软件学报 ISSN 1000-9825, CODEN RUXUEW
2026,37(3):1393−1412 [doi: 10.13328/j.cnki.jos.007428] [CSTR: 32375.14.jos.007428]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

支持索引动态更新的高效可搜索属性加密方案*

张明川[1], 万千雪[1], 刘牧华[2], 朱军龙[1], 吴庆涛[1,3]

[1](河南科技大学 信息工程学院, 河南 洛阳 471023)

[2](河南科技大学 数学与统计学院, 河南 洛阳 471023)

[3](龙门实验室, 河南 洛阳 471000)

通信作者: 刘牧华, E-mail: liumuhua@haust.edu.cn

摘　要: 基于属性的可搜索加密技术实现了多用户场景下加密数据安全且细粒度的共享, 但往往面临着加解密计算开销大、查询效率低、索引无法更新等问题. 为同时解决上述问题, 在可搜索属性加密技术的基础上提出了一个支持索引动态更新的高效检索方案. 具体而言, 通过复用相同的访问策略, 减少加密过程中因策略重复带来的计算开销, 并将大部分解密运算安全外包给云服务器, 减轻了本地设备的解密计算负担. 结合哈希表和跳表构建了一个支持多关键词检索的倒排索引结构, 使用 BLS 短签名技术实现了索引更新的权限验证. 形式化的安全分析证明, 该方案能够有效抵御合谋攻击、选择明文攻击、伪造更新令牌和解密私钥等多种攻击方式. 实验结果显示, 该方案兼具高效的检索和索引更新性能, 在策略重复时能有效降低加密计算开销.

关键词: 属性基加密; 可搜索加密; 动态索引; 多关键词检索; 策略复用

中图法分类号: TP309

中文引用格式: 张明川, 万千雪, 刘牧华, 朱军龙, 吴庆涛. 支持索引动态更新的高效可搜索属性加密方案. 软件学报, 2026, 37(3): 1393–1412. http://www.jos.org.cn/1000-9825/7428.htm

英文引用格式: Zhang MC, Wan QX, Liu MH, Zhu JL, Wu QT. Efficient Searchable Attribute Encryption Scheme Supporting Dynamic Index Updates. Ruan Jian Xue Bao/Journal of Software, 2026, 37(3): 1393–1412 (in Chinese). http://www.jos.org.cn/1000-9825/7428.htm

Efficient Searchable Attribute Encryption Scheme Supporting Dynamic Index Updates

ZHANG Ming-Chuan[1], WAN Qian-Xue[1], LIU Mu-Hua[2], ZHU Jun-Long[1], WU Qing-Tao[1,3]

[1](School of Information Engineering, Henan University of Science and Technology, Luoyang 471023, China)

[2](School of Mathematics and Statistics, Henan University of Science and Technology, Luoyang 471023, China)

[3](Longmen Laboratory, Luoyang 471000, China)

Abstract: Attribute-based searchable encryption (ABSE) enables secure and fine-grained sharing of encrypted data in multi-user environments. However, it typically encounters challenges such as high computational overhead for encryption and decryption, limited query efficiency, and the inability to update indexes dynamically. To address these limitations, this study proposes an efficient searchable scheme based on ABSE that supports dynamic index updates. The reuse of identical access policies minimizes redundant computation during encryption. Most decryption operations are securely outsourced to the cloud, thus reducing the local device's computational load. An inverted index structure supporting multi-keyword Boolean retrieval is constructed by integrating hash tables with skip lists. BLS short signature technology is employed to verify the permissions for index updates, ensuring data owners can manage the retrieval of encrypted data. Formal security analysis confirms that the proposed scheme effectively defends against collusion attacks, chosen plaintext attacks, forged update tokens, and decryption key forgery. Experimental results demonstrate high efficiency in both retrieval and index update

* 基金项目: 国家自然科学基金 (62102134); 河南省科技计划 (231111222600, 231100220600); 河南省高校科技创新团队支持计划 (24IRTSTHN022)
收稿时间: 2024-07-17; 修改时间: 2024-11-22; 采用时间: 2025-03-18; jos 在线出版时间: 2025-07-09
CNKI 网络首发时间: 2025-07-10

operations, along with a significant reduction in encryption overhead when access policy reuse occurs.

Key words: attribute-based encryption (ABE); searchable encryption (SE); dynamic index; multi-keyword retrieval; policy reuse

云计算的兴起和发展令外包型的数据存储变得越来越普遍[1]. 为了确保数据的机密性, 数据所有者通常会将明文数据加密后再存储到云服务器上. 传统的加密存储方式虽然保护了数据安全, 却带来了数据可用性的挑战. 为解决这一问题, Song 等人[2]首次研究了可搜索加密技术 (searchable encryption, SE), 实现了基于关键词对加密数据的检索操作, 开创了密码学中一个新的研究方向. 但多数 SE 方案旨在为工业级应用提供高效的检索策略[3], 且通常以一定程度的访问模式和检索模式泄露为代价[4,5], 容易受到泄露滥用攻击 (leakage abuse attack, LAA)[6]. 此外, SE 在实现加密数据的精细化访问控制方面也存在局限性, 难以有效支持多用户数据共享的应用需求[7], 在数据量庞大且动态变化的云环境中不具备良好的适用性. 属性基加密技术 (attribute-based encryption, ABE)[8,9]通过用户属性灵活地控制数据访问权限, 提供了更为精细和安全的数据访问机制[10]. 受此启发, Wang 等人[11]将属性基加密技术与可搜索加密方案相结合, 提出了基于属性的可搜索加密技术 (attribute-based searchable encryption, ABSE). 该方案同时实现了对加密数据的有效检索和基于属性的细粒度访问控制, 为云环境中加密数据的精确检索和细粒度访问控制提供了新的技术路径.

尽管 ABSE 技术已得到广泛研究, 现有方案仍存在一些待优化的方面: 首先, 在检索机制设计方面, 主流方案通过将数据访问者的搜索能力嵌入检索陷门, 并利用双线性配对运算对陷门和加密文件进行逐一匹配验证. 该类方案虽然实现了细粒度访问控制和精确检索, 但在大规模数据场景下的计算效率仍需优化. 其次, 在访问策略处理效率方面, 现有方案通常需要对访问策略中的所有属性进行逐个处理. 考虑到在实际应用场景中不同数据对象的访问策略经常出现重叠特性, 尤其是在需要精细化访问控制且属性域相对稳定的安全场景下, 这种策略重叠现象更为普遍. 如何针对这种策略重叠现象优化加密算法的处理效率, 也是一个关键问题. 此外, 在索引动态更新方面, 数据所有者初始上传加密数据时提取的关键词往往难以完全满足后续的检索需求, 这就要求云端索引能够随着检索需求进行动态调整[12]. 如何在提高检索效率的同时实现轻量级的索引更新并降低重构开销, 仍面临着技术挑战. 针对上述问题, 我们提出了一种支持索引更新的高效可搜索属性加密方案. 该方案将加密数据存储、检索以及属性解密等操作部署到可信的私有云服务器上执行, 以确保数据处理的准确性与安全性. 主要贡献如下.

(1) 优化的访问策略处理机制: 针对不同访问策略间存在属性重叠导致计算资源浪费的问题, 提出访问策略复用机制, 通过加密过程中对重复策略的动态复用, 有效降低了访问策略重复场景下的加密计算开销.

(2) 高效的动态索引结构: 设计了一种由哈希表和跳表构造的动态倒排索引结构, 该结构利用哈希表存储关键词, 跳表维护加密数据标识, 检索开销仅与目标关键词关联的数据项数量有关, 而无需遍历整个索引. 通过引入 BLS 短签名机制验证数据所有者的更新权限, 实现了对索引的安全动态更新.

(3) 安全性分析: 基于离散对数问题 (discrete logarithm problem, DLP) 证明了方案的解密私钥具有不可计算性. 基于判定型 Diffie-Hellman 困难问题 (decisional bilinear Diffie-Hellman, DBDH) 证明了方案的中间密文具有选择明文攻击下的不可区分性. 基于计算型 Diffie-Hellman 假设 (computational Diffie-Hellman, CDH) 证明了方案的更新令牌具有不可伪造性. 该方案还能抵御恶意数据访问者对复用策略的合谋攻击.

(4) 实验验证: 基于 JPBC 库 (Java pairing based cryptography) 实现了本文方案, 在真实数据集上进行了相关测试. 实验结果表明, 本文方案具备高效的检索性能、能够准确执行索引更新操作, 在复用策略时能有效缩短数据的加解密时间.

1 相关研究工作

可搜索属性加密技术结合了属性基加密的细粒度访问控制特性和可搜索加密技术中基于关键词的检索机制. 自 Wang 等人[11]在 2013 年提出基础理论构造以来, 研究人员在索引结构设计、访问控制机制优化和加解密效率提升等方面不断深化研究, 并取得了一定的进展[12−32].

在索引结构优化与检索性能提升方面, Liang 等人[12]引入了隐藏矢量加密技术构建检索策略, 该方法不仅支

持多关键词检索还允许数据所有者和授权访问者对云端加密数据的关键词进行动态管理. 但该方案只支持与门访问策略且关键词更新操作局限于新旧关键词集合的交集. Miao 等人[13]设计了一种针对单一关键词的分层数据检索的方案, 而后又在此方案的基础上得到支持多个检索关键词和撤销数据访问者权限两种方案. 但都采用了先访问权限验证后检索的方式, 检索延迟较高. 随后在文献 [14] 中, Miao 等人又设计出了一种支持多关键词检索、多密钥加密和支持数据动态更新的增强型方案. 该方案通过设计索引切换器和倒排索引的双索引结构实现了可验证的动态更新功能, 但更新操作需要多方交互进行, 存在通信开销问题. Liang 等人[15]根据隐私性要求将外包给云服务器的数据分成非隐私数据和隐私数据两类, 对其分别构造支持关键词和索引的动态更新的倒排索引结构, 实现了对加密数据的细粒度检索. 该方案在更新过程中的身份验证缺乏安全性分析, 可能导致恶意用户通过伪造身份进行未授权的关键词更新操作. Yin 等人[16]利用数据所有者的主密钥为数据访问者生成聚合私钥, 将每个索引关键词都关联一个访问策略并指定数据访问者的检索权限, 显著提升了检索效率, 但方案的索引结构依赖于 3 个独立的哈希映射表实现, 存储开销较大. Zhang 等人[17]设计了一种支持缩小检索范围且能对检索结果进行验证的多关键词检索方案, 该方案令多个数据所有者对同一个明文进行加密, 只有数据访问者同时满足多个数据所有者设定的访问策略时才能进行解密. 然而, 该方案在处理多关键词搜索时存在关键方程的设计缺陷, 无法有效实现多关键词场景下的检索功能. 针对这一问题, Guo 等人[18]通过增加陷门生成的计算开销修正了文献 [17] 中关键方程的设计错误, 在保持原有方案安全性的基础上实现了多关键词的有效搜索, 但大规模关键词场景下计算成本仍然较高. Huang 等人[19]通过将密文的关键词加密映射到布隆过滤器而后向上求并集的方式构造出一个检索二叉树, 通过遍历树节点完成对特定关键词的查找, 该方案实现了次线性的检索时间复杂度. 然而, 该方案未能有效处理布隆过滤器固有的假阳性问题, 可能会对检索结果出现误判现象.

在访问控制机制与策略优化方面, Zhang 等人[20]通过隐藏访问策略的属性集保护用户隐私信息, 利用区块链存放检索索引, 分布式云服务器存放数据, 有效防止了数据篡改和单点故障问题. 受限于区块链的不可篡改性, 该方案难以对数据和索引实现更新. Chen 等人[21]考虑到数据访问者的权限是处于动态变化中的, 设计了一种撤销数据访问者权限后数据仍然完整的方案. 该方案的检索机制仍采用传统的属性验证后匹配关键词的两阶段策略, 需要对密文进行线性扫描验证, 在面对海量加密数据时检索效率较低. Miao 等人[22]将时间概念引入索引的构造, 设计出一种时间范围可控且支持用户更新的多关键词搜索方案. 该方案在陷门生成和检索匹配阶段的计算复杂度与关键词数量同样表现为线性关系. 冯涛等人[23]利用差分隐私技术模糊数据所有者的身份, 结合区块链和云服务器协同存储索引及加密数据, 设计出一个高效且防篡改的数据共享方案. 该方案将计算密集的检索操作部署在区块链节点上会导致节点负载严重, 下载解密的方式会返回大量不符合访问策略的检索结果, 终端设备需要承担所有的解密开销. 牛淑芬等人[24]在隐藏访问策略的同时, 通过设计数据标签实现了对云端加密数据去重和检索结果验证. 但其检索和验证过程需要数据使用者和云服务器进行多轮交互, 增加了计算和通信开销. 王经纬等人[25]引入二叉树管理用户撤销列表并更新节点随机值实现复用, 缓解了系统中用户数量上限的问题. 通过在搜索算法中设置验证机制, 确保被撤销的用户无法检索加密数据. 然而该方案受限于二叉树的预设规模, 即使通过更新机制在一定程度上提高了节点的复用性, 仍难以动态扩充用户容量. 同时双线性配对运算的高计算检索开销也制约了方案在大规模加密数据场景下的部署效率. Xu 等人[26]将数据访问者的属性进行分层, 使每个属性都对应一系列的属性值, 实现了更加灵活的访问控制. 该方案同样使用了先验证访问权限再匹配关键词的线性处理模式, 计算开销较大. 为了解决个人健康记录共享过程中的权限委托问题, Lee 等人[27]提出了一种基于密钥聚合可搜索属性加密和区块链结合的方案, 该方案利用区块链的不可篡改性为数据索引提供了安全可靠的存储环境, 但也因此限制了索引的动态更新能力. Zhang 等人[28]通过聚合属性的哈希值表示访问策略, 实现了对个人健康记录中访问策略的隐藏. 但该方案仅支持与门访问策略和单关键词检索, 存在一定局限性.

在提升数据加解密效率, Niu 等人[29]将属性解密计算部分外包给边缘计算以降低轻量级设备的运营成本, 但该方案只支持单关键词检索. Rao 等人[30]结合属性基签密技术和云服务器辅助解密思想, 该方案支持布尔公式查询且无需与权威机构交互就能验证检索结果, 但陷门生成和检索算法的时间复杂度与关键词数量呈线性相关. Meng 等人[31]提出了一种适用于雾计算环境的方案, 该方案将大部分的解密计算转移到了雾节点上, 降低了数据

访问者的本地计算开销. 但未考虑关键词之间的关联机制, 无法支持多关键词查询需求. Zou 等人[32]利用区块链生成属性令牌, 在智能合约的帮助下实现属性授权和多关键词的模糊搜索, 在保证检索结果可靠的同时降低了计算成本. 该方案支持云服务器对加密数据进行预解密操作, 但由于使用了双线性配对的检索机制, 其搜索效率随关键词数量增加显著下降.

