软件学报 ISSN 1000-9825, CODEN RUXUEW
*Journal of Software*, 2023,34(4):1907−1925 [doi: 10.13328/j.cnki.jos.006698]

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

# 面向可变用户群体的可搜索属性基加密方案*

王经纬[1], 宁建廷[2,3], 许胜民[2], 殷新春[1,4], 陈海霞[2]

[1](扬州大学 信息工程学院, 江苏 扬州 225127)

[2](福建师范大学 计算机与网络空间安全学院, 福建 福州 350007)

[3](信息安全国家重点实验室(中国科学院 信息工程研究所), 北京 100093)

[4](扬州大学广陵学院, 江苏 扬州 225000)

通信作者: 宁建廷, E-mail: jtning@fjnu.edu.cn

**摘 要**: 为解决属性基加密方案中用户撤销繁琐、密文更新计算开销大的问题, 提出一种面向可变用户群体的可搜索属性基加密方案. 利用二叉树管理撤销列表, 当需要撤销用户时, 可信中心只要将其加入撤销列表, 并通知云服务器更新部分密文, 提高了用户撤销的效率. 考虑到利用二叉树实现用户撤销会导致系统中用户数量存在上限, 当某个二叉树叶结点所代表的用户被撤销后, 只要更新二叉树中设置的随机值, 其他用户就可以重复使用该结点. 基于配对计算为用户提供密文搜索功能, 并保证被撤销的用户无法搜索密文. 安全性分析表明, 该方案在随机谕言模型下满足选择明文不可区分安全性. 性能分析和实验数据表明, 该方案相比于同类方案, 计算开销更小.

**关键词**: 用户撤销; 访问控制; 属性基加密; 秘密共享; 二叉树

**中图法分类号**: TP309

中文引用格式: 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

英文引用格式: 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). http://www.jos.org.cn/1000-9825/6698.htm

## Searchable Attribute-based Encryption Scheme for Dynamic User Groups

WANG Jing-Wei[1], NING Jian-Ting[2,3], XU Sheng-Min[2], YIN Xin-Chun[1,4], CHEN Hai-Xia[2]

[1](School of Information Engineering, Yangzhou University, Yangzhou 225127, China)

[2](College of Computer and Cyber Security, Fujian Normal University, Fuzhou 350007, China)

[3](State Key Laboratory of Information Security (Institute of Information Engineering, Chinese Academy of Sciences), Beijing 100093, China)

[4](Guangling College of Yangzhou University, Yangzhou 225000, China)

**Abstract**: To improve the performance of user revocation and ciphertext update, a searchable attribute-based encryption scheme for dynamic user groups is proposed. A binary tree is applied to manage the revocation list. The user revocation will be achieved by adding revoked users to the revocation list and informing the cloud server to update the ciphertexts. To relieve the limitation of the number of system users, the nodes of the user binary tree will be re-used by new users if the random values in the nodes could be updated when user revocation occurs. Besides, a ciphertext search function, based on bilinear pairing, is provided and all revoked users are not allowed to perform the search algorithm. The security analysis proves that the proposed scheme is IND-CPA secure under the random oracle model. The performance analysis shows that the proposed scheme outperforms other existing solutions in terms of computational overhead.

**Key words**: user revocation; access control; attribute-based encryption; secret sharing; binary tree

* 基金项目: 国家自然科学基金(62102090, 62032005, 61972094, 61902070); 福建省科协第二届青年人才托举工程

收稿时间: 2021-11-04; 修改时间: 2022-01-03, 2022-02-17, 2022-03-26; 采用时间: 2022-04-14

得益于云计算技术的发展和应用, 用户只需支付较少的服务费用就能获取云端提供的各种资源和服务. 因此, 相比于将数据存储在本地, 云服务器所提供的计算与存储能力对用户更具有吸引力. 根据预测, 到 2025 年, 全球数据规模将增长至 163 ZT[1]. 对于如此庞大规模的数据, 如何在保障数据安全的基础上提供更优质的数据服务是云服务提供商们首先需要考虑的问题[2,3]. 在数据安全方面, 为了防止他人恶意窃取用户数据而造成隐私泄露或财产损失, 一个最简单直接的方法就是由用户自己加密数据, 并将加密后的数据上传至云服务器保存. 然而, 这种做法存在诸多隐患: 用户可能需要保管多种密钥以满足不同的使用需求, 一旦密钥遭到泄露, 会对数据安全造成巨大的威胁; 以密钥作为数据访问的唯一凭证, 缺乏对访问者身份的控制能力, 更无法抵抗恶意云服务器的攻击. 而在数据使用方面, 用户每次访问数据都需要将完整的密文下载到本地之后再执行解密, 使用起来极为不便; 当需要与他人共享数据时, 需要事先明确共享成员的身份, 一旦出现人员变动的情况, 无法灵活地修改访问权限. 此外, 当面对海量的加密数据时, 缺少合适的检索机制快速准确地定位相关的密文.

属性基加密(attribute-based encryption, ABE)作为一种可以提供细粒度访问控制的工具, 完全满足上述安全和使用需求. 属性基加密的起源是 Sahai 等人于 2005 年提出的一种基于用户生物特征信息的身份基加密方案[4]. 在他们的方案中, 用户身份由 $n$ 个与用户生物特征相关的属性表示. 在匹配时, 只要 $n$ 个属性中存在 $t$ 个属性匹配成功, 即可完成认证. 随后, Waters 等人在文献[5]中进一步对文献[4]中的模糊身份基加密方案进行了扩展, 正式提出了属性基加密的概念. 与模糊身份基加密方案不同, 属性基加密方案中的用户身份可以使用任意与个人身份相关的属性来表示, 并且根据应用场景的不同, 属性基加密可以分为密文策略属性基加密(ciphertext-policy attribute-based encryption, CP-ABE)[6−8]和密钥策略属性基加密(key-policy attribute-based encryption, KP-ABE)[9−11]. 在密文策略属性基加密中, 用户的密钥和一个属性集合相关, 密文与一个访问策略相关. 用户完成解密的唯一条件是用户密钥中的属性集合可以满足密文中的访问策略. 而在密钥策略属性基加密中, 用户的密钥和一个访问策略相关, 密文与一个属性集合相关. 只有当密文中的属性集合可以满足密钥中的访问策略时, 才能完成解密. 虽然这两种属性基加密方案都可以实现数据共享, 但由于密文策略属性基加密方案中的访问策略可以由数据拥有者制定, 因此更加适用于云环境下的数据共享模式[12].

