ID}_{\mathrm{com}}$：链上身份承诺；
- 交易哈希：待授权的交易哈希；
- 域：链/应用域标签；

- 目标：对上下文派生的授权密钥材料的哈希承诺，即 target $=H$(Derive(REV，上下文));

- rp_com：防重放承诺（随机数承诺或空化符）。

所有公开输入均为共识可见，且必须绑定到证明，以防止替换和重放攻击。

7.3 核心约束

我们从高层描述电路约束，其形式旨在自然映射到标准算术化。令 $H(\cdot)$ 表示用于绑定的所选电路内哈希。

（C1）承诺一致性。电路重新计算身份承诺，并强制其与 $\mathsf{ID}_{\mathsf{com}}$ 相等：

$$H(\mathsf{REV} \| \mathsf{salt} \| \mathsf{domain}) = \mathsf{ID}_{\mathsf{com}}. \tag{3}$$

This 确保证明者控制与链上锚一致的身份根 r.

（C2）确定性推导正确性（目标绑定）。电路强制要求公开目标哈希对应于从身份根派生出的特定上下文密钥：

$$\mathsf{target} = H(\mathsf{Derive}(\mathsf{REV}, \mathsf{Ctx})). \tag{4}$$

外层哈希确保派生出的密钥本身绝不会作为公开输入暴露；链上仅公开其哈希承诺（目标），从而保护身份相关密码学材料的机密性。根据约束（C5），Ctx 的域分量必须等于公开输入中的域，确保派生正确性与声明的域绑定。

实现说明：为尽量减少约束数量，Derive 在电路内部实例化为一系列确定性的零知识友好哈希（例如 Poseidon），避免在电路中模拟非原生域算术或执行开销高昂的传统编码（例如 Bech32、Base58）。上下文元组 Ctx $=$(AlgID，Domain，Index) 被编码为固定长度的域元素输入，供哈希函数使用。

实例化范围：（C2）中提到的函数 Derive 指的是电路原生派生函数，即在证明系统内部计算的具体零知识友好哈希序列，而非外部的 DIDP Derive 实现；后者可能使用非电路友好型原语（例如 HKDF、HMAC-SHA-256）。安全性分析（第 8 节）完全基于电路原生函数：证明系统强制要求证明者知道一个 REV，可通过该电路原生函数派生出所声明的目标。确保功能等价性，即电路原生派生与链下 DIDP 派生在相同的 (REV，Ctx) 输入下产生相同输出，是一项部署时实例化要求。只要链下 DIDP 流程采用与电路中相同的哈希函数和编码约定，即可满足这一要求（这是自然的选择，因为零知识友好型哈希函数在链下同样适用）。这种分离遵循零知识协议设计中的标准实践：电路定义“规范”计算，外部组件则与之保持一致。

（C3）与 交易哈希 **的授权绑定**。电路通过强制执行授权令牌关系，将授权绑定到特定的交易哈希和上下文：

$$\mathsf{Auth} = H(\mathsf{REV} \| \mathsf{Ctx} \| \mathsf{TxHash} \| \mathsf{domain} \| \mathsf{nonce}). \tag{5}$$

（C4）防重放（随机数/无效标识符规则）。 电路以一种验证者能够强制执行唯一性或单调性的方式，对防重放状态进行承诺。支持两种规范形式：

- 随机数承诺：
$$\mathsf{rp_com} = H(\mathsf{ID_{com}} \| \mathsf{nonce}), \tag{6}$$
使区块链能够执行每身份随机数规则。
- 空化符（一次性令牌）：
$$\mathsf{rp_com} = H(\mathsf{Auth} \| \mathsf{domain}), \tag{7}$$
使区块链能够通过维护无效标识符集合来拒绝重复使用。

（C5）域分离与上下文一致性。电路会强制要求，作为公开输入声明的域值在所有承诺和绑定计算中得到一致使用，从而防止跨链或跨应用的证明重用。这包括两项要求：

（a）公开输入一致性：域必须作为显式输入出现在承诺绑定（C1）、授权绑定（C3）和防重放计算（C4）中。（b）上下文—域绑定：电路会强制要求，上下文元组中的域组件与公开声明的域相匹配：

$$\mathsf{Ctx.Domain} = \mathsf{domain}. \tag{8}$$

此约束确保 (C2) 中的派生与所有其他绑定使用相同的域，从而建立上下文特定派生与公共域标识符之间的关联。

这些约束确保有效证明能够同时证明身份所有权（通过知晓 $\mathsf{ID_{com}}$ 对应的 REV 原像）、上下文正确性（通过在 Ctx 下进行确定性推导）、交易特定授权（通过绑定到 TxHash）以及防止重放（通过 rp_com）。

约束数量估算。 电路开销主要来自电路内哈希计算。表1列出了参考实现测得的约束明细：使用 Poseidon，宽度为 $t = 3$、S 盒指数为 $\alpha = 17$、8 个完整轮次和 57 个部分轮次，并在 BN254 标量域上运行。[2]

每次调用的开销随输入元数而变化：arkworks 海绵组件（速率为 $r = 2$）处理 k 个域元素输入需要 $\lceil k/r \rceil$ 个吸收轮数，每轮需要 ≈ 400 个 R1CS 约束。(C2) 需要两次哈希调用——一次对 4 个输入进行内层哈希，另一次对 1 个输入进行外层哈希。(C3) 吸收 7 个域元素（REV‖AlgID‖Domain‖Index‖TxHash‖domain‖nonce），需要 4 个吸收轮数，占电路总量的 40%。

两种重放模式（随机数注册表和空值符集合）产生的约束数量相同。域分离（C5）通过对 Ctx 的域组件施加等式约束来实现，开销可以忽略不计。

作为比较，在零知识电路中验证 ML-DSA 这类基于格的签名，需要表达：(i) 在 $\mathbb{Z}_q$ 中对次数为 256 的多项式环执行数论变换（NTT），其中 $q = 2^{23} - 2^{13} + 1$，每次需要 $O(n \log n)$ 次非原生域乘法；(ii) 高位宽模约简（在 ≈ 254 位

[2]本文的原始版本引用了 Poseidon 论文中关于每次哈希调用需要 ≈ 250–300 个约束的 [6] 理论估算，该估算假设使用 $\alpha = 5$，且部分轮次较少。参考实现针对 BN254 标量域使用 $\alpha = 17$，并采用 57 个部分轮次。