通过分析相关工作, 我们发现 ABSE 方案主要应用于医疗、金融、政府等对数据安全要求较高的领域. 现实情况是, 这些机构出于数据隐私法规、监管要求和数据敏感性等因素考虑, 通常会采用严格的私有部署模式将系统部署在本地数据中心或私有云环境中. 因此我们认为传统 ABSE 方案中基于对云服务器的半可信敌手假设在这些场景下可能过于保守, 可以适当弱化并关注性能优化以符合实际需求. 基于此, 我们提出了一种面向私有云环境的优化方案: 通过合理放宽关键词安全性假设构建了支持多关键词检索和动态更新的高效索引结构; 引入访问策略复用机制降低重复加密开销, 同时将属性解密运算安全地外包执行, 从而提升可搜索属性加密方案在实际应用场景中的实用性.

2 预备知识

本文方案中常用的符号如表 1 所示.

表 1 文中常见符号说明

符号	说明
G, G_T	阶为大素数p的乘法循环群
$e(a,b)$	双线性配对运算
λ	安全参数
$\mathbb{Z}_p^*$	由整数1, 2,…, p–1组成的集合
可忽略概率 ε	对任意的多项式函数 $poly(\lambda)$, 有 $\varepsilon < 1/poly(\lambda)$
属性集合 U	数据访问者的属性域
伪属性集合 V	与真实属性无关的随机属性域
$a\in_R\mathbb{Z}_p^*$, $b\in_R G$	从 $\mathbb{Z}_p^*$, G 中分别选取随机元素 a, b
$Index^{\text{Invert}}$	倒排检索索引结构
$FileID$	加密数据的唯一标识号
$\{kw_w\}_{w\in W}$	随加密数据上传到云的索引关键词
$\{sw_s\}_{s\in S}$	数据访问者搜索数据时定义的检索关键词
$\{rw_s\}_{i\in I}$	数据所有者更新索引时选定的目标关键词

2.1 双线性映射

令 G, G_T 为 p 阶的乘法循环群, 群 G 的生成元为 g, 阶数 p 为一个大素数, 双线性映射 $e: G\times G\to G_T$ 具有下列 3 个特性.

(1) 双线性: $\forall x, y\in G$, $\forall a, b\in\mathbb{Z}_p^*$, 等式 $e(x^a,y^b)=e(x,y)^{a\cdot b}$ 恒成立.

(2) 非退化性: $\exists x, y\in G$, 使得 $e(x,y)\neq 1$.

(3) 可计算性: $\forall x, y\in G$, 都可以有效计算出 $e(x,y)$.

2.2 困难性问题

(1) 判定型双线性 Diffie-Hellman 假设 (DBDH)

令 G, G_T 为满足双线性映射 $e: G\times G=G_T$ 的乘法循环群, 生成元 $g\in G$, 阶为素数 p. 选取 a, b, c, $z\in_R\mathbb{Z}_p^*$ 和 $\mu\in_R\{0,1\}$. 若 $\mu=0$, 输出 $T_0=(g,g^a,g^b,g^c,Z=e(g,g)^z)$, 否则输出 $T_1=(g,g^a,g^b,g^c,Z=e(g,g)^{a\cdot b\cdot c})$. 敌手 $\mathcal{B}$ 收到元组 $T_\mu=(g,g^a,g^b,g^c,Z)$ 后, 输出对 μ 的猜测 $\mu'\in\{0,1\}$. 若对于任意概率多项式时间敌手 $\mathcal{B}$, 成功区分 $e(g,g)^{a\cdot b\cdot c}$ 和随机元素 $e(g,g)^z$ 的优势 $Adv_{\mathcal{B}}=|\Pr[\mu'=\mu]-1/2|<\varepsilon$, 其中 ε 为可忽略概率, 则称 DBDH 假设是困难的.

(2) 离散对数问题 (DLP)

令 G, G_T 为满足双线性映射 $e: G\times G=G_T$ 的乘法循环群, 生成元 $g\in G$, 阶为素数 p. 选取 $a\in_R\mathbb{Z}_p^*$, 将 $T=(g,g^a)$ 发送给敌手 $\mathcal{B}$, 若对于任意概率多项式时间敌手 $\mathcal{B}$, 通过 g^a 计算得到 a 的概率 ε 是可忽略的, 则称 DLP 是困难的.

(3) 计算型 Diffie-Hellman 假设 (CDH)

令 G, G_T 为满足双线性映射 $e: G\times G=G_T$ 的乘法循环群, 生成元 $g\in G$, 阶为素数 p. 选取 a, $b\in_R\mathbb{Z}_p^*$, 将 $T=(g,g^a,g^b)$ 发送给敌手 $\mathcal{B}$, 若对于任意概率多项式时间敌手 $\mathcal{B}$, 通过 $T=(g,g^a,g^b)$ 得到 $g^{a\cdot b}$ 的概率 ε 是可以忽略的, 则称 CDH 假设是困难的.

2.3 BLS 短签名

选择素数阶为 p 的乘法循环群 G, G_T, 生成元 $g\in G$, 双线性映射 $e: G\times G=G_T$, 抗碰撞的哈希函数 $H:\{0,1\}^*\to G$, BLS 短签名方案包含以下 3 个步骤.

(1) 密钥生成: 签名者选取随机数 $\kappa\in_R\mathbb{Z}_p^*$ 作为私钥, 计算得 g^κ 作为公钥.

(2) 签名算法: 对要签名的消息 $m\in\{0,1\}^*$ 进行映射 $H_m=H(m)\to G$, 输出消息 m 的签名 H_m^κ.

(3) 签名验证: 若 $e(g,H_m^\kappa)=e(g^\kappa,H_m)$ 成立, 则签名验证成功, 否则验证失败.

2.4 策略复用及判定

策略复用是指预处理和保存常用的访问策略, 在后续加密时直接调用这些计算结果, 以降低加密开销. 当某个访问策略需要在后续多个数据的加密过程中需要重复使用时, 考虑将该访问策略生成一个复用策略, 在后续数据的加密过程中对其进行复用以减少重复计算, 从而提高密文的生成效率.

(1) 转换访问策略

若存在一个访问策略 $\Lambda=(a_1\vee(a_2\wedge(a_3\vee a_4\vee(a_5\wedge a_6\wedge a_7))))\vee(b_1\wedge b_2)\vee(c_1\wedge c_2\wedge c_3)$, 其中 $a_1,a_2,\ldots,c_3$ 为数据所有者设定的不同类别的属性, 访问策略 Λ 对应的常规访问控制树如图 1 的左半部分所示.

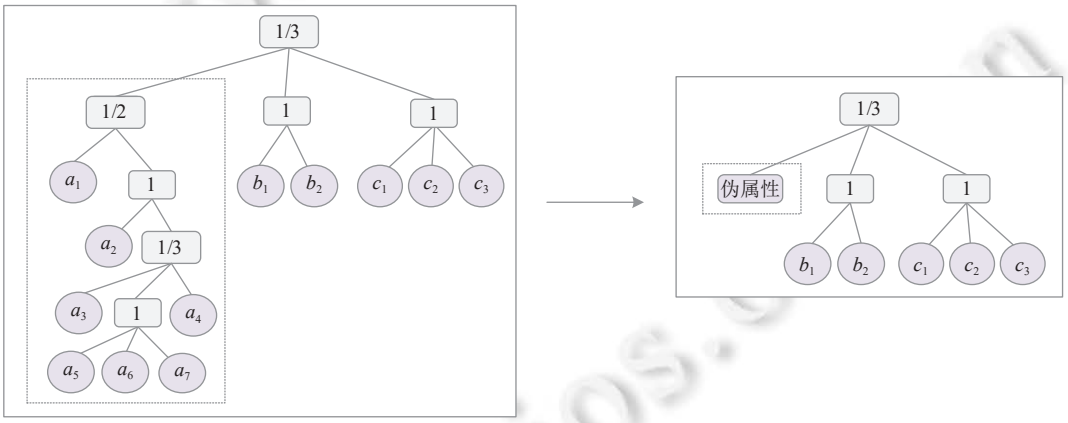


图 1 转换访问策略

假定数据所有者需要重复使用虚线框内的策略加密不同的数据, 可以将这部分策略设为复用策略 $\Lambda_{sub}=(a_1\vee(a_2\wedge(a_3\vee a_4\vee(a_5\wedge a_6\wedge a_7))))$, 并将其对应的子树视为一个叶子节点, 随机选取一个伪属性对其进行替换. 如图 1 右半部分所示.

(2) 对真子句构造布隆过滤器

① 对复用策略进行析取运算得到 4 个真子句:

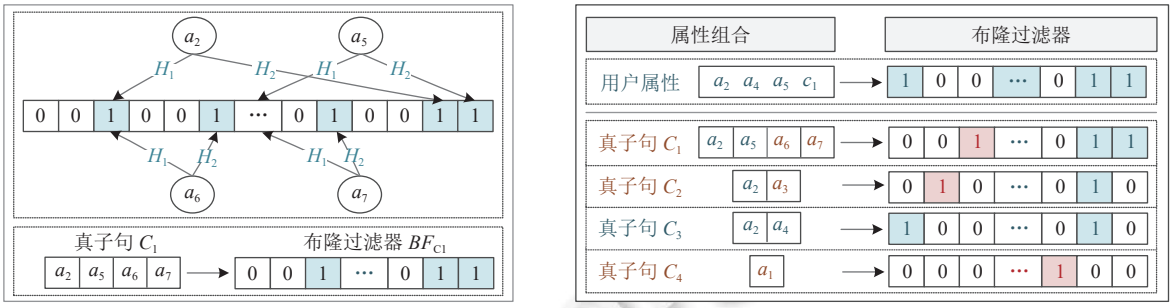
$$\Lambda_{sub1}=(a_5\wedge a_6\wedge a_7)\wedge a_2,\ \Lambda_{sub2}=a_3\wedge a_2,\ \Lambda_{sub3}=a_4\wedge a_2,\ \Lambda_{sub4}=a_1.$$

将真子句的属性集合分别记为: $C_1=\{a_5,a_6,a_7,a_2\}$, $C_2=\{a_3,a_2\}$, $C_3=\{a_4,a_2\}$, $C_4=\{a_1\}$.

② 为每个真子句分别构造一个子句布隆过滤器

初始化 4 个长度为 M, 值均为 0 的布隆过滤器 $BF[m]=0, \forall m \in [1,M]$.

使用一组相互独立的抗碰撞的哈希函数 $H=\{h_1,h_2,\ldots,h_t\}$ 将各个子句 C_j 中包含的属性分别映射到布隆过滤器的 l 个不同的位置上 $BF[h_t(a_{i,J})_{t\in l,i\in I}]=1$, 并将这些位置上的值由 0 置为 1, 其余位置上的值不变. 当某个位置多次被置 1 时, 其值仍取 1. 真子句 $C_1=\{a_5,a_6,a_7,a_2\}$ 的构造如图 2(a) 所示, 为了直观展示, 对图中布隆过滤器的位数和哈希函数个数进行了简化. 其余真子句的构造过程与 C_1 相同.



(a) 真子句的哈希映射与布隆过滤器构建示例　　(b) 基于布隆过滤器的策略判定过程示例

图 2　复用策略生成及判定过程

(3) 复用策略判定

云服务器根据密文中的复用策略标识符提取相应的子句布隆过滤器和属性值组件, 并进行如下判定.

① 对数据访问者的属性布隆过滤器和子句布隆过滤器进行等位判定.

要求子句布隆过滤器中值为 1 的位在数据访问者属性布隆过滤器中也必须为 1, 否则判定失败. 其中, 数据访问者的属性布隆过滤器由权威中心生成. 当且仅当真子句是用户属性的子集时, 等位判定才能通过. 判定过程如图 2(b) 所示, 只有 C_3 满足条件.

② 对属性密钥和子句的属性值组件进行配对验证.

利用布隆过滤器加快属性的判定过程, 快速过滤掉不匹配的属性访问组合. 考虑到布隆过滤器存在假阳性, 为进一步保证访问的合法性, 在布隆过滤器初步判定完成后, 需要结合具体的算法进行双线性配对验证.

当步骤①和②都判定成功时, 云服务器返回判定结果 $FR_v=0$ 并进行后续的外包属性解密.

2.5　支持多关键词检索和更新的倒排索引结构

数据索引主要用于建立数据与检索关键词间的映射关系, 根据映射方向的不同, 通常可以分为正向索引和倒排索引两种基本形式: 传统的索引结构采用正向映射方式, 建立数据到关键词的映射关系. 这种索引的构建方式简单, 便于数据的添加和删除操作, 但查找关键词时需要线性遍历整个索引结构, 效率较低. 倒排索引采用将关键词映射到数据, 通过访问关键词即可获取相关的数据集合, 提升了查找速度. 常规的倒排索引采用静态结构设计, 在构建完成后难以进行局部更新.

为解决这一局限性, 我们引入了一个两层索引架构: 外层采用哈希表实现关键词到数据集合的快速映射, 内层则利用跳表组织具体的数据项. 哈希表提供了 O(1) 的平均访问时间, 确保了检索操作的高效性; 跳表的概率平衡特性使得对数据项的插入、删除和更新操作都能在 $O(\log n)$ 时间内完成, 这一特性有效解决了传统静态索引难以应对频繁数据更新的实际需求. 为确保更新操作的安全性, 我们引入了 BLS 签名机制用以验证更新权限, 确保操作只有合法的数据所有者才能执行对指定关键词的更新操作. 同时, 得益于该索引结构的精确寻址能力, 所有的更新操作都被严格限制在目标数据项范围内, 有效避免了对无关数据的干扰.

(1) 索引构建

① 数据所有者选用性能良好且密码安全的哈希函数 (如 SHA-256、BLAKE2 等) 将关键词映射为一个固定长度的大整数 KW^* 并随着密文发送给云服务器. 该步骤支持自定义哈希函数以适应不同的性能需求.

② 云服务器选用一个分布均匀的散列函数 (如 Murmur Hash、City Hash、Cuckoo Hash 等) 将大整数 KW^* 映射到哈希表的相应槽位上, 得到关键词的索引位置 $H(KW^*)$. 由于哈希函数对同一关键词总是产生相同的输出, 因此在索引构建过程中, 可以在此步骤过滤重复的关键词. 该步骤利用哈希函数确定性的输出特性, 实现对重复关键词的自然去重操作.

③ 对哈希表的每个非空槽位, 当关键词对应多个数据项时, 采用概率平衡的跳表结构来组织这些关联数据. 每个跳表节点存储一对值 $< Fi, FMi >$ (即该加密数据的编号 *FileID* 和标识组件 *FileMatch*), 其中 *FileID* 是云服务器按照加密数据接收时间顺序生成的唯一标识号, *FileMatch* 是数据所有者的身份标识组件. 通过随机化的层次分配机制, 使每个节点以 1/2 的概率向上提升层级, 从而自动维持跳表的平衡状态.

(2) 检索操作

① 数据访问者用相同的哈希函数对要查询的关键词进行处理, 生成检索令牌发送给云服务器.

② 云服务器对检索令牌中的每个查询关键词使用相同的散列函数计算出相应的槽位, 直接访问跳表底层链表以获取与检索关键词对应的数据编号集合. 对于包含了多个关键词的复合查询, 还需要根据查询谓词对获取的数据编号集合执行相应的集合运算.