基于大型公司范围内数据共享的应用需求, Wang 等人提出了一个支持用户撤销和高效密文搜索的属性基加密方案[13]. 一方面, 他们使用“版本控制”的方法来实现用户撤销. 简单来说, 当用户撤销发生之后, 可信中心会选择一个版本参数 $ver$ 并通知云服务器和系统中未撤销的用户更新密文和密钥. 而未撤销的用户由于无法获得更新密钥, 从而丧失解密能力. 另一方面, 为了提供更好的数据共享服务, 文献[13]还提供了密文搜索功能. 通过为每个密文附加一个关键词索引, 用户可以花费很小的计算开销, 通过关键词匹配的方法在密文中搜索. 只有当用户拥有足够的访问权限时, 云服务器才会将搜索结果返还给用户.

虽然上述方案可以解决公司内部的数据共享问题, 但为了防止撤销用户继续访问系统中的密文, 每当有用户需要离开系统时, 不仅负责存储的云服务器需要更新密文, 系统中所有未被撤销的用户也需要更新其个人密钥, 因此, 使用“版本控制”来实现用户撤销会带来较大的计算开销. 当使用计算资源受限的物联网设备时, 密钥更新所带来的计算开销过大的问题将更加突出. 随着物联网设备的普及, 设计低计算开销的属性基加密方案逐渐受到人们的重视. 在文献[14]中, Han 等人使用二叉树来实现用户撤销(见第 2.6 节), 用户撤销时, 只需要将撤销用户加入撤销列表并更新密文, 无需更新系统中用户的密钥. 然而在该方案中, 系统用户数量受制于二叉树结点的数量, 存在上限. 当新用户加入时, 如何充分利用那些与撤销用户相关联的结点却鲜受关注. 此外, 文献[14]中的方案并不具备密文搜索功能. 当云服务器中存在大量密文数据时, 用户只能通过多次下载并尝试解密的方法寻找所需数据, 使用起来十分不便. 因此, 为了降低用户撤销带来的计算开销、实现无限制的用户加入与退出管理、提高用户的使用体验, 本文提出了一个面向可变用户群体的可搜索属性基加密方案, 贡献如下:

(1) 在撤销机制方面, 使用二叉树来实现用户撤销. 当需要撤销用户时, 只要将待撤销的用户加入撤销列表并更新密文, 简化了用户撤销的流程. 此外, 本文还通过更新二叉树中结点随机值的方式, 提

高了二叉树结点的复用性, 缓解了该类方案用户数量存在上限的问题;

(2) 在计算开销方面, 降低了密文更新的计算开销. 系统只需要更新密文中与用户身份相关的部分密文, 无需更新完整的密文;

(3) 考虑到云服务器中可能存在大量的加密数据, 数据用户需要耗费大量的时间获取所需的信息, 本文设计了一个高效的密文搜索算法. 一方面, 数据拥有者在上传密文前会为每个密文附加相应的搜索索引. 数据用户只需要输入搜索关键词即可获取相应的陷门. 当云服务器收到搜索请求后, 进行 2 次双线性配对操作即可完成搜索. 此外, 为了防止被撤销用户通过搜索功能获取与密文相关的信息, 本文还在搜索算法中设置了验证机制;

(4) 本文方案基于判定双线性 Diffie Hellman (decisional bilinear Diffie-Hellman, DBDH)困难性假设和离散对数(discrete logarithm, DL)困难性假设, 能有效抵抗选择明文攻击, 并可以保证关键词不可区分安全. 此外, 本文还讨论了前后向安全以及当面对关键词猜测攻击和共谋攻击时方案的安全性. 性能分析表明: 本文方案在没有增加额外计算开销的情况下, 提高了系统的实用性.

## 1 相关工作

撤销是指由于各种原因(检测到用户的恶意行为或用户服务过期等)而需要收回系统中某些用户访问权限的操作. 在属性基加密中, 对于撤销机制的研究最早由 Pirretti 等人[15]提出. 由于现实生活中人员变动的情况十分普遍, 因此提供方便、高效的撤销功能十分重要. 为了解决上述问题, 研究人员基于属性基加密提出了许多行之有效的解决方案.

在属性撤销方面, 文献[16]提出了一个支持多中心和属性撤销的属性基加密系统. 每当撤销发生后, 该系统需要更新所有系统中未撤销用户的密钥并重加密相关的密文, 带来了大量的计算开销. 文献[17]的方案中同样支持属性撤销. 在他们的系统中, 可信中心负责为所有的属性生成一对属性公私钥. 每当数据拥有者向可信中心发送撤销请求, 可信中心首先生成一个撤销属性的版本密钥并更新相应的属性公钥. 同时, 可信中心还需要向系统中的用户以及云服务器发送该版本密钥, 以完成用户密钥和密文的更新. 整个撤销过程不仅繁琐, 而且同样存在计算开销大的问题. 在文献[18]中, Hoang 等人提出了一个支持前向安全的可撤销属性基加密方案. 考虑到可能存在需要同时撤销多个用户共有的某个属性的情况, 他们的方案可以同时撤销 $k$ 个用户的属性. 类似于其他属性撤销方案, 撤销过程中同样需要第三方可信机构的参与, 且需要为所有未撤销用户生成更新密钥. 文献[19]的方案中, 云服务器需要存储系统中所有密钥的版本信息. 当发生撤销之后, 未撤销的用户通过云服务器提供的版本密钥来更新自己的密钥. 一旦用户的密钥丢失, 将直接失去访问系统中数据的能力. 此外, 将所有更新密钥存储在云服务器中也会增加系统的安全风险.

相比于属性撤销, 用户撤销更多地受到了研究人员的青睐. 文献[20]基于变色龙哈希提出了医疗数据共享场景下支持用户撤销的密文策略属性基加密方案. 当用户被撤销后, 系统会为所有未撤销的用户分配一个陷门, 以更新他们的用户密钥. 由于用户撤销之后往往需要更新相应的密文以保障系统的前向安全性, 为防止不可信的云服务器提供错误的更新结果, 文献[21]提出了一个旨在保障数据完整性的属性基加密方案. 文献[22]在一个层次属性基加密方案中实现了用户撤销功能, 以适应医疗环境下的数据共享. 文献[23]通过为用户密钥设置有效期的方法, 实现了基于时间的数据访问控制方案. 在此基础上, 他们通过结合白盒追踪和二叉树实现了高效的用户追踪和撤销功能. 但文献[23]中的方案并没有考虑系统用户存在上限的问题. 文献[24]同样提出了一个支持用户撤销的属性基加密方案, 同时, 他们的方案还支持密文搜索、用户追责、外包解密等功能. 然而, 以上所有方案在用户撤销之后, 都需要在密文更新阶段花费大量的计算开销.

