中国科学:信息科学 在审文章 onlinelatex

SCIENTIA SINICA Informationis

论文

面向自毁数据胶囊的可追踪访问控制方法

洪克用[1], 王祥宇[1*], 徐国文[2], 梁岩荣[3], 田创[1], 李兴华[1,4], 马建峰[1,4]

1. 西安电子科技大学网络与信息安全学院, 西安 710126
2. 电子科技大学计算机科学与工程学院, 成都 611731
3. 西北大学计算机学院, 西安 710127
4. 空天地一体化综合业务网全国重点实验室, 西安 710071

* 通信作者. E-mail: wangxiangyu01@xidian.edu.cn

国家自然科学基金项目 (批准号:62572368、62502075、U23A20303、62232013、62125205)、国家自然科学基金创新研究群体项目 (批准号:62121001)、中国博士后科学基金项目 (批准号:2024M752534) 和中央高校基本科研业务费专项资金资助项目 (批准号:YJSJ26003)

摘要 作为数据要素流通中的重要载体, 数据胶囊通过封装数据主体及附属策略, 实现了数据全生命周期的临时性访问控制. 在脱离发布方的管控边界后, 胶囊被迁移至云服务集群, 并按照既定策略向使用方提供受控访问. 当胶囊的管理权与所有权相分离时, 传统临时性访问控制方法不仅将执行策略的临时区间显式暴露于密文结构中, 致使区间的范围结构泄露, 且难以防止授权方在特定时隙内蓄意出卖解密能力的行为. 针对以上问题, 本文提出一种兼具胶囊自毁和叛逆者追踪特性的临时性访问控制方案. 该方案将临时区间映射为互斥节点的编码集合以准确判定特定时隙是否处于该区间, 实现数据胶囊的自毁机制. 此外, 该方案通过合理性检验、全局密钥追踪和胶囊更新流程, 支持密钥泄露责任主体的准确定位与权限撤销. 基于 l 型 co-Strong Diffie-Hellman (l-co-SDH) 假设和判定性线性 (DLIN) 假设, 该方案可分别达到可追踪性模型下的不可伪造性和选择明文攻击模型下的语义安全性. 基于 Java 配对密码学库 (JPBC) 及真实数据集的评估结果显示, 该方案在数据胶囊流通中具备良好的可扩展性和实用性.

关键词 数据胶囊, 叛逆者追踪, 自毁机制, 临时性访问控制, 数据要素流通

1 背景介绍

在高度互联的数字经济时代, 数据要素流通推进分散异构的数据资源在不同主体之间有序流转, 不仅充分释放其蕴含的潜在价值, 更显著提升了整体利用效率[1,2]. 然而, 日益严峻的隐私泄露与权限滥用始终是制约数据要素安全流通的热点问题[3]: 一方面, 针对密文的访问控制难以彻底阻断越权访问行为[4]; 另一方面, 针对密钥的恶意泄露使非授权方非法获取解密能力[5]. 出于对权限管理和密钥安全的重视, 众多组织相继通过立法手段强化数据访问与使用行为的规范. 例如, 欧盟的《通用数据保护条例》(General Data Protection Regulation, GDPR) [6] 自 2018 年实施以来, 凭借其严格的合规标准成为数据治理的立法标杆. 其中, “存储限制原则”要求数据使用方仅可在满足合法目的所需期限内存储数据, 而“控制者责任”则明确应采取技术与组织措施防止数据非授权访问与意外损毁. 近期, 中国国家数据局出台的《可信数据空间发展行动计划 (2024—2028 年)》[7]

引用格式: 洪克用, 王祥宇, 徐国文, 等. 面向自毁数据胶囊的可追踪访问控制方法. 中国科学: 信息科学, 在审文章
Keyong Hong, Xiangyu Wang, Guowen Xu, et al. Traceable access control method for self-destructive data capsules. Sci Sin Inform, for review

Accepted

https://www.sciengine.com/doi/10.1360/SSI-2026-0005

指出, 需建立健全可信数据空间合规管理指引, 清晰界定数据发布方、使用方与平台方的权责边界. 在国内外政策的推动下, 数据胶囊作为一种安全可控的数据流通单元, 已引发广泛关注 [3]. 数据胶囊通过将数据主体与处理逻辑、溯源信息以及生命周期等附属策略统一封装, 构建自包含的临时隔离容器以持续保护敏感对象 [8,9], 并强调依照约束规则可控使用, 为缓解越权访问与密钥泄露提供了内生约束的天然途径.

在适用于数据胶囊的密态访问控制中, 基于临时性访问控制的自毁机制允许数据发布方定义时间限制的执行策略, 从而以不可恢复的方式自动中止使用方权限, 由此防止敏感数据持续暴露 [10]. 该思想可追溯至 Bertino 等的早期研究 [11], 通常设置原始密钥不再适配密文内嵌的临时区间, 强调数据主体的自主与可控. 事实上, 传统临时性访问控制方法可细化为两类：第一类基于时间特定加密 (Time-Specific Encryption, TSE) [10,12~16], 将临时区间显式暴露于密文结构中, 不仅削弱策略本身的隐私性, 还致使区间的范围结构面临泄露风险. 第二类方法则依托合数阶群构造可比较属性加密 (Comparable Attribute-Based Encryption, Comparable ABE) 方案 [17~21]. 然而, 合数阶群的固有特性使此类方法在同等安全等级下难以兼顾计算效率与存储开销. 因此, **如何通过密态方式隐藏临时区间并准确判定特定时隙在区间中的归属**仍是数据胶囊中亟待解决的关键挑战.

在密钥安全方面, 目前的 TSE 构造仅在授权逻辑上针对使用方是否具备某一时间区间内的临时性权限进行判定, 而在密钥恶意泄露的追踪与问责方面尚显匮乏. 因此, 传统临时性访问控制无法有效遏制非授权方滥用临时性权限的行为. 而传统可追踪方法的研究更侧重于约束潜在叛逆者对属性密钥的蓄意泄露与权限撤销 [22~24], 缺乏对时间维度解密能力泄露的责任溯源, 因此难以界定非授权方在特定时隙内非法获取密钥的行为. 另外, 传统访问控制普遍设定云服务集群遵循“诚实且好奇”模型 [25], 并默认数据使用方不与云合谋. 在数据胶囊的可信流通场景中, 一旦叛逆者与云在胶囊管理权与所有权相分离的情况下形成合谋威胁即可轻易泄露外包密钥. 敌手无需遵循策略便可独立解封数据胶囊, 从而规避访问权限与临时性权限的双重判定. 因此, **如何同时实施针对时隙密钥和属性密钥的全局追踪并及时撤销叛逆者权限**成为当前研究的另一技术问题.

综上, 面向自毁数据胶囊的可信流通场景, 以下问题亟待解决:

(1) 如何密态隐藏执行策略内嵌的临时区间并判定当前密钥中的特定时隙属于该区间? (2) 如何全局定位蓄意泄露时隙密钥和属性密钥的潜在叛逆者并撤销其权限?

1.1 本文贡献

针对上述问题, 本文提出了面向自毁数据胶囊的可追踪访问控制方法架构, 并由此构建支持编码结构优化的时间特定 ABE 构造, 主要贡献如下：

(1) 通过完全二叉树与二进制编码结构, 将封装过程中设定的临时区间映射为互斥节点集合, 并在密钥生成过程中将时隙路径内嵌至时隙密钥与客户端密钥, 实现对时隙归属的密态判定. 若该时隙不属于预设区间, 则临时性权限判定条件无法满足, 由此实现数据胶囊的逻辑自毁, 同时避免临时区间范围结构的显式暴露;

(2) 提出针对全局密钥的合理性检验和叛逆者追踪方法, 将域约束、良构性验证和身份搜索应用于全局密钥中, 支持责任主体的准确定位与权限撤销并遏制非授权方滥用解密权限的行为, 有效解决叛逆者与云服务集群泄露解密能力的问题;

(3) 基于 l-co-SDH 和 DLIN 困难假设, 证明所提方案在可追踪性模型下具备不可伪造性, 并在选择明文攻击模型下保证语义安全性. JPBC 库 [26] 中 g149 和 d224 曲线的评估结果显示, 当属性数量与时隙树深度分别达到 100 与 10 时, 本方案的封装性能相较现有方案 HPL+22 [23] 和 YLZ+25 [27] 分别提升 2.017× ∼ 2.203× 和 2.249× ∼ 2.471×, 密文存储则分别降低 2.943× 和 3.122×. 此外, 其封装与解封装的时间开销在多个真实数据集中均维持秒级水平.

Accepted

https://www.sciengine.com/doi/10.1360/SSI-2026-0005

表 1　相关工作的特性对比.

Table 1　Feature comparison of related works.

Features	[10]	[12]	[16]	[19]	[20]	[21]	[23]	[27]	[29]	[30]	[32]	Ours
Temporal access control	✓	✓	✓	✓	✓	✓	×	×	×	×	×	✓
Fine-grained access control	✓	×	✓	✓	✓	×	✓	✓	✓	✓	✓	✓
Tracing for attribute keys	×	×	×	×	×	×	✓	✓	✓	✓	✓	✓
Tracing for time-slot keys	×	×	×	×	×	×	×	×	×	×	×	✓
Privilege revocation for traitors	×	×	×	×	×	×	✓	✓	×	×	✓	✓
Privacy of time ranges	×	×	×	✓	✓	×	×	×	×	×	×	✓
Access policy hidden	×	×	×	×	×	×	✓	×	✓	×	×	✓
Group structure	*PR*	*PR*	*PR*	*CO*	*CO*	*CO*	*PR*	*PR*	*CO*	*PR*	*CO*	*PR*
Security model	*CPA*	*CPA*	*CPA*	*CKA*	*CPA*	*KCA*	*CPA*	*CPA*	*CPA*	*R/W*	*CPA*	*CPA*
Access structure	*AT*	—	*LS*	*AT*	*AT*	—	*LS*	*LS*	*LS*	*LS*	*LS*	*LS*

† ***Remark.*** *PR*: Prime-order group; *CO*: Composite-order group; *CPA*: Chosen plaintext attack; *CKA*: Chosen keyword attack; *KCA*: Keyword collusion attack; *R/W*: No-read/No-write rule; *AT*: Tree-based access structure; *LS*: Linear secret sharing scheme.

1.2　相关工作

本节聚焦于**临时性访问控制**与**叛逆者追踪**技术的相关工作. 特性对比详见表 1.

临时性访问控制. 早在 2002 年, Bertino 等 [11] 就提出了自主访问控制方法, 允许授权仅在特定临时区间内有效, 并可通过预定义规则设定权限依赖关系. 受时间因素的启发, 具备动态约束能力的临时性访问控制方法相继出现. Paterson 与 Quaglia [12] 提出了将解密能力与时间约束绑定的时间特定加密原语. 当时隙属于密文中预定义的临时区间内时, 接收方才具备数据的访问权限. 在此基础上, 若干时间可控的公钥加密 [13,14] 和公钥签名 [15] 方案被提出, 但受限于表达能力, 它们在访问控制粒度方面仍较为粗略, 难以达到大规模用户场景下访问权限的细粒度管控.

为解决粗粒度访问控制的局限性, Xiong 等 [10] 提出了支持时间特定属性的密钥策略 ABE 方案, 使所有属性均在预定区间结束后自毁. 类似地, Ma 等 [16] 所提出的细粒度访问控制方法将临时区间和属性集合嵌入密文中. 只有当属性集合满足访问策略且时隙属于临时区间时, 密文才可成功解密. 然而, 文献 [10] 与文献 [16] 将临时区间直接作为组件暴露于密文中, 使得敌手可轻易观测到临时区间的范围结构, 因此显著增加了隐私泄露风险. 为了授予用户对资源的临时访问, 若干可比较访问控制方法将时间因素作为可比较的子属性引入. Zhu 等 [17] 将时间因素引入整数区间比较, 并结合代理重加密实现云计算中策略灵活的时隙可控访问. Wang 等 [18] 则将可比较区间嵌入密钥中, 并将所有属性集成至单一密文, 具备常数规模的密文结构. 受文献 [17] 与文献 [18] 的启发, Miao 等 [19] 实现了一种基于时间可比较的关键词搜索方案, 支持细粒度的临时管理和动态用户撤销. Tong 等 [20] 提出分别具有线性与亚线性搜索复杂度的两种布尔区间可比较可搜索方案, 缓解了粗粒度且非顺序性的访问控制问题. Liu 等 [21] 在双服务器架构下提出时间可控的分层多关键词搜索方案, 可有效抵御关键词猜测攻击. 尽管如此, 前述方案均基于合数阶群, 而由于其自身结构特性, 在相同安全等级下, 前述方案的实际效率难以达到素数阶群访问控制方案的水准.