从性能角度分析, 哈希表定位关键词的时间复杂度为 O(1), 获取跳表中对应的完整数据编号列表需要 O(n) 的遍历时间, 对单关键词的检索复杂度为 O(n). 对于包含了 k 个关键词的复合查询, 总体检索复杂度为 O($k \cdot n$). 其中 n 仅代表检索关键词关联的数据条目数而非整个数据集, 因而在大规模数据中具有效率优势.

(3) 索引更新

为实现对检索索引的安全动态更新, 引入了在检索索引中加入 BLS 短签名验证机制, 在保持云端加密数据关键词数量不变的情况下, 实现对索引中关键词的安全替换操作. 这种更新操作不需要重建整个索引, 同时能够保证其他加密数据的索引不受更新操作的影响. 相关验证组件由数据所有者的唯一标识号生成, 确保只有合法的数据所有者才能完成对指定加密数据的关键词更新. 具体过程如下.

① 数据所有者将包含了原关键词、目标关键词、标识组件和签名组件的更新令牌发送给云服务器.

② 云服务器根据更新令牌的原关键词定位对应的跳表, 在跳表中查找与标识组件相等的加密数据并进行签名验证, 完成更新请求的合法判定.

③ 若更新请求合法, 云服务器修改相应的跳表指针, 将加密数据从原关键词对应的跳表中移除. 而后检查目标关键词在索引中是否存在, 若目标关键词尚未建立索引, 则在全局哈希表中创建该关键词的新索引项. 若已存在, 则将加密数据按编号顺序 *FileID* 加入目标关键词的跳表, 完成索引更新操作.

后文图 3 展示了更新请求合法时, 将数据 $F7$ 中的关键词从 $KW1$ 更新为 $KW2$ 时跳表指针的变换过程: 首先在 $KW1$ 对应的跳表中, 从最高层开始, 通过标识组件 $FM7$ 在跳表中查找数据 $F7$ 的位置并记录每一层的前驱节点, 而后从上到下依次解除 $F7$ 在各层与其相邻节点的链接关系 ($F4$ 和 $F10$、$F2$ 和 $F9$). 其次在 $KW2$ 对应的跳表中, 同样从最高层开始查找 $F7$ 的适当插入位置并记录每层前驱节点, 然后根据记录的前驱信息从上到下依次建立 $F7$ 与新相邻节点 ($F4$ 和 $F10$) 的链接关系. 其中, 检索过程用蓝色虚线表示, 指针的重构过程用红色实线表示. 从性能角度分析: 由于跳表的层级结构特性, 因而在 $KW1$ 和 $KW2$ 中查找数据 $F7$ 的操作时间复杂度均为 O($\log n$), 链接关系更新操作时间复杂度均为 O(1). 因此, 整个更新过程的总时间复杂度为 O($\log n$). 其中 n 仅代表与检索关键词关联的数据数量而非整个数据集.

3 方案构造

3.1 系统模型

本文方案的系统模型如图 4 所示, 包含 4 个实体: 权威中心、数据所有者、数据访问者和云服务器.

(1) 权威中心: 可信的属性授权实体. 负责生成系统主私钥和系统主公钥, 根据数据访问者的属性生成相应的属性密钥和解密私钥.

(2) 云服务器: 提供可信计算环境. 负责存储数据所有者上传的密文和关键词并生成相应的倒排检索索引结构. 当收到检索请求时, 云服务器对数据访问者上传的检索令牌在倒排索引中进行匹配查找, 并将检索结果基于数据访问者的属性密钥进行外包解密, 最终返回给数据访问者解密权限内的中间密文. 在收到数据所有者的更新请求时, 云服务器验证更新令牌的签名, 仅当验证通过时才执行相应的更新操作.

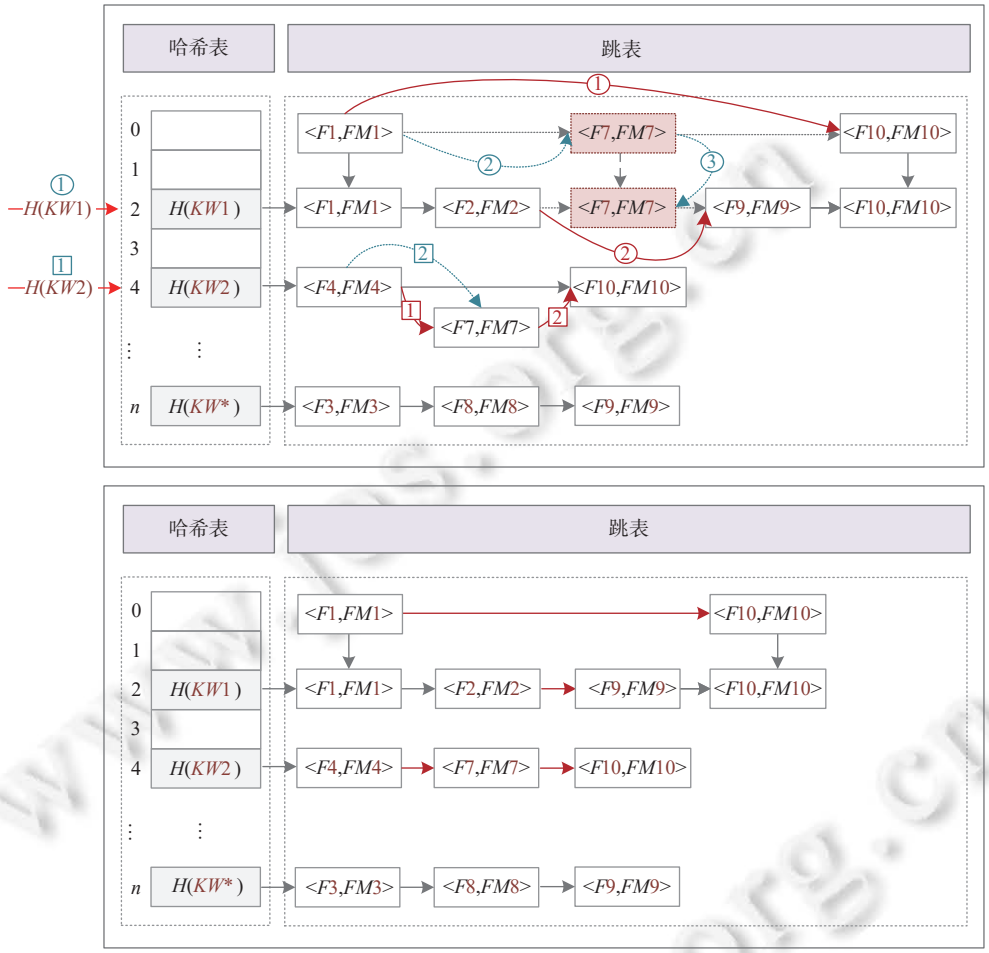


图 3 关键词替换过程

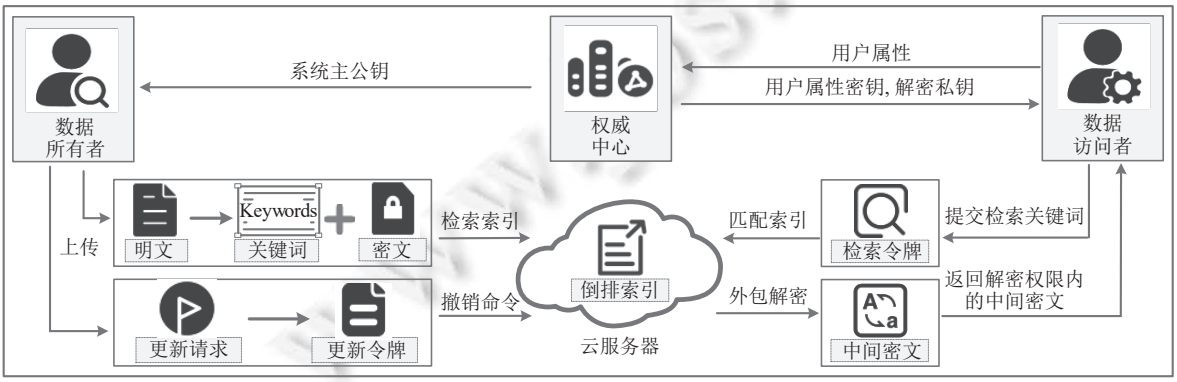


图 4 系统模型图

(3) 数据所有者: 云端加密数据的提供方, 可信实体. 负责生成需要复用的复用策略、处理关键词和加密明文并发送给云服务器. 当数据所有者需要更新数据在云端的索引时, 生成更新令牌发送给云服务器.

(4) 数据访问者: 云端加密数据的使用方, 不完全可信. 负责生成检索令牌发送给云服务器进行相应的检索. 只有当数据访问者的属性符合检索结果中密文的访问策略时, 云服务器才能对该加密数据进行属性外包解密, 并返回相应的中间密文. 数据访问者使用解密私钥对中间密文进行少量计算得到最终的明文.

3.2 算法定义

方案涉及的主要算法如下.

(1) $Setup(1^{\lambda},U,V)\rightarrow(MSK,MPK)$: 系统初始化算法由权威中心 CA 执行. 输入安全参数 λ, 属性集合 U, 伪属性集合 V. 输出系统的主公钥 MPK, 主私钥 MSK.

(2) $KeyGen(Y,DU_{\mathrm{id}},MPK,MSK)\rightarrow(UK,TK)$: 数据访问者的密钥生成算法由权威中心 CA 执行. 输入数据访问者的属性集合 Y、唯一标识号 DU_{id}、主公钥 MPK、主私钥 MSK. 输出属性密钥 UK 和解密私钥 TK.

(3) $SPGen(\Lambda_{\mathrm{sub}},DO_{\mathrm{id}},MPK)\rightarrow(SP_{\mathrm{Tag}}^{DO},SP_{\mathrm{Tag}}^{CSP})$: 复用策略生成算法由数据所有者 DO 执行. 输入复用策略 Λ_{sub}, 数据所有者的唯一标识号 DO_{id}, 主公钥 MPK. 输出复用策略的标识符 SP_{Tag}^{DO} 和 SP_{Tag}^{CSP}, 其中 SP_{Tag}^{DO} 由数据所有者在本地保存, SP_{Tag}^{CSP} 上传到云服务器.

(4) $Encrypt(m,MPK,\Lambda,SP_{\mathrm{Tag}}^{DO})\rightarrow(CT_m^{\delta},\{KW_k\}_{k\in K})$: 加密算法由数据所有者 DO 执行. 输入明文数据 m, 主公钥 MPK, 访问策略 Λ, 复用策略标识符 SP_{Tag}^{DO}. 输出密文 CT_m^{δ} 和关键词集合 $\{KW_k\}_{k\in K}$.

(5) $IndexGen(CT_m^{\delta},\{KW_k\}_{k\in K},MPK)\rightarrow Index^{\mathrm{Invert}}$: 索引生成算法由云服务器 CSP 执行. 输入数据所有者上传的密文 CT_m^{δ}, 关键词集合 $\{KW_k\}_{k\in K}$, 主公钥 MPK. 输出检索索引 $Index^{\mathrm{Invert}}$.

(6) $TrapGen(MPK,UK,\{sw_s\}_{s\in S})\rightarrow Q_s$: 检索陷门生成算法由数据访问者 DU 执行. 输入主公钥 MPK、数据访问者的属性密钥 UK 和检索关键词集合 $\{sw_s\}_{s\in S}$. 输出检索陷门 Q_s.

(7) $Search(MPK,Q_s,Index^{\mathrm{Invert}})\rightarrow SR$: 检索算法由云服务器 CSP 执行. 输入系统主公钥 MPK, 检索陷门 Q_s, 检索索引 $Index^{\mathrm{Invert}}$. 输出检索结果 SR.

(8) $Decrypt_1(SR,Q_s,SP_{\mathrm{Tag}}^{CSP},MPK)\rightarrow CT_m$: 外包解密算法由云服务器 CSP 执行. 输入检索结果 SR, 检索陷门 Q_s, 复用策略的标识符 SP_{Tag}^{CSP}, 主公钥 MPK. 输出中间密文 CT_m.

$Decrypt_2(CT_m,TK)\rightarrow m$: 本地解密算法由数据访问者 DU 执行. 输入解密私钥 TK, 中间密文 CT_m. 输出明文 m.

(9) $UTGen(m',MPK,DO_{\mathrm{id}})\rightarrow UT$: 更新令牌生成算法由数据所有者 DO 执行. 输入明文消息 m', 主公钥 MPK, 数据所有者标识号 DO_{id}. 输出更新令牌 UT.

(10) $ReIndex(MPK,UT,CT_m^{\delta},Index^{\mathrm{Invert}})\rightarrow UR$: 索引更新算法由云服务器 CSP 执行. 输入系统主公钥 MPK, 更新令牌 DK, 密文 CT_m^{δ}, 检索索引 $Index^{\mathrm{Invert}}$. 输出更新结果 UR.

3.3 方案安全性定义

本文方案的云服务器作为部署在组织内部的专有计算环境, 在检索和外包解密算法中提供安全可靠的运算支持. 在此基础上, 考虑以下 3 种类型的恶意敌手. 第 1 类敌手以窃取解密私钥为目标, 试图破解某个数据访问者泄露的属性密钥以获取相应的解密私钥, 称其为敌手 $\mathcal{A}_1$. 第 2 类敌手通过选择性地控制提交给数据所有者加密的明文, 并分析从云服务器获得的中间解密结果, 试图推导出外包解密算法的有用信息, 称其为敌手 $\mathcal{A}_2$. 第 3 类敌手通过伪造特定数据的更新令牌来非法篡改云端索引, 称其为敌手 $\mathcal{A}_3$. 通过描述这 3 类敌手与对应模拟器之间的安全游戏给出本文方案的安全性定义.

(1) 解密私钥的不可计算性

通过模拟器 $\mathcal{B}$ 和敌手 $\mathcal{A}_1$ 之间的安全游戏描述本文方案中的解密私钥具有不可计算的安全性.

初始化阶段: 模拟器 $\mathcal{B}$ 接收来自敌手 $\mathcal{A}_1$ 定义的挑战属性集合 Y^*.

系统建立阶段: 模拟器 $\mathcal{B}$ 执行 $Setup(\cdot)$ 算法得到系统主公钥 MPK 和系统主私钥 MSK, 将主公钥 MPK 发送给敌手 $\mathcal{A}_1$.

询问阶段 1: 模拟器 $\mathcal{B}$ 构造一个初始为空的询问列表 T, 一个空字典 D 及一个计数器 $j=0$. 敌手 $\mathcal{A}_1$ 进行下列自适应查询, 该过程可以重复多项式次.

属性密钥查询: 敌手 $\mathcal{A}_1$ 向模拟器 $\mathcal{B}$ 提交自己的标识号 $\mathcal{A}id$ 和一个属性集合 Y, 其中 $Y \neq Y^*$. 当 Y 在询问列表 T 中时, 将组合 $(j, \mathcal{A}id, Y, UK_Y, TK_Y)$ 返回给敌手 $\mathcal{A}_1$. 否则, 模拟器 $\mathcal{B}$ 执行 $KeyGen(\cdot)$ 算法, 生成相应的属性密钥 UK_Y 和解密私钥 TK_Y, 将属性密钥 UK_Y 发送给敌手 $\mathcal{A}_1$. 并将组合 $(j, \mathcal{A}id, Y, UK_Y, TK_Y)$ 添加到询问列表 T, 计数器 $j=j+1$.