表1：ZK-ACE 电路的测量约束分解（Poseidon、 $t = 3$、 $\alpha = 17$、BN254、arkworks）。

约束	输入	哈希调用	R1CS
（C1）承诺一致性	3	1	805
（C2）推导正确性	$4 + 1$	2	1,200
(C3) 授权绑定	7	1	1,615
(C4) 重放防护	2	1	400
(C5) 域分离 + `enforce_equal`	—	—	4
总计		**5**	**4,024**

证明系统域中模拟 23 位模数，需要为每个 limb 配备范围检查小工具）；以及 (iii) 拒绝采样和提示重构逻辑，其中包含依赖于数据的分支。NIST 安全级别 2 下的 ML-DSA 验证涉及 $k \times \ell = 4 \times 4$ 次矩阵—向量数论变换和 $k = 4$ 次多项式重构，即使采用优化的非原生算术，R1CS 约束数量的结构性下限仍达数百万。因而，实测的 ZK-ACE 电路（4,024 个 R1CS 约束；表1）**比在电路内进行的后量子签名验证小约三个数量级**。这一差距反映了少量零知识友好型哈希调用与基于格的签名验证所需完整代数机制之间的结构性差异。由此得到的电路规模使得在消费级硬件上生成证明成为可行（单线程耗时 ≈ 63 毫秒；见第 12 节），并支持实用的批处理和递归组合（第 11 节）。

7.4 重放保护设计

ZK-AC E 支持多种防重放策略。我们介绍两种典型方案 .

方案 A：随机数注册表（账户式模型）。在账户式模型中，每个身份承诺都会在链上维护一个随机数状态。证明将随机数承诺（式 6）作为公开输入，验证者则根据协议规则（例如，随机数必须严格递增，或等于预期的下一个随机数）确保该随机数是新鲜的。

权衡。随机数注册表可提供简单、确定性的重放保护，并能自然地与账户抽象集成。不过，它们会引入每个身份对应的状态更新，并泄露顺序信息，这在隐私保护场景中可能并不理想。此外，基于随机数的模型要求依次提交，从而限制并行处理能力。

方案 B：空化符集合（隐私导向模型）。在隐私导向模型中，证明会输出一个由授权令牌派生的空化符（式 7）。验证者维护一个全局的已使用空化符集合，并拒绝任何空化符曾经出现过的交易。

权衡。 空化符支持一次性授权，无需为每个身份维护随机数计数器，并且更有利于实现不同授权之间的不可链接性。然而，维护和检查无效标识符集合会导致全局状态增长，因此必须谨慎设计，以防止不同域之间出现非预期的关联性。

备注 7.1. 随机数模型与空值符模型之间的选择取决于具体部署，且不会影响 *ZK-ACE* 的核心安全属性。在各自的验证者强制执行规则下，这两种模型均满足抗重放定义（定义 *8.2*）。

8 安全定义与分析

本节通过明确的基于博弈的定义，对 ZK-ACE 的安全属性进行形式化，并按照基于代码的博弈玩法方法，给出基于标准假设的归约证明

学方法 [23]。关于底层密码学工具的全面介绍，请参阅 [24]。全文中， $\mathcal{A}$ 表示一个概率多项式时间对手。

注释 8.1（将实际攻击场景映射到形式化博弈）。 威胁模型（第 *3.5* 节）赋予 $\mathcal{A}$ 提交、重新排序、重放和替换交易负载的能力。下文定义的四种安全博弈分别刻画了这些能力：

- 未授权的授权（身份盗用）：不持有 REV 的对手试图为链上身份承诺生成有效证明。这由授权可靠性游戏（定义 *8.1*）刻画。
- 抢跑和交易替换：对手在内存池中观察到有效的授权证明后，试图将其重新绑定到不同的交易哈希。由于该证明通过约束 *(C3)* 与 交易哈希 进行密码学绑定，因此这一行为由替换抵抗性博弈（定义 *8.3*）刻画。
- 重放和状态竞争攻击：对手试图重新提交先前已接受的授权，或利用并发提交造成双花。这两种情况均由抗重放游戏（定义 *8.2*）刻画，该游戏的分析涵盖随机数（顺序排列）和空化符（集合成员关系）执行模型。
- 跨链或跨应用证明迁移：对手从一个域获取有效证明，并试图在另一个域中使用该证明。跨域分离博弈（定义*8.4*）涵盖了此类情况，它通过在所有承诺和绑定计算（*C1–C5*）中纳入域来强制实现。

这四种博弈共同覆盖了威胁模型中描述的攻击面。未涵盖的攻击（例如底层REV 遭到破坏或共识层遭受拒绝服务）不在讨论范围内，正如第*3.5*节所述。

8.1 授权可靠性

定义 8.1（授权可靠性游戏）。该游戏$\mathsf{Game}^{\mathsf{auth}}_{\mathcal{A}}(\lambda)$ 按如下方式进行：
1. 设置。挑战者生成一个新的身份：$REV \$ \leftarrow \{0,1\}^{256}$、盐值 $\$ \leftarrow \{0,1\}^{\lambda}$，以及 $ID_{\mathsf{com}} = H(REV \| salt \| domain)$。验证密钥 vk 和承诺 ID_{com} 提供给$\mathcal{A}$。*REV* 和盐值不会公开。 *2.* 预言机访问。 $\mathcal{A}$ 可以观察所有公开链上数据，包括此前被接受的证明及其公开输入。 *3.* 输出。 $\mathcal{A}$ 输出一个候选对 $(\pi^*, \pi^*_{\mathsf{pub}})$，其中 π^*_{pub} 包含作为身份承诺的 ID_{com} 。 *4.* 获胜条件。若 $ZKVerify(vk\ \pi^*, \pi^*_{\mathsf{pub}}) = 1$ 成立，且提交元组通过重放谓词，则 $\mathcal{A}$ 获胜。

的优势为 $\mathcal{A}$ $\mathsf{Adv}^{\mathsf{auth}}(\lambda) = \Pr[\text{博弈}\ \mathsf{Auth}(\lambda) = 1]$。

注记 8.2（对抗模型的强度）。 定义 *8.1* 将对被接受的证明进行被动观察作为对抗接口的基准。此模型足以支持我们的核心归约。可以扩展为显式的自适应选择消息授权预言机，但这需要额外的证明系统级假设（例如，授权接口的模拟可靠性/不可延展性），才能保持界限不变。我们将这一强化留待未来形式化。