叛逆者追踪. 为了解决授权滥用的问题, 叛逆者追踪技术通过在密钥中嵌入可验证追踪信息精确定位密钥泄露责任主体, 可实现针对性访问隔离. Liu 等 [22] 构造了支持任意单调访问结构的合数阶可追踪 ABE 方案. 在兼顾表达能力、安全性与效率的前提下, 该方案突破了传统叛逆者追踪技术策略表达受限的瓶颈. 受文献 [22] 的启发, Ning 等 [24] 提出了白盒可追踪 ABE 方案. 该方案依赖素数阶群且具备恒定追踪存储开销. Zhang 等 [28] 构建了双授权机构协同 ABE 方案, 通过分布式密钥生成机制解决了密钥托管问题, 同时实现对密钥滥用行为的公开追踪. 针对医疗物联网场景, Zeng 等 [29] 提出了公开可追踪的大属性总体 ABE 方案, 可

Accepted

https://www.sciengine.com/doi/10.1360/SSI-2026-0005

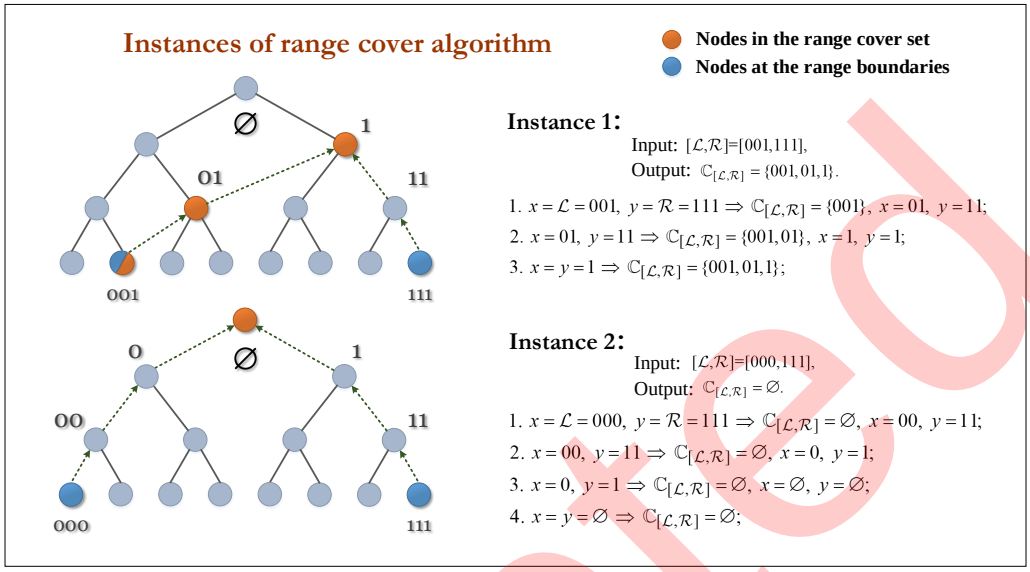


图 1　(网络版彩图) 区间覆盖算法实例.

Figure 1　(Color Online) Instances of range cover algorithm.

在保护策略隐藏的同时实施对恶意用户的高效追踪. 近期, 张等 [30] 提出的跨域访问控制加密方案利用 Shamir 门限秘密共享生成可嵌入密钥的身份份额, 故无需维护撤销列表即可精确追踪叛逆者. 尽管前述工作可密态判定密钥权属, 却难以在追踪后阻止叛逆者使用既有密钥, 因而无法实现叛逆者权限撤销.

为解决上述问题, Han 等 [23] 利用树型撤销结构设计了具备半结构策略隐藏、用户撤销与白盒追踪的 ABE 方案, 在保证策略隐私的同时实现对叛逆者的精确定位与权限动态管理. 然而, 用户在解密中必须完成属性集合与访问策略的匹配运算, 使本地计算负担随属性数量增加而显著上升. 通过安全聚合函数与双阶段解密设计, Yin 等 [27] 所设计的 ABE 方案将叛逆者写入撤销列表实现选择性密文更新, 保证了撤销中合法用户的隐私. 该方案将身份总体在初始化中静态限定并将撤销身份分别嵌入密钥和密文中, 故难以支持新增用户的动态管理. 近期, Meng 等 [31] 提出了服务器辅助的可追踪可撤销 ABE 构造, 不仅修复了传统方案中选择关键词攻击的安全缺陷, 还兼具恒定密钥与陷门规模以及快速追踪的特性. 基于前述工作, Meng 等 [32] 提出基于静态假设的新型 ABE 方案, 利用服务器辅助兼容撤销与追踪功能, 同时在密钥规模方面具备实用性. 尽管如此, 文献 [31] 与文献 [32] 仅侧重本地密钥的追踪, 未考虑叛逆者与服务器合谋泄露解密能力的行为.

2　预备知识

定义 1 (区间覆盖算法 [12])　该算法以区间 $[\mathcal{L},\mathcal{R}]$ 为输入, 通过端点奇偶性判定是否将其添加至覆盖集合 $\mathbb{C}_{[\mathcal{L},\mathcal{R}]}$: 若左端点为偶数, 则上移至父节点, 否则将其添加至 $\mathbb{C}_{[\mathcal{L},\mathcal{R}]}$ 并移至父节点的右兄弟节点; 若右端点为偶数, 则将其添加至 $\mathbb{C}_{[\mathcal{L},\mathcal{R}]}$ 并移至父节点的左兄弟节点, 否则向上移至父节点. 通过动态添加边界并迭代上移直至端点相遇, 该算法将 $[\mathcal{L},\mathcal{R}]$ 分解为完全二叉树中的区间覆盖集合 $\mathbb{C}_{[\mathcal{L},\mathcal{R}]}$. 各覆盖节点对应子区间互斥, 其并集构成完整区间 $[\mathcal{L},\mathcal{R}]$. 具体示例与流程详见图 1.

定义 2 (l-co-SDH 假设 [33])　选取素数 p 阶源群 $\mathbb{G}_1$ 和 $\mathbb{G}_2$ 以及目标群 $\mathbb{G}_T$, 由此设定非对称双线性映射 $e:\mathbb{G}_1\times\mathbb{G}_2\to\mathbb{G}_T$. g_1、g_2 和 x 分别从 $\mathbb{G}_1$、$\mathbb{G}_2$ 和 $\mathbb{Z}_p$ 中随机选取. 给定元组 $\mathcal{D}$ 如下:

$$\mathcal{D}=\left(g_1,g_1^x,g_1^{x^2},\ldots,g_1^{x^l},g_2,g_2^x,g_2^{x^2},\ldots,g_2^{x^l}\right).$$

若任意概率多项式时间 (Probabilistic Polynomial-Time, P.P.T.) 伪造者 $\mathcal{F}$ 无法以不可忽略的优势计算 c 与

Accepted

https://www.sciengine.com/doi/10.1360/SSI-2026-0005

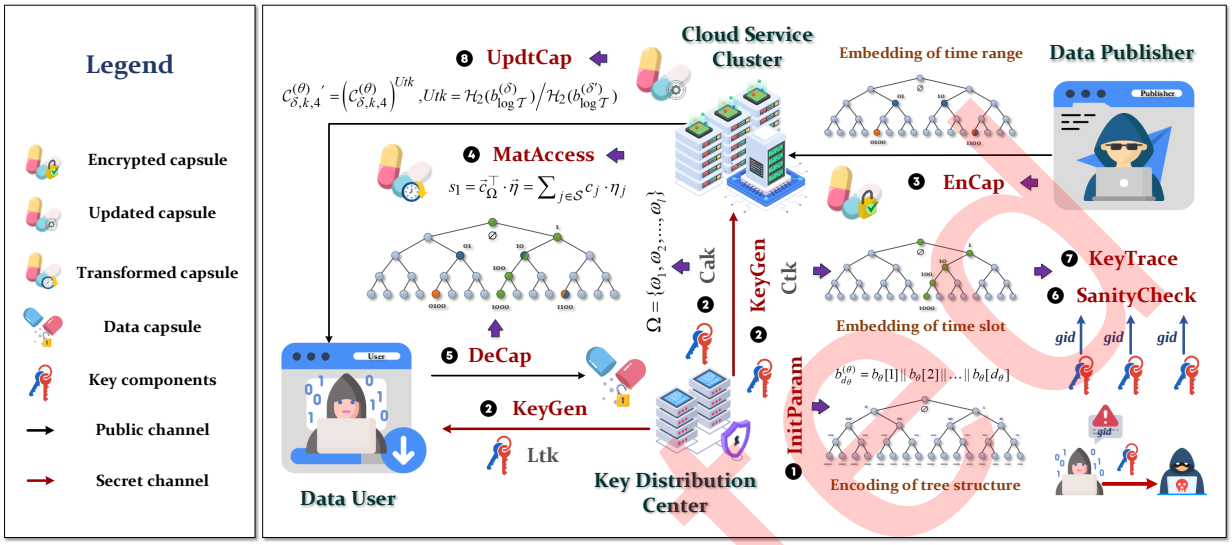


图 2　(网络版彩图) 可追踪访问控制方法架构.

Figure 2　(Color Online) Architecture of the traceable access control method.

$g_1^{\frac{1}{c+x}}$, 则称 l-co-SDH 假设成立. 伪造者 $\mathcal{F}$ 的优势定义为 $Adv_{\mathcal{F}}^{l\text{-}co\text{-}SDH} = \Pr\left[\mathcal{F}(\mathcal{D}) = (c, g_1^{\frac{1}{c+x}})\right]$.

定义 3 (DLIN 假设 [34])　选取素数 p 阶源群 $\mathbb{G}_1$ 和 $\mathbb{G}_2$ 以及目标群 $\mathbb{G}_T$, 设定非对称双线性映射 $e: \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. 在选取 $g_1 \in \mathbb{G}_1$、$g_2 \in \mathbb{G}_2$ 和 $x_1, x_2, y_1, y_2, z \in \mathbb{Z}_p$ 后, 给定元组 $\mathcal{D}$ 如下:

$$\mathcal{D} = \left(g_1, g_1^{x_1}, g_1^{x_2}, g_1^{x_1 y_1}, g_1^{x_2 y_2}, g_2, g_2^{x_1}, g_2^{x_2}, g_2^{x_1 y_1}, g_2^{x_2 y_2}\right).$$

若任意 P.P.T. 敌手 $\mathcal{A}$ 无法以不可忽略的优势区分 $\mathcal{T}_0 = \left(g_1^{y_1+y_2}, g_2^{y_1+y_2}\right)$ 与 $\mathcal{T}_1 = \left(g_1^z, g_2^z\right)$, 则称判定性线性假设 (Decisional Linear Assumption, DLIN) 成立. 敌手 $\mathcal{A}$ 的优势定义为 $Adv_{\mathcal{A}}^{DLIN} = |\Pr[\mathcal{A}(\mathcal{T}_0, \mathcal{D}) = 1] - \Pr[\mathcal{A}(\mathcal{T}_1, \mathcal{D}) = 1]|$.

3　面向自毁数据胶囊的可追踪访问控制系统模型与安全模型

本章阐述面向自毁数据胶囊的可追踪访问控制方法的系统模型、形式化方案框架, 以及可追踪性模型和语义安全模型.

3.1　系统模型

面向自毁数据胶囊的可追踪访问控制方法架构如图 2 所示, 包含四个实体：密钥分发中心 (Key Distribution Center)、云服务集群 (Cloud Service Cluster)、数据发布方 (Data Publisher) 与数据使用方 (Data User).

作为可信实体, 密钥分发中心负责系统初始化 (步骤 1, ***InitParam***), 为大规模数据使用方生成属性密钥 Cak、时隙密钥 Ctk 和客户端密钥 Ltk (步骤 2, ***KeyGen***). 数据发布方通过预定义的访问策略和临时区间封装数据胶囊 (步骤 3, ***EnCap***) 并传输至云服务集群. 云服务集群设定为半可信实体且可能与使用方合谋泄露密钥, 负责为数据使用方提供细粒度访问权限判定并生成转换胶囊 (步骤 4, ***MatAccess***). 数据使用方在获取转换胶囊后, 由 Ltk 判定临时性访问权限并解封胶囊 (步骤 5, ***DeCap***). 此外, 密钥分发中心通过密钥合理性检验 (步骤 6, ***SanityCheck***) 和身份标识搜索 (步骤 7, ***KeyTrace***), 对出卖解密能力的恶意行为实施溯源, 并更新胶囊以撤销其访问权限 (步骤 8, ***UpdtCap***). 该方案的形式化框架如图 3 所示.

Accepted

https://www.sciengine.com/doi/10.1360/SSI-2026-0005

Formal Framework

(1) ***InitParam*** $(\mathcal{T}, \mathcal{U}, 1^{\kappa}) \rightarrow (\text{mpk}, \text{msk}, \Pi_{\mathcal{T}}, \mathcal{KL})^{a)}$. This algorithm takes as input epoch parameter $\mathcal{T}$, attribute universe $\mathcal{U}$, and security parameter κ, then outputs public parameter mpk, master secret key msk, encoding state $\Pi_{\mathcal{T}}$, and an empty key list $\mathcal{KL}$. It is invoked by Key Distribution Center.

(2) ***KeyGen*** $(\text{msk}, gid, \Omega = \{u_i : v_i\}_{i\in[1,l]}, \delta \in (0, \mathcal{T}]) \rightarrow (K, \text{Cak}, \text{Ctk}, \text{Ltk}, \mathcal{KL})$. This algorithm takes as input msk, current time slot δ, global identity gid, and attribute set Ω, then outputs tracing component K, attribute key Cak, time-slot key Ctk, client key Ltk, and updated $\mathcal{KL}$. It is invoked by Key Distribution Center.

(3) ***EnCap*** $(\mathcal{M}, \mathbb{P} = (\mathbb{M}, \rho(\cdot), \{u_{\rho(j)} : v_{\rho(j)}\}_{j\in[1,m]}), [\mathcal{L}, \mathcal{R}], \delta \in (0, \mathcal{T}]) \rightarrow EC$. This algorithm takes as input message $\mathcal{M}$, access policy $\mathbb{P}$, time range $[\mathcal{L}, \mathcal{R}]$, and current δ, then outputs encrypted capsule EC. It is invoked by Data Publisher.