腐败私钥查询: 敌手 $\mathcal{A}_1$ 对询问列表 T 中的第 i 项查询解密私钥, 模拟器 $\mathcal{B}$ 将组合 $(i, \mathcal{A}id, Y, UK_Y, TK_Y)$ 返回给敌手 $\mathcal{A}_1$, 更新字典 $D=D\cup Y$. 若询问列表 T 中第 i 项不存在则算法终止.

挑战阶段: 模拟器 $\mathcal{B}$ 对挑战属性集合 Y^* 运行 $KeyGen(\cdot)$ 算法, 生成相应的属性密钥 UK_{Y^*} 和解密私钥 TK_{Y^*}, 并将属性密钥 UK_{Y^*} 发送给敌手 $\mathcal{A}_1$.

询问阶段 2: 与询问阶段 1 相同, 但不允许对属性集合 Y 进行重复询问.

输出阶段: 敌手 $\mathcal{A}_1$ 输出与属性密钥 UK_{Y^*} 相关的解密私钥 TK'_{Y^*}, 若 $TK'_{Y^*}=TK_{Y^*}$, 则敌手 $\mathcal{A}_1$ 赢得游戏.

定义 1. 对于任意概率多项式时间内的敌手 $\mathcal{A}_1$, 若 $\mathcal{A}_1$ 输出正确解密私钥的概率是可忽略的, 则称该方案具有解密私钥不可计算性.

(2) 中间密文不可区分性

通过模拟器 $\mathcal{B}$ 和敌手 $\mathcal{A}_2$ 之间的安全游戏描述中间密文在选择明文攻击下具有不可区分性.

初始化阶段: 敌手 $\mathcal{A}_2$ 定义一个挑战访问策略 Λ^* 发送给模拟器 $\mathcal{B}$.

系统建立阶段: 模拟器 $\mathcal{B}$ 执行 $Setup(\cdot)$ 算法得到系统主公钥 MPK 和系统主私钥 MSK, 将系统主公钥 MPK 发送给敌手 $\mathcal{A}_2$.

询问阶段 1: 敌手 $\mathcal{A}_2$ 进行下列自适应查询, 该过程可以重复多项式次.

属性密钥查询: 敌手 $\mathcal{A}_2$ 向模拟器 $\mathcal{B}$ 提交自己的标识号 $\mathcal{A}id$ 和一个属性集合 Y, 其中 $Y \nvDash \Lambda^*$. 当 Y 在询问列表 T 时, 将组合 $(j, \mathcal{A}id, Y, UK_Y, TK_Y)$ 返回给敌手 $\mathcal{A}_2$. 否则, 模拟器 $\mathcal{B}$ 执行 $KeyGen(\cdot)$ 算法, 生成相应的属性密钥 UK_Y 和解密私钥 TK_Y, 并将属性密钥 UK_Y 发送给敌手 $\mathcal{A}_2$.

中间密文查询: 敌手 $\mathcal{A}_2$ 向模拟器 $\mathcal{B}$ 提交一个明文 m、访问策略 $\Lambda \neq \Lambda^*$ 和一个满足 Λ 的属性密钥 UK_Y. 模拟器 $\mathcal{B}$ 根据访问策略 Λ 对明文 m 执行 $Encrypt(\cdot)$ 得到密文 CT_m^{δ}, 而后使用属性密钥 UK_Y 对密文 CT_m^{δ} 执行外包解密 $Decrypt_1(\cdot)$ 得到中间密文 CT_m. 最后将中间密文 CT_m 发送给敌手 $\mathcal{A}_1$.

挑战阶段: 敌手 $\mathcal{A}_2$ 将两个相同长度的明文消息 m_0, m_1 和一个属性集合 Y^* 提交给模拟器 $\mathcal{B}$, 其中 $Y^* \vDash \Lambda^*$. 模拟器 $\mathcal{B}$ 对属性集合 Y^* 生成相应的属性密钥 UK_{Y^*}, 而后随机选取一个明文消息 m_ς, $\varsigma \in \{0,1\}$ 用挑战访问策略 Λ^* 执行加密算法 $Encrypt(\cdot)$ 得到密文 $CT_{m\varsigma}^{\delta}$, 接着使用属性密钥 UK_{Y^*} 执行 $Decrypt_1(\cdot)$ 得到挑战中间密文 $CT_{m\varsigma}$. 最后, 模拟器 $\mathcal{B}$ 将挑战中间密文 $CT_{m\varsigma}$ 发送给敌手 $\mathcal{A}_2$.

询问阶段 2: 与询问阶段 1 相同, 但不允许对属性集合 Y 和明文 m 进行重复询问.

猜测阶段: 敌手 $\mathcal{A}_2$ 输出对 ς 的猜测结果 ς'. 若 $\varsigma'=\varsigma$, 则敌手 $\mathcal{A}_2$ 以 $Adv_{\mathcal{A}_2}=|\Pr[\varsigma'=\varsigma]-1/2|=\varepsilon$ 的优势赢得该游戏.

定义 2. 对于任意概率多项式时间敌手 $\mathcal{A}_2$, 若 $\mathcal{A}_2$ 赢得游戏优势的概率是可忽略的, 则称方案在选择明文攻击下具有中间密文不可区分安全性.

(3) 更新令牌的不可伪造性

初始化阶段: 模拟器 $\mathcal{B}$ 执行 $Setup(\cdot)$ 算法得到系统主公钥 MPK 和系统主私钥 MSK, 将系统主公钥 MPK 发送给敌手 $\mathcal{A}_3$.

哈希询问阶段: 敌手 $\mathcal{A}_3$ 在该阶段最多进行 qH 次询问, 模拟器 $\mathcal{B}$ 随机选择 $k \in \{1,2,\ldots,qH\}$ 作为对敌手 $\mathcal{A}_3$ 伪造更新令牌的猜测值. 构造一个初始为空的询问列表 Q_{list} 存放敌手 $\mathcal{A}_3$ 的询问实例及结果. 若 $i \neq k$ 则计算 H_{m_i} 回复

给敌手 $\mathcal{A}_3$ 并将 $(i,m_i,H_{m_i},\{DW_{d_i}\}_{d_i\in D})$ 添加到询问列表 Q_{list} 中, 若 $i=k$ 则将 H_{m_k} 回复给敌手 $\mathcal{A}_3$.

更新令牌请求阶段: 敌手 $\mathcal{A}_3$ 在该阶段对选定的消息 m_i 请求生成相应的更新令牌, 若 $i=k$ 则算法终止. 当 $i\neq k$ 时, 模拟器 $\mathcal{B}$ 从询问列表 Q_{list} 中取出 m_i 对应的 $(i,m_i,H_{m_i},\{DW_{d_i}\}_{d_i\in D})$, 计算得到更新令牌 $DK(i)=\{FileMatch(i),E^{\delta}_{m_i},\{DW_{d_i}\}_{d_i\in D}\}$ 返回给敌手 $\mathcal{A}_3$.

更新令牌伪造阶段: 敌手 $\mathcal{A}_3$ 选择一个在上个阶段中未查询过的消息 m_j 进行更新令牌的伪造, 当 $j\neq k$ 时算法终止, 当 $j=k$ 时计算相应伪造的更新令牌 $DK(j)=\{FileMatch(j),E^{\delta}_{m_j},\{DW_{d_j}\}_{d_j\in D}\}$.

定义 3. 对于任意概率多项式时间内的敌手 $\mathcal{A}_3$, 若 $\mathcal{A}_3$ 输出合法更新令牌的概率是可忽略的, 则称方案的更新令牌不可伪造性.

4 算法设计及安全性分析

4.1 方案具体设计

(1) $Setup(1^{\lambda},U,V)\rightarrow(MSK,MPK)$: 初始化算法

输入系统安全参数 λ, 属性域 U, 伪属性域 V. 输出双线性映射群 $e:G\times G=G_T$, 其中 G, G_T 为 p 阶乘法循环群, 生成元 $g\in G$, p 为大素数. 对真实属性域中的每个属性 $u\in U$ 选取 $\alpha_u\in_R\mathbb{Z}_p^*$, 对伪属性域中的每个伪属性 $v\in V$ 选取 $\beta_v\in_R\mathbb{Z}_p^*$. 选取抗碰撞的哈希函数 $H_1:\{0,1\}^*\rightarrow\mathbb{Z}_p^*$, $H_2:\{0,1\}^*\rightarrow G$, $H_3:\{0,1\}^*\times G\rightarrow\mathbb{Z}_p^*$, 一组相互独立的抗碰撞的哈希函数 $H=\{h_1,h_2,\ldots,h_t\}$, 密码安全的哈希函数 $\mathcal{H}_1$, 均匀的散列哈希函数 $\mathcal{H}_2$. 选取 b, $c\in_R\mathbb{Z}_p^*$, 计算系统主公钥 $MPK=\{G,G_T,e:G\times G=G_T,g,p,H_1,H_2,H_3,H,\mathcal{H}_1,\mathcal{H}_2,g^b,e(g,g)^c,\{g^{\alpha_u}\}_{u\in U},\{g^{\beta_v}\}_{v\in V}\}$ 和系统主私钥 $MSK=\{b,c,\alpha_u,\beta_v\}_{u\in U,v\in V}$.

(2) $KeyGen(Y,DU_{id},MPK,MSK)\rightarrow(UK,TK)$: 数据访问者的密钥生成算法

设数据访问者的属性集合为 $y\in Y$, 唯一标识号 $DU_{\mathrm{id}}\in\{0,1\}^*$. 权威中心随机选取 $h\in_R G$, $z'\in_R\mathbb{Z}_p^*$, 计算 $K_0=H_3(DU_{\mathrm{id}},h)=z$, $K_1=g^{\alpha_y\cdot z'}_{y\in Y}$, $K_2=g^{c/(b\cdot z)}$, $K_3=g^{b\cdot z'}$, $K_4=g^{z'}$, 通过函数集合 $H=\{h_1,h_2,\ldots,h_t\}$ 将属性集合 Y 映射到布隆过滤器的位数组中得到 $K_5=BF_Y$. 最后将属性密钥 $UK=\{K_1,K_2,K_3,K_4,K_5\}$ 和解密私钥 $TK=K_0$ 通过安全信道返回给数据访问者.

(3) $SPGen(\Lambda_{\mathrm{sub}},DO_{\mathrm{id}},MPK)\rightarrow(SP^{DO}_{\mathrm{Tag}},SP^{CSP}_{\mathrm{Tag}})$: 复用策略生成算法

数据所有者计算 $\vartheta=H_1(DO_{\mathrm{id}})\rightarrow\mathbb{Z}_p^*$, 其中 DO_{id} 为数据所有者的唯一标识号. 对复用策略 Λ_{sub} 进行析取得到 J 个满足复用策略的真子句, 将真子句的属性映射到布隆过滤器的位数组中得到 $BF_{j,j\in J}$. 对每个真子句计算 $l_{i,j}=g^{\vartheta\cdot(\alpha_{1,j}+\alpha_{2,j}+\ldots+\alpha_{i,j})}$, $i\in I$ 为每个真子句中包含的属性数量. 随后从系统主公钥 MPK 中随机选取一个伪属性值 g^{β_x} 计算 $T_{\mathrm{tag}}=g^{\beta_x}\cdot g^{b\cdot\vartheta\cdot FR_v}$, FR_v 的初始值为 1, 仅当数据访问者的复用策略验证通过时, 将 FR_v 的值置为 0. 最后, 数据所有者将复用策略标识符 $SP^{CSP}_{\mathrm{Tag}}=\{BF_j,l_{i,j}\}_{i\in I,j\in J}$ 上传至云服务器, 将 $SP^{DO}_{\mathrm{Tag}}=\{T_{\mathrm{tag}},g^{\beta_x}\}$ 存储在本地.

(4) $Encrypt(m,MPK,\Lambda,SP^{DO}_{\mathrm{Tag}})\rightarrow(CT^{\delta}_m,\{KW_k\}_{k\in K})$: 加密算法

数据所有者首先对明文数据 m 提取关键词集合 $\{kw_k\}_{k\in K}$, 计算 $\{KW_k=\mathcal{H}_1(kw_k)\}_{k\in K}$. 设访问结构树 Λ 的叶子节点集合为 $x\in X$, 包含根节点在内的非叶子节点为阈值节点 $n\in N$. 根据 Λ 中是否需要包含复用策略, 将加密过程分为两种情况.

① 不包含复用策略

对访问结构树中的每个节点 υ 选取一个阶数为 $o_\upsilon=k_\upsilon-1$ 的随机多项式 q_υ, 其中 k_υ 为 q_υ 的门限值. 选取根节点秘密值 $r'\in_R\mathbb{Z}_p^*$, 即 $q_{r(0)}=r'$. 对访问结构树自上而下分发根节点秘密值 r', 节点 υ 对应的秘密值为 $q_{\upsilon(0)}=q_{parent(\upsilon)}{}^{index(\upsilon)}$, 其中 $index(\upsilon)$ 为节点 υ 在其父节点 $parent(\upsilon)$ 处的位序. 选取 $r\in_R\mathbb{Z}_p^*$, 输入消息 $m\in\{0,1\}^*$, 随机选取一个安全的对称密钥 CK 计算: $M=Enc_{CK}(m)$, $E_m=CK\cdot e(g,g)^{c\cdot r\cdot r'}$, $\mathfrak{h}=H_2(DO_{\mathrm{id}})\rightarrow G$, $H_m=H_3(m,\mathfrak{h})\rightarrow\mathbb{Z}_p^*$, $FileMatch=g^{H_m}$, $E_0=g^r$, $E_1=g^{b\cdot r}$, $E_4=g^{\vartheta}$, 对 $\forall x\in X$ 有: $E_2=g^{b\cdot r\cdot q_{x(0)}}$, $E_3=g^{\alpha_x\cdot r}$. 得到密文: $CT^{\delta}_m=\{\Lambda,M,E_m,FileMatch,E_0,E_1,E_2,E_3,E_4\}$. 并将 CT^{δ}_m 和 $\{KW_k\}_{k\in K}$ 发送给云服务器.

② 包含复用策略

若需要包含的复用策略未定义则执行算法 $STGen(\cdot)$.

如果复用策略已经生成过, 数据所有者从本地取出该复用策略的标识符 $SP_{\mathrm{Tag}}^{DO}=\{T_{\mathrm{tag}},g^{\beta_x}\}$, 并将其中的组件 $T_{\mathrm{tag}}=g^{\beta_x}\cdot g^{b\cdot\vartheta\cdot FR_v}$ 嵌入密文. 此时复用策略对应的子树被伪属性 χ 替代, 访问策略 Λ 被转换成 Λ', 具体过程可参考第 2.4 节. 将转换访问策略 Λ' 的叶子节点集合记为 $X'=(X-\tau)\cup\chi$, τ 为复用策略的属性集合. 对 X' 中包含伪属性和真实属性在内的节点, 计算相应的组件 $E_2'=g^{r\cdot q_{x(0)}}\cup g^{r\cdot q_{x(0)}}$, $E_3'=g^{\alpha_x\cdot r}\cup(T_{\mathrm{tag}})^r$, 其余组件的计算过程同步骤①中一致. 最后得到密文: $CT_m^{\delta}=\{\Lambda',M,E_m,FileMatch,E_0,E_1,E_2',E_3',E_4,T_{\mathrm{tag}}\}$, 数据所有者将 CT_m^{δ} 和 $\{KW_k\}_{k\in K}$ 发送给云服务器.