为了缓解上述问题、提高属性基加密方案中用户撤销的实用性, 文献[25]提出利用二叉树实现用户撤销, 并成功将用户撤销的开销由线性增长降低为对数级增长. 基于文献[25]的工作, 研究人员提出了许多可撤销属性基加密方案. 然而, 这些方案中的用户数量均存在上限, 不适用于用户数量动态变化的场景. 此外, 当与某个叶子结点相关联的用户被撤销后, 如何在不影响撤销列表中用户的情况下将这些叶子结点与其他用户相

关联, 同样值得研究.

另一方面, 由于属性基加密所产生的共享数据以密文的形式存储在云服务器中, 为了协助用户方便快捷地获取他们所需要的数据, 可搜索加密(searchable encryption, SE)技术逐渐受到了人们的关注. 最初的可搜索加密方案大多是以对称加密机制为背景设计的, 因此只能支持“一对一”的数据共享模式[26], 但在该模式下, 用户需要提前知晓数据共享对象的身份. 为了提高可搜索加密技术的实用性, 2006 年, Curtmola 等人基于广播加密技术构造了第一个支持“一对多”模式的可搜索加密方案[27]. 但在该方案中, 所有的用户依然需要共享一个用户密钥. 而在密文策略属性基加密中, 由于密钥与用户的身份信息相关, 每个用户都有属于自己的密钥, 因此更便于数据的共享. Li 等人[28]利用 $k$ 近邻($k$-nearest neighbor, $k$NN)算法和 ABE 技术提出了一个医疗数据共享场景下, 支持搜索的数据加密方案. 在他们的方案中, 搜索功能同时具备前向安全性和后向安全性. Awad 等人[29]提出了一个支持模糊关键词搜索和搜索结果排序的可搜索加密方案, 然而他们的方案并没有解决布隆过滤器可能会误报的问题. Liu 等人[30]基于区块链技术提出了一个可搜索属性基加密方案. 结合智能合约技术, 使用区块链取代传统属性基加密方案中的可信中心, 从而达到去中心化的目的, 且密文搜索仅需执行一次群上的求幂运算. He 等人[31]为了丰富属性基加密中密文搜索关键词的表达能力, 提出了一个支持混合布尔关键词搜索的属性基加密方案. 该方案中, 用于搜索的关键词可以使用布尔运算符连接并形成一个布尔表达式, 使得用户搜索更为灵活. 文献[32]提出了一个针对分级数据的可搜索属性基加密方案, 该方案针对单关键词搜索可能导致搜索结果不准确的问题, 提出了一个支持多关键词搜索的属性基加密方案. 同时, 为了保证方案的前向安全性, 该方案还提出了一个安全的用户撤销方案. Sultan 等人[33]提出了在云环境下支持授权关键字搜索的属性基加密方案, 用户交互式地向云服务器提交搜索陷门, 从而摆脱了搜索时对于安全信道的依赖. 此外, 该方案还支持多关键词搜索以获得更好的搜索结果. 以上的方案各有特点, 但考虑到在云环境下海量的数据中进行搜索时, 搜索效率将会直接影响用户的使用体验, 因此, 如何提高密文搜索效率更值得研究人员关注.

表 1 列出了本文方案与现存的一些研究工作在功能、特性方面的对比情况, 其中, $F1$ 表示用户撤销, $F2$ 表示可重用的用户二叉树, $F3$ 表示密文更新, $F4$ 表示密文搜索, $F5$ 表示前后向安全.

表 1 功能、特性对比

| 方案 | $F1$ | $F2$ | $F3$ | $F4$ | $F5$ |
|---|---|---|---|---|---|
| 文献[13] | √ | × | √ | √ | × |
| 文献[14] | √ | × | √ | × | √ |
| 文献[18] | √ | × | √ | × | √ |
| 文献[30] | √ | × | × | √ | × |
| 文献[32] | √ | × | × | √ | √ |
| 本文方案 | √ | √ | √ | √ | √ |

## 2 基础知识

### 2.1 常用符号说明

表 2 列出了一些在本文中常见符号的说明.

表 2 常见符号说明

| 符号 | 含义 |
|---|---|
| $G$, $G_T$ | 阶为质数 $p$ 的双线性循环群 |
| $UT$ | 用户二叉树 |
| $Y$ | 叶子结点集合 |
| $L$ | 系统属性空间 |
| $S$ | 用户属性集合 |
| $T$ | 密文访问结构 |
| $e$ | 双线性映射 |
| $U$ | 系统中用户的集合 |
| $\lambda$ | 系统安全参数 |

### 2.2 双线性映射

令 $G$ 和 $G_T$ 表示阶为质数 $p$ 的双线性循环群, $g$ 为群 $G$ 的一个生成元. 双线性映射 $e$ 具有以下特性.

(1) 双线性: 对于任意 $u,v\in G$ 以及 $a,b\in Z_p^*$, $e(u^a,v^b)=e(u,v)^{ab}$;

(2) 非退化性: 存在 $u,v\in G$, 使得 $e(u,v)\neq 1$;

(3) 可计算性: 对于任意 $u,v\in G$, 都可以有效地计算 $e(u,v)$.

### 2.3 困难性问题

在 DBDH 问题中, 挑战者根据安全参数选择阶为质数 $p$ 的双线性循环群 $G$ 和 $G_T$, 令 $g$ 表示群 $G$ 的生成元. 随机选择 $a,b,s\in Z_p^*$. 当挑战者向攻击者提供元组 $\delta=(g,g^a,g^b,g^s)$ 时, 攻击者很难将 $e(g,g)^{abs}\in G_T$ 与一个群 $G_T$ 中的随机元素 $R$ 区分开来. 假设一个算法 $B$ 可以解决 DBDH 问题, 那么将它在群中解决这个问题的优势定义为 $\varepsilon$, 其中, $|\Pr[B(\delta,e(g,g)^{abs})]-\Pr[B(\delta,R)]|\geqslant\varepsilon$.