定理 8.1（授权可靠性）。 在满足以下条件时：*(i)* 证明系统具有知识可靠性；*(ii)* H具有抗碰撞性；*(iii) DIDP*（第 *3.3* 节）具有身份根恢复困难性；对于任意概率多项式时间 对手 $\mathcal{A}$：

$$\mathsf{Adv}^{\mathsf{auth}}_{\mathcal{A}}(\lambda) \le \mathsf{Adv}^{\mathsf{ks}}_{\mathcal{B}_1}(\lambda) + \mathsf{Adv}^{\mathsf{cr}}_{\mathcal{B}_2}(\lambda) + \mathsf{Adv}^{\mathsf{rec\text{-}didp}}_{\mathcal{B}_3}(\lambda) + \mathsf{negl}(\lambda).$$

证明。 假设 $\mathcal{A}$输出了验证者接受的 $(\pi^*,\ \pi^*_{\mathsf{pub}})$。我们按如下方式构造归约 $\mathcal{B}_1$， $\mathcal{B}_2$， $\mathcal{B}_3$。

步骤 1：见证提取。 根据证明系统的知识可靠性性质，存在一个概率多项式时间提取器 $\mathcal{E}$，使得给定 $\mathcal{A}$的可接受证明及随机性， $\mathcal{E}$ 会输出一个见证 $w^* = (\mathsf{REV}^*, \text{盐值}^*, \text{上下文}^*, \text{随机数}^*, \text{辅助信息}^*)$，满足所有电路约束 (C1)–(C5)，除非概率至多为 $\mathsf{Adv}^{\mathsf{ks}}_{\mathcal{B}_1}(\lambda)$。我们构造 $\mathcal{B}_1$ 来模拟针对 $\mathcal{A}$ 的博弈，并在收到可接受的证明后调用 $\mathcal{E}$ 以恢复见证。

步骤 2：对提取出的见证进行案例分析。假设提取成功。根据约束 (C1)，提取出的见证满足：

$$H(\mathsf{REV}^* \| \mathsf{salt}^* \| \mathsf{domain}) = \mathsf{ID}_{\mathsf{com}} = H(\mathsf{REV} \| \mathsf{salt} \| \mathsf{domain}).$$

我们考虑两种情况：

情况 *1*：$(\mathsf{REV}^*, \mathsf{salt}^*) \neq (\mathsf{REV}, \mathsf{salt})$。在这种情况下， $\mathcal{A}$ 为 $\mathsf{ID}_{\mathsf{com}}$在 H下生成了两个不同的原像，构成一次碰撞。我们构造 $\mathcal{B}_2$，将抗碰撞性挑战嵌入 H，并将 $\mathcal{A}$ 用作子程序：$\mathcal{B}_2$ 使用挑战哈希诚实地模拟博弈；当提取结果为 $(\mathsf{REV}^*, \mathsf{salt}^*) \neq (\mathsf{REV}, \mathsf{salt})$ 时，输出两个原像作为碰撞。因此 $\mathsf{Adv}^{\mathsf{cr}}_{\mathcal{B}_2}(\lambda) \geq \Pr[\text{情况 } 1]$。

情况 *2*：$\mathsf{REV}^* = \mathsf{REV}$ 且$\mathsf{salt}^* = \mathsf{salt}$。在这种情况下，提取出的见证包含正确的 REV。由于 $\mathcal{A}$ 并未获知 REV，这构成了对 DIDP 身份根恢复困难性的破坏（第 3.3 节）。我们按如下方式构造 $\mathcal{B}_3$。 $\mathcal{B}_3$ 参与恢复游戏 $\mathsf{Game}^{\mathsf{rec\text{-}didp}}$，从 DIDP 挑战者处获得对 $\mathcal{O}_{\mathsf{pub}}$ 和 $\mathcal{O}_{\mathsf{derive}}$ 的预言机访问。 $\mathcal{B}_3$采样盐值 $\$ \leftarrow \{0,1\}^\lambda$，并查询 $\mathcal{O}_{\mathsf{pub}}(\mathsf{salt}, \mathsf{domain})$，以获得 $\mathsf{ID}_{\mathsf{com}} = H(\mathsf{REV} \| \mathsf{salt} \| \mathsf{domain})$，而无需获知 REV。同样， $\mathcal{B}_3$ 查询 $\mathcal{O}_{\mathsf{derive}}(\mathsf{Ctx})$，以获得任意模拟公开链上数据所需上下文的目标绑定值$H(\mathsf{Derive}(\mathsf{REV}, \mathsf{Ctx}))$。 $\mathcal{B}_3$ 使用 $\mathsf{ID}_{\mathsf{com}}$和推导出的公开值，为 $\mathcal{A}$忠实地模拟授权可靠性游戏。当提取成功并得到 REV^* =REV 时， $\mathcal{B}_3$ 在 DIDP 恢复游戏中输出 REV^* 作为答案。因此 $\mathsf{Adv}^{\mathsf{rec\text{-}didp}}_{\mathcal{B}_3}(\lambda) \geq \Pr[\text{情况 } 2]$。

步骤 3：组合。 根据所有失败事件的并集界：

$$\mathsf{Adv}^{\mathsf{auth}}_{\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{ks}}_{\mathcal{B}_1}(\lambda) + \mathsf{Adv}^{\mathsf{cr}}_{\mathcal{B}_2}(\lambda) + \mathsf{Adv}^{\mathsf{rec\text{-}didp}}_{\mathcal{B}_3}(\lambda) + \mathsf{negl}(\lambda). \qquad \square$$

8.2 抗重放性

De定义 8.2 (抗重放游戏)。 博弈 $\mathsf{Game}^{\mathsf{replay}}_{\mathcal{A}}(\lambda)$ 按如下方式进行 :

1. 设置。 如定义 *8.1* 所述。诚实证明者为哈希为 TxHash_1，的交易生成一个有效的授权元组$(\pi_1, \pi_{1,\mathsf{pub}})$，该交易随后被验证者接受并应用于验证者状态（随机数递增或插入空化符）。

2. 预言机访问。 $\mathcal{A}$ 接收$(\pi_1, \pi_{1,\mathsf{pub}})$，并可完全访问链上状态。

3. 获胜条件。 若 $\mathcal{A}$ 导致第二个授权元组$(\pi^*, \pi^*_{\mathsf{pub}})$被接受，则 $\mathcal{A}$ 获胜，其中 π^*_{pub} 包含ID com 以及相同的逻辑授权事件（即，该元组违反了协议的抗重放不变量）。