(4) ***MatAccess*** $(EC, \text{Cak}, \text{Ctk}) \rightarrow \mathcal{TC}/\perp$. This algorithm takes as input EC, Cak, and Ctk. If Ω embedded in Cak satisfies $\mathbb{P}$, it outputs transformed capsule $\mathcal{TC}$. Otherwise, it outputs $\perp$. It is invoked by Cloud Service Cluster.

(5) ***DeCap*** $(\mathcal{TC}, \text{Ltk}) \rightarrow \mathcal{M}/\perp$. This algorithm takes as input $\mathcal{TC}$ and Ltk. If δ embedded in Ltk belongs to $[\mathcal{L}, \mathcal{R}]$, it outputs $\mathcal{M}$; otherwise, it outputs $\perp$. It is invoked by Data User.

(6) ***SanityCheck*** $(\text{msk}, K, \text{Cak}, \text{Ctk}, \text{Ltk}) \rightarrow 1/0$. This algorithm takes as input msk, K, Cak, Ctk, and Ltk. It outputs 1 if all the above keys pass domain-constraint and well-formedness checks, then outputs 0 otherwise. It is invoked by Key Distribution Center.

(7) ***KeyTrace*** $(k_1, \mathcal{KL}) \rightarrow gid/\perp$. This algorithm takes as input tracing component k_1 and $\mathcal{KL}$. If k_1 exists, it outputs traitor identity gid; otherwise, it outputs $\perp$. It is invoked by Key Distribution Center.

(8) ***UpdtCap*** $(EC, \{\delta, \delta'\} \in (0, \mathcal{T}]) \rightarrow EC'$. This algorithm takes as input EC, original δ, and current δ', then outputs updated EC'. It is invoked by Cloud Service Cluster.

a) *Remark.* The public parameter mpk is taken as the default input of subsequent algorithms.

图 3　(网络版彩图) 可追踪访问控制方案形式化框架.

Figure 3　(Color Online) Formal framework of the traceable access control scheme.

3.2　安全模型

3.2.1　可追踪性

在安全参数 κ 下, P.P.T. 伪造者 $\mathcal{F}$ 与挑战者 $\mathcal{C}$ 之间的可追踪性模型如下定义：

(1) 初始化阶段. $\mathcal{C}$ 运行 ***InitParam*** $(\mathcal{T}, \mathcal{U}, 1^{\kappa})$ 完成系统初始化. 它保留 msk、$\Pi_{\mathcal{T}}$ 和 $\mathcal{KL}$ 并将 mpk 发送至 $\mathcal{F}$.

(2) 密钥询问阶段. $\mathcal{C}$ 维护密钥列表 $\mathcal{KL}$, 并向 $\mathcal{F}$ 提供 q 次对 $(K, \text{Cak}, \text{Ctk}, \text{Ltk})$ 的询问：当 $\mathcal{F}$ 针对 $\{gid_i, \Omega_i, \delta_i\}_{i\leqslant q}$ 请求 Cak、Ctk 和 Ltk 时, $\mathcal{C}$ 首先查询 $\mathcal{KL}$. 若 $(gid_i, \Omega_i, \delta_i, K, \text{Cak}, \text{Ctk}, \text{Ltk})$ 已存在, $\mathcal{C}$ 直接返回 K、Cak、Ctk 和 Ltk. 否则, $\mathcal{C}$ 通过 ***KeyGen*** $(\text{msk}, \{gid_i, \Omega_i, \delta_i\}_{i\leqslant q})$ 计算 $(K, \text{Cak}, \text{Ctk}, \text{Ltk})$, 最后返回元组 $(K, \text{Cak}, \text{Ctk}, \text{Ltk})$ 并写入 $\mathcal{KL}$.

(3) 密钥伪造阶段. $\mathcal{F}$ 输出伪造密钥元组 $(K^*, \text{Cak}^*, \text{Ctk}^*, \text{Ltk}^*)$. 若 ***KeyTrace*** $(k_1, \mathcal{KL}) \neq \perp$ 且 ***KeyTrace*** $(k_1, \mathcal{KL}) \notin \{gid_1, \ldots, gid_q\}$, 则 $\mathcal{F}$ 赢得可追踪性游戏. $\mathcal{F}$ 的优势定义为 $Adv_{\mathcal{F}} = \Pr\left[\textbf{\textit{KeyTrace}}(k_1, \mathcal{KL}) \notin \{gid_1, \ldots, gid_q, \perp\}\right]$.

定义 4 (可追踪性)　若不存在 P.P.T. 伪造者 $\mathcal{F}$ 以不可忽略的优势 $Adv_{\mathcal{F}}$ 赢得前述游戏, 则称面向自毁数据胶囊的可追踪访问控制方法具备可追踪性.

3.2.2　语义安全性

在安全参数 κ 下, P.P.T. 敌手 $\mathcal{A}$ 与挑战者 $\mathcal{C}$ 之间的语义安全模型如下定义：

(1) 初始化阶段. $\mathcal{A}$ 向 $\mathcal{C}$ 声明挑战临时区间 $[\mathcal{L}^*, \mathcal{R}^*] \subseteq (0, \mathcal{T}]$ 和挑战访问策略 $\mathbb{P}^* = (\mathbb{M}^*, \rho^*(\cdot), \{u^*_{\rho^*(j)} : v^*_{\rho^*(j)}\}_{j\in[1,m^*]})$. 其中, $\mathbb{M}^*$ 为 m^* 行 n^* 列的份额矩阵, $\rho^*(\cdot)$ 则将 $\mathbb{M}^*$ 的行集映射至 $\mathcal{U}$ 中的属性名 $u^*_{\rho^*(j)}$, 而 $v^*_{\rho^*(j)}$ 表示关联属性值. 随后, $\mathcal{C}$ 执行 ***InitParam*** $(\mathcal{T}, \mathcal{U}, 1^{\kappa})$ 完成初始化. 它将 mpk 发送至 $\mathcal{A}$ 并保留 msk、

Accepted

https://www.sciengine.com/doi/10.1360/SSI-2026-0005

$\Pi_{\mathcal{T}}$ 和 $\mathcal{KL}$.

(2) 询问阶段 1. $\mathcal{C}$ 维护密钥列表 $\mathcal{KL}$, 并通过下述方式向 $\mathcal{A}$ 提供 Cak、Ctk 与 Ltk：当 $\mathcal{A}$ 针对全局身份 gid、时隙 δ 和属性集合 Ω 请求 Cak、Ctk 和 Ltk 时, $\mathcal{C}$ 首先查询 $\mathcal{KL}$. 其中, $\Omega \not\models \mathbb{P}^*$ 且 $\delta \notin [\mathcal{L}^*, \mathcal{R}^*]$. 若 $(gid, \Omega, \delta, K, \text{Cak}, \text{Ctk}, \text{Ltk})$ 已存在, $\mathcal{C}$ 直接返回 Cak、Ctk 和 Ltk. 否则, $\mathcal{C}$ 通过 ***KeyGen*** $(\text{msk}, gid, \Omega, \delta)$ 计算 $(K, \text{Cak}, \text{Ctk}, \text{Ltk})$, 最后将元组 $(K, \text{Cak}, \text{Ctk}, \text{Ltk})$ 发送至 $\mathcal{A}$.

(3) 挑战阶段. $\mathcal{A}$ 向 $\mathcal{C}$ 发送两个等长消息 $\mathcal{M}_0^*$ 与 $\mathcal{M}_1^*$.

$\mathcal{C}$ 随机选取 $\psi \in \{0,1\}$, 并使用挑战访问策略与临时区间封装 $\mathcal{M}_\psi^*$. 它选择 $\delta \in [\mathcal{L}^*, \mathcal{R}^*]$ 并调用 ***EnCap*** $(\mathcal{M}_\psi^*, \mathbb{P}^*, [\mathcal{L}^*, \mathcal{R}^*], \delta)$, 由此计算挑战胶囊 EC_ψ^*. 最后, $\mathcal{C}$ 将 EC_ψ^* 发送给 $\mathcal{A}$.

(4) 询问阶段 2. $\mathcal{A}$ 继续执行与阶段 1 中的询问, 但具有限制 $\Omega \not\models \mathbb{P}^*$ 且 $\delta \notin [\mathcal{L}^*, \mathcal{R}^*]$.

(5) 猜测. $\mathcal{A}$ 输出对随机比特 ψ 的猜测 ψ'. $\mathcal{A}$ 获胜的优势定义为 $Adv_{\mathcal{A}} = |\Pr[\psi = \psi'] - \frac{1}{2}|$.

定义 5 (**语义安全性**) 若任意 P.P.T. 敌手 $\mathcal{A}$ 在前述语义安全性游戏中的优势 $Adv_{\mathcal{A}}$ 至多可忽略, 则面向自毁数据胶囊的可追踪访问控制方法可达到选择明文攻击下的语义安全性.

4 面向自毁数据胶囊的可追踪访问控制方案构造

本章阐述面向自毁数据胶囊的可追踪访问控制方案的具体构造.

• ***InitParam*** $(\mathcal{T}, \mathcal{U}, 1^\kappa) \to (\text{mpk}, \text{msk}, \Pi_{\mathcal{T}}, \mathcal{KL})$. 该算法首先定义三个素数 p 阶乘法循环群 $\mathbb{G}_1$、$\mathbb{G}_2$、$\mathbb{G}_T$, 以及非对称双线性映射 $e: \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. 随后, 该算法随机选取 $g_1, g_3 \in \mathbb{G}_1$、$g_2 \in \mathbb{G}_2$ 和 $\alpha, a \in \mathbb{Z}_p$, 并定义单向抗碰撞哈希函数 $\mathcal{H}_1: \{0,1\}^* \to \mathbb{G}_1$ 和 $\mathcal{H}_2: \{0,1\}^* \to \mathbb{Z}_p$. 根据纪元参数 $\mathcal{T}$, 它初始化一棵完全二叉树 $\Pi_{\mathcal{T}}$ 并按照下述方式执行编码：**1)** 对于非根节点 θ, 迭代定义编码 $b_{d_\theta} = b_\theta[1] \| b_\theta[2] \| \ldots \| b_\theta[d_\theta]$, 其中, $d_\theta \in [1, \log \mathcal{T}]$ 为节点 θ 的深度; **2)** 对于根节点 θ, 定义其编码为 $b_{d_\theta} = \varnothing$. 该算法依照树深 $\log \mathcal{T}$ 均匀选择一组标量 $\beta, \beta_1^{(0)}, \beta_1^{(1)}, \ldots, \beta_{\log \mathcal{T}}^{(0)}, \beta_{\log \mathcal{T}}^{(1)} \in \mathbb{G}_1^{2 \log \mathcal{T}+1}$. 最后, 密钥分发中心输出公开参数 $\text{mpk} = \{\beta, \beta_1^{(0)}, \beta_1^{(1)}, \ldots, \beta_{\log \mathcal{T}}^{(0)}, \beta_{\log \mathcal{T}}^{(1)}, g_1, g_2, g_3, g_2^a, e(g_1, g_2)^\alpha, \mathcal{H}_1(\cdot), \mathcal{H}_2(\cdot), e\}$、保留主私钥 $\text{msk} = \{\alpha, a\}$ 和编码状态 $\Pi_{\mathcal{T}}$, 并且初始化密钥列表 $\mathcal{KL}$.

• ***KeyGen*** $(\text{msk}, gid, \Omega = \{u_i : v_i\}_{i \in [1,l]}, \delta \in (0, \mathcal{T}]) \to (K, \text{Cak}, \text{Ctk}, \text{Ltk}, \mathcal{KL})$. 该算法首先从 $\mathbb{Z}_p$ 中随机选取 γ, τ_{id}. 接着, 该算法设置追踪组件 $K = (k_1 = \gamma, k_2 = g_2^{\tau_{id}}, k_3 = (g_2^a)^{\tau_{id}})$ 并如下设定属性密钥 $\text{Cak} = (\{u_i\}_{i \in [1,l]}, ak_1, \{ak_{i,2}\}_{i \in [1,l]})$. 其中, $ak_1 = g_1^{\frac{\alpha - \tau_{id}}{a+\gamma}} \cdot g_3^{\tau_{id}}$, $ak_{i,2} = \mathcal{H}_1(u_i || v_i)^{\tau_{id}}$. 该算法解析当前时隙 $\delta \in (0, \mathcal{T}]$ 编码 $b_{\log \mathcal{T}}^{(\delta)} = b_\delta[1] \| b_\delta[2] \| \ldots \| b_\delta[\log \mathcal{T}]$ 并计算 $\mathcal{H}_2(b_{\log \mathcal{T}}^{(\delta)})$. 对于 $\vartheta \in Path(\delta)$, 它如下设置时隙密钥 $\text{Ctk} = tk_{\delta,k}^{(\vartheta)}$ 和客户端密钥 $\text{Ltk} = (lk_1, lk_{\delta,k,2}^{(\vartheta)})$：

$$tk_{\delta,k}^{(\vartheta)} = g_1^{a \cdot \tau_{id}} \cdot \left(\beta \cdot \prod_{k=1}^{d_\vartheta} \beta_\vartheta^{(b_\delta[k])} \right)^{\frac{\tau_{id}}{\mathcal{H}_2(b_{\log \mathcal{T}}^{(\delta)})}}, \quad lk_1 = g_1^{a \cdot \tau_{id}}, \quad lk_{\delta,k,2}^{(\vartheta)} = g_1^{\frac{\tau_{id}}{a+\gamma}} \left(\beta \cdot \prod_{k=1}^{d_\vartheta} \beta_\vartheta^{(b_\delta[k])} \right)^{\frac{\tau_{id}}{\mathcal{H}_2(b_{\log \mathcal{T}}^{(\delta)})}}.$$

其中, d_ϑ 表示路径节点 ϑ 的深度. 最后, 密钥分发中心将元组 $(gid, K, \text{Cak}, \text{Ctk}, \text{Ltk})$ 记录至 $\mathcal{KL}$ 中, 并通过安全信道将 (Ltk, K) 传输至数据使用方、将 $(\text{Cak}, \text{Ctk}, K)$ 传输至云服务集群.