**定义 1**. 对于任何多项式时间的算法, 其成功解决上述 DBDH 问题的优势均是可忽略的, 则称 DBDH 困难性假设成立.

在 DL 问题中, 群 $G$ 是阶为质数 $p$ 的双线性循环群. 令 $g$ 表示群 $G$ 的生成元, 给定 $g,g^s\in G$, 对任意正整数 $s\in Z_p^*$, DL 问题的目标是计算 $s$. 任意概率多项式时间(probabilistic polynomial time, PPT)算法成功解决 DL 问题的概率 $Adv^{DL}(A)=\Pr[A(g,g^s)]$ 是可忽略的, 其中, 概率来源于 $s$ 在 $Z_p^*$ 上的随机选取和算法的随机选择.

**定义 2**. 对于任何多项式时间算法, 其成功解决上述 DL 问题的概率均是可忽略的, 则称 DL 困难性假设成立.

### 2.4 访问结构

令 $L=\{A_1,A_2,\ldots,A_n\}$ 表示属性空间, 则称一个非空的属性集合 $A\subseteq 2^L$ 为访问结构. 此外, 对于任意 $B,C\subseteq L$, 如果 $B\in A$ 并且 $B\subseteq C$, 那么 $C\in A$, 则 $A$ 是单调的. 对于任意 $D\in A$, $D$ 是授权集合.

### 2.5 访问结构树

$T$ 是一个访问结构树, $Root$ 是 $T$ 的根结点, $x$ 代表树 $T$ 的一个结点. 如果 $x$ 是内部结点, 令 $num_x$ 表示 $x$ 的孩子结点的个数, $l_x$ 表示其阈值. 若 $l_x=1$, 说明 $x$ 是一个或门结点; 若 $l_x=num_x$, 说明 $x$ 是一个与门结点. 若 $x$ 是叶子结点, 则其阈值 $l_x=1$ 且与一个属性相关. 此外, 本文还定义了以下函数: $parent(x)$ 表示结点 $x$ 的父亲结点; $attr(x)$ 表示叶子结点 $x$ 的关联属性; $index(x)$ 表示结点 $x$ 的唯一编号; $T(S)=1$ 表示属性集合 $S$ 满足访问结构树 $T$.

### 2.6 用户二叉树

令 $U$ 表示系统中用户的集合, $Y$ 表示叶子结点的集合, $R$ 表示撤销列表, 则对用户二叉树 $UT$ 的定义如下.

(1) 用户二叉树 $UT$ 的每一个叶子结点分别与一个系统中的用户相关联. 令 $|U|$ 代表系统中用户的数量, $|Y|$ 代表叶子结点的数量, 则用户二叉树中所有结点的数量为 $2|Y|-1$. 按广度优先搜索的顺序对树中的结点进行编号, 根结点编号为 0, 最后一个结点编号为 $2|Y|-2$;

(2) $path(i)$ 是一条从结点 0 到结点 $i$ 的路径;

(3) $cover(R')$ 是与所有不在撤销列表 $R$ 中与用户相关结点的最小结点集合, 称为与撤销列表 $R$ 相关的最小覆盖集合[25]. 令 $x.lchild$ 表示结点 $x$ 的左孩子, $x.rchild$ 表示结点 $x$ 的右孩子, 最小覆盖集合生成算法的定义如算法 1 所示;

(4) 如果一个用户 $u$ 不在撤销列表 $R$ 中, 则有且仅有一个结点 $j$, 使得 $j=cover(R)\cap path(u)$.

**算法 1**. 最小覆盖集合生成算法.

function $cover(R)$

$X,Y\leftarrow\varnothing$

**for** $u\in R$

$X \leftarrow X \cup path(u)$
**end for**
**for** $x \in X$
**if** $x.lchild \notin X$ **then** $Y \leftarrow Y \cup x.lchild$
**end if**
**if** $x.rchild \notin X$ **then** $Y \leftarrow Y \cup x.rchild$
**end if**
**end for**
**if** $Y=\varnothing$ **then** $Y \leftarrow Root$
**end if**
**return** $Y$
**end** function

如图 1 所示, 撤销列表 $R=\{u2,u5,u6\}$, 因此与撤销列表 $R$ 相关的最小覆盖集合 $cover(R)=\{7,4,6\}$. 以 $u4$ 为例, 从结点 0 到结点 $u4$ 的路径为 $path(u4)=path(10)=\{0,1,4,10\}$. 因此, $path(u4)$与 $cover(R)$相交得到的唯一结点为: $j=cover(R) \cap path(u4)=\{4\}$.

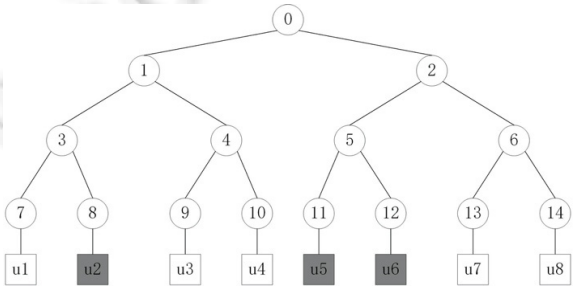



图 1 用户二叉树

## 3 形式化定义及安全模型

本文所提的方案中共包含 4 个实体, 分别是可信中心、云服务器、数据拥有者和数据用户. 可信中心和数据拥有者是完全可信的, 云服务器半可信, 而数据用户不可信. 本节首先介绍方案的系统模型, 包括各个参与方以及他们的职责; 其次, 算法定义给出了构成本文方案的 8 个算法; 最后, 在安全模型部分, 我们分别考虑了选择明文攻击安全和关键词不可区分安全并给出相应的说明.

### 3.1 系统模型

本文方案的系统模型图如图 2 所示.

(1) 可信中心是系统中完全可信的机构, 主要负责初始化整个系统和为数据用户分发用户密钥. 此外, 可信中心还需要维护撤销列表. 通过更新用户二叉树中结点随机值的方式, 实现用户二叉树结点的多次使用, 保障系统中数据的安全;

(2) 云服务器是半可信的机构. 由于云服务器通常具备强大的计算与存储资源, 因此在系统中负责存储和维护用户上传的数据. 当可信中心完成撤销操作之后, 为了防止被撤销的用户继续访问密文, 云服务器需要根据可信中心提供的参数, 更新存储在系统中的密文. 云服务器是半可信的机构, 它虽然会按照设定的流程为用户提供服务, 但也会在不与数据用户共谋的情况下尝试进行解密;

(3) 数据拥有者希望共享部分数据给某个特定的群体, 但是由于这些数据可能包含某些敏感信息, 无法直接以明文的形式共享. 利用属性基加密, 数据拥有者可以细粒度地控制数据的共享范围. 由于数据拥有者是数据共享的主体, 因此是可信的;

(4) 数据用户可以通过在可信中心注册, 获得一个与自身属性集合相关的用户密钥. 凭借用户密钥, 数据用户可以选择云服务器中合适的密文进行解密. 由于数据用户并不可信, 因此一部分权限不足的用户可能会尝试通过合并他们的用户密钥来获取他们原本无法解密的信息.