定理8.2 (重放抗性)。 在定理 *8.1* 的假设以及验证者正确强制执行重放谓词的前提下，对于任意概率多项式时间 对手 $\mathcal{A}$：

$$\mathsf{Adv}^{\mathsf{replay}}_{\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{auth}}_{\mathcal{A}}(\lambda) + \mathsf{negl}(\lambda).$$

证明。 我们分别考虑每种重放模型：

随机数模型。 在使用随机数 n 完成诚实授权后，验证者状态将推进至等待随机数 $n+1$。

- 直接重放：重新提交（π_1，$\pi_{1,\ \mathsf{pub}}$）会生成使用随机数 n 计算得到的 $\mathrm{rp_com}$。验证者拒绝该请求，因为预期的随机数是 $n+1$。

- 使用过期随机数的新证明：任何使用随机数 $\leq n$ 计算 $\mathrm{rp_com}$ 的新证明都会被单调性检查拒绝。
- 使用新鲜随机数的新证明：要生成随机数为 $n+1$ 的有效证明，必须有满足 (C1)–(C5) 的有效见证；这可归约为授权可靠性游戏（定理 8.1）。

空值符模型。诚实授权后，空化符 $\mathrm{rp_com}_1 = H(\mathrm{Auth}_1 \| \text{域})$ 会被加入无效标识符集合 $\mathcal{N}$。

- 直接重放：重新提交 $(\pi_1,\ \pi_{1,\ \text{公钥}})$ 会生成相同的 $\mathrm{rp_com}_1 \in \mathcal{N}$，因此会被拒绝。
- 相同授权，不同证明：针对相同的 (REV，上下文，交易哈希，域，随机数) 的任何证明都会生成相同的 Auth_1，因此空化符也相同，故会被拒绝。
- 不同的授权参数：更改 Auth 计算中的任何组件，都会产生不同的 Auth' 和新的空化符 $\mathrm{rp_com}' \notin \mathcal{N}$。这代表一个真正的新授权事件（交易、上下文或随机数不同），因此会被接受——这是正确的，因为它并非重放。

在两种模型中，$\mathcal{A}$ 导致重放的授权事件被接受的唯一方法，是使用一个新的重放令牌生成有效证明，而这需要掌握有效见证。这可归约为授权可靠性。

□

8.3 抗替换性

定义 8.3（替换抵抗性博弈）。博弈$\mathrm{Game}^{\mathsf{subst}}_{\mathcal{A}}(\lambda)$ 按如下方式进行：

1. 设置。如定义 *8.1*。生成一个诚实授权元组（π,π_{pub}），其公开输入为（身份标识$_{\mathsf{com}}$,交易哈希,域,目标, $\mathrm{rp_com}$）。*2.* 预言机访问。$\mathcal{A}$ 接收（π,π_{pub}）。*3.* 获胜条件。$\mathcal{A}$ 若验证者接受（π, π'_{pub}）且 $\pi'_{\mathsf{pub}} \neq \pi_{\mathsf{pub}}$，则获胜；也就是说，同一证明在修改后的公开输入下仍被接受（交易哈希、目标或域不同）。

定理 8.3（抗替换性）。在证明系统满足公开输入绑定性质的前提下，对于任意概率多项式时间对手 $\mathcal{A}$：

$$\mathsf{Adv}^{\mathsf{subst}}_{\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{pib}}_{\mathcal{B}}(\lambda) + \mathsf{negl}(\lambda).$$

证明。假设 $\mathcal{A}$输出了 $\pi'_{\mathsf{pub}} \neq \pi_{\mathsf{pub}}$，使得 $\mathsf{ZKVerify}(\mathrm{vk}\ \pi,\ \pi'_{\mathsf{pub}}) = 1$。那么，同一个证明 π 在两个不同的公开输入元组下均可验证。我们构造 $\mathcal{B}$，将此情形嵌入公开输入绑定挑战中：$\mathcal{B}$ 模拟针对 $\mathcal{A}$的替换博弈；若 $\mathcal{A}$成功，则输出 $(\pi,\ \pi_{\mathsf{pub}},\ \pi'_{\mathsf{pub}})$，作为违反公开输入绑定性质的证据。

根据公开输入绑定性质（第3.4节），这种情况发生的概率至多为可忽略的le概率，因此可得：

$$\mathsf{Adv}^{\mathsf{subst}}_{\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{pib}}_{\mathcal{B}}(\lambda) + \mathsf{negl}(\lambda).$$ □

8.4 跨域分离

定义8.4 (跨域分离博弈)。博弈 $\mathrm{Game}^{\mathsf{domain}}_{\mathcal{A}}(\lambda)$ 按如下方式进行：

*1.*设置。挑战者在两个不同的域上为同一身份注册域$_1$ $\neq$域$_2$，并分别使用承诺身份$_{\mathsf{com}}{}^{(1)} = H(\mathsf{REV}\|\text{盐值}_1\|\text{域}_1)$ 和身份$_{\mathsf{com}}{}^{(2)} = H(\mathsf{REV}\|\text{盐值}_2\|\text{域}_2)$，其中盐值$_1$,盐值$_2$ 是独立抽样的。针对域域$_1$生成一个诚实的授权证明(π,π_{pub})。

2. 获胜条件。$\mathcal{A}$ 若使某个域域$_2$上的授权元组被接受，且使用了来自域域$_1$ 授权的材料（通过证明重用或派生），则获胜。

定理8.4 (跨域分离)。在 H 具备抗碰撞性且证明系统具备公开输入绑定的前提下，对于任意概率多项式时间 对手 $\mathcal{A}$：

$$\mathsf{Adv}^{\mathsf{domain}}_{\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{pib}}_{\mathcal{B}_1}(\lambda) + \mathsf{Adv}^{\mathsf{cr}}_{\mathcal{B}_2}(\lambda) + \mathsf{negl}(\lambda).$$

证明。 我们考虑两种攻击策略：

证明重用。 如果 $\mathcal{A}$ 将原始证明 π （为域 domain_1生成）提交给域 domain_2 的验证者，则公开输入必须包含 $\mathrm{ID}_{\mathsf{com}}{}^{(2)}$ 和 domain_2。由于 π 是针对包含 $\mathrm{ID}_{\mathsf{com}}{}^{(1)}$ 和 domain_1, 的公开输入生成的，而 $(\mathrm{ID}_{\mathsf{com}}{}^{(1)},\ \mathrm{domain}_1) \neq (\mathrm{ID}_{\mathsf{com}}{}^{(2)},\ \mathrm{domain}_2)$，因此根据公开输入绑定，验证将失败。