• ***EnCap*** $(\mathcal{M}, \mathbb{P} = (\mathbb{M}, \rho(\cdot), \{u_{\rho(j)} : v_{\rho(j)}\}_{j \in [1,m]}), [\mathcal{L}, \mathcal{R}], \delta \in (0, \mathcal{T}]) \to EC$. 该算法首先选择秘密向量 $\vec{S} = (s_1, \sigma_2, \ldots, \sigma_n)^\top \in \mathbb{Z}_p^{n \times 1}$ 和标量 $s_2 \in \mathbb{Z}_p$ 并依据预定义布尔公式指定 m 行 n 列的份额矩阵 $\mathbb{M}$. 随后, 它将 $\mathbb{M} \times \vec{S} = (\eta_1, \eta_2, \ldots, \eta_m)^\top \in \mathbb{Z}_p^{m \times 1}$ 作为份额向量 $\vec{\eta}$, 据此生成表达性访问策略 $\mathbb{P} = (\mathbb{M}, \rho(\cdot), \{u_{\rho(j)} : v_{\rho(j)}\}_{j \in [1,m]})$. 接着, 该算法调用区间覆盖算法获得 $[\mathcal{L}, \mathcal{R}]$ 的覆盖集合 $\mathbb{C}_{[\mathcal{L},\mathcal{R}]}$, 并逐一解析 $\tau_\theta \in \mathbb{C}_{[\mathcal{L},\mathcal{R}]}$ 编码 $b_{\log \mathcal{T}}^{(\mathbb{C})} = b_{\tau_\theta}[1] \| b_{\tau_\theta}[2] \| \ldots \| b_{\tau_\theta}[\log \mathcal{T}]$ 和当前时隙 δ 编码 $b_{\log \mathcal{T}}^{(\delta)} = b_\delta[1] \| b_\delta[2] \| \ldots \| b_\delta[\log \mathcal{T}]$. 对于 $j \in [1, m]$ 和

Accepted

https://www.sciengine.com/doi/10.1360/SSI-2026-0005

$\tau_\theta \in \mathbb{C}_{[\mathcal{L},\mathcal{R}]}$, 数据发布方将明文 $\mathcal{M}$ 封装为密态胶囊 $EC = \{u_{\rho(j)}, \mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_1', \mathcal{C}_2, \mathcal{C}_2', \mathcal{C}_{\rho(j),3}, \mathcal{C}_{\delta,k,4}^{(\tau_\theta)}\}$：

$$\mathcal{C}_0 = \mathcal{M} \cdot e(g_1, g_2)^{\alpha \cdot s_1}, \quad \mathcal{C}_1 = g_2^{s_1}, \quad \mathcal{C}_1' = (g_2^a)^{s_1}, \quad \mathcal{C}_2 = g_2^{s_2}, \quad \mathcal{C}_2' = (g_2^a)^{s_2},$$

$$\mathcal{C}_{\rho(j),3} = g_3^{\eta_j} \cdot \mathcal{H}_1(u_{\rho(j)} || v_{\rho(j)})^{s_2}, \quad \mathcal{C}_{\delta,k,4}^{(\tau_\theta)} = \left(\beta \cdot \prod_{k=1}^{d_{\tau_\theta}} \beta_k^{(b_{\tau_\theta}[k])}\right)^{\frac{s_1}{\mathcal{H}_2(b_{\log \mathcal{T}}^{(\delta)})}}.$$

其中, d_{τ_θ} 表示 $\mathbb{C}_{[\mathcal{L},\mathcal{R}]}$ 中元素 θ 的深度. 最终, 数据发布方将 EC 传输至云服务集群.

• ***MatAccess*** $(EC, \mathrm{Cak}, \mathrm{Ctk}, \delta \in (0, \mathcal{T}]) \to \mathcal{TC}/\perp$. 该算法首先判断 Cak 中所嵌入的属性集合 $\Omega = \{u_i : v_i\}_{i\in[1,l]}$ 是否与 EC 中的访问策略 $\mathbb{P}$ 相匹配. 若 Ω 不满足 $\mathbb{P}$, 即 Ω 为非授权集合, 则该算法输出 $\perp$. 否则, 它可在概率多项式时间内计算重构常数向量 $\vec{c}_\Omega \in \mathbb{Z}_p^{|\Omega|}$, 并且满足 $s_1 = \vec{c}_\Omega^{\top} \cdot \vec{\eta} = \sum_{j\in\mathcal{S}} c_j \cdot \eta_j$, 其中, $\mathcal{S} = \{j \mid j \in \Omega\} \subset [1, m]$. 该算法按照下述计算得到 $\mathcal{TC}_1$：

$$\mathcal{TC}_1 = e\left(ak_1, {\mathcal{C}_1}^{k_1} \cdot \mathcal{C}_1'\right) \cdot e\left(\prod_{j\in\mathcal{S}} (ak_{j,2})^{c_j}, {\mathcal{C}_2}^{k_1} \cdot \mathcal{C}_2'\right) \cdot e\left(\prod_{j\in\mathcal{S}} (\mathcal{C}_{\rho(j),3})^{c_j}, {k_2}^{k_1} \cdot k_3\right)^{-1}.$$

对于 $\{\mathcal{C}_{\delta,k,4}^{(\tau_\theta)}\}_{\theta\in\mathbb{C}_{[\mathcal{L},\mathcal{R}]}}$ 和 $\{tk_{\delta,k}^{(\vartheta)}\}_{\vartheta\in Path(\delta)}$, 计算 $\mathcal{TC}_{\delta,k,2}^{(\tau_\theta,\vartheta)} = e(tk_{\delta,k}^{(\vartheta)}, \mathcal{C}_1) \cdot (\mathcal{C}_{\delta,k,4}^{(\tau_\theta)}, k_2)^{-1}$. 最后, 云服务集群将转换胶囊 $\mathcal{TC} = \{\vartheta, \mathcal{TC}_1, \mathcal{TC}_{\delta,k,2}^{(\tau_\theta,\vartheta)}, \mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_{\delta,k,4}^{(\tau_\theta)}\}$ 发送至数据使用方.

• ***DeCap*** $(\mathcal{TC}, \mathrm{Ltk}) \to \mathcal{M}/\perp$. 该算法首先检查等式 $\mathcal{TC}_{\delta,k,2}^{(\tau_\theta,\vartheta)} \stackrel{?}{=} e(lk_1, \mathcal{C}_1)$ 是否成立. 若不成立, 则 $Path(\delta) \cap \mathbb{C}_{[\mathcal{L},\mathcal{R}]} = \varnothing$, 该算法中止. 否则, 根据 $\{\hat{\vartheta}, \mathcal{TC}_{\delta,k,2}^{(\hat{\vartheta})}\}$ 解析唯一的交集节点编码 $b_{d_{\hat{\vartheta}}}^{(\hat{\vartheta})}$. 算法确定 $\mathcal{C}_{\delta,4} \leftarrow \mathcal{C}_{\delta,k,4}^{(\hat{\vartheta})}$ 和 $lk_{\delta,2} \leftarrow lk_{\delta,k,2}^{(\hat{\vartheta})}$, 并通过 Ltk 解封 $\mathcal{TC}$[1]：

$$\mathcal{M} = \mathcal{C}_0 \cdot \mathcal{TC}_1^{-1} \cdot e\left(\mathcal{C}_{\delta,4}, {k_2}^{k_1} \cdot k_3\right) \cdot e\left(lk_{\delta,2}, {\mathcal{C}_1}^{k_1} \cdot \mathcal{C}_1'\right)^{-1}.$$

• ***SanityCheck*** $(\mathrm{msk}, K, \mathrm{Cak}, \mathrm{Ctk}, \mathrm{Ltk}) \to 1/0$. 该算法按照下述四个步骤验证密钥合理性：**1)** 首先考虑密钥组件 K：① $k_1 \in \mathbb{Z}_p$ 且 $k_2, k_3 \in \mathbb{G}_2$; ② $e(g_1^a, k_2) = e(g_1, k_3) \neq 1$; **2)** 该算法检查属性密钥 Cak 各组件：① $ak_1, ak_{i,2} \in \mathbb{G}_1$; ② $e(ak_1, g_2^a \cdot g_2^{k_1}) = e(g_1, g_2)^\alpha \cdot e(g_1, k_2)^{-1} \cdot e(g_3, k_2^{k_1} \cdot k_3) \neq 1$; ③ $\exists\ i \in [1, l]$, $s.t.$ 特定属性元素 $\{u_i : v_i\}_{i\in[1,l]}$ 满足 $e(ak_{i,2}, g_2^a) = e(\mathcal{H}_1(u_i||v_i), k_3) \neq 1$; **3)** 该算法如下检查时隙密钥 Ctk：① $tk_{\delta,k}^{(\vartheta)} \in \mathbb{G}_1$; ② $\exists\ d_\vartheta \in [1, \log \mathcal{T}]$, $s.t.$ 特定路径节点 $\vartheta \in Path(\delta)$ 满足 $e(tk_{\delta,k}^{(\vartheta)}, g_2) \cdot e(g_1, k_3)^{-1} = e\left((\beta \prod_{k=1}^{d_\vartheta} \beta_\vartheta^{(b_\delta[k])})^{\frac{1}{\mathcal{H}_2(b_{\log \mathcal{T}}^{(\delta)})}}, k_2\right) \neq 1$; **4)** 该算法检查客户端密钥 Ltk 各组件：① $lk_1, lk_{\delta,k,2}^{(\vartheta)} \in \mathbb{G}_1$; ② $e(lk_1, g_2) = e(g_1, k_3) \neq 1$; ③ $\exists\ d_\vartheta \in [1, \log \mathcal{T}]$, $s.t.$ 特定路径节点 $\vartheta \in Path(\delta)$ 满足 $e(lk_{\delta,k,2}^{(\vartheta)}, g_2^a \cdot g_2^{k_1}) \cdot e(g_1, k_2)^{-1} = e\left((\beta \prod_{k=1}^{d_\vartheta} \beta_\vartheta^{(b_\delta[k])})^{\frac{1}{\mathcal{H}_2(b_{\log \mathcal{T}}^{(\delta)})}}, k_2^{k_1} \cdot k_3\right) \neq 1$.

若 $\{K, \mathrm{Cak}, \mathrm{Ctk}, \mathrm{Ltk}\}$ 均可通过本算法的域约束与良构性验证, 则输出 1, 否则输出 0.

• ***KeyTrace*** $(k_1, \mathcal{KL}) \to gid/\perp$. 若 ***SanityCheck*** $(\mathrm{msk}, K, \mathrm{Cak}, \mathrm{Ctk}, \mathrm{Ltk})$ 输出 0, 则本算法输出 $\perp$; 否则, $\{K, \mathrm{Cak}, \mathrm{Ctk}, \mathrm{Ltk}\}$ 为良构密钥. 由此, 密钥分发中心在 $\mathcal{KL}$ 中搜索 k_1. 若 k_1 存在, 则算法输出叛逆者全局身份 gid; 否则输出 $\perp$. 对于追踪出的叛逆者 gid, 密钥分发中心在下一时隙 $\delta' \in (0, \mathcal{T}]$ 不再设置 Ctk 和 Ltk.

• ***UpdtCap*** $(EC, \{\delta, \delta'\} \in (0, \mathcal{T}]) \to EC'$. 该算法解析当前时隙编码 $b_{\log \mathcal{T}}^{(\delta')} = b_{\delta'}[1] \| b_{\delta'}[2] \| \ldots \| b_{\delta'}[\log \mathcal{T}]$ 并计算 $\mathcal{H}_2(b_{\log \mathcal{T}}^{(\delta')})$. 该算法在 $\mathbb{Z}_p$ 中设置更新令牌 $\mathrm{Utk} = \frac{\mathcal{H}_2(b_{\log \mathcal{T}}^{(\delta)})}{\mathcal{H}_2(b_{\log \mathcal{T}}^{(\delta')})}$. 最后, 云服务集群计算组件 ${\mathcal{C}_{\delta,k,4}^{(\tau_\theta)}}'$ 并更新数据胶囊为 EC'：${\mathcal{C}_{\delta,k,4}^{(\tau_\theta)}}' = \left(\mathcal{C}_{\delta,k,4}^{(\tau_\theta)}\right)^{\mathrm{Utk}}$.

1) **注释**. 若 $\delta \in [\mathcal{L}, \mathcal{R}]$, 必存在 $Path(\delta)$ 和 $\mathbb{C}_{[\mathcal{L},\mathcal{R}]}$ 的唯一交集 $\hat{\vartheta}$, 使 Ltk 的内嵌路径信息 $\{\beta \cdot \prod_{k=1}^{d_\vartheta} \beta_\vartheta^{(b_\delta[k])}\}_{\vartheta\in Path(\delta)}$ 与 $\mathbb{C}_{[\mathcal{L},\mathcal{R}]}$ 和 EC 的区间覆盖信息 $\{\beta \cdot \prod_{k=1}^{d_{\tau_\theta}} \beta_k^{(b_{\tau_\theta}[k])}\}_{\tau_\theta\in\mathbb{C}_{[\mathcal{L},\mathcal{R}]}}$ 同余, 故 $\mathcal{TC}_{\delta,k,2}^{(\tau_\theta,\vartheta)} = e(lk_1, \mathcal{C}_1)$ 成立. 反之, 若 $\delta \notin [\mathcal{L}, \mathcal{R}]$, 则 ***DeCap*** 中止, 从临时性层面阻断非授权方解封 $\mathcal{TC}$, 从而实现数据胶囊的自毁机制.