以公司内部数据共享的应用场景为例: 某公司人数相对固定, 但随时存在人员流动的情况. 公司的 IT 部门作为可信中心, 为所有员工分发与其职业属性相关的用户密钥以及系统参数, 公司中高级员工作为数据拥有者可以制定访问策略, 并上传数据到第三方云服务器中, 以便数据共享; 其他员工只要自身拥有的属性满足访问策略且未被撤销, 都能以数据用户的身份向云服务器申请数据. 当有员工离开公司时, 出于商业利益考虑, 需要将该员工加入撤销列表以防止其继续访问公司内部的商业文件. 为了保证系统的前后向安全,撤销员工后, 需要及时更新云服务器中的加密数据.

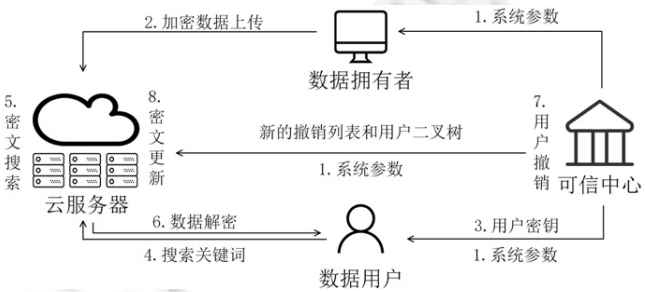



图 2 系统模型图

### 3.2 算法定义

本文方案由 8 个算法组成, 具体描述如下.

(1) $Setup(\lambda,L,n)\rightarrow(PK,MSK,UT)$. 该算法由可信中心执行, 负责系统的初始化并生成系统参数. 算法的输入为安全参数$\lambda$、属性空间 $L$ 以及系统最大用户数 $n$, 输出系统公钥 $PK$、系统主密钥 $MSK$ 和用户二叉树 $UT$;

(2) $Encryption(PK,ck,R,T,KW)\rightarrow CT$. 该算法由数据拥有者执行, 负责根据制定的访问策略加密数据用户共享的数据. 基于密钥封装的思想, 本文方案并不直接使用属性基加密, 而是先使用对称加密算法来加密共享数据, 然后只需要加密用于加密共享数据的对称密钥, 就可以实现对共享数据的访问控制. 此外, 为了实现密文搜索功能, 数据拥有者还需要为密文设置相应的关键词索引. 加密算法的输入为系统公钥 $PK$、对称密钥 $ck$、撤销列表 $R$、访问策略 $T$ 以及关键词 $KW$, 输出密文 $CT$;

(3) $KeyGen(PK,MSK,u,S)\rightarrow SK$. 该算法由可信中心执行, 负责根据系统中数据用户的属性集合生成相应的密钥. 算法的输入为系统公钥 $PK$、系统主密钥 $MSK$、用户身份 $u$ 以及用户属性集合 $S$, 输出密钥 $SK$;

(4) $Trapdoor(PK,SK,SKW)\rightarrow trapdoor$. 该算法由数据用户执行, 负责根据数据用户提供的搜索关键词生成搜索陷门. 算法的输入为系统公钥 $PK$、用户密钥 $SK$ 和搜索关键词 $SKW$, 输出搜索陷门 $trapdoor$;

(5) $Search(CT,trapdoor)\rightarrow 0$ or 1. 该算法由云服务器执行, 负责根据用户提供的搜索陷门在密文中进行匹配. 算法的输入为密文 $CT$ 和搜索陷门 $trapdoor$, 如果匹配成功输出 1, 否则输出 0;

(6) $Decryption(PK,UT,CT,SK)\rightarrow ck$. 该算法由数据用户执行, 负责解密属性基加密算法生成的密文, 恢复对称密钥用于进一步解密. 算法的输入为系统公钥 $PK$、用户二叉树 $UT$、密文 $CT$ 以及用户密钥 $SK$, 输出对称密钥 $ck$;

(7) $Revocation(R',u,MSK,UT')\rightarrow(R,UT)$. 该算法由可信中心运行, 可以实现对用户访问权限的撤销. 算法的输入参数为当前撤销列表 $R'$、待撤销用户 $u$ 的身份、系统主密钥 $MSK$ 以及旧的用户二叉树 $UT'$, 输出新的撤销列表 $R$ 和新的用户二叉树 $UT$;

(8) $CTUpdate(PK,CT',R)\rightarrow CT$. 该算法由云服务器执行, 当系统中有用户被撤销, 云服务器执行密文更

新算法, 更新所存储的相关密文. 算法的输入参数为系统公钥 $PK$、待更新的密文 $CT'$以及撤销列表 $R$, 输出新密文 $CT$.

### 3.3 安全模型

(1) 选择明文攻击安全

本文提出的方案是选择明文不可区分(selectively indistinguishable choose plaintext attack, sIND-CPA)安全的, 其安全模型可以由以下挑战者 $C$ 和攻击者 $A$ 之间的安全游戏表示.