承诺碰撞。 若 $\mathcal{A}$ 试图找到(REV′，盐值′， 域$_2$)，使得 H(REV′||盐值′||域$_2$) $=\mathrm{ID}_{\mathsf{com}}{}^{(1)}$，则会产生两个不同的域标记输入映射到同一哈希值，因此构成 H下的碰撞事件。

此外，根据约束(C5)，域包含在所有电路绑定中。以域 $=$域$_1$ 生成的证明，在声明域为域$_2$, 时无法通过验证者的约束检查，因为承诺、授权令牌和防重放承诺均纳入了域。综合可得：$\mathrm{Adv}_{\mathcal{A}}^{\mathsf{domain}}(\lambda) \leq \mathrm{Adv}_{\mathcal{B}_1}^{\mathsf{pib}}(\lambda) + \mathrm{Adv}_{\mathcal{B}_2}^{\mathsf{cr}}(\lambda) + \mathsf{negl}(\lambda)$。

□

8.5 组合

推论 8.1 （组合安全性）。 在定理 *8.1–8.4* 的假设下，*ZK-ACE* 同时满足针对概率多项式时间对手的授权可靠性、抗重放性、抗替换性和跨域分离。具体而言，对于任何概率多项式时间 对手 $\mathcal{A}$ 攻击任一单一性质，存在概率多项式时间 归约 $\mathcal{B}_1,\ldots,\mathcal{B}_5$，使得：

$$\mathrm{Adv}_{\mathcal{A}}^{\mathsf{combined}}(\lambda) \leq \mathrm{Adv}_{\mathcal{B}_1}^{\mathsf{ks}}(\lambda) + 2 \cdot \mathrm{Adv}_{\mathcal{B}_2}^{\mathsf{cr}}(\lambda) + \mathrm{Adv}_{\mathcal{B}_3}^{\mathsf{rec\text{-}didp}}(\lambda) + 2 \cdot \mathrm{Adv}_{\mathcal{B}_4}^{\mathsf{pib}}(\lambda) + \mathsf{negl}(\lambda),$$

其中，2 倍的$\mathrm{Adv}^{\mathsf{cr}}$ 源于授权可靠性证明（定理 *8.1*，情形 *1*）和跨域分离证明（定理 *8.4*）都归约为抗碰撞性；2 倍的$\mathrm{Adv}^{\mathsf{pib}}$ 则源于抗替换性证明（定理 *8.3*）和跨域分离证明都归约为公开输入绑定。在上述假设下，该界是可忽略的。

命题 8.1 (完备性)。 如果诚实证明者持有满足所有约束 *(C1)–(C5)* 的有效见证 $w =$(REV、盐值、上下文、随机数、aux)，且防重放状态允许提交该令牌（即随机数是新鲜的，或空化符尚未使用），则验证者以 $1 - \mathsf{negl}(\lambda)$ 的概率接受所生成的证明。

证明。 根据底层证明系统的完备性，对于满足约束的见证而诚实生成的证明，以压倒性概率被接受。根据假设，防重放谓词会因所提交的随机数或空化符具有新鲜性而通过。所有公开输入在构造时均根据见证一致地计算，因此验证者的约束检查均能通过。

□

注释 8.3 (隐私保证)。根据底层证明系统的零知识性质，*ZK-ACE* 授权证明不会透露有关私有见证(REV、盐值、上下文、随机数、aux)的任何信息，除非这些信息可由公开输入(ID$_{\mathsf{com}}$、交易哈希、域、目标、rp_com)推知。具体而言：

- *256*位根熵值REV 仍处于隐藏状态。由于REV 是唯一的身份根，所有密码学材料均由其派生，因此对其保密可确保观察者无法计算任何派生密钥。
- 承诺盐盐值不会公开，从而确保使用不同盐值生成的承诺之间的身份不可链接性。
- 派生上下文 上下文 和随机数 随机数 均处于隐藏状态，因此观察者无法推断目标算法、应用域、派生索引或重放防护状态；公开的 rp_com 所揭示的信息除外。

观察者能够获取的信息仅限于：*(i)* 使用了哪个身份承诺，*(ii)* 授权了哪笔交易，*(iii)* 域和目标上下文，以及 *(iv)* 重放防护令牌。这种信息泄露特征是任何必须将证明绑定到特定交易并实施重放防护的授权系统所固有的，也是共识验证所需的最少信息披露。

9 链上验证与部署

9.1 验证接口

在共识层，ZK-ACE 提供一个最小化的验证接口，接受零知识证明以及一组固定的公开输入。从概念上讲，验证者会评估如下形式的谓词：

$$\mathsf{ZKVerify}(\mathsf{proof}, \mathsf{public_inputs}) \in \{\mathsf{accept}, \mathsf{reject}\}.$$

关键在于，验证者不会直接接受证明者所声明的公开输入。在评估证明之前，验证者必须独立验证上下文一致性（算法 1，第 6–7 步）：(i) 根据提交的交易负载重新计算交易哈希，并确认其与 π_{pub} 中声明的值一致；(ii) 确认 $\mathrm{ID}_{\mathsf{com}}$ 对应于已注册的链上身份承诺；以及 (iii) 确认域和目标与预期的交易上下文一致。只有通过这些检查后，验证者才会评估零知识证明，并在验证成功后执行与重放防护相关的必要状态更新：

- 在每个身份的随机数注册表中记录已消耗的随机数；
- 将空化符插入全局无效标识符集合以防止重复使用；或
- 更新授权索引或类似的单调状态变量。

这些更新是确定性的，且可由所有共识参与者验证。身份根材料（REV）或派生密钥绝不会写入链状态。

9.2 与账户抽象的兼容性

ZK-ACE 天然兼容账户抽象（AA）架构，包括与 ERC-4337 风格交易流程相契合的设计[16]。

验证器模块集成。 在 AA 场景中，ZK-ACE 可作为与抽象账户关联的验证器模块 来实现。账户验证逻辑不再检查经典签名或后量子签名，而是调用 ZK-ACE 验证接口，以确定该交易是否已由已承诺身份授权。