Accepted

https://www.sciengine.com/doi/10.1360/SSI-2026-0005

5 安全性证明

5.1 可追踪性

定理 1 (可追踪性) 若 l-co-SDH 假设成立, 则当 $q \leqslant l$ 时, 本方案的通用构造具备可追踪性. 其中, 任意 P.P.T. 伪造者 $\mathcal{F}$ 至多具备可忽略的优势 ε.

证明. 假设存在 P.P.T. 伪造者 $\mathcal{F}$ 可在执行 q 次密钥询问后, 以不可忽略的优势 ε 攻破本方案的可追踪性, 则可作为子程序并构造模拟器 $\mathcal{B}$ 破解 l-co-SDH 困难假设. 形式化证明描述如下：

(1) 初始化阶段. $\mathcal{B}$ 获取一组 l-co-SDH 的问题实例 $\mathcal{D} = (\hat{g}_1, \hat{g}_1{}^{x}, \hat{g}_1{}^{x^2}, \ldots, \hat{g}_1{}^{x^l}, \hat{g}_2, \hat{g}_2{}^{x}, \hat{g}_2{}^{x^2}, \ldots, \hat{g}_2{}^{x^l})$. 其中, $l = q+1$. 随后, $\mathcal{B}$ 从 $\mathbb{Z}_p$ 中随机选取 q 个互不同余的整数 $c_1, c_2, \ldots, c_q$ 并定义多项式 $f(y) = \prod_{i=1}^{q}(y+c_i) = \sum_{i=0}^{q} z_i y^i$. 其中, 多项式 $f(y)$ 的系数被定义为 $\{z_i\}_{i\in[0,q]}$. 令 $g_1 = \prod_{i=0}^{q}(\hat{g}_1{}^{x^i})^{z_i} = \hat{g}_1{}^{f(a)}$ 且 $g_2 = \prod_{i=0}^{q}(\hat{g}_2{}^{x^i})^{z_i} = \hat{g}_2{}^{f(a)}$, 接着计算 $g_1^a = \prod_{i=1}^{q+1}(\hat{g}_1{}^{x^i})^{z_{i-1}} = \hat{g}_1{}^{a\cdot f(a)}$ 和 $g_2^a = \prod_{i=1}^{q+1}(\hat{g}_2{}^{x^i})^{z_{i-1}} = \hat{g}_2{}^{a\cdot f(a)}$. $\mathcal{B}$ 从 $\mathbb{G}_1$ 中随机选择 $\beta, \beta_1^{(0)}, \beta_1^{(1)}, \ldots, \beta_{\log\mathcal{T}}^{(0)}, \beta_{\log\mathcal{T}}^{(1)}$、从 $\mathbb{Z}_p$ 中选择 α 和 μ 并计算 $e(g_1,g_2)^\alpha$ 和 $g_3 = (\hat{g}_1{}^{a\cdot f(a)})^\mu = (g_1^a)^\mu$. 最后, 它输出 $\mathrm{mpk} = \{\beta, \beta_1^{(0)}, \beta_1^{(1)}, \ldots, \beta_{\log\mathcal{T}}^{(0)}, \beta_{\log\mathcal{T}}^{(1)}, g_1, g_2, g_3, g_2^a, e(g_1,g_2)^\alpha, \mathcal{H}_1(\cdot), \mathcal{H}_2(\cdot), e\}$ 并初始化密钥列表 $\mathcal{KL}$.

(2) 密钥询问阶段. 当 $\mathcal{F}$ 发起 $q \leqslant l$ 次针对 $\{gid_i, \Omega_i, \delta_i\}_{i\in[0,q]}$ 的密钥询问时, $\mathcal{B}$ 首先查询 $\mathcal{KL}$. 若相应密钥元组已存在, $\mathcal{B}$ 直接返回 K、Cak、Ctk 和 Ltk. 否则, $\mathcal{B}$ 设置多项式 $f_i(y) = f(y)/(y+c_i) = \prod_{j=1\neq i}^{q}(y+c_j)$. $f_i(y)$ 可进一步展开为 $f_i(y) = \sum_{j=0}^{q-1}\zeta_j y^j$. 随后, $\mathcal{B}$ 计算 $F_j = \prod_{j=0}^{q-1}(\hat{g}_1{}^{x^j})^{\zeta_j} = \hat{g}_1{}^{f_j(a)} = \hat{g}_1{}^{f(a)/(c_j+a)} = g_1{}^{1/(c_j+a)}$. $\mathcal{B}$ 从 $\mathbb{Z}_p$ 中选取 τ_{id} 并设置 $K = (k_1 = c_j = \gamma, k_2 = (\hat{g}_2{}^{f(a)})^{\tau_{id}} = g_2^{\tau_{id}}, k_3 = (\hat{g}_2{}^{a\cdot f(a)})^{\tau_{id}} = (g_2^a)^{\tau_{id}})$. 它如下计算 $\mathrm{Cak} = (ak_1 = (F_j)^{\alpha-\tau_{id}}\cdot g_3^{\tau_{id}}, ak_{i,2} = \mathcal{H}_1(u_i||v_i)^{\tau_{id}})$. $\mathcal{B}$ 解析 $\{\delta_i\}_{i\in[0,q]}$ 编码 $b_{\log\mathcal{T}}^{(\delta_i)} = b_{\delta_i}[1]\|\ldots\|b_{\delta_i}[\log\mathcal{T}]$ 并计算 $\mathcal{H}_2(b_{\log\mathcal{T}}^{(\delta_i)})$. 对于 $\vartheta_i \in Path(\delta_i)$, 它如下设置 Ctk 和 Ltk：

$$tk_{\delta_i,k}^{(\vartheta_i)} = (\hat{g}_1{}^{f(a)})^{\tau_{id}}\cdot\left(\beta\cdot\prod_{k=1}^{d_{\vartheta_i}}\beta_k^{(b_{\delta_i}[k])}\right)^{\frac{\tau_{id}}{\mathcal{H}_2(b_{\log\mathcal{T}}^{(\delta_i)})}},$$

$$lk_1 = (\hat{g}_1{}^{a\cdot f(a)})^{\tau_{id}},\qquad lk_{\delta_i,k,2}^{(\vartheta_i)} = (F_j)^{\tau_{id}}\left(\beta\cdot\prod_{k=1}^{d_{\vartheta_i}}\beta_k^{(b_{\delta_i}[k])}\right)^{\frac{\tau_{id}}{\mathcal{H}_2(b_{\log\mathcal{T}}^{(\delta_i)})}}.$$

$\mathcal{B}$ 最后返回密钥元组 $(K, \mathrm{Cak}, \mathrm{Ctk}, \mathrm{Ltk})$ 并将其写入 $\mathcal{KL}$.

(3) 密钥伪造阶段. $\mathcal{F}$ 输出一个伪造密钥元组 $(K^*, \mathrm{Cak}^*, \mathrm{Ctk}^*, \mathrm{Ltk}^*)$ 均可通过合理性检验, 并且满足 $k_1 \notin \{c_1, \ldots, c_q\}$. 若 $\mathcal{F}$ 成功伪造密钥, $\mathcal{B}$ 将 $f(y)$ 转化为 $f(y) = \Psi(y)(y+k_1) + o(y)$, 其中 $\Psi(y)$ 扩展为某一特定多项式 $\sum_{j=0}^{q-1}\xi_j y^j$ 且 $o_j \neq 0$. $\mathcal{B}$ 可如下设定元组 (c^*, x^*)：① 计算 $c^* \equiv k_1 \ (mod\ p)$; ② 计算 $Temp = \big(ak_1\cdot lk_{\delta,k,2}^{(\vartheta)}\cdot lk_1^{-\mu}\cdot tk_{\delta,k}^{(\vartheta)-1}\cdot lk_1\big)^{\frac{1}{\alpha}} = g_1^{\frac{1}{a+k_1}} = \hat{g}_1{}^{\frac{f(a)}{a+k_1}} = \hat{g}_1{}^{\Psi(a)}\cdot\hat{g}_1{}^{\frac{o(a)}{a+k_1}}$; ③ 计算 $x^* = \big(Temp\cdot\prod_{j=0}^{q-1}(\hat{g}_1{}^{x^j})^{\xi_j}\big)^{\frac{1}{o_j}} = \hat{g}_1{}^{\frac{1}{a+k_1}}$. 其中, $gcd\ (o_j, p) = 1$, 故可定义 $\frac{1}{o_j}$ 为 o_j 在模 p 意义下的逆. 否则, $\mathcal{B}$ 随机选择 $(c^*, x^*) \in \mathbb{Z}_p\times\mathbb{G}_1$ 作为 l-co-SDH 的解.

由于 $e(x^*, \hat{g}_2{}^{a}\cdot\hat{g}_2{}^{c^*}) = e(\hat{g}_1{}^{\frac{1}{a+k_1}}, \hat{g}_2{}^{a}\cdot\hat{g}_2{}^{k_1}) = e(\hat{g}_1, \hat{g}_2)$, 故元组 (c^*, x^*) 是 l-co-SDH 困难问题的可行解. 若 $\mathcal{F}$ 以不可忽略的优势 ε 成功伪造密钥元组, 则 $\mathcal{B}$ 可利用其能力解决 l-co-SDH 困难问题. 记 $\tilde{\mathcal{F}}$ 为 $\mathcal{F}$ 成功伪造合法密钥事件, 则 $\mathcal{B}$ 解决 l-co-SDH 问题的优势计算如下：$Adv_{\mathcal{F}}^{l-co-SDH} = \Pr[\mathcal{F}(c^*,x^*)|\tilde{\mathcal{F}}, gcd(o_j,p) \neq 1]\cdot\Pr[\tilde{\mathcal{F}}, gcd(o_j,p)\neq 1] + \Pr[\mathcal{F}(c^*,x^*)|\tilde{\mathcal{F}}, gcd(o_j,p)=1]\cdot\Pr[\tilde{\mathcal{F}}, gcd(o_j,p)=1] + \Pr[\mathcal{F}(c^*,x^*)|\neg\tilde{\mathcal{F}}]\cdot\Pr[\neg\tilde{\mathcal{F}}] = \Pr[\tilde{\mathcal{F}}, gcd(o_j,p)=1] = \Pr[\tilde{\mathcal{F}}]\cdot\Pr[gcd(o_j,p)=1] = \varepsilon$.

5.2 语义安全性

定理 2 (语义安全性) 若 DLIN 假设成立, 则本方案的通用构造具备语义安全性. 其中, 任意 P.P.T. 敌手

Accepted

https://www.sciengine.com/doi/10.1360/SSI-2026-0005

$\mathcal{A}$ 至多具备可忽略的优势 $\varepsilon/2$.

证明. 假设存在 P.P.T. 敌手 $\mathcal{A}$ 能够以不可忽略的优势 ε 攻破本方案的语义安全性, 则可作为子程序并构造模拟器 $\mathcal{B}$ 破解 DLIN 困难假设. 形式化证明描述如下：

(1) 初始化阶段. $\mathcal{A}$ 声明挑战临时区间 $[\mathcal{L}^*,\mathcal{R}^*]\subseteq(0,\mathcal{T}]$ 和挑战访问策略 $\mathbb{P}^*=(\mathbb{M}^*,\rho^*(\cdot),\{u^*_{\rho^*(j)}:v^*_{\rho^*(j)}\}_{j\in[1,m^*]})$. 随后, $\mathcal{B}$ 获取一组 DLIN 假设的实例 $\mathcal{D}=(g_1,g_1^{x_1},g_1^{x_2},g_1^{x_1y_1},g_1^{x_2y_2},g_2,g_2^{x_1},g_2^{x_2},g_2^{x_1y_1},g_2^{x_2y_2})$. $\mathcal{B}$ 通过计算 $e(g_1,g_2)^{\alpha}=e(g_1^{x_1},g_2)\cdot e(g_1,g_2^{x_2})=e(g_1,g_2)^{x_1+x_2}$ 隐含设置 $\alpha=x_1+x_2$. $\mathcal{B}$ 设定 $g_3=g_1^{x_1y_1}$ 并均匀选择一组标量 $\tilde{t},\tilde{t}_1^{(0)},\tilde{t}_1^{(1)},\ldots,\tilde{t}_{\log\mathcal{T}}^{(0)},\tilde{t}_{\log\mathcal{T}}^{(1)}\in\mathbb{Z}_p^{2\log\mathcal{T}+1}$, 随后设定 $\beta=(g_1^{x_1})^{\tilde{t}},\beta_1^{(0)}=(g_1^{x_1})^{\tilde{t}_1^{(0)}},\beta_1^{(1)}=(g_1^{x_1})^{\tilde{t}_1^{(1)}},\ldots,\beta_{\log\mathcal{T}}^{(0)}=(g_1^{x_1})^{\tilde{t}_{\log\mathcal{T}}^{(0)}},\beta_{\log\mathcal{T}}^{(1)}=(g_1^{x_1})^{\tilde{t}_{\log\mathcal{T}}^{(1)}}$. $\mathcal{B}$ 选取标量 $a\in\mathbb{Z}_p$ 并计算 g_2^a, 最后返回 $\mathrm{mpk}=\{\beta,\beta_1^{(0)},\beta_1^{(1)},\ldots,\beta_{\log\mathcal{T}}^{(0)},\beta_{\log\mathcal{T}}^{(1)},g_1,g_2,g_3,g_2^a,e(g_1,g_2)^{\alpha},\mathcal{H}_1(\cdot),\mathcal{H}_2(\cdot),e\}$.