(5) $IndexGen(CT_m^{\delta},\{KW_k\}_{k\in K},MPK)\rightarrow Index^{\mathrm{Invert}}$: 索引生成算法

云服务器根据关键词集合 $\{KW_k\}_{k\in K}$ 计算各个关键词在哈希表中的位置 $[L_k]_{k\in K}=\mathcal{H}_2(KW_k)_{k\in K}$. 并按照接收 CT_m^{δ} 的时间顺序对其生成一个唯一的标识号 $FileID$, 而后从 CT_m^{δ} 中取出验证组件 $FileMatch=g^{H_m}$. 将 $<FileID,FileMatch>$ 放到关键词对应的跳表上, 形成 $FileID$ 有序的索引结构 $Index^{\mathrm{Invert}}$.

(6) $TrapGen(MPK,UK,\{sw_s\}_{s\in S})\rightarrow Q_s$: 检索陷门生成算法

数据访问者对检索关键词集合 $\{sw_s\}_{s\in S}$ 计算 $\{SW_s=\mathcal{H}_1(sw_s)\}_{s\in S}$, 形成检索陷门 $Q_s=\{UK,\{SW_s\}_{s\in S}\}$ 发送给云服务器. 检索陷门中的多个关键词可以用布尔逻辑进行连接, 例如对 3 个检索关键词求检索数据的交集: $\{SW_1\ \&\ SW_2\ \&\ SW_3\}$, 或对两个检索关键词求检索数据的并集 $\{SW_2\ \|SW_4\}$ 等.

(7) $Search(MPK,Q_s,Index^{\mathrm{Invert}})\rightarrow SR$: 检索算法

云服务器计算 $\mathcal{H}_2(SW_s)$ 得到每个检索关键词 SW_s 在哈希表中的位置. 对该位置上存放的 KW_k 和检索关键词 SW_s 进行等值比较, 相等则返回相应的加密数据编号. 根据检索请求中关键词的逻辑连接方式, 对所有检索关键词的数据编号集合 $\{FileID_s\}_{s\in S}$ 进行交集或并集的计算, 得到检索结果 SR.

(8) $Decrypt_1(SR,Q_s,SP_{\mathrm{Tag}}^{CSP},MPK)\rightarrow CT_m$: 外包解密算法

云服务器根据检索结果 SR 中的加密数据编号 $FileID$ 提取相应的密文 CT_m^{δ}, 并基于检索令牌 Q_s 中数据访问者的属性密钥 UK 对 CT_m^{δ} 进行外包解密. 根据 CT_m^{δ} 中是否包含 T_{tag}, 将叶子节点的计算分成以下两种情况.

① 密文中不包含 T_{tag}, 即访问策略中不包含复用策略.

对访问策略中的叶子节点 $x\in X$ 执行 $DecLeaf(CT_m^{\delta},UK,x)$ 解密算法: $D_x=e(K_1,E_1)\cdot e(K_2,E_2)/e(K_3,E_3)=e(g,g)^{\alpha_y\cdot z'\cdot b\cdot r}\cdot e(g,g)^{c\cdot r\cdot q_{x(0)}/z}/e(g,g)^{b\cdot z'\cdot r\cdot\alpha_x}$, 对属性相同的节点有 $D_x=e(g,g)^{c\cdot r\cdot q_{x(0)}/z}$.

② 密文中包含 T_{tag}, 即访问策略中包含复用策略.

对普通属性叶子节点的计算同步骤①. 当按序计算到组件 $E_2'=g^{r\cdot q_{x(0)}}\cup g^{r\cdot q_{x(0)}}$ 和 $E_3'=g^{\alpha_x\cdot r}\cup(T_{\mathrm{tag}})^r$ 的最后一个元素时, 云服务器首先根据密文组件中的 T_{tag} 找到对应的 $SP_{\mathrm{Tag}}^{CSP}=\{BF_j,l_{i,j}\}_{i\in I,j\in J}$ 并判定数据访问者是否满足复用策略. 该判定过程分为两步, 首先基于布隆过滤器进行初步判定, 具体过程如第 2.4 节所示. 而后对通过布隆过滤器判定的真子句 $l_{i,j}=g^{\vartheta\cdot(\alpha_{1,j}+\alpha_{2,j}+\ldots+\alpha_{i,j})}$ 和 UK 基于双线性配对进行二次验证, 计算 $e(K_4,l_{i,j})/e(K_1,E_4)=e(g^{z'},g^{\vartheta\cdot(\alpha_{1,j}+\alpha_{2,j}+\ldots+\alpha_{i,j})})/e(g^{\alpha_y\cdot z'},g^{\vartheta})\stackrel{?}{=}1$. 若该等式成立则返回 $FR_v=0$, 此时 E_3' 中 $(T_{\mathrm{tag}})^r=g^{\beta_x\cdot r}$. 由解密算法 $DecLeaf(CT_m^{\delta},UK,\chi)$ 计算得到伪属性叶子节点的秘密值 $D_{\chi}'=e(g,g)^{c\cdot r\cdot q_{x(0)}/z}$. 若判定失败则返回 $FR_v=1$, 此时在 E_3' 中 $(T_{\mathrm{tag}})^r=g^{\beta_x\cdot r}\cdot g^{b\cdot\vartheta\cdot r}$, $D_{\chi'}=e(g,g)^{c\cdot r\cdot q_{x(0)}/z}/e(g,g)^{b\cdot z'\cdot b\cdot\vartheta\cdot r}$, 存在不能消除的组件 $e(g,g)^{b\cdot z'\cdot b\cdot\vartheta\cdot r}$ 无法向上递归重构其父节点的秘密值.

对上述两种情况中包含伪属性在内满足阈值条件的叶子节点, 通过拉格朗日插值函数重构其父节点的秘密值. 假设阈值节点 n 的子节点 x 集合为 n_x, 对访问策略中阈值节点的计算如下: 将 x 在 κ 处的次序记为 $\kappa=index(x_{\kappa})$, $\mathcal{K}_n=\{index(x_{\kappa}),x\in n_x\}$. 执行 $DecNode(CT_m^{\delta},UK,n)$ 算法, 将输出结果记为 D_n. 当集合 $\mathcal{K}_n$ 不存在时 $D_n=\perp$, 反之计算: $D_n=\prod_{x\in n_x}\left(e(g,g)^{c\cdot r\cdot q_{parent(x)}^{index(x_{\kappa})}/z}\right)^{\Delta_{\kappa,\mathcal{K}_n(0)}}=\prod_{x\in n_x}e(g,g)^{c\cdot r\cdot q_n(\kappa)\cdot\Delta_{\kappa,\mathcal{K}_n(0)}/z}=e(g,g)^{c\cdot r\cdot q_n(0)/z}$. 递归计算到根节点时得到 $D_r=e(g,g)^{c\cdot r\cdot r'/z}$. 最后将中间密文 $CT_m=\{E_m,D_r,M\}$ 发送给数据访问者.

若数据访问者不满足密文的访问策略, 则无法恢复访问控制树的根节点秘密值, 外包解密失败. 此时, 云服务器中止对外包解密算法且不向数据访问者返回该密文的任何信息.

(9) $Decrypt_2(CT_m,TK)\rightarrow m$: 本地解密算法

数据访问者收到中间密文 $CT_m=\{E_m,D_r,M\}$ 后用自己的解密私钥 $TK=K_0=z$ 计算得到对称密钥 $CK=E_m/(D_r)^{TK}=CK\cdot e(g,g)^{c\cdot r\cdot r'}/(e(g,g)^{c\cdot r\cdot r'/z})^z$. 接着使用对称密钥 CK 解密得到明文数据 $m=Dec_{CK}(M)$.

(10) $UTGen(m',MPK,DO_{\mathrm{id}}) \to UT$: 更新令牌生成算法

当数据所有者要更新云端加密数据 m' 中 i 个关键词对应的索引时, 首先对这 i 个原关键词 $\{kw_i\}_{i\in I}$ 和目标关键词集合 $\{rw_i\}_{i\in I}$ 计算 $KW_i=\{\mathcal{H}_1(kw_i)\}_{i\in I}$, $RW_i=\{\mathcal{H}_1(rw_i)\}_{i\in I}$. 与 BLS 签名相关的组件: $H_{m'}=H_3(m',\mathfrak{h})\to\mathbb{Z}_p^*$, $FileMatch'=g^{H_{m'}}$, $E_{m'}^{\delta}=(FileMatch')^{\vartheta}=g^{H_{m'}\cdot\vartheta}$, 其中 $\mathfrak{h}=H_2(DO_{\mathrm{id}})\to G$, $\vartheta=H_1(DO_{\mathrm{id}})\to\mathbb{Z}_p^*$. 得到更新令牌 $UT=\{\{KW_i\}_{i\in I},\{RW_i\}_{i\in I},FileMatch',E_{m'}^{\delta}\}$ 发送给云服务器.

(11) $ReIndex(MPK,UT,CT_m^{\delta},Index^{\mathrm{Invert}})\to UR$: 索引更新算法

云服务器收到索引更新请求后, 首先判定数据所有者对更新令牌 UT 中要被替换的原关键词集合 $KW_i=\{\mathcal{H}_1(kw_i)\}_{i\in I}$ 的更新权限: 计算各个原关键词 KW_i 在索引中的位置 $L[KW_i]=\mathcal{H}_2(KW_i)$, 在相应跳表中查找与更新令牌中匹配组件 $FileMatch'$ 相等的加密数据匹配组件 $FileMatch$, 取出 $FileMatch$ 对应的密文 CT_m^{δ} 得到组件 $E_4=g^{\theta}$ 并进行等价判定: $e(g,E_{m'}^{\delta})\stackrel{?}{=}e(FileMatch,E_4)$, 当且仅当该等式成立, 权限验证完成. 否则算法中止. 对包含多个原关键词 KW_i 的同一密文, 权限验证操作仅需执行一次. 而后计算各个新插入的目标关键词 RW_i 在索引中的位置 $L[RW_i]=\mathcal{H}_2(RW_i)$ 修改相应的跳表指针, 并将组件 $<FileID,FileMatch>$ 按 $FileID$ 的顺序加入目标关键词的索引. 若该目标关键词不存在, 则创建一个新的索引条目. 最后返回成功结果 $UR=1$ 或中止结果 $UR=\perp$.

4.2 正确性分析

对于检索到的密文, 云服务器基于数据访问者的属性密钥执行外包解密算法 $Decrypt_1(\cdot)$. 假设数据访问者满足密文中的访问策略, 根据密文中是否包含复用策略的标识符分两种情况进行分析.

(1) 不包含复用策略

对访问结构树的叶子节点有:

$$D_{leaf}=e(K_1,E_1)\cdot e(K_2,E_2)/e(K_3,E_3)=e(g^{\alpha_y\cdot z'},g^{b\cdot r})\cdot e(g^{c/(b\cdot z)},g^{r\cdot q_{x(0)}})/e(g^{b\cdot z'},g^{r\cdot\alpha_x})$$

$$=e(g,g)^{\alpha_y\cdot z'\cdot b\cdot r}\cdot e(g,g)^{c\cdot r\cdot q_{x(0)}/z}/e(g,g)^{b\cdot z'\cdot r\cdot\alpha_x}=e(g,g)^{c\cdot r\cdot q_{x(0)}/z}.$$

对阈值节点有: $D_n=\prod_{x\in n_x}\left(e(g,g)^{c\cdot r\cdot q_{parent(x)}^{index(x_k)}/z}\right)^{\Delta_{k,\mathcal{K}_n(0)}}=\prod_{x\in n_x}e(g,g)^{c\cdot r\cdot q_{n(k)}\cdot\Delta_{k,\mathcal{K}_n(0)}/z}=e(g,g)^{c\cdot r\cdot q_n(0)/z}$, 递归计算到根节点时得 $D_r=e(g,g)^{c\cdot r\cdot r'/z}$. 云服务器将中间密文 $CT_m=\{M,E_m,D_r\}$ 发送给数据访问者.

数据访问者执行解密算法 $Decrypt_2(\cdot)$, 用解密私钥 $TK=z$ 还原对称密钥 $CK=E_m/(D_r)^{TK}=CK\cdot e(g,g)^{c\cdot r\cdot r'}/(e(g,g)^{c\cdot r\cdot r'/z})^z$, 最后再用对称密钥 CK 计算得到明文数据 $m=Dec_{CK}(M)$.

(2) 包含复用策略

对普通叶子节点的计算过程同上.

对于伪属性叶子节点, 首先需要判定数据访问者的属性是否满足该节点对应的复用策略: 云服务器根据密文中标识组件 T_{tag} 找到关联的 $SP_{\mathrm{Tag}}^{CSP}=\{BF_j,l_{i,j}\}_{i\in I,j\in J}$, 将子句布隆过滤器 BF_J 与数据访问者属性密钥中的 C_5 组件进行等位比较. 若存在某个子句布隆过滤器 BF_j 的位数组是 K_5 组件的位数组的子集 $BF_j\subseteq BF_Y$, 则取出该子句的属性值组件 $l_{i,j}=g^{\vartheta\cdot(\alpha_{1,j}+\alpha_{2,j}+\ldots+\alpha_{i,j})}$ 并与数据访问者属性密钥中的 $K_1=g^{\alpha_y\cdot z'}{}_{y\in Y}$, $K_4=g^{z'}$ 及密文中的 $E_4=g^{\theta}$ 组件进行判定 $e(K_4,l_{i,j})\stackrel{?}{=}e(K_1,E_4)$. 若该等式成立则判定数据访问者满足伪属性叶子节点对应的复用策略, 云服务器返回 $FR_v=0$. 此时 $(T_{\mathrm{tag}})^r=g^{\beta_x\cdot r}\cdot g^{b\cdot\vartheta\cdot FR_v\cdot r}=g^{\beta_x\cdot r}$, 计算得到伪属性叶子节点的秘密分量 $D_{leaf'}=e(K_1,E_1)\cdot e(K_2,E_2')/e(K_3,E_3')=e(g^{\beta_x\cdot z'},g^{b\cdot r})\cdot e(g^{c/(b\cdot z)},g^{b\cdot r\cdot q_{x(0)}})/e(g^{b\cdot z'},g^{\alpha_x\cdot r})=e(g,g)^{\beta_x\cdot z'\cdot b\cdot r}\cdot e(g,g)^{c\cdot r\cdot q_{x(0)}/z}/e(g,g)^{b\cdot z'\cdot\alpha_x\cdot r}=e(g,g)^{c\cdot r\cdot q_{x(0)}/z}$. 后续计算过程同 (1) 中一致.