打包器与提交流程。零知识证明的生成在链下进行，由控制该身份的实体执行，通常是用户的钱包软件。打包器或中继者可以收集用户操作并将其提交到区块链，但无需访问任何秘密身份材料。打包器只负责传输证明和公开输入，不会影响授权正确性。这种分离保留了账户抽象的最小化信任设计，同时允许灵活分配证明生成和交易提交的职责。

9.3 实例化灵活性

证明系统选择。 ZK-ACE 可采用不同类别的零知识证明系统来实现。基于SNARK的系统（例如 Groth16 [8]）可提供最小的证明大小，

并实现最快的验证速度，但代价是需要可信设置。通用系统（例如 PLONK [7]）无需针对每个电路分别进行可信设置。透明系统（例如 STARK [9]）完全不需要可信设置，但生成的证明更大。基于 IPA 的系统（例如 Bulletproofs [10]）则在证明大小与验证时间之间提供了另一种权衡。除可靠性、零知识性和公开输入绑定外，该框架不依赖任何特定系统独有的属性。

哈希函数参数化。 零知识友好型哈希函数（如 Poseidon [6], ）用于电路中的承诺与绑定。参数选择（例如元数、轮数、域大小）交由部署环境决定。ZK-ACE 的安全性论证仅依赖于标准的抗碰撞性与抗原像性假设，而不依赖于任何特定参数集。

证明系统迁移。 由于身份承诺与证明系统无关（它们仅依赖于哈希函数，而非零知识证明系统），验证者与证明者的升级无需进行身份轮换。区块链可以从一种证明系统迁移到另一种证明系统，而不会使现有身份承诺失效。

10 结构性链上数据核算

本节对不同授权模型下的链上授权数据进行结构性核算。其目的是比较基于后量子签名的授权与 ZK-ACE 授权所要求的协议规定的数据对象，且独立于具体实现或性能基准测试。

10.1 核算方法

该分析并非基于实验测量、运行时基准测试或 Gas 分析。相反，它依赖于对各模型下为交易授权而必须纳入链上的数据对象进行协议级核算。具体而言，我们比较：

- 共识参与者必须接收、验证并存储的强制性授权凭证；以及
- 这些凭证的渐近大小，这由密码学标准和协议结构决定。

10.2 基线：后量子签名数据对象

在基于后量子签名的授权模型中，每笔交易都必须携带一个由共识层验证的密码学签名对象。对于基于格的签名方案，标准化参数集所对应的签名大小通常为数千字节。表 2 总结了具有代表性的大小。

其显著特点是，后量子签名对象的大小为千字节级。这些对象必须传输、验证并作为账本的一部分永久记录，因此其大小是链上数据增长的首要影响因素。

10.3 ZK-ACE 授权凭证

在 ZK-ACE 下，显式的后量子签名对象不会置于链上。相反，授权由紧凑承诺与简洁零知识证明的组合来表示。每笔交易所需的链上产物为：

- **身份承诺**：针对确定性身份（$\mathrm{ID}_{\mathrm{com}}$）的基于哈希的承诺，通常占用 32 字节。

表 2：标准化方案的后量子签名工件大小。

方案	NIST 安全级别	签名大小	公钥
ML-DSA-44 [2]	2	2,420 字节	1,312 字节
ML-DSA-65 [2]	3	3,309 字节	1,952 字节
ML-DSA-87 [2]	5	4,627 字节	2,592 字节
SLH-DSA-128f [3]	1	17,088 字节	32 字节
FN-DSA-512 [4]	1	≈666 字节	897 字节
Ed25519（经典）		64 字节	32 字节
secp256k1 ECDSA（经典）		≈71 字节	33 字节

表3：每笔交易授权数据的结构对比。后量子密码签名模型列假设首次使用时传输公钥（实际中，对于重复发送方，这部分开销可摊销至零）。ZK-ACE 列假设使用 Groth16 证明大小；采用 PLONK 或 STARK 实例时，证明部分会增大。两列均未计入特定链的编码开销（例如 RLP、SSZ 或调用数据填充）。

组件	后量子密码签名模型	ZK-ACE
签名 / 证明	2,420–4,627 B	128–256 B（Groth16）
公钥（摊销后）	1,312–2,592 B	0 B（重复使用承诺）
身份承诺	—	32 B
公开输入	—	≈160 B
每笔交易总计	**3,732–7,219 B**	**≈320–448 B**

- **零知识证明**：用于证明授权正确性的简洁证明。现代零知识证明系统生成的证明大小固定，通常约为 128–256 字节（Groth16）至约 1 KB（Plonk/STARK），且不受底层授权语义复杂度的影响。
- **公开输入**：一组少量的共识可见值，包括交易哈希、域、目标和 rp_com。这些输入通常占用约 128–192 字节。

关键在于，这些凭证都不会随着后量子签名方案的大小或内部结构而变化。

10.4 数量级比较

比较两种模型的结构性要求，可以看出链上授权数据存在明显的数量级差异，汇总见表3：

根据标准化的后量子签名大小（ML-DSA 2–5级，表 2）和具有代表性的零知识证明大小（Groth16），这种结构性替换意味着授权凭证的单笔交易链上数据可减少约 *10–20*× 。这些数据直接来自表 3 中的字节数比较，并不依赖于特定于实现的优化。重要的是，这种减少并非源于对后量子签名进行压缩，而是通过消除将后量子签名材料置于链上的必要性，并以授权语义的简洁证明取而代之。

比较注意事项。 表3展示了一项结构性、协议层比较，旨在说明授权制品大小的数量级差异。在任何具体部署中，若干因素都会影响精确比例：(i) 公钥摊销取决于发送方的密钥是否已为区块链所知（在后量子密码学模型中，重复发送方的摊销成本为零，从而缩小差距）；

；(ii) 零知识证明大小因证明系统而异（Groth16 生成的证明大小为 ≈ 128–256B，而 PLONK 或 STARK 证明更大）；以及 (iii) 特定链编码格式（RLP、SSZ、ABI 编码）会给两种模型都带来开销。因此，与其将此比较视为对任何特定区块链上实际 Gas 成本或字节数的预测，不如将其理解为对各授权模型所要求的密码学产物大小 进行的结构性核算。

11 可扩展性扩展

11.1 批量授权

来自不同身份的多个独立授权语句，可以通过批量验证或证明聚合技术组合成一个聚合证明。这能摊销验证者的工作量，并减少链上产物的数量：无需使用 n 个独立证明（每个 $\approx$128–256 字节），单个批量证明即可证明 n 条授权语句，从而降低每笔交易的验证成本和数据开销。