(2) 询问阶段 1. 在接收到 $\mathcal{A}$ 针对 gid、$\delta\notin[\mathcal{L}^*,\mathcal{R}^*]$ 和 $\Omega\not\models\mathbb{P}^*$ 请求 K、Cak、Ctk 和 Ltk 的询问后, $\mathcal{B}$ 查询密钥列表 $\mathcal{KL}$, 若存在相应条目则返回至 $\mathcal{A}$. 否则如下响应：$\mathcal{B}$ 根据 gid 选择 $\gamma,\tilde{\tau},\tau_{id}\in\mathbb{Z}_p$ 并设置追踪组件 $K=(k_1=\gamma,k_2=g_2^{\tilde{\tau}\cdot\tau_{id}},k_3=(g_2^a)^{\tilde{\tau}\cdot\tau_{id}})$. 随后, $\mathcal{B}$ 根据 Ω 计算 Cak：

$$ak_1=(g_1^{x_1}\cdot g_1^{x_2}\cdot g_1^{-\tilde{\tau}\cdot\tau_{id}})^{\frac{1}{a+\gamma}}\cdot(g_1^{x_2})^{\tilde{\tau}\cdot\tau_{id}}=g_1^{\frac{x_1+x_2-\tilde{\tau}\cdot\tau_{id}}{a+\gamma}+x_2\cdot\tilde{\tau}\cdot\tau_{id}},\quad ak_{i,2}=\mathcal{H}_1(u_i||v_i)^{\tilde{\tau}\cdot\tau_{id}}.$$

$\mathcal{B}$ 解析 $\delta\notin[\mathcal{L}^*,\mathcal{R}^*]$ 编码 $b^{(\delta)}_{\log\mathcal{T}}=b_\delta[1]\|\ldots\|b_\delta[\log\mathcal{T}]$. 接着, $\mathcal{B}$ 针对 $\vartheta\in Path(\delta)$ 定义 Ctk：

$$tk^{(\vartheta)}_{\delta,k}=g_1^{a\cdot\tilde{\tau}\cdot\tau_{id}}\left((g_1^{x_1})^{\tilde{t}}\cdot\prod_{k=1}^{d_\vartheta}(g_1^{x_1})^{\tilde{t}_k^{(b_\delta[k])}}\right)^{\frac{\tau_{id}}{\mathcal{H}_2(b^{(\delta)}_{\log\mathcal{T}})}}=g_1^{a\cdot\tilde{\tau}\cdot\tau_{id}+\frac{x_1\left(\tilde{t}+\sum_{k=1}^{d_\vartheta}\tilde{t}_k^{(b_\delta[k])}\right)\cdot\gamma}{\mathcal{H}_2(b^{(\delta)}_{\log\mathcal{T}})}}.$$

类似地, $\mathcal{B}$ 计算 Ltk 的组件 $lk_1=g_1^{a\cdot\tilde{\tau}\cdot\tau_{id}}$, 并如下定义 $lk^{(\vartheta)}_{\delta,k,2}$：

$$lk^{(\vartheta)}_{\delta,k,2}=g_1^{\frac{\tilde{\tau}\cdot\tau_{id}}{a+\gamma}}\left((g_1^{x_1})^{\tilde{t}}\cdot\prod_{k=1}^{d_\vartheta}(g_1^{x_1})^{\tilde{t}_k^{(b_\delta[k])}}\right)^{\frac{\tilde{\tau}\cdot\tau_{id}}{\mathcal{H}_2(b^{(\delta)}_{\log\mathcal{T}})}}=g_1^{\tilde{\tau}\cdot\tau_{id}\left(\frac{1}{a+\gamma}+\frac{x_1\left(\tilde{t}+\sum_{k=1}^{d_\vartheta}\tilde{t}_k^{(b_\delta[k])}\right)}{\mathcal{H}_2(b^{(\delta)}_{\log\mathcal{T}})}\right)}.$$

最后, $\mathcal{B}$ 记录 $\mathcal{H}_1(\cdot)$ 和 $\mathcal{H}_2(\cdot)$ 的应答 $h_{i,1}$ 和 $h_{i,2}$, 并将密钥元组 $(K,\mathrm{Cak},\mathrm{Ctk},\mathrm{Ltk})$ 记录并发送至 $\mathcal{A}$.

(3) 挑战阶段. 在接收到 $\mathcal{A}$ 发送的两个等长消息 $\mathcal{M}_0^*$ 与 $\mathcal{M}_1^*$ 后, $\mathcal{B}$ 随机选取 $\psi\in\{0,1\}$. $\mathcal{B}$ 计算获得挑战临时区间 $[\mathcal{L}^*,\mathcal{R}^*]$ 的覆盖集合 $\mathbb{C}_{[\mathcal{L}^*,\mathcal{R}^*]}$, 接着随机选择 $s_2,\tilde{\gamma}\in\mathbb{Z}_p$ 并解析当前编码 $b^{(\delta)}_{\log\mathcal{T}}$. 依据挑战访问策略 $\mathbb{P}^*$, $\mathcal{B}$ 将 $\mathcal{M}^*_\psi$ 如下封装为 EC^*_ψ：

$$\mathcal{C}^*_{\psi,0}=\mathcal{M}^*_\psi\cdot e(g_1^{x_1},g_2^{z_\psi})\cdot e(g_1^{z_\psi},g_2^{x_2})=\mathcal{M}^*_\psi\cdot e(g_1,g_2)^{\alpha\cdot s_1^*},\quad \mathcal{C}_1^*=g_2^{z_\psi}=g_2^{s_1^*},\quad \mathcal{C}_1'^*=(g_2^{z_\psi})^a=(g_2^a)^{s_1^*},$$

$$\mathcal{C}_2^*=g_2^{s_2},\qquad \mathcal{C}_2'^*=\mathcal{C}_2^a=(g_2^a)^{s_2},\qquad \mathcal{C}^*_{\rho(j),3}=(g_1^{x_1y_1})^{\eta_j^*}\cdot(h_{j,1}^*)^{\tilde{\gamma}\cdot s_2}=g_3^{\eta_j^*}\cdot\mathcal{H}_1(u_{\rho^*(j)}||v_{\rho^*(j)})^{s_2},$$

$$\mathcal{C}^{(\tau_\theta)*}_{\delta,k,4}=\left((g_1^{z_\psi})^{\tilde{t}}\cdot\prod_{k=1}^{d_{\tau^*_\theta}}(g_1^{z_\psi})^{\tilde{t}_k^{(b_{\tau^*_\theta}[k])}\cdot\tilde{\gamma}}\right)^{\frac{1}{\mathcal{H}_2(b^{(\delta)}_{\log\mathcal{T}})}}=(g_1^{z_\psi})^{\frac{\tilde{t}+\tilde{\gamma}\sum_{k=1}^{d_{\tau^*_\theta}}\tilde{t}_k^{(b_{\tau^*_\theta}[k])}}{\mathcal{H}_2(b^{(\delta)}_{\log\mathcal{T}})}}=\left(\beta\cdot\prod_{k=1}^{d_{\tau^*_\theta}}\beta_k^{(b_{\tau^*_\theta}[k])}\right)^{\frac{s_1^*}{\mathcal{H}_2(b^{(\delta)}_{\log\mathcal{T}})}}.$$

最后, $\mathcal{B}$ 将 $EC^*_\psi=\{\mathcal{C}^*_{\psi,0},\mathcal{C}_1^*,\mathcal{C}_1'^*,\mathcal{C}_2^*,\mathcal{C}_2'^*,\mathcal{C}^*_{\rho(j),3},\mathcal{C}^{(\tau_\theta)*}_{\delta,k,4}\}$ 发送至 $\mathcal{A}$.

(4) 询问阶段 2. $\mathcal{A}$ 继续执行与阶段 1 中的询问, 但必须保证 $\Omega\not\models\mathbb{P}^*$ 且 $\delta\notin[\mathcal{L}^*,\mathcal{R}^*]$.

(5) 猜测. $\mathcal{B}$ 接收来自 $\mathcal{A}$ 对随机比特 ψ 的猜测 $\psi'\in\{0,1\}$.

若 $\psi'=\psi$, 则 $\mathcal{B}$ 猜测视图 $vw'=0$, 即判定挑战项 $\mathcal{T}_\psi$ 有效, 否则 $\mathcal{B}$ 输出猜测视图 $vw'=1$, 即判定挑战项 $\mathcal{T}_\psi$ 为 $\mathcal{T}_1$. 因此, $\mathcal{A}$ 的语义安全性优势计算如下：$Adv^{DLIN}_{\mathcal{A}}=1/2\cdot\Pr[vw'=vw|vw=0]+1/2\cdot\Pr[vw'=vw|vw=1]-1/2=1/2\cdot(\varepsilon+1/2)+1/2\cdot1/2-1/2=\varepsilon/2$.

Accepted

https://www.sciengine.com/doi/10.1360/SSI-2026-0005

表 2 时间与空间复杂度理论对比.

Table 2 Theoretical comparison of time and space complexity.

Schemes	*Time complexity*				*Space complexity*		
	KeyGen	***EnCap***	***DeCap***	***SanityCheck***	Cak	Ctk & Ltk	*EC*
XLY+14 [10]	$\mathcal{O}(\mathbb{M}\times\mathcal{T})$	$\mathcal{O}(\Omega\times\mathbb{C})$	$\mathcal{O}(\Omega\times\mathbb{C})$	—	$\mathcal{O}(\mathbb{M}\times\mathcal{T})$		$\mathcal{O}(\Omega\times\mathbb{C})$
MLD+17 [16]	$\mathcal{O}(\mathbb{M}+\mathcal{T})$	$\mathcal{O}(\Omega+\mathbb{C})$	$\mathcal{O}(\Omega+\mathbb{C})$	—	$\mathcal{O}(\mathbb{M})$	$\mathcal{O}(\mathcal{T})$	$\mathcal{O}(\Omega+\mathbb{C})$
MLL+23 [19]	$\mathcal{O}(\Omega+\mathcal{T})$	$\mathcal{O}(\mathbb{M}+\mathbb{C})$	$\mathcal{O}(\Omega+\mathbb{C})$	—	$\mathcal{O}(\Omega\times\mathcal{T})$		$\mathcal{O}(\mathbb{M}+\mathbb{C})$
HPL+22 [23]	$\mathcal{O}(\Omega+\log\mathcal{T})$	$\mathcal{O}(\mathbb{M}+\hat{\mathbb{C}})$	$\mathcal{O}(\Omega)$	$\mathcal{O}(\Omega)$	$\mathcal{O}(\Omega)$	$\mathcal{O}(\log\mathcal{T})$	$\mathcal{O}(\mathbb{M}+\hat{\mathbb{C}})$
YLZ+25 [27]	$\mathcal{O}(\Omega+\mathcal{R})$	$\mathcal{O}(\mathbb{M}+\mathcal{R})$	$\mathcal{O}(\Omega+\mathcal{R})$	$\mathcal{O}(\Omega)$	$\mathcal{O}(\Omega+\mathcal{R})$	—	$\mathcal{O}(\mathbb{M}+\mathcal{R})$
MC25 [32]	$\mathcal{O}(\Omega+\log\mathcal{T})$	$\mathcal{O}(\mathbb{M})$	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(\Omega)$	$\mathcal{O}(\log\mathcal{T})$	$\mathcal{O}(\mathbb{M})$
Ours	$\mathcal{O}(\Omega+\log\mathcal{T})$	$\mathcal{O}(\mathbb{M}+\hat{\mathbb{C}})$	$\mathcal{O}(1)$	$\mathcal{O}(\Omega+\log\mathcal{T})$	$\mathcal{O}(\Omega)$	$\mathcal{O}(\log\mathcal{T})$	$\mathcal{O}(\mathbb{M}+\hat{\mathbb{C}})$

6 性能评估

在表 2 所示的理论对比中, 我们考虑若干代表性工作与本文方案在 ***KeyGen***、***EnCap***、***DeCap*** 和 ***SanityCheck*** 上的时间复杂度比较, 以及属性密钥、时隙密钥和密态数据胶囊的空间复杂度比较. 对比工作包括 XLY+14 [10]、MLD+17 [16]、MLL+23 [19]、HPL+22 [23]、YLZ+25 [27] 和 MC25 [32].