- 初始化阶段: 一个概率多项式时间攻击者 $A$ 将要挑战的访问策略 $T^*$和用户身份 $u^*$发送给挑战者 $C$;
- 系统建立阶段: 挑战者 $C$ 运行系统建立算法 $Setup(\cdot)$, 生成系统公钥 $PK$ 和主私钥 $MSK$, 将系统公钥 $PK$、用户二叉树 $UT$ 发送给攻击者 $A$, 并保存系统主私钥 $MSK$;
- 询问阶段 1: 攻击者 $A$ 在该阶段向挑战者 $C$ 发起密钥询问、撤销询问和密文更新询问, 直至主动结束该阶段. 3 种询问的定义如下.
  - 密钥询问: 攻击者 $A$ 选择一个用户身份 $u$ 和一个属性集合 $S$ 提交给挑战者 $C$. 挑战者 $C$ 运行密钥生成算法 $KeyGen(\cdot)$, 获取并返回相应的用户密钥 $SK_u$;
  - 撤销询问: 攻击者 $A$ 选择一个用户身份 $u$ 并提交给挑战者 $C$. 挑战者 $C$ 运行撤销算法 $Revocation(\cdot)$, 获取并返回新的撤销列表 $R$ 和用户二叉树 $UT$;
  - 密文更新询问: 攻击者 $A$ 选择一个撤销列表 $R$ 和密文 $CT$ 提交给挑战者 $C$. 挑战者 $C$ 运行密文更新算法 $CTUpdate(\cdot)$, 生成并返回更新的密文 $CT$;
- 挑战阶段: 攻击者 $A$ 向挑战者 $C$ 提交 2 个等长的消息 $M_0$ 和 $M_1$. 此外, 在挑战阶段, 挑战用户的身份需要满足以下条件.
  1) 如果挑战用户 $u^*$是未撤销用户(普通用户), 那么攻击者 $A$ 禁止向挑战者提交任何满足挑战访问策略 $T^*$的属性集合, 以获取相应的密钥;
  2) 如果挑战用户 $u^*$是撤销用户, 那么攻击者可以在密钥询问中任意询问与用户 $u$ 相关的用户密钥, 但在询问完成后, 必须对用户 $u$ 执行撤销询问和密文更新询问;

  挑战者 $C$ 随机选择 $c\in\{0,1\}$, 运行加密算法 $Encryption(\cdot)$, 并根据参数 $T^*$和 $M_c$ 生成挑战密文 $CT_c$;
- 询问阶段 2: 攻击者 $A$ 执行询问阶段 1 的密钥询问、撤销询问以及密文更新询问, 并遵从相应的限制条件;
- 猜测阶段: 当攻击者 $A$ 结束询问阶段 2 之后, 输出一个猜测结果 $c'\in\{0,1\}$. 如果 $c=c'$, 那么称攻击者 $A$ 以 $Adv=|\Pr[c=c']-1/2|$的优势赢得该游戏.

**定义 3**. 如果不存在多项式时间的攻击者能以不可忽略的优势赢得这个游戏, 那么本文方案在选择明文攻击下满足不可区分安全性.

(2) 关键词不可区分安全

本文提出的方案是选择模型下关键词不可区分安全的, 其安全模型可以由以下挑战者 $C$ 和攻击者 $A$ 之间的安全游戏表示.

- 系统建立阶段: 挑战者 $C$ 运行系统建立算法 $Setup(\cdot)$, 生成系统公钥 $PK$ 和主私钥 $MSK$, 然后, 挑战者 $C$ 将系统公钥 $PK$ 发送给攻击者 $A$, 保存系统主私钥 $MSK$;
- 询问阶段 1: 攻击者 $A$ 在该阶段向挑战者 $C$ 发起关键词询问, 直至主动结束该阶段.
  - 关键词询问: 攻击者 $A$ 选择一个用户身份 $u$、一个属性集合 $S$ 和关键词 $SKW$ 提交给挑战者 $C$. 挑战者 $C$ 运行密钥生成算法 $KeyGen(\cdot)$, 生成相应的用户密钥 $SK_u$, 运行陷门生成算法 $Trapdoor(\cdot)$, 获取并返回陷门 $trapdoor$;
- 挑战阶段: 攻击者 $A$ 向挑战者 $C$ 提交 2 个等长的关键词 $SKW_1$ 和 $SKW_2$. 挑战者 $C$ 随机选择 $c\in\{0,1\}$, 并运行密钥生成算法 $KeyGen(\cdot)$和陷门生成算法 $Trapdoor(\cdot)$, 获取挑战关键词 $trapdoor$. 要求 $SKW_1$ 和 $SKW_2$ 不能是在询问阶段 1 中出现过的关键词;

- 询问阶段 2: 攻击者 $A$ 执行询问阶段 1 的关键词询问, 其中的限制条件为不能询问关键词 $SKW_1$ 和 $SKW_2$;
- 猜测阶段: 当攻击者 $A$ 结束询问阶段 2 之后, 输出一个猜测结果 $c'\in\{0,1\}$. 如果 $c=c'$, 那么称攻击者 $A$ 以 $Adv=|\Pr[c=c']-1/2|$的优势赢得该游戏.

**定义 4**. 如果不存在多项式时间的攻击者可以以不可忽略的优势赢得这个游戏, 那么本文方案在选择关键词攻击下满足不可区分安全性.

## 4 方案设计及安全性分析

### 4.1 方案设计

(1) $Setup(\lambda,L,n)\rightarrow(PK,MSK,UT)$

该算法由可信中心运行, 输入参数为安全参数$\lambda$、属性空间 $L$ 和系统最大用户数 $n$. 算法生成双线性映射群$(G,G_T,p,e)$, 随机选择 $g\in G$, $\alpha,\beta\in Z_p^*$. 对于二叉树中的结点$\forall i\in UT$, 算法选择 $x_i\in Z_p^*$, 计算 $y_i=g^{x_i}$, 并将 $y_i$ 存储在结点 $i$ 中. 对于所有的属性 $j\in L$, 算法随机选择 $v_j\in Z_p^*$, 并计算 $A_j=g^{v_j}$, 设置哈希函数 $H_0:\{0,1\}^*\rightarrow Z_p^*$, $H_1:\{0,1\}^*\rightarrow G$, 令 $hash=(H_0,H_1)$. 将系统最大用户数量 $n$ 作为用户二叉树 $UT$ 叶子结点的个数. 算法输出系统公钥 $PK$、系统主密钥 $MSK$ 以及用户二叉树 $UT$, 其中, 系统公钥 $PK$、系统主密钥 $MSK$ 如下:

$$PK=\{g,e(g,g)^{\alpha},g^{\beta},\{A_j\}_{j\in L},hash\},$$
$$MSK=\{\alpha,\beta,\{x_i\}_{i\in UT},\{v_j\}_{j\in L}\}.$$