批量授权保留了每个组件语句各自的安全保证：聚合证明的知识可靠性意味着每个组成语句也具备知识可靠性。聚合器（负责收集各个证明并生成批量证明）并不可信，也无法获知任何私有见证材料。

11.2 递归证明组合

递归组合允许顶层证明证明多个基础层证明 [11, 12] 的有效性。应用于 ZK-ACE 时，递归证明可以验证 k 个独立授权证明，并生成一个大小恒定的证明，无论 k 如何。这在 Rollup 架构中特别有用，因为单个递归证明即可授权将整批交易提交到基础层。

递归保留了 ZK-ACE 的所有安全属性：如果递归证明系统具备知识可靠性，那么存在一个可接受的递归证明，就意味着所有组成授权都存在有效的见证。由于域标签会在递归过程中传播，因此跨域分离得以保持。

12 参考实现与基准测试

为验证第 7.3 节和第 10 节的结构分析，我们提供了 ZK-ACE 电路的参考实现，并报告了单线程 CPU 基准测试结果。该实现采用开源方案，并使用 arkworks 库生态系统[26]进行 BN254 算术和 Groth16 证明。

12.1 实现

该参考电路使用 arkworks `PoseidonSpongeVar`小工具实现全部五个约束（C1）–（C5），在 BN254 标量域上采用宽度为 $t = 3$、S 盒指数为 $\alpha = 17$、8 个完整轮次和 57 个部分轮次。Groth16 [8] 用作证明系统。在随机数注册表和空值符集合重放模式下，该电路均产生相同的约束数量（4,024 个 R1CS），这证实了重放模式的选择不会影响电路复杂度。

实现还包含一个使用相同参数的原生（链下）Poseidon 哈希，以确保电路内计算与链下见证生成在功能上等效。

表 4：ZK-ACE 的 Groth16 证明系统基准测试（单线程，Apple M3 Pro）。

操作	中位数	95% 置信区间
可信设置（一次性）	120 ms	[111, 138]ms
证明（每笔交易）	63 毫秒	[61, 65]毫秒
验证（每笔交易）	651 μ秒	[618, 703] μ秒

表 5：ACE 运行时流水线基准测试（单线程，Apple M3 Pro）。阶段 2 使用模拟证明器；实际 Groth16 耗时见表 4。

操作	批量大小		中位时间
HKDF 密钥派生（单次）	1		1.16 μ秒
身份承诺（$\mathrm{ID}_{\mathrm{com}}$）	1		270 ns
证明生成	1		2.25 μs
证明验证	1		0.82 μs
阶段 1a：AttestCheck	2,000	247 μ秒	（124 纳秒/笔交易）
阶段 1b：执行	2,000	134 μ秒	（67 纳秒/笔交易）
状态根（默克尔）	2,000		537 μ秒
区块构建	2,000		1.97 毫秒
阶段 2：模拟证明†	2,000		4.16 毫秒
端到端流水线†	2,000		7.56 毫秒
REV Argon2id 封存/解封	1		3.5 毫秒

†模拟证明器：仅为编排开销，不包含实际的零知识证明生成。

12.2 基准测试

所有测量均使用 Criterion.rs 在运行 macOS 15.7、Rust 1.92 的 Apple M3 Pro（12 核、36GB RAM）上采集，执行采用单线程模式，并使用默认编译器优化设置（`--release`）。

零知识证明操作。 表 4 报告了单个授权证明的 Groth16 核心操作。所有计时数据均为 Criterion.rs 中位数，来自 `cargo bench -p zk-ace --benchprove_verify_bench`。

每份证明 $\approx$0.65 ms 的验证时间完全处于 L1 与 Rollup 场景下链上验证的预算之内。$\approx$63ms 的证明时间对应单线程 CPU 执行；先前关于 GPU 加速的 Groth16 证明器（例如 Icicle、Bellman-GPU）的研究表明，在可比电路规模上可获得 5–15$\times$ 倍加速，据此可将单证明延迟推算至 $\approx$4–13 ms，但尚未针对该特定电路进行实测。

证明与流水线操作。表 5 报告了身份层原语与共识关键流水线阶段的计时，这些数据在 ACE 运行时上使用 `cargo bench -p ace-runtime --features test-utils --bench pipeline_bench` 测得。**注：**阶段 2（证明生成）使用确定性模拟证明器，模拟每笔交易的 SHA-256 哈希与树聚合；它并不调用真实的 Groth16 证明。模拟证明器计时反映的是证明编排的开销——见证值编组、树构建与数据序列化——不包括零知识证明计算本身，后者在表 4 中单独进行基准测试。

在批量大小为 2,000 时，共识关键 Attest–Execute 路径（阶段 1a + 1b + 默克尔根）在 $\approx$0.92ms 内完成，远低于 400 ms 时隙窗口的 1%。证明检查

阶段处理每笔交易耗时 ≈ 124ns，与设计预期的亚微秒级 HMAC-SHA256 验证一致。端到端流水线（包含模拟证明）耗时 ≈ 7.56ms；将模拟证明器替换为真实的 Groth16 证明将增加 $\approx 63\,\text{ms} \times 2{,}000$ 的证明者工作量，该工作可在 GPU 核心之间并行执行，且不在共识关键路径上（第 11.1 节）。

13 局限性与未来工作

- **DIDP 依赖性**。 ZK-ACE 的安全性取决于底层 DIDP（第 3.3 节）所假定的安全属性。这些属性——确定性、上下文隔离和身份根恢复困难性——在将派生函数建模为随机预言机时，可由标准假设（REV 的抗碰撞性和高最小熵）推导得出。ACE-GF [1] 是一个具体实例；对任何候选 DIDP 进行独立分析，将进一步增强本工作的基础。

- **空值符集合状态增长**。 在空值符模型中，全局空值符集合会随着授权事件数量的增加而单调增长。长期状态管理策略（例如基于纪元的剪枝、默克尔树压缩）仍是一个有待解决的工程问题。

- **单平台基准测试**。 第 12 节中的基准测试是在单一平台（Apple M3 Pro、单线程 CPU）上收集的。精确到 Gas 的、特定于区块链的成本估算和 GPU 加速证明时间仍有待测量。跨平台验证（例如 x86、ARM 服务器、WASM）和形式化电路审计将留待后续工作。