5 安全性分析

5.1 解密私钥的不可计算性

若离散对数问题 (DLP) 成立, 则本文方案的解密私钥具有不可计算性.

证明: 假定存在一个敌手 $\mathcal{A}_1$ 在概率多项式时间内能计算出解密私钥, 则可以构造出一个模拟器 $\mathcal{B}$ 在概率多项式时间内解决 DLP 问题.

初始化阶段: 选择满足双线性映射 $e: G\times G=G_T$, 阶为素数 p 的乘法循环群 G, G_T, 生成元 $g\in G$. 模拟器 $\mathcal{B}$ 接收一个 DL 问题实例 $(g,g^{1/z})\in G$ 和一个敌手 $\mathcal{A}_1$ 定义的挑战属性集合 Y^*.

系统建立: 模拟器 $\mathcal{B}$ 对属性域中的每个属性 $u\in U$ 选取 $\alpha_u\in_R\mathbb{Z}_p^*$, 对伪属性域中的每个伪属性 $v\in V$ 选取 $\beta_v\in_R\mathbb{Z}_p^*$. 选取抗碰撞的哈希函数 $H_1:\{0,1\}^*\to\mathbb{Z}_p^*$, $H_2:\{0,1\}^*\to G$, $H_3:\{0,1\}^*\times G\to\mathbb{Z}_p^*$, 随机元素 b, $c\in_R\mathbb{Z}_p^*$, 计算得到主公钥 $MPK=\{G,G_T,e:G\times G=G_T,g,p,H_1,H_2,H_3,g,g^b,g^c,\mathcal{G}=g^{b/c},g^{1/z},\{g^{\alpha_u}\}_{u\in U},\{g^{\beta_v}\}_{v\in V}\}$ 发送给敌手 $\mathcal{A}_1$, 自己保存主私钥 $MSK=\{b,c,\alpha_u,\beta_v\}_{u\in U,v\in V}$.

询问阶段 1: 模拟器 $\mathcal{B}$ 构造一个初始为空的询问列表 T, 一个空字典 D 以及一个计数器 $j=0$. 敌手 $\mathcal{A}_1$ 进行下列自适应查询, 该查询过程可以重复多项式有界次.

属性密钥询问: 敌手 $\mathcal{A}_1$ 向模拟器 $\mathcal{B}$ 提交自己的标识符 $\mathcal{A}id$ 和一个属性集合 Y, 其中 $Y\neq Y^*$. 若 Y 在询问列表 T 中则将相应的组合 $(j,\mathcal{A}id,Y,UK_Y,TK_Y)$ 返回给敌手 $\mathcal{A}_1$. 否则, 模拟器 $\mathcal{B}$ 执行 $KeyGen(\cdot)$ 算法, 选取 $h\in_R G$, $z'\in_R\mathbb{Z}_p^*$, 计算 $K_0=H_3(\mathcal{A}id,h)=z^*$, $K_1=g_{y\in Y^*}^{\alpha_y\cdot z'}$, $K_2=g^{c/(b\cdot z^*)}$, $K_3=g^{b\cdot z'}$, $K_4=g^{z'}$, $K_5=BF_{y,y\in Y^*}$. 将属性密钥 $UK_Y=\{K_1,K_2,K_3,K_4,K_5\}$ 发送给敌手 $\mathcal{A}_1$, 并将组合 $(j,\mathcal{A}id,Y,UK_Y,TK_Y)$ 添加至询问列表 T 中, 计数器 $j=j+1$.

腐败私钥查询: 敌手 $\mathcal{A}_1$ 对询问列表 T 中的第 i 项查询解密私钥, 模拟器 $\mathcal{B}$ 将 $(i,\mathcal{A}id,Y,UK_Y,TK_Y)$ 中的解密私钥 $TK_Y=K_0$ 返回给敌手 $\mathcal{A}_1$, 并在字典 D 中添加属性集合 Y. 若询问列表 T 中第 i 项不存在则该算法终止.

挑战阶段: 模拟器 $\mathcal{B}$ 用 DL 实例 $(g,g^{1/z})\in G$ 对挑战属性集合 Y^* 生成属性密钥 $UK_{Y^*}=\{K_1,K_2,K_3,K_4,K_5\}$ 发送给敌手 $\mathcal{A}_1$, 令 $\mathcal{G}=g^{b/c}$ 则 $K_1=\mathcal{G}_{y\in Y}^{b\cdot\alpha_y\cdot z'/c}$, $K_2=(g^{1/z})^{c/b}=\mathcal{G}^{1/z}$, $K_3=\mathcal{G}^{b\cdot b\cdot z'/c}$, $K_4=\mathcal{G}^{b\cdot z'/c}$, $K_5=BF_{y,y\in Y}$, $z'\in_R\mathbb{Z}_p^*$.

询问阶段 2: 与询问阶段 1 相同, 但不允许对属性集合 Y 进行重复询问.

输出阶段: 敌手 $\mathcal{A}_1$ 输出属性密钥 UK_{Y^*} 相关的解密私钥 TK'_{Y^*}, 若 $TK'_{Y^*}=TK_{Y^*}$ 则敌手 $\mathcal{A}_1$ 赢得上述游戏.

若在概率多项式时间内敌手 $\mathcal{A}_1$ 输出正确的解密私钥 $TK'_{Y^*}=TK_{Y^*}=z$, 则模拟器 $\mathcal{B}$ 可以利用敌手 $\mathcal{A}_1$ 的输出得到 $1/z$ 从而解决 DLP 问题 $(g,g^{1/z})\in G$. 由于 DLP 问题在概率多项式时间内不存在有效解, 故而不存在概率多项式时间的敌手 $\mathcal{A}_1$ 能以不可忽略的优势赢得上述游戏, 因此可以认为本文方案中数据访问者的解密私钥具有不可计算的安全性.

5.2 中间密文的不可区分性

若 DBDH 假设成立, 则本文方案的中间密文在选择明文攻击下具有不可区分安全性.

证明: 假设存在一个敌手 $\mathcal{A}_2$ 能在概率多项式时间内以不可忽略的优势 ε 区分出中间密文, 则可以构造一个模拟器 $\mathcal{B}$ 能在概率多项式时间内以 $\varepsilon/2$ 的优势解决 DBDH 困难假设.

初始化阶段: 选择满足双线性映射 $e: G\times G=G_T$, 阶为素数 p 的乘法循环群 G, G_T, 生成元 $g\in G$, 随机元素 c, r, r', $\gamma\in_R\mathbb{Z}_p^*$. 模拟器 $\mathcal{B}$ 接收一个挑战元组 $(g,g^c,g^r,g^{r'},T_\varsigma)$, 其中 $\varsigma\in_R\{0,1\}$. 当 $\mu=1$ 时, $T_\varsigma=T_1=e(g,g)^{c\cdot r\cdot r'}$. 当 $\mu=0$ 时, $T_\varsigma=T_0=e(g,g)^{\gamma}$. 敌手 $\mathcal{A}_2$ 定义一个挑战访问策略 Λ^* 发送给模拟器 $\mathcal{B}$.

系统建立阶段: 模拟器 $\mathcal{B}$ 对属性域中的每个属性 $u\in U$ 选取 $\alpha_u\in_R\mathbb{Z}_p^*$, 对伪属性域中的每个伪属性 $v\in V$ 选取 $\beta_v\in_R\mathbb{Z}_p^*$. 选取 $b\in_R\mathbb{Z}_p^*$, 抗碰撞的哈希函数 $H_1:\{0,1\}^*\to\mathbb{Z}_p^*$, $H_2:\{0,1\}^*\to G$, $H_3:\{0,1\}^*\times G\to\mathbb{Z}_p^*$. 得主公钥 $MPK=\{G,G_T,e:G\times G=G_T,g,p,H_1,H_2,H_3,g^b,g^c,g^r,g^{r'},\{g^{\alpha_u}\}_{u\in U},\{g^{\beta_v}\}_{v\in V}\}$, 主私钥 $MSK=\{b,\alpha_u,\beta_v\}_{u\in U,v\in V}$, 模拟器 $\mathcal{B}$ 将主公钥 MPK 发送给敌手 $\mathcal{A}_2$, 自己保存主私钥 MSK.

询问阶段 1: 敌手 $\mathcal{A}_2$ 进行下列自适应查询, 该查询过程可以重复多项式有界次.

属性密钥询问: 敌手 $\mathcal{A}_2$ 向模拟器 $\mathcal{B}$ 提交自己的标识符 $\mathcal{A}id$ 和一个属性集合 Y, 其中 $Y\not\models\Lambda^*$. 模拟器 $\mathcal{B}$ 执行 $KeyGen(\cdot)$ 算法, 选取 $h\in_R G$, $z'\in\mathbb{Z}_p^*$, 计算 $K_0=H_3(\mathcal{A}id,h)=z^*$, $K_1=g_{y\in Y^*}^{\alpha_y\cdot z'}$, $K_2=g^{c/(b\cdot z^*)}$, $K_3=g^{b\cdot z'}$, $K_4=g^{z'}$, $K_5=BF_{y,y\in Y^*}$, 得到属性密钥 $UK_Y=\{K_1,K_2,K_3,K_4,K_5\}$ 并发送给敌手 $\mathcal{A}_2$.

中间密文查询: 敌手 $\mathcal{A}_2$ 向模拟器 $\mathcal{B}$ 提交一个明文 m、访问策略 $\Lambda\neq\Lambda^*$ 和一个满足 Λ 的属性密钥 UK_Y. 模拟器 $\mathcal{B}$ 为自己生成一个标识号 $\mathcal{B}_{\mathrm{id}}$, 对明文消息 m 执行 $Encrypt(\cdot)$ 算法. 随机选取 $\mathfrak{r}$, $\mathfrak{r}'\in_R\mathbb{Z}_p^*$ 和一个安全的对称密钥 CK 计算得到: $M=Enc_{CK}(m)$, $E_m=CK\cdot e(g,g)^{c\cdot\mathfrak{r}\cdot\mathfrak{r}'}$, $\mathfrak{h}=H_2(\mathcal{B}_{\mathrm{id}})\to G$, $H_m=H_3(m,\mathfrak{h})\to\mathbb{Z}_p^*$, $FileMatch=g^{H_m}$, $E_0=g^{\mathfrak{r}}$, $E_1=g^{b\cdot\mathfrak{r}}$, $\vartheta=H_1(\mathcal{B}_{\mathrm{id}})$, $E_4=g^{\vartheta}$, 对访问策略 Λ 中的叶节点计算: $E_2=g^{b\cdot\mathfrak{r}\cdot q_{x(0)}}$, $E_3=g^{\alpha_x\cdot\mathfrak{r}}$, 得到密文: $CT_m^{\delta}=\{\Lambda,M,E_m,FileMatch,$

$E_0, E_1, E_2, E_3, E_4\}$.

模拟器 $\mathcal{B}$ 根据属性密钥 UK_Y 对密文 CT_m^{δ} 执行外包解密算法 $Decrypt_1(\cdot)$: 对叶子节点 x 计算 $D_{leaf} = e(K_1, E_1)\cdot e(K_2, E_2)/e(K_3, E_3) = e(g,g)^{c\cdot\tau\cdot q_{x(0)}/z^*}$.

对内部节点计算 $D_n = \prod_{x\in n_x}\left(e(g,g)^{c\cdot r\cdot q_{parent(x)}^{index(x_k)}/z}\right)^{\Delta_{k,Kn(0)}} = e(g,g)^{c\cdot\tau\cdot q_{n(0)}/z^*}$, 递归到根节点时得 $D_r = e(g,g)^{c\cdot\tau\cdot r'/z^*}$. 最后模拟器 $\mathcal{B}$ 将中间密文 $CT_m = \{E_m, D_r, M\}$ 发送给敌手 $\mathcal{A}_2$.

挑战阶段: 敌手 $\mathcal{A}_2$ 将两个相同长度的明文消息 m_0, m_1 和一个属性集合 Y^*, 其中 $Y^* \models \Lambda^*$ 提交给模拟器 $\mathcal{B}$. 模拟器 $\mathcal{B}$ 对属性集合 Y 执行 $KeyGen(\cdot)$ 算法, 生成相应的属性密钥 UK_{Y^*}, 而后随机选取一个明文消息 $m_{\varsigma,\varsigma\in\{0,1\}}$ 用挑战元组 $(g, g^c, g^r, g^{r'}, T_\mu)$ 执行加密算法 $Encrypt(\cdot)$. 选取一个安全的对称密钥 CK 计算: $M_\varsigma = Enc_{CK}(m_\varsigma)$, $E_{m\varsigma} = CK\cdot T_\mu$, $\mathfrak{h} = H_2(\mathcal{B}_{\mathrm{id}}) \to G$, $H_{m_\varsigma} = H_3(m_\varsigma, \mathfrak{h}) \to \mathbb{Z}_p^*$, $FileMatch = g^{H_{m\varsigma}}$, $E_0 = g^r$, $E_1 = g^{b\cdot r}$, $\vartheta = H_1(\mathcal{B}_{\mathrm{id}})$, $E_4 = g^{\vartheta}$, 对属性节点 x 有: $E_2 = g^{b\cdot r\cdot q_{x(0)}}$, $E_3 = g^{\alpha_x\cdot r}$, 得到密文 $CT_{m\varsigma}^{\delta} = \{\Lambda, M, E_{m_\varsigma}, FileMatch\}$, 接着使用属性密钥 UK_Y 对密文 $CT_{m\varsigma}^{\delta}$ 执行外包解密算法 $Decrypt_1(\cdot)$ 得到 $D_r = e(g,g)^{c\cdot r\cdot r'/z}$. 最后, 模拟器 $\mathcal{B}$ 将挑战中间密文 $CT_{m\varsigma} = \{E_{m\varsigma}, D_r, M_\varsigma\}$ 返回给敌手 $\mathcal{A}_2$.

询问阶段 2: 与询问阶段 1 相同, 但不允许对明文 m 和属性集合 Y 进行重复询问.

猜测阶段: 敌手 $\mathcal{A}_2$ 输出对 ς 的猜测结果 $\varsigma' \in \{0,1\}$, 若 $\varsigma' = \varsigma$ 则判定敌手 $\mathcal{A}_2$ 赢得上述游戏.

当 $\mu = 0$ 时, 挑战中间密文 $CT_{m\varsigma}$ 在群 G_T 上具有随机性. 敌手 $\mathcal{A}_2$ 无法得到关于 m_ς 的任何信息, 只能随机猜测 ς' 的值, 因而赢得游戏的概率为 $\Pr[\varsigma' = \varsigma|\varsigma = 0] = 1/2$. 当 $\mu = 1$ 时, $CT_{m\varsigma}^{\delta}$ 是由 DBDH 实例进行的有效加密. 相较于随机猜测, 敌手 $\mathcal{A}_2$ 能更加准确地猜测出 ς 的值, 将这个优势设为 ε, 则敌手 $\mathcal{A}_2$ 赢得游戏的概率为 $\Pr[\varsigma' = \varsigma|\varsigma = 1] = \varepsilon + 1/2$. 此时模拟器 $\mathcal{B}$ 可以利用敌手 $\mathcal{A}_2$ 的输出 ς' 对挑战元组 $(g, g^c, g^r, g^{r'}, T_\varsigma)$ 中的 ς 进行判断, 从而以 $Adv_{\mathcal{B}} = \Pr[\varsigma = \varsigma\prime] - 1/2 = 1/2\cdot(1/2) + 1/2\cdot(1/2+\varepsilon) - 1/2 = \varepsilon/2$ 的优势在概率多项式时间内解决 DBDH 困难问题. 由于在概率多项式时间内不存在有效的判定算法验证 DBDH 假设成立, 故而不存在概率多项式时间的敌手 $\mathcal{A}_2$ 能以不可忽略的优势破坏上述方案的安全性, 即本文方案的中间密文在选择明文攻击下具有不可区分安全性.