其中, $\mathbb{M}$ 和 Ω 分别表示份额与属性数量; $\mathbb{C}$ 和 $\hat{\mathbb{C}}$ 分别表示临时区间长度和区间覆盖集合规模/最小覆盖子集规模; $\mathcal{T}$ 表示时隙节点数量; $\log\mathcal{T}$ 表示时隙树/撤销树的深度; $\mathcal{R}$ 表示撤销用户数量. 具体而言, 本方案在 ***KeyGen*** 中将时隙组件 Ctk 和 Ltk 的复杂度从 $\mathcal{O}(\mathcal{T})$ 降低至亚线性 $\mathcal{O}(\log\mathcal{T})$, 在 ***EnCap*** 中将临时区间复杂度从 $\mathcal{O}(\mathbb{C})$ 降低至区间覆盖集合规模 $\mathcal{O}(\hat{\mathbb{C}})$, 并在 ***DeCap*** 中保持常数复杂度 $\mathcal{O}(1)$. 值得注意的是, 本方案在 ***SanityCheck*** 中可同时对 Cak、Ctk 和 Ltk 执行合理性检验, 故相较 HPL+22 [23]、YLZ+25 [27] 和 MC25 [32]

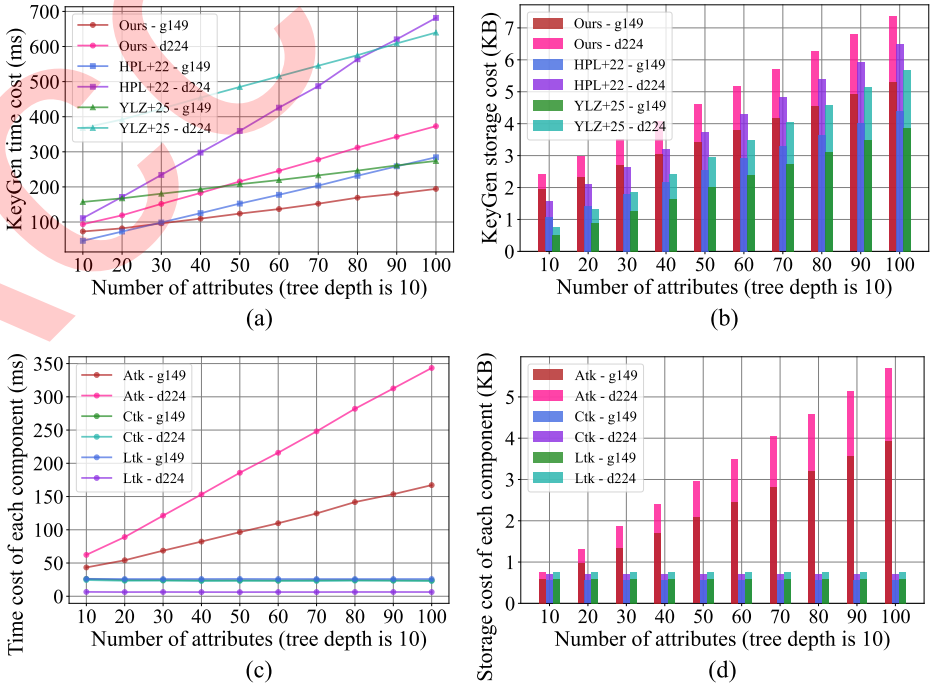


图 4 (网络版彩图) 密钥生成过程时间与存储开销 (时隙树深度为 10): (a) KeyGen 整体时间开销 (毫秒). (b) KeyGen 整体存储开销 (千字节). (c) KeyGen 组件时间开销 (毫秒). (d) KeyGen 组件存储开销 (千字节).

Figure 4 (Color Online) Time and storage cost of the key generation process (tree depth is 10): (a) KeyGen time cost (ms). (b) KeyGen storage cost (KB). (c) Time cost of each component (ms). (d) Storage cost of each component (KB).

https://www.sciengine.com/doi/10.1360/SSI-2026-0005

Accepted

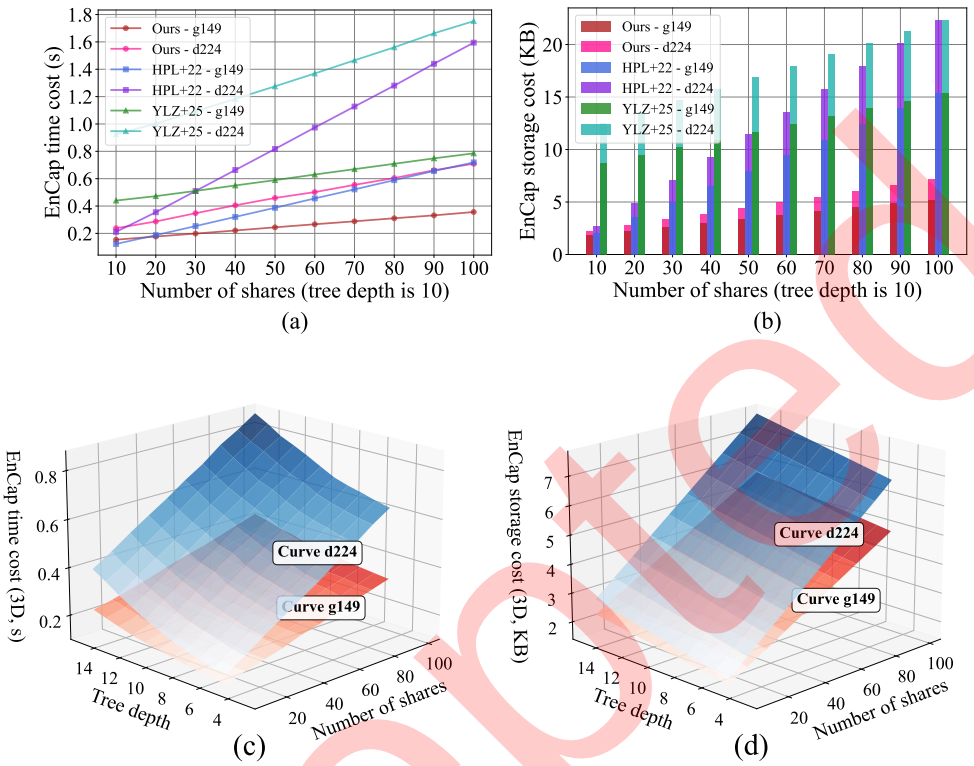


图 5 (网络版彩图) 封装过程时间与存储开销: (a) EnCap 时间开销 (秒). (b) EnCap 存储开销 (千字节). (c) EnCap 时间开销 (三维视图, 秒). (d) EnCap 存储开销 (三维视图, 千字节).

Figure 5 (Color Online) Time and storage cost of the encapsulation process: (a) EnCap time cost (s). (b) EnCap storage cost (KB). (c) EnCap time cost (3D, s). (d) EnCap storage cost (3D, KB).

具有较高的时间复杂度 $\mathcal{O}(\Omega + \log \mathcal{T})$.

实验评估在一台搭载频率为 2.10 GHz 的 Intel i7-12700 处理器、配备运行内存为 16 GB 的台式机上运行. 由于本文构造基于非对称双线性映射, 仿真选用 JPBC 库 [26] 中嵌入度为 6 的 d224 曲线与嵌入度为 10 的 g149 曲线, 在 Java SE Development Kit 21 (JDK 21) 环境下衡量本文构造的性能. 为系统对比 HPL+22 [23]、YLZ+25 [27] 与本方案的性能, 属性总体规模 $\mathcal{U}$、属性数量 Ω 和份额数量 $\mathbb{M}$ 的范围均设为 10 至 100, 时隙树深度 $\log \mathcal{T}$ 为 5 至 15.

图 4 清晰地刻画了密钥生成过程中 ***KeyGen*** 的整体性能. 其中, 图 4(a) 和图 4(b) 对比了各方案的整体性能, 图 4(c) 和图 4(d) 则详细分析了三种密钥组件 Cak、Ctk 和 Ltk 的开销分布. 由图 4(a) 与图 4(b) 可知, 三种方案的时间与存储成本均随 Ω 的增加呈线性增长. HPL+22 [23] 在关联属性时采用撤销树结构中路径值的承诺定义密钥组件. YLZ+25 [27] 的总成本不仅只关联属性集合, 还受撤销列表中用户数量的影响. 换言之, 该方案将动态撤销特性实施于密钥, 导致每次撤销均需执行全局密钥更新与分发. 本方案将 u_i 与 v_i 拼接嵌入 Cak、将时隙编码 $b_{\log \mathcal{T}}^{(\delta)}$ 嵌入 Ctk 和 Ltk, 虽具备较高存储成本, 但在耗时方面相较于 HPL+22 [23] 和 YLZ+25 [27] 具有明显优势. 具体而言, 当 $\mathcal{U} = 100$ 且 $\log \mathcal{T} = 10$ 时, 本方案在 g149 和 d224 曲线下总耗时分别约为 194.273 毫秒与 373.102 毫秒, 其存储总开销则分别达到 5.288 千字节 (K、Cak、Ctk 和 Ltk 分别为 0.143、3.934、0.557 和 0.594 千字节) 与 7.355 千字节 (K、Cak、Ctk 和 Ltk 分别为 0.190、5.688、0.711 和 0.766 千字节), 均处于可接受范围.

图 5(a) 和图 5(b) 分别展示了 ***EnCap*** 算法以 $\mathbb{M}$ 为自变量的时间与存储成本, 而以 $\log \mathcal{T}$ 和 $\mathbb{M}$ 作为独立变量的三维视图则分别在如图 5(c) 和图 5(d) 中描述. 在本方案中, 组件 $\mathcal{C}_{\delta,k,4}^{(\tau_\theta)}$ 的成本取决于 $[\mathcal{L}, \mathcal{R}]$ 的规

https://www.sciengine.com/doi/10.1360/SSI-2026-0005

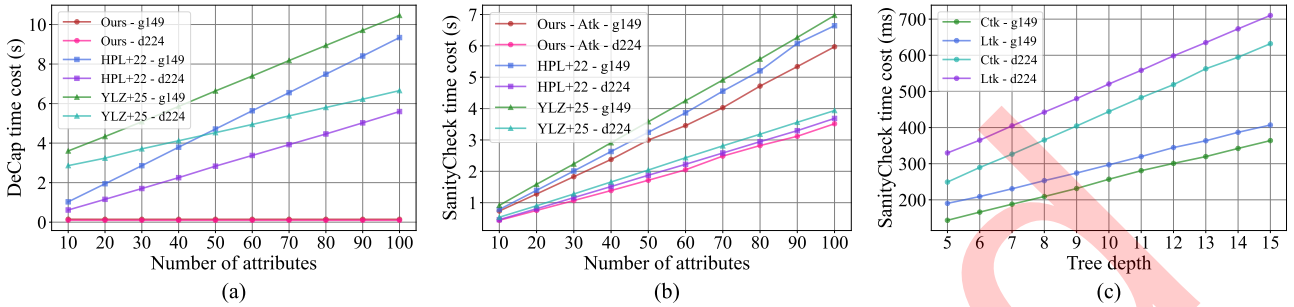


图 6 (网络版彩图) 解封装与追踪过程开销: (a) DeCap 时间开销 (秒). (b) SanityCheck 时间开销 (秒). (c) SanityCheck 时间开销 (毫秒).

Figure 6 (Color Online) Cost of the decapsulation and tracing processes: (a) DeCap time cost (s). (b) SanityCheck time cost (s). (c) SanityCheck time cost (ms).

表 3 *EnCap* 和 *DeCap* 算法中真实数据集下的时间与存储消耗 (秒, 兆字节).

Table 3 Time and storage consumption of ***EnCap*** and ***DeCap*** algorithms on real-world datasets (s, MB).

Algorithms	*Curves*	*soc-orkut*		*flickr*		*sc-ldoor*		*yahoo-msg*		*sc-msdoor*	
		Time	**Storage**	**Time**	**Storage**	**Time**	**Storage**	**Time**	**Storage**	**Time**	**Storage**
EnCap $\log\mathcal{T}=20$ $\mathbb{M}=500$	g149	27.741	405.901	15.941	222.958	5.541	67.029	4.258	40.287	3.234	30.589
	d159	14.893	256.372	7.930	140.829	3.498	42.348	2.710	25.458	2.373	19.333
	d201	15.893	333.284	8.780	183.077	4.401	55.052	3.560	33.095	3.259	25.132
	d224	16.502	358.921	9.743	197.160	4.902	59.287	4.202	35.641	3.965	27.066
	e	58.756	273.643	56.306	150.396	50.759	45.350	50.211	27.334	48.715	20.801
	a	18.696	273.535	15.823	150.289	12.673	45.242	12.494	27.227	12.637	20.693
DeCap	g149	34.933	405.869	15.754	222.925	4.907	66.997	3.292	40.255	2.394	30.556
	d159	12.905	256.338	7.483	140.795	2.439	42.314	1.315	25.424	1.102	19.299
	d201	13.486	333.240	7.503	183.033	2.289	55.008	1.619	33.051	1.344	25.088
	d224	16.497	358.873	10.186	197.113	2.879	59.239	1.939	35.594	1.653	27.018
	e	7.309	273.427	3.992	150.181	1.083	45.135	0.702	27.119	0.566	20.585
	a	6.293	273.427	4.033	150.181	0.957	45.135	0.748	27.119	0.493	20.585

模. 不失一般性, 我们采用区间 $[1,\mathcal{T}-1]$ 作为临时区间以获得尽可能多的覆盖集合节点. 相较于包含多个 $\mathbb{M}$ 规模组件的 HPL+22 [23] 和 YLZ+25 [27], 本方案仅具备份额相关的单一组件 $\mathcal{C}_{\rho(j),3}$, 故其时间与存储成本显著优于 HPL+22 [23] 和 YLZ+25 [27]. 另一方面, 本方案在 $\log\mathcal{T}$ 的每轮迭代中仅引入两个新增覆盖节点, 因此临时性层面的开销增长微乎其微. 在 $\log\mathcal{T}=10$ 且 $\mathcal{U}=100$ 的条件下, 本方案在 g149 和 d224 曲线下封装时间分别约为 356.353 与 709.033 毫秒, 其存储总开销则分别达到 5.232 与 7.164 千字节. 在相同条件下, HPL+22 [23] 和 YLZ+25 [27] 在前述配置曲线中分别耗时 {719.047, 1594.412} 和 {785.289, 1751.839} 毫秒, 且均具有 {15.400, 22.367} 千字节的存储成本. 因此, 相较于 HPL+22 [23] 和 YLZ+25 [27], 本方案的封装性能分别提升了 2.017× ∼ 2.203× 和 2.249× ∼ 2.471×, 密文存储分别降低了 2.943× 和 3.122×.