- **证明者成本**。 虽然验证者成本较低（≈ 0.65 毫秒），但在资源受限的设备上，证明者成本（单线程 CPU 上为 ≈ 63 毫秒）可能不容忽视。关于 GPU 加速的 Groth16 证明器的既有研究报告称，在相近的电路规模下可实现 5–15$\times$ 倍加速，但尚未针对 ZK-ACE 电路进行测量。

- **身份撤销与轮换**。 按当前规范，ZK-ACE 不包含用于身份撤销或承诺轮换的链上机制。未来的研究方向之一，是扩展该框架，使其支持与身份承诺模型兼容的撤销语义。

- **隐私与吞吐量权衡**。 在随机数模型与空化符模型之间进行选择，涉及排序透明性与隐私之间的根本权衡。值得进一步研究针对不同授权类别结合这两种策略的混合模型。

14 结论

ZK-ACE 将后量子安全区块链授权重新定义为身份绑定证明问题，而非签名对象验证问题。通过以确定性授权的零知识证明取代随交易携带的后量子签名，该模型在保留验证者侧语义保证的同时，消除了共识对千字节级签名产物的依赖。

该构造通过一组最小组件实现这一点：一个锚定在链上的身份承诺、一个强制执行承诺一致性、目标绑定、交易授权、重放防护和域分离的电路，以及一个与账户抽象和 Rollup 架构兼容的验证接口。形式化安全分析表明，授权可靠性、抗重放性、抗替换性和跨域分离均可归约为标准密码学假设。

结构化数据核算表明，与直接部署后量子签名相比，每笔交易授权数据减少了一个数量级；这并非通过压缩签名实现，而是完全消除了共识路径中对签名对象的需求。参考实现证实，电路规模总计为 4,024 个 R1CS 约束

在消费级硬件上，验证耗时不到 1 毫秒，单线程证明生成耗时不到 100 毫秒。凭借对批量聚合和递归组合的支持，ZK-ACE 为后量子区块链系统提供了一种可扩展的授权架构。

参考文献

[1] J. S. Wang。ACE-GF：原子密码实体生成框架。arXiv 预印本 arXiv:2511.20505，2025年。 https://arxiv.org/abs/2511.20505.[2] NIST。FIPS 204：基于模格的数字签名标准（ML-DSA）。联邦信息处理标准，2024年。[3] NIST。FIPS 205：无状态哈希签名标准（SLH-DSA）。联邦信息处理标准，2024年。[4] NIST。FIPS 206草案：基于NTRU格的FFT数字签名标准（FN-DSA）。联邦信息处理标准（草案），2024年。[5] A. Biryukov、D. Dinu 和 D. Khovratovich。Argon2：用于密码哈希及其他应用的内存困难函数。发表于 *IEEE EuroS&P*研讨会，2016年。[6] L. Grassi、D. Kales、D. Khovratovich、A. Roy、C. Rechberger 和 M. Schofnegger。Poseidon：一种用于零知识证明系统的新型哈希函数。发表于*USENIX*安全研讨会，2021年。[7] A. Gabizon、Z. J. Williamson 和 O. Ciobotaru。PLONK：用于普世非交互式知识论证的拉格朗日基置换。*IACR* 电子印本 2019/953，2019年。[8] J. Groth。论基于配对的非交互式论证的大小。发表于$EUROCRYPT$，2016年。[9] E. Ben-Sasson、I. Bentov、Y. Horesh 和 M. Riabzev。可扩展、透明且抗后量子攻击的计算完整性。*IACR* 电子印本 2018/046，2018年。[10] B. Bünz、J. Bootle、D. Boneh、A. Poelstra、P. Wuille 和 G. Maxwell。Bulletproofs：用于机密交易等的简短证明。发表于 $IEEE\ S\&P$，2018年。[11] S. Bowe、J. Grigg 和 D. Hopwood。无需可信设置的递归证明组合。*IACR* 电子印本 2019/1021，2019年。[12] A. Kothapalli、S. Setty 和 I. Tzialla。Nova：基于折叠方案的递归零知识论证。发表于$CRYPTO$，2022年。[13] E. B. Sasson、A. Chiesa、C. Garman、M. Green、I. Miers、E. Tromer 和 M. Virza。Zerocash：基于比特币的去中心化匿名支付。发表于$IEEE\ S\&P$，2014年。[14] A. Pertsev、R. Semenov 和 R. Storm。Tornado Cash 隐私解决方案。白皮书，2019年。[15]隐私与扩展探索（PSE）、以太坊基金会。Semaphore：以太坊上的零知识信号。 https://semaphore.pse.dev/，2020年至2024年。[16] V. Buterin、Y. Weiss、D. Kristjansson、N. Luca、T. Silberstein 和 M. Tirosh。ERC-4337：使用替代内存池实现账户抽象。以太坊改进提案，2021年。

[17] V. Buterin、D. Feist、D. Loerakker、G. Kadianakis、M. Garnett、M. Taiwo Adeniyi 和 A. Schoedon。EIP-4844：分片 Blob 交易。以太坊改进提案，2022年。[18] 0xPARC。circom-ecdsa：使用零知识简洁非交互式知识论证证明椭圆曲线数字签名算法。 https://github.com/0xPARC/circom-ecdsa，2022年。[19] D. Boneh 和 M. Franklin。基于 Weil 配对的基于身份的加密。发表于$CRYPTO$，2001年。[20] A. Shamir。基于身份的密码系统和签名方案。发表于$CRYPTO$，1984年。[21] H. Krawczyk 和 P. Eronen。基于 HMAC 的提取-扩展密钥派生函数（HKDF）。RFC 5869，2010年。[22] S. Goldwasser、S. Micali 和 C. Rackoff。交互式证明系统的知识复杂度。*SIAM* 计算杂志，18(1)：186–208，1989年。[23] M. Bellare 和 P. Rogaway。三重加密的安全性及基于代码博弈的证明框架。发表于$EUROCRYPT$，2006年。[24] D. Boneh 和 V. Shoup。应用密码学研究生课程。0.6版，2023年。https://toc.cryptobook.us/.[25] A. Chiesa、G. Fenzi 和 J. Weissenberg。用于受限交错码的零知识交互式Oracle可证明系统。*IACR* 电子印本 2026/391，2026年。[26] arkworks 贡献者。arkworks：用于开发和编写零知识简洁非交互式知识论证的生态系统。 https://arkworks.rs/，2021年至2024年。