5.3 更新令牌的不可伪造性

当云服务器收到数据所有者上传的对明文 m' 的更新令牌 $UT = \{\{KW_i\}_{i\in I}, \{RW_i\}_{i\in I}, FileMatch', E_{m'}^{\delta}\}$ 后, 需要对 UT 中的 BLS 签名组件 $E_{m'}^{\delta} = g^{H_{m'}\cdot\vartheta}$ 和云端加密数据的组件 $FileMatch = g^{H_m}$ 和 $E_4 = g^{\vartheta}$ 进行双线性配对运算, 完成对数据所有者更新权限的验证 $e(g, E_{m'}^{\delta}) \stackrel{?}{=} e(FileMatch, E_4)$. 将 BLS 签名组件 $E_{m'}^{\delta}$ 的不可伪造性归约到 CDH 困难问题上: 若存在一个概率多项式时间的敌手 $\mathcal{A}_3$ 能以概率 ε 伪造出 m' 的签名组件 $E_{m'}^{\delta}$, 则可以构造出一个模拟器 $\mathcal{B}$ 以 ε/qH 的概率解决 CDH 问题.

初始化: 选择满足双线性映射 $e: G\times G = G_T$, 阶为素数 p 的乘法循环群 G, 生成元 $g \in G$, 将一个 CDH 问题实例 $(g, g^a, g^b) \in G$ 发送给模拟器 $\mathcal{B}$. 模拟器 $\mathcal{B}$ 将主公钥 $MPK\{G, G_T, e, g, p, H_1, H_2, H_3, g^a, g^b\}$ 发送给敌手 $\mathcal{A}_3$, 将主私钥设为 $MSK = \vartheta = a$ 自己保存.

哈希询问阶段: 敌手 $\mathcal{A}_3$ 在该阶段最多进行 qH 次询问, 模拟器 $\mathcal{B}$ 随机选择 $k \in \{1, 2, \ldots, qH\}$ 作为对敌手 $\mathcal{A}_3$ 伪造更新令牌中签名组件的猜测值. 构造一个初始为空的询问列表 Q_{list} 存放敌手 $\mathcal{A}_3$ 的询问实例及结果, 将敌手 $\mathcal{A}_3$ 的第 i 次询问的消息实例记为 m_i. 当 $i \neq k$ 时, 计算 $\mathfrak{h} = H_2(\mathcal{A}id) \to G$, $H_{m_i} = H_3(m_i, \mathfrak{h})$, 得到组件 $FileMatch(i) = g^{H_{m_i}}$ 回复给敌手 $\mathcal{A}_3$, 并将 $(i, m_i, H_{m_i}, FileMatch(i), \{DW_{d_i}\}_{d_i\in D})$ 添加到询问列表 Q_{list} 中, 其中 $\mathcal{A}id$ 为敌手 $\mathcal{A}_3$ 的标识号. 当 $i = k$ 时, 计算得到 $FileMatch(k) = g^{H_{m_k}}\cdot g^b$ 回复给敌手 $\mathcal{A}_3$.

更新令牌请求阶段: 敌手 $\mathcal{A}_3$ 在该阶段对选定的消息 m_i 请求生成相应的更新令牌, 若 $i = k$ 则算法终止. 当 $i \neq k$ 时, 模拟器 $\mathcal{B}$ 从询问列表 Q_{list} 中取出 m_i 对应的 $(i, m_i, H_{m_i}, FileMatch(i), \{KW_r\}_{r\in R}, \{RW_r\}_{r\in R})$, 得到相应的更新令牌 $UT(i) = \{FileMatch(i), E_{m_i}^{\delta} = (g^a)^{H_{m_i}} = g^{H_{m_i}\cdot\vartheta}, \{KW_r\}_{r\in R}, \{RW_r\}_{r\in R}\}$ 返回给敌手 $\mathcal{A}_3$.

更新令牌伪造阶段: 敌手 $\mathcal{A}_3$ 选择一个上个阶段中未向模拟器 $\mathcal{B}$ 查询过的消息 m_j 进行更新令牌的伪造, 当 $j \neq k$ 时算法终止, 否则 $UT(j) = \{FileMatch(j), E_{m_j}^{\delta} = FileMatch(k)^{\vartheta} = (g^{H_{m_k}+b})^a = (g^a)^{H_{m_k}+b}, \{KW_r\}_{r\in R}, \{RW_r\}_{r\in R}\}$. 此时模拟器 $\mathcal{B}$ 可以通过敌手 $\mathcal{A}_3$ 的伪造更新令牌组件 $E_{m_j}^{\delta}$ 计算 $E_{m_j}^{\delta}/(g^a)^{H_{m_j}}$ 得到 $g^{a\cdot b}$, 从而解决 CDH 困难问题. 这与 CDH

问题的困难性相矛盾, 因为已知 CDH 问题在概率多项式时间内不存在有效解, 即模拟器 $\mathcal{B}$ 无法找到一个私钥 ϑ 使得 $MSK=\vartheta=a$ 成立并据此解决 CDH 问题, 故而本方案的更新令牌具有不可伪造性.

5.4 其他安全性

(1) 检索安全性

当数据访问者基于关键词进行加密数据的检索时, 云服务器不会将检索结果中的密文直接返回给数据访问者, 而是根据检索令牌中数据访问者的属性密钥对这些密文进行外包解密操作. 当且仅当数据访问者的属性与密文的访问策略相匹配, 云服务器才将密文转化为中间密文并发送给数据访问者.

若数据访问者的属性不满足访问策略则认定其无权限访问该密文, 并继续对检索结果中的其他密文执行外包解密算法, 直至对检索结果中所有密文都完成判定. 通过这种检索方式, 云服务器最终仅向数据访问者返回其解密权限内的中间密文, 对于无权限的密文, 云服务器不提供任何信息. 从而保证了数据访问者对云端数据的合法访问, 维护了云端数据的安全和隐私性.

(2) 策略复用安全性

回顾第 2.4 节, 构造子句布隆过滤器旨在提升密文和数据访问者的属性匹配效率, 该判定过程不具备相关安全性. 我们假设云服务器不会篡改存储内容, 则策略复用的安全性取决于能否阻止数据访问者对属性密钥进行非法组合. 本文方案中数据访问者的属性密钥中, 与属性判定有关的组件为 $K_1=g^{\alpha_y\cdot z'}$ $(y\in Y)$ 和 $K_4=g^{z'}$, 由于未授权的数据访问者无法得到权威中心为每位数据访问者分配的随机数 $z'\in_R\mathbb{Z}_p^*$, 故而无法有效地进行合谋攻击.

6 性能分析

6.1 理论分析

将本文方案和 EABSE[1]、FAME[10]、VMSE[12]进行对比分析, 表 2 列出了系统初始化、属性密钥生成、加解密、检索、更新索引等主要算法的计算开销. 其中, G, G_T 代表群上指数运算, C, C_T 代表群上的乘法运算, P 为配对运算, CK 为对称加密运算, f 为伪随机函数, H 为对关键词的哈希运算. n 为权威中心的数量, a 为访问控制策略的属性数量, l 为 LSSS 矩阵的列, b 为属性密钥的属性数量, s 为属性密钥中满足访问策略的属性数量. K 为系统中的所有关键词, κ 为更新关键词数量, n 为与检索关键词关联的加密数据的数量.

表 2 计算开销对比

方案	密钥生成	加密算法	检索开销	外包解密	本地解密	索引更新
EABSE[1]	$(b+1)G+2bC+bC_T$	$4G+(a+2)C+CK$	$(2+3s)P+G_T+3C_T$	—	CK	—
FAME[10]	$(9b+9)G+(8b+9)C$	$(12al+6a)C+6aG$	—	—	$(6s+3)G+6G_T+6P$	—
VMSE[12]	$G+2bf$	$(2a+k+4)G+C+CK$	$3P+(K+1)C$	—	CK	$(\kappa+s+1)P+\kappa H$
本文	$(b+7)G$	$(2a+4)G+2G_T+C_T+CK$	$2H$	$3sP+2sC_T$	G_T+C_T+CK	$2\kappa H+G+2P$

密钥生成算法中, VMSE[12]对属性的计算只有伪随机函数且仅支持与门策略, 因而开销最小. 其余方案的计算开销都随属性数量线性增加, 本文的属性密钥仅用于外包解密, 计算复杂度相对较低. 在加密算法中, EABSE[1]、VMSE[12]和本文方案都采用了对明文数据进行对称加密后再对密钥进行属性加密的密钥封装技术, 加密计算开销随访问控制策略中属性数量的增加而增大. 快速加密方案 FAME[10]对明文拆分加密, 实现了高效的加密和数据传输操作. 属性数量相等条件下, 单门限加密方案 FAME[10]的开销最小. 本文方案的策略复用算法允许制定访问策略时重复使用某些策略, 加密开销会随着复用策略中属性数量的增多而下降.

检索部分对比了基于双线性配对进行检索的常规方案 EABSE[1]和 VMSE[12], 可以看到常规检索方案的开销较大. 本文方案对云端加密数据的检索不依赖于双线性配对运算而是基于哈希运算, 降低了检索开销. 在解密算法中, EABSE[1]和 VMSE[12]在检索时就完成了属性判定, 因而访问者仅需对检索结果中的加密数据执行对称解密算法. 本文采用了外包解密的方式将大部分计算任务移植到了云服务器上, 降低了数据访问者的本地解密计算开销.

VMSE[12]和本文方案支持对检索索引的有效更新, 但更新操作都受限于首次上传时确定的关键词数量. VMSE[12]方案通过双线性配对运算匹配新旧关键词集合实现更新索引, 更新后的索引只保留了新旧关键词的交集部分, 无法引入新的关键词. 本文方案基于 BLS 签名对数据所有者进行权限验证, 若验证通过则将原有关键词替换为目标关键词从而实现索引更新. 对于包含多个替换关键词的同一密文, 仅需执行一次验证操作.

6.2 实验分析

实验平台为 Intel® Core™ i5-13600KF 处理器、16 GB 内存、Windows 11 操作系统. 选用 $y^2 = x^3 + ax + b$ 构造的 A 类椭圆曲线, 用 Java 语言调用 JPBC 库编程实现各个方案, 在安然电子邮件数据集[33]上对相关算法执行 30 次, 取平均值作为最终实验结果, 以下是具体的实验分析.

初始化算法的执行时间如图 5(a) 所示. 对 EABSE[1]进行 4 倍缩放, 可以看到 EABSE[1]和本文方案初始化算法的执行时间随着属性域的大小呈线性增长趋势. 方案 FAME[10]和 VMSE[12]初始化算法的运行时间则近似恒定, 可支持大属性域. 属性密钥生成算法的执行时间如图 5(b) 所示, 可以看到由于 VMSE[12]只用伪随机函数处理各个属性, 密钥生成时间最少, 但只支持与门策略. 其他方案的属性密钥的生成时间都与属性数量正相关, EABSE[1]由于在初始化算法中对所有属性预计算了一部分组件值, 故而属性密钥的生成开销同样较小. 加密算法和解密算法的运行对比时间如图 5(c) 所示. 在数据集中随机选择一个明文数据进行加密, 可以看到 EABSE[1]和 VMSE[12]采用了对称加密的方式, 加密开销近似恒定, VMSE[12]需要对明文数据计算签名用于后续执行关键词更新操作因而开销略大. 将方案 FAME[10]的加密时间缩放 10 倍, 可以看到该方案加密耗时与访问策略中属性数量线性相关. 在解密算法中, 方案 FAME[10]执行双线性配对的次数不依赖属性的数量, 解密开销趋于一个定值. VMSE[12]和 EABSE[1]在处理明文数据时同样采用了对称加密算法, 计算开销相对较小.

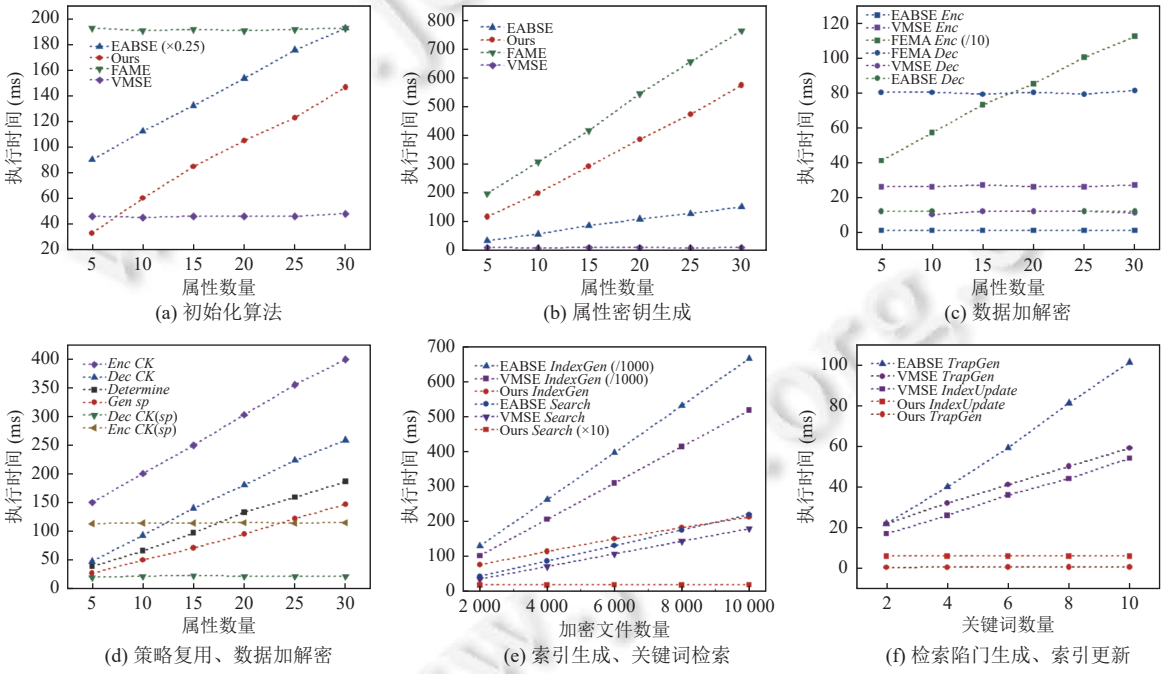


图 5 算法性能对比

本文提出了一种复用访问策略减少加密计算的方法, 对相关加解密操作的计算效率进行独立比较分析. 如图 5(d) 所示, 给出了访问策略中属性数量一致时, 使用常规属性加密算法加密对称密钥 *Enc CK* 和使用复用策略加密对称密钥的运行时间 *Enc CK*(*sp*) 对比, 使用常规属性加密算法解密对称密钥 *Dec CK* 和使用复用策略解密对称密钥的运行时间 *Dec CK*(*sp*) 对比以及复用策略生成时间 *Gen sp* 和复用策略判定时间 *Determine*. 可以看到采用

复用策略方式的加密时间更短, 且复用策略生成后可以重复用于加密后续数据. 当复用策略中包含全部加密属性时, 加密算法和解密算法的执行时间近似恒定开销.