(2) $Encryption(PK,ck,R,T,KW)\rightarrow CT$

该算法由数据拥有者运行, 输入参数为系统公钥 $PK$、对称密钥 $ck$、撤销列表 $R$、访问策略 $T$ 以及关键字 $KW$, 其中, 撤销列表 $R=\varnothing$. 加密算法产生的密文分为 2 个部分, 分别与访问策略和撤销列表相关. 其中, 与访问策略相关的部分负责验证用户密钥中的属性是否满足要求, 而与撤销列表相关的部分负责防止被撤销的用户解密密文. 对于任意访问策略 $T$ 中的结点 $x$, 算法随机选择一个阶为 $d_x=l_x-1$ 的多项式 $q_x$, 其中, $l_x$ 为结点 $x$ 的门限值. 从根结点 $Root$ 开始, 随机选择秘密值 $s\in Z_p^*$, 并令 $q_{Root}(0)=s$. 然后, 算法再随机选择 $d_{Root}$ 个参数完成对多项式 $q_{Root}$ 的定义. 对于其他的结点 $x$, 算法按照自顶向下的顺序设置 $q_x(0)=q_{parent(x)}(index(x))$, 并同样选择 $d_x$ 个参数来完成对多项式 $q_x$ 的定义. 算法计算 $C=ck\cdot(g,g)^{s\alpha}$, $C_0=g^s$, 令 $Y$ 表示叶子结点的集合, 则与访问策略相关的部分密文为$\{C,C_0,\{C_{i,1},C_{i,2}\}_{i\in Y}\}$, 其中, $C_{i,1}=g^{q_i(0)}$, $C_{i,2}=g^{v_{attr(i)}q_i(0)}$. 令 $cover(R)$表示与撤销列表 $R$ 相关的最小覆盖集合(见第 2.6 节). 对$\forall j\in cover(R)$, 算法计算与撤销列表相关的密文部分 $\{T_j=y_j^s\}_{j\in cover(R)}$. 令密文关键词索引 $Ind_{kw}=g^{sH_0(KW)}$. 综合上述密文, 数据拥有者最终将如下密文上传至云服务器:

$$CT=\{T,C,C_0,\{C_{i,1},C_{i,2}\}_{i\in Y},\{T_j\}_{j\in cover(R)},Ind_{kw}\}.$$

(3) $KeyGen(PK,MSK,u,S)\rightarrow SK$

该算法由可信中心运行, 输入参数为系统公钥 $PK$、系统主密钥 $MSK$、用户身份 $u\in Z_p^*$ 以及用户属性集合 $S$. 对应密文中的两个部分, 密钥同样包含两个部分.

- 第 1 部分与用户属性相关: 算法选择随机值 $r,t,\delta\in Z_p^*$, 计算 $D=g^{(ab+\beta t)/\delta}H_1(u)^r$, 对于$\forall\tau\in S$, 计算:

$$D_{\tau,1}=H_1(u)^{\delta r}g^{v_{\tau}r},\ D_2=g^r;$$

- 第 2 部分与撤销列表相关: 令 $path(i_u)=\{i_0,\ldots,i_u\}$表示从根结点 $i_0$ 到与用户 $u$ 相关的结点 $i_u$ 的路径, 算法计算 $K_u=g^{\beta t/x_{i_u}}$.

综合以上密钥,可信中心最终将如下密钥秘密发送给用户 $u$:

$$SK=\{D,\{D_{\tau,1}\}_{\tau\in S},D_2,K_u,\{x_i\}_{i\in path(i_u)},\delta\}.$$

(4) $Trapdoor(PK,SK,SKW)\rightarrow trapdoor$

该算法由数据用户执行, 输入参数为系统公钥 *PK*、用户密钥 *SK* 和搜索关键词 *SKW*. 算法选择随机数 $d \in Z_p^*$, 计算关键词陷门 $trapdoor=\{td_1,\{td_{2,i}\}_{i\in path(i_u)}\}$, 其中, $td_1=g^{dH_0(SKW)/\delta}$, $td_{2,i}=\{g^{dx_i/\delta}\}_{i\in path(i_u)}$.

(5) *Search*(*CT*,*trapdoor*)→0 or 1

该算法由云服务器执行, 算法输入为密文和搜索陷门 *trapdoor*. 云服务器接收到数据用户的搜索请求后, 验证密文 *CT* 中是否存在 $T_j$ 使得等式 $e(td_1,T_j)=e(td_2,Ind_{kw})$成立: 如果匹配成功输出 1, 云服务器将密文 *CT* 发送给数据用户进行进一步的解密操作; 否则算法输出 0, 表示搜索失败. 本文在 *trapdoor* 中设置了验证机制, 按照第 2.6 节对用户二叉树的介绍, 只要用户 *u* 不在撤销列表 *R* 中, 则有且仅有一个结点 *j* 使得 $j=cover(R)\cap path(i_u)$. 因此, 被撤销的用户将无法完成搜索操作.

(6) *Decryption*(*PK*,*UT*,*CT*,*SK*)→*ck*

该算法由数据用户运行, 输入参数为系统公钥 *PK*、用户二叉树 *UT*、密文 *CT* 以及用户密钥 *SK*. 对于用户 *u*, 如果 $u\in R$ 或该数据用户的属性集合 *S* 不满足访问策略 *T*, 则算法停止; 否则, 用户二叉树 *UT* 中有且仅有一个结点 *j* 满足 $j=cover(R)\cap path(i_u)$. 令 $path(u)=\{i_0,\ldots,i_{temp(j)},\ldots,i_u\}$, 其中, $i_{temp(j)}=j$, $i_u$ 是用户二叉树中与用户 *u* 相关的叶子结点, 算法计算 $\theta=x_{i_u}/x_j$ 以及 $B=e(K_u,T_j)^\theta=e(g^{\beta t/x_{i_u}},y_j^s)^\theta=e(g,g)^{\beta ts}$. 然后, 解密程序递归地执行 *DecryptNode*(*CT*,*SK*,*x*), 其中, *x* 是访问策略 *T* 中的结点. 如果 *x* 是叶子结点, 令 $\tau=attr(x)$, 算法计算如下:

$$DecryptNode(CT,SK,x)=\frac{e(D_{\tau,1},C_{x,1})}{e(D_2,C_{x,2})}=\frac{e(H_1(u)^{\delta r}g^{v_\tau r},g^{q_x(0)})}{e(g^r,g^{v_{attr(x)}q_x(0)})}=\frac{e(H_1(u),g)^{\delta rq_x(0)}e(g,g)^{rv_\tau q_x(0)}}{e(g,g)^{rv_{attr(x)}q_x(0)}}=e\big(H_1(u),g\big)^{\delta rq_x(0)}.$$