在图 6(a) 所示的解封装过程中, 本方案的 ***DeCap*** 展现出更优的常数时间复杂度, 在 g149 和 d224 曲线下的耗时分别控制在 150 和 100 毫秒以内. 而 HPL+22 [23] 和 YLZ+25 [27] 在解密时均需执行属性集合 Ω 和访问策略 $\mathbb{P}$ 之间的线性匹配, 显著增加了数据使用方的本地负担. 如图 6(b) 所示, 在属性密钥追踪的最坏时间复杂度下, HPL+22 [23]、YLZ+25 [27] 和本方案保持相近的计算成本. 由图 6(c) 可知, 本方案对于时隙密钥 Ctk 和 Ltk 的追踪过程仅产生毫秒级开销.

为直观验证本方案在数据胶囊流通中的可行性, 我们在表 3 中进一步评估了 ***EnCap*** 和 ***DeCap*** 在五个

Accepted

https://www.sciengine.com/doi/10.1360/SSI-2026-0005

公开数据集上的测试结果. 所用数据集包括 ***soc-orkut***、***flickr***、***sc-ldoor***、***yahoo-msg*** 和 ***sc-msdoor*** [35], 其存储体量依次递减. 由表 3 可知, 各配置曲线 {g149, d159, d201, d224, e, a}.properties 对 ***EnCap*** 与 ***DeCap*** 的性能影响具有显著差异. 针对 ***EnCap*** 的时间开销, 不同曲线对 $\mathbb{M}$、$\log\mathcal{T}$ 及数据集体量的敏感性各异：e 曲线在哈希与模乘运算中对 $\mathbb{M}$ 规模尤为敏感, g149 与 a 曲线次之, 而 d 系列曲线相对稳定, 封装效率较高. 在 ***DeCap*** 中, 各曲线耗时受数据集体量的影响展现出明显的收敛趋势. 其中, e 与 a 曲线效率突出, 其解封装开销远低于 g149 及 d 系列曲线. 虽存在前述差异, 本文构造的封装与解封装均消耗秒级资源, 在数据要素流通中具备可扩展性与实用性.

7 总结

为应对数据要素流通中胶囊自毁与叛逆者追踪的实际需求, 本文提出一种兼具自毁数据胶囊和叛逆者追踪特性的临时性访问控制方法. 它通过完全二叉树结构与二进制编码, 将临时区间映射为互斥节点集合, 实现对时隙归属的准确判定. 此外, 该方案还通过全局密钥的合理性检验, 对属性密钥、时隙密钥和客户端密钥的恶意泄露行为实施溯源, 防止非授权方滥用访问权限. 本文提供了可追踪访问控制方法的构造实例并完成语义安全性和可追踪性的形式化证明. 与传统密态访问控制方法相比, 本方案在数据要素流通中具有更优的临时管控与责任溯源能力. 未来研究将基于本文的可追踪访问控制方法, 聚焦在数据胶囊密态流通过程中延伸控制与时隙密钥管理的技术实现, 探索基于临时性访问控制的策略可更新属性基构造及其面向端云协同架构的实际部署能力.

参考文献

1 China Academy of Information and Communications Technology, Cloud Computing and Big Data Research Institute. Data element development report. In: 2025 Data Element Development Conference, 2025 [中国信息通信研究院云计算与大数据研究所. 数据要素发展报告. 2025 年数据要素发展大会, 2025] Available: https://www.caict.ac.cn/kxyj/qwfb/ztbg/202511/P020251128616212191150.pdf

2 Lyu A L, Wang D Y, Yang X Y, et al. Hot spots and trend analysis of data element circulation development. Inf. Commun. Technol. Policy, 2025, 51(04): 2–7 [吕艾临, 王丹阳, 杨晓芸, 等. 数据要素流通发展热点与趋势分析. 信息通信技术与政策, 2025, 51(04): 2–7]

3 Li H, Li F H, Shi X Y, et al. Research on access control for secure cross-domain data circulation. J. Commun., 2025, 46(4): 238–254 [李恒, 李凤华, 史欣怡, 等. 面向数据跨域安全流通的访问控制研究综述. 通信学报, 2025, 46(4): 238–254]

4 Xiao M Y, Huang Q, Chen W Y, et al. Domain-specific fine-grained access control for cloud-edge collaborative IoT. IEEE Trans. Inf. Forensics Secur., 2024, 19: 6499–6513

5 Jing J W. Challenges in key management. J. Cryptol. Res., 2025, 12(5): 957–958 [荆继武. 密钥管理的挑战. 密码学报, 2025, 12(5): 957–958]

6 European Union. General data protection regulation (GDPR). 2018. Available: https://gdpr-info.eu/

7 Chinese Government Portal. Action plan for the development of trusted data spaces (2024–2028). 2024 [中国政府网. 可信数据空间发展行动计划 (2024—2028 年). 2024] Available: https://www.gov.cn/zhengce/zhengceku/202411/P020250105293637969784.pdf

8 Lyu Q Y, Zhou Y L, Ren Y Z, et al. Secure personal data sharing for simultaneous, parallel or sequential processing service: Autonomously and controllably. Future Gener. Comput. Syst., 2025: 107879

9 Lyu Q Y, Zhou Y L, Ren Y Z, et al. Toward personal data sharing autonomy: a task-driven data capsule sharing system. IEEE Trans. Inf. Forensics Secur., 2024, 19: 9760–9774

10 Xiong J B, Liu X M, Yao Z Q, et al. A secure data self-destructing scheme in cloud computing. IEEE Trans. Cloud Comput., 2014, 2(4): 448–458

11 Bertino E, Bettini C, Ferrari E, et al. A temporal access control mechanism for database systems. IEEE Trans. Knowl. Data Eng., 2002, 8(1): 67–80

12 Paterson K G, Quaglia E A. Time-specific encryption. In: Proceedings of the 7th Security and Cryptography for Networks (SCN 2010). Amalfi, Italy: Springer, 2010: 1–16

13 Ishizaka M, Kiyomoto S. Time-specific encryption with constant-size secret-keys secure under standard assumption. Cryptol. ePrint Arch., 2020: 595

14 Kasamatsu K, Matsuda T, Emura K, et al. Time-specific encryption from forward-secure encryption: generic and direct constructions. Int. J. Inf. Secur., 2016, 15: 549–571

15 Ishizaka M, Kiyomoto S. Time-specific signatures. In: Proceedings of the 23rd Information Security Conference (ISC 2020). Bali, Indonesia: Springer, 2020: 20–38

Accepted

https://www.sciengine.com/doi/10.1360/SSI-2026-0005

16 Ma S Q, Lai J Z, Deng H R, et al. Adaptable key-policy attribute-based encryption with time interval. Soft Comput., 2017, 21(20): 6191–6200
17 Zhu Y, Hu H X, Ahn G J, et al. Towards temporal access control in cloud computing. In: Proceedings of the 31st IEEE International Conference on Computer Communications (INFOCOM 2012). Orlando, FL, USA: IEEE, 2012: 2576–2580
18 Wang Z J, Huang D J, Zhu Y, et al. Efficient attribute-based comparable data access control. IEEE Trans. Comput., 2015, 64(12): 3430–3443
19 Miao Y B, Li F, Li X H, et al. Time-controllable keyword search scheme with efficient revocation in mobile e-health cloud. IEEE Trans. Mob. Comput., 2023, 23(5): 3650–3665
20 Tong Q Y, Li X H, Miao Y B, et al. Privacy-preserving boolean range query with temporal access control in mobile computing. IEEE Trans. Knowl. Data Eng., 2022, 35(5): 5159–5172
21 Liu T, Miao Y B, Choo K K R, et al. Time-controlled hierarchical multi-keyword search over encrypted data in cloud-assisted IoT. IEEE Internet Things J., 2022, 9(13): 11017–11029
22 Liu Z, Cao Z F, Wong D S. White-box traceable ciphertext-policy attribute-based encryption supporting any monotone access structures. IEEE Trans. Inf. Forensics Secur., 2012, 8(1): 76–88
23 Han D Z, Pan N N, Li K-C, et al. A traceable and revocable ciphertext-policy attribute-based encryption scheme based on privacy protection. IEEE Trans. Dependable Secure Comput., 2022, 19(1): 316–327
24 Ning J T, Dong X L, Cao Z F, et al. White-box traceable ciphertext-policy attribute-based encryption supporting flexible attributes. IEEE Trans. Inf. Forensics Secur., 2015, 10(6): 1274–1288
25 Qian X Y, Yuan S, Xu G W, et al. A lightweight function-hiding IPE Scheme with support for dual outsourcing. Sci. China Inf. Sci., 2025, 55(11): 2817–2833 [钱心缘, 袁帅, 徐国文, 等. 支持双外包的轻量级函数隐藏内积加密方案. 中国科学: 信息科学, 2025, 55(11): 2817–2833]
26 Angelo D C. JPBC - Java pairing-based cryptography library: introduction, 2013. Available: http://gas.dia.unisa.it/projects/jpbc/index.html
27 Yin H J, Li J X, Zhu Y, et al. A traceable CP-ABE scheme supporting dynamic revocation and efficient decryption for medical data sharing. IEEE Internet Things J., 2025, 12(24): 53610–53622
28 Zhang Z T, Zeng P, Pan B F, et al. Large-universe attribute-based encryption with public traceability for cloud storage. IEEE Internet Things J., 2020, 7(10): 10314–10323
29 Zeng P, Zhang Z T, Lu R, et al. Efficient policy-hiding and large universe attribute-based encryption with public traceability for internet of medical things. IEEE Internet Things J., 2021, 8(13): 10963–10972
30 Zhang Y H, Yang J Y, Zheng X K, et al. Key traceable attribute-based cross-domain access control encryption. J. Cryptol. Res., 2025, 12(5): 1031–1046 [张应辉, 杨嘉雨, 郑晓坤, 等. 密钥可追踪的属性基跨域访问控制加密方案. 密码学报, 2025, 12(5): 1031–1046]
31 Meng F, Cheng L X. STR-ABKS: Server-aided traceable and revocable attribute-based encryption with keyword search. IEEE Internet Things J., 2024, 11(7): 12649–12659
32 Meng F, Cheng L X. TSR-ABE: Traceable and server-aided revocable ciphertext-policy attribute-based encryption under static assumptions. IEEE Trans. Inf. Forensics Secur., 2025, 20: 955–967
33 Tan S Y, Gross T. MoniPoly—an expressive q-SDH-based anonymous attribute-based credential system with constant-size proofs. IEEE Trans. Dependable Secure Comput., 2024, 22(3): 1993–2010
34 Agrawal S, Chase M. FAME: Fast attribute-based message encryption. In: Proceedings of the 24th ACM SIGSAC Conference on Computer and Communications Security (CCS 2017). Dallas, TX, USA: ACM, 2017: 665–682
35 Leskovec J, Sosic R. Snap: A general-purpose network analysis and graph-mining library. ACM Trans. Intell. Syst. Technol., 2016, 8(1): 1–20. Available: https://snap.stanford.edu/data/other.html

Accepted

https://www.sciengine.com/doi/10.1360/SSI-2026-0005

Traceable access control method for self-destructive data capsules

Keyong Hong[1], Xiangyu Wang[1*], Guowen Xu[2], Yanrong Liang[3], Chuang Tian[1], Xinghua Li[1,4] & Jianfeng Ma[1,4]

1. *School of Cyber Engineering, Xidian University, Xi'an 710126, China*
2. *School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China*
3. *School of Computer Science, Northwest University, Xi'an 710127, China*
4. *State Key Laboratory of Integrated Service Networks (ISN), Xidian University, Xi'an 710071, China*

* Corresponding author. E-mail: wangxiangyu01@xidian.edu.cn

Abstract As an important carrier for data element circulation, data capsule realizes temporal access control over the entire lifecycle by encapsulating data subjects with their associated policies. After leaving the control boundary of its publisher, the capsule is transferred to the cloud cluster and provides controlled access to users based on the predefined policies. When the management of capsules is separated from ownership, traditional temporal access control methods not only explicitly embed the time range of execution policies within the ciphertext, which leads to the leakage of range structures, but also struggle to effectively prevent authorized entities from deliberately leaking decryption capabilities within specific time slots. To tackle the above issues, this paper proposes a temporal access control scheme that features capsule self-destruction and traitor tracing. It maps the execution policies into an encoded set of mutually exclusive nodes to accurately determine whether the specific time slot falls within this range, thereby implementing the self-destructive mechanism. Furthermore, this scheme supports accurate identification and privilege revocation of leakage responsibility through sanity check, global key tracing, and capsule update processes. Based on the l-type co-Strong Diffie-Hellman (l-co-SDH) assumption and the Decisional Linear (DLIN) assumption, our scheme achieves unforgeability under the traceability model and semantic security under the chosen plaintext attack model, respectively. Evaluation results based on the Java Pairing-Based Cryptography (JPBC) library and real-world datasets show that the proposed scheme displays excellent scalability and practicality in data capsule circulation.

Keywords data capsule, traitor tracing, self-destructive mechanism, temporal access control, data element circulation

Accepted

https://www.sciengine.com/doi/10.1360/SSI-2026-0005