索引生成和检索操作相关算法的执行时间如图 5(e) 所示, 对每个数据随机选取 10 个关键词, 而后随机选取关键词在索引中进行检索. EABSE[1]和 VMSE[12]都是对关键词都是执行一次群上的指数运算后绑定随机数, 并结合哈希函数和双线性配对运算判定检索关键词和索引中的关键词是否相等, 完成相应的检索操作. 其中, VMSE[12]通过隐藏向量加密 (hidden vector encryption, HVE) 算法生成数据访问者的属性集和密文的访问策略, 遍历索引匹配得到满足访问策略的密文, 再基于双线性配对运算逐一验证检索关键词和密文中的关键词, 降低了检索时访问属性判定操作的计算开销. 而 EABSE[1]则采用了传统的双线性配对运算验证访问属性及关键词, 计算开销较大. 本文采用哈希表实现对检索关键词的快速定位, 通过直接访问目标关键词对应的跳表底层数据以获取完整匹配结果而无需遍历全部索引, 同时支持检索关键词的布尔逻辑组合. 将方案 EABSE[1]和 VMSE[12]的索引生成运行时间缩放 1 000 倍与本文方案进行对比, 可以看到在加密文件数量较多时本文方案索引生成和检索操作的执行时间最少. 检索陷门生成和索引更新算法执行时间如图 5(f) 所示, 本文方案在陷门生成时只涉及高效的哈希运算, 因而开销最小. EABSE[1]和 VMSE[12]生成检索陷门时需要在群上计算用于检索配对的关键词组件, 因而随着关键词数量的增多, 运行时间也随之增大. VMSE[12]和本文支持对索引中的关键词进行更新, VMSE[12]在生成更新索引时至少需要与云服务器进行 3 轮交互, 且更新时间与替换的关键词数量线性相关. 本文方案更新索引是非交互式的, 对于包含了多个替换关键词的同一密文, 只需要执行一次基于双线性配对的权限验证操作.

通过上述对比实验可以得出本文方案在索引更新和检索效率方面具有一定的优势, 在经常使用访问策略的场景下, 支持数据所有者通过复用加密策略的方式更加高效地处理数据. 通过将解密过程外包给云服务器, 大幅降低了数据访问者在本地的解密时间. 此外, 本文还设计了一种支持更新操作的倒排索引结构, 能够在较低的开销下实现对加密数据的快速检索和索引更新.

7 总 结

为了提高数据的加解密效率、实现高效灵活且支持索引更新的可搜索属性加密方案, 本文提出了一种云服务器辅助计算的高效可搜索属性加密方案. 首先, 为了减少数据所有者的加密计算开销, 考虑对经常使用的访问策略进行复用, 实现对后续数据的快速加密操作. 其次, 结合哈希表和跳表设计了一种倒排索引结构, 哈希表用于存储关键词实现快速定位, 对应的跳表中顺序存放相关的加密数据标号和验证组件, 在检索时依次取出. 该方法对单关键词检索的时间复杂度为对数级, 对多关键词检索的时间复杂度为次线性级. 检索速度只取决于与关键词相关的数据量, 无需遍历整个索引. 再次, 为了实现数据访问者对检索索引的安全更新, 引入 BLS 短签名验证数据所有者的相关权限, 仅当验证通过后才修改相应的跳表指针进行更新操作. 此外, 我们将大部分属性解密计算外包到了云服务器上, 数据访问者仅需在本地完成少量计算就能得到明文. 最后, 在 DLP、DBDH、CDH 困难假设下分别针对不同类型的敌手证明了本文方案的安全性, 并在真实数据集上进行了相关测试. 实验表明本文方案具备高效的检索性能, 可以准确执行更新索引操作, 在复用策略时能有效缩短数据加密和解密时间.

References

[1] Guo WF, Dong XL, Cao ZF, Shen JC. Efficient attribute-based searchable encryption on cloud storage. Journal of Physics: Conf. Series, 2018, 1087(5): 052001. [doi: 10.1088/1742-6596/1087/5/052001]

[2] Song DX, Wagner D, Perrig A. Practical techniques for searches on encrypted data. In: Proc. of the 2000 IEEE Symp. on Security and Privacy. Berkeley: IEEE, 2000. 44–55. [doi: 10.1109/SECPRI.2000.848445]

[3] Xu L, Zheng LQ, Xu CZ, Yuan XL, Wang C. Leakage-abuse attacks against forward and backward private searchable symmetric encryption. In: Proc. of the 2023 ACM SIGSAC Conf. on Computer and Communications Security. Copenhagen: Association for Computing Machinery, 2023. 3003–3017. [doi: 10.1145/3576915.3623085]

[4] Kornaropoulos EM, Moyer N, Papamanthou C, Psomas A. Leakage inversion: Towards quantifying privacy in searchable encryption. In: Proc. of the 2022 ACM SIGSAC Conf. on Computer and Communications Security. Los Angeles: ACM, 2022. 1829–1842. [doi: 10.1145/

3548606.3560593]

[5] Oya S, Kerschbaum F. IHOP: Improved statistical query recovery against searchable symmetric encryption through quadratic optimization. In: Proc. of the 31st USENIX Security Symp. Boston: USENIX Association, 2022. 2407–2424.

[6] Ning JT, Huang XY, Poh GS, Yuan JM, Li YJ, Weng J, Deng RH. LEAP: Leakage-abuse attack on efficiently deployable, efficiently searchable encryption with partially known dataset. In: Proc. of the 2021 ACM SIGSAC Conf. on Computer and Communications Security. New York: ACM, 2021. 2307–2320. [doi: 10.1145/3460120.3484540]

[7] Bossuat A, Bost R, Fouque PA, Minaud B, Reichle M. SSE and SSD: Page-efficient searchable symmetric encryption. In: Malkin T, Peikert C, eds. Proc. of the 41st Advances in Cryptology. Cham: Springer, 2021. 157–184. [doi: 10.1007/978-3-030-84252-9_6]

[8] Goyal V, Pandey O, Sahai A, Waters B. Attribute-based encryption for fine-grained access control of encrypted data. In: Proc. of the 13th ACM Conf. on Computer and Communications Security. Alexandria: ACM, 2006. 89–98. [doi: 10.1145/1180405.1180418]

[9] Bethencourt J, Sahai A, Waters B. Ciphertext-policy attribute-based encryption. In: Proc. of the 2007 IEEE Symp. on Security and Privacy. Berkeley: IEEE, 2007. 321–334. [doi: 10.1109/SP.2007.11]

[10] Agrawal S, Chase M. FAME: Fast attribute-based message encryption. In: Proc. of the 2017 ACM SIGSAC Conf. on Computer and Communications Security. Dallas: ACM, 2017. 665–682. [doi: 10.1145/3133956.3134014]

[11] Wang CJ, Li WT, Li Y, Xu XL. A ciphertext-policy attribute-based encryption scheme supporting keyword search function. In: Wang GJ, Ray I, Feng DG, Rajarajan M, eds. Proc. of the 5th Cyberspace Safety and Security. Cham: Springer, 2013. 377–386. [doi: 10.1007/978-3-319-03584-0_28]

[12] Liang YR, Li YP, Zhang K, Wu ZQ. VMSE: Verifiable multi-keyword searchable encryption in multi-user setting supporting keywords updating. Journal of Information Security and Applications, 2023, 76: 103518. [doi: 10.1016/j.jisa.2023.103518]

[13] Miao YB, Ma JF, Liu XM, Li XH, Jiang Q, Zhang JW. Attribute-based keyword search over hierarchical data in cloud computing. IEEE Trans. on Services Computing, 2020, 13(6): 985–998. [doi: 10.1109/TSC.2017.2757467]

[14] Miao YB, Tong QY, Deng RH, Choo KKR, Liu XM, Li HW. Verifiable searchable encryption framework against insider keyword-guessing attack in cloud storage. IEEE Trans. on Cloud Computing, 2022, 10(2): 835–848. [doi: 10.1109/TCC.2020.2989296]

[15] Liang YR, Li YP, Zhang K, Ma LN. DMSE: Dynamic multi-keyword search encryption based on inverted index. Journal of Systems Architecture, 2021, 119: 102255. [doi: 10.1016/j.sysarc.2021.102255]

[16] Yin H, Zhang W, Deng H, Qin Z, Li KQ. An attribute-based searchable encryption scheme for cloud-assisted IIoT. IEEE Internet of Things Journal, 2023, 10(12): 11014–11023. [doi: 10.1109/JIOT.2023.3242964]

[17] Zhang YH, Zhu T, Guo R, Xu SM, Cui H, Cao J. Multi-keyword searchable and verifiable attribute-based encryption over cloud data. IEEE Trans. on Cloud Computing, 2023, 11(1): 971–983. [doi: 10.1109/TCC.2021.3119407]

[18] Guo WP, Shi RH, Zhang XX. Comment on “Multi-keyword searchable and verifiable attribute-based encryption over cloud data”. IEEE Trans. on Cloud Computing, 2023, 11(4): 3797–3798. [doi: 10.1109/TCC.2023.3312918]

[19] Huang QL, Wei QL, Yan GY, Zou L, Yang YX. Fast and privacy-preserving attribute-based keyword search in cloud document services. IEEE Trans. on Services Computing, 2023, 16(5): 3348–3360. [doi: 10.1109/TSC.2023.3265270]

[20] Zhang K, Zhang Y, Li YP, Liu XM, Lu LF. A blockchain-based anonymous attribute-based searchable encryption scheme for data sharing. IEEE Internet of Things Journal, 2024, 11(1): 1685–1697. [doi: 10.1109/JIOT.2023.3290975]

[21] Chen LQ, Xu SG, Zhang H, Weng J. Fair-and-exculpable-attribute-based searchable encryption with revocation and verifiable outsourced decryption using smart contract. IEEE Internet of Things Journal, 2025, 12(4): 4302–4317. [doi: 10.1109/JIOT.2024.3484227]

[22] Miao YB, Li F, Li XH, Liu ZQ, Ning JT, Li HW, Choo KKR, Deng RH. Time-controllable keyword search scheme with efficient revocation in mobile E-health cloud. IEEE Trans. on Mobile Computing, 2024, 23(5): 3650–3665. [doi: 10.1109/TMC.2023.3277702]

[23] Feng T, Chen LQ, Fang JL, Shi JM. Blockchain data sharing scheme based on localized difference privacy and attribute-based searchable encryption. Journal on Communications, 2023, 44(5): 224–233 (in Chinese with English abstract). [doi: 10.11959/j.issn.1000-436x.2023103]

[24] Niu SF, Song M, Fang LZ, Wang CF. Cloud storage data sharing based on attribute encryption in smart healthcare. Journal of Electronics & Information Technology, 2022, 44(1): 107–117 (in Chinese with English abstract). [doi: 10.11999/JEIT210858]

[25] Wang JW, Ning JT, Xu SM, Yin XC, Chen HX. Searchable attribute-based encryption scheme for dynamic user groups. Ruan Jian Xue Bao/Journal of Software, 2023, 34(4): 1907–1925 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/6698.htm [doi: 10.13328/j.cnki.jos.006698]

[26] Xu LL, Chen XF, Zhang FG, Li WH, Wu HT, Tang SH, Xiang Y. ASBKS: Towards attribute set based keyword search over encrypted personal health records. IEEE Trans. on Dependable and Secure Computing, 2021, 18(6): 2941–2952. [doi: 10.1109/TDSC.2020.2970928]

[27] Lee J, Oh J, Kwon D, Kim M, Kim K, Park Y. Blockchain-enabled key aggregate searchable encryption scheme for personal health record sharing with multidelegation. IEEE Internet of Things Journal, 2024, 11(10): 17482–17494. [doi: 10.1109/JIOT.2024.3357802]

[28] Zhang BY, Yang WJ, Zhang FT, Ning JT. Efficient attribute-based searchable encryption with policy hiding over personal health records. IEEE Trans. on Dependable and Secure Computing, 2025, 22(2): 1299–1312. [doi: 10.1109/TDSC.2024.3432769]

[29] Niu SF, Hu Y, Zhou SW, Shao HL, Wang CF. Attribute-based searchable encryption in edge computing for lightweight devices. IEEE Systems Journal, 2023, 17(3): 3503–3514. [doi: 10.1109/JSYST.2023.3283389]

[30] Rao YS, Prasad S, Bera S, Das AK, Susilo W. Boolean searchable attribute-based signcryption with search results self-verifiability mechanism for data storage and retrieval in clouds. IEEE Trans. on Services Computing, 2024, 17(4): 1382–1399. [doi: 10.1109/TSC.2023.3327816]

[31] Meng F, Cheng LX, Wang MQ. ABDKS: Attribute-based encryption with dynamic keyword search in fog computing. Frontiers of Computer Science, 2021, 15(5): 155810. [doi: 10.1007/s11704-020-9472-7]

[32] Zou YP, Peng T, Wang GJ, Luo ET, Xiong JB. Blockchain-assisted multi-keyword fuzzy search encryption for secure data sharing. Journal of Systems Architecture, 2023, 144: 102984. [doi: 10.1016/j.sysarc.2023.102984]

[33] Enron Corporation. Enron Email Dataset. 2002. https://www.cs.cmu.edu/~./enron/

附中文参考文献

[23] 冯涛, 陈李秋, 方君丽, 石建明. 基于本地化差分隐私和属性基可搜索加密的区块链数据共享方案. 通信学报, 2023, 44(5): 224–233. [doi: 10.11959/j.issn.1000-436x.2023103]

[24] 牛淑芬, 宋蜜, 方丽芝, 王彩芬. 智慧医疗中基于属性加密的云存储数据共享. 电子与信息学报, 2022, 44(1): 107–117. [doi: 10.11999/JEIT210858]

[25] 王经纬, 宁建廷, 许胜民, 殷新春, 陈海霞. 面向可变用户群体的可搜索属性基加密方案. 软件学报, 2023, 34(4): 1907–1925. http://www.jos.org.cn/1000-9825/6698.htm [doi: 10.13328/j.cnki.jos.006698]

作者简介

张明川, 博士, 教授, 博士生导师, CCF 专业会员, 主要研究领域为信息安全, 未来互联网, 工业互联网, 网络安全, 机器学习.

万千雪, 硕士生, 主要研究领域为信息安全, 属性加密.

刘牧华, 博士, 副教授, 主要研究领域为信息安全, 密码学与安全协议.

朱军龙, 博士, 副教授, CCF 专业会员, 主要研究领域为机器学习, 优化理论.

吴庆涛, 博士, 教授, 博士生导师, CCF 高级会员, 主要研究领域为网络与信息安全, 云计算与物联网, 智能信息处理.