若 *x* 是一个非叶子结点, 且 *z* 是 *x* 的孩子结点, $S_x$ 是结点 *x* 的孩子结点的集合, 则解密程序 *DecryptNode*(*CT*, *SK*,*x*)的计算过程定义如下:

$$DecryptNode(CT,SK,x)=\prod_{z\in S_x}F_z^{\Delta_{j,S_x'}(0)}=\prod_{z\in S_x}e(H(u),g)^{\delta rq_z(0)\Delta_{j,S_x'}(0)}=e(H_1(u),g)^{\delta rq_{parent(z)}(index(z))^{\Delta_{j,S_x'}(0)}}=e(H_1(u),g)^{\delta rq_x(0)}.$$

其中, $j=index(x)$, $S_x'=\{index(x):z\in S_x\}$. 如果数据用户的属性集合 *S* 满足访问策略 *T*, 则最终解密结果如下:

$$F_{Root}=DecryptNode(CT,SK,R)=e(H_1(u),g)^{\delta rq_{Root}(0)}=e(H_1(u),g)^{\delta rs}.$$

接着, 计算 *ck* 如下:

$$\frac{C\cdot F_{Root}\cdot B}{e(C_0,D)^\delta}=\frac{ck\cdot e(g,g)^{\alpha s}e(H_1(u),g)^{\delta rs}e(g,g)^{\beta ts}}{e(g^s,g^{(\alpha+\beta t)/\delta}H_1(u)^r)^\delta}=\frac{ck\cdot e(g,g)^{\alpha s}e(H_1(u),g)^{\delta rs}e(g,g)^{\beta ts}}{e(g,g)^{\alpha s}e(g,g)^{\beta ts}e(g,H_1(u))^{\delta rs}}=ck.$$

(7) *Revocation*(*R*′,*u*,*MSK*,*UT*′)→(*R*,*UT*)

该算法由可信中心运行, 输入参数为当前撤销列表 *R*′、待撤销的用户 *u*、系统主密钥 *MSK* 以及当前用户二叉树 *UT*′. 选择随机数 $x\in Z_p^*$ 替换主密钥 *MSK* 中相应的 $x_u$, 计算 $y=g^x$ 替换 *UT* 中相应的 $y_u$. 可信中心将 *u* 添加到撤销列表 *R*′中, 形成新的撤销列表 *R*, 同时公布新的用户二叉树 *UT*.

(8) *CTUpdate*(*PK*,*CT*′,*R*)→*CT*

该算法由云服务器运行, 输入参数为系统公钥 *PK*、待更新的密文 *CT*′以及撤销列表 *R*. 当用户撤销发生之后, 云服务器根据最新的撤销列表 *R* 以及密文 *CT*′, 计算更新后的密文 *CT*. 令 *cover*(*R*)表示与最新撤销列表相关的最小覆盖集合, *cover*(*R*′)表示与更新前的撤销列表相关的最小覆盖集合. 此时, *cover*(*R*)中的结点 *j* 存在以下两种情况.

1) 如果存在一个结点 $j'\in cover(R')$使得 $j=j'$, 则令 $T_j=T_{j'}$;

2) 如果存在一个结点 $j'\in cover(R')$使得 $j'$是 $j$ 的一个祖先结点, 令 $path(j)=path(j')\cup\{i_{temp(j')+1},\ldots,i_{temp(j)}\}$, 其中, $i_{temp(j)}=j$, $i_{temp(j')}=j'$. 令 $Y_{j'}=T_{j'}$, 并依次计算 $Y_{i_{k+1}}=(Y_{i_k})^{x_{i_{k+1}}/x_{i_k}}=y_{i_{k+1}}^s$, 其中, $k=temp(j'),\ldots,temp(j)$, 再设置 $T_j=Y_j$. 因此, 与访问策略相关的密文部分不会变化, 更新后的密文如下:

$$CT=\{T,C,C_0,\{C_{i,1},C_{i,2}\}_{i\in Y},\{T_j\}_{j\in cover(R)},Ind_{kw}\}.$$

### 4.2 安全性分析

(1) 选择明文攻击安全

定理 1 保证了本文方案在选择明文攻击下满足不可区分安全性, 其证明过程通过模拟攻击者 $A$ 与挑战者 $C$ 之间的安全游戏来实现. 游戏共分为 6 个阶段, 游戏中的攻击者分为未撤销与撤销这两种情况. 首先, 在初始化阶段, 攻击者 $A$ 将要挑战的访问策略和一个随机选择的挑战用户身份发送给挑战者 $C$. 在系统建立阶段, 挑战者先确认攻击者是否为撤销用户, 然后运行 $Setup(\cdot)$算法生成必要的系统公钥、用户二叉树以及系统主私钥. 挑战者 $C$ 公开系统公钥与用户二叉树, 秘密保存系统主私钥. 在询问阶段 1, 攻击者 $A$ 可以向挑战者 $C$ 发起密钥询问、撤销询问以及密文更新询问来模拟其在攻击之前的信息收集行为, 其中, 撤销询问与密文更新询问分别对应方案中的撤销与密文更新算法, 而密钥询问则根据攻击者 $A$ 身份的不同会产生不同的响应机制. 如果攻击者 $A$ 是非撤销用户, 那么他只能提交属性集合不满足挑战访问策略的密钥询问. 如果攻击者 $A$ 是撤销用户, 那么他可以选择提交任意属性集合进行密钥询问. 在挑战阶段, 攻击者 $A$ 向挑战者 $C$ 提交 2 个等长的消息, 挑战者 $C$ 随机选择一个消息, 以 50%的概率生成一个合法密文. 随后, 攻击者 $A$ 在询问阶段 2 中继续向挑战者 $C$ 进行询问. 最后, 在猜测阶段, 攻击者 $A$ 输出对挑战目标的猜测结果, 如果此时密文不是一个合法的密文, 则攻击者有 1/2 的概率猜测正确; 否则, 令攻击者猜测正确的概率为$\varepsilon$+1/2, 其中, $\varepsilon$为攻击者 $A$ 赢得游戏的优势. 由于挑战密文的构造是基于 DBDH 问题实现的, 如果攻击者 $A$ 在多次游戏中获胜的优势是不可忽略的, 那么就能同样以这个优势破解 DBDH 问题.

**定理 1**. 如果 DBDH 假设成立, 那么本文方案在选择明文攻击下满足不可区分安全性.

证明: 假设存在一个概率多项式时间攻击者 $A$ 能在随机谕言模型下以不可忽略的优势$\varepsilon$攻破本文方案,那么就可以构造一个概率多项式时间算法 $B$, 以$\varepsilon$/2 的优势攻破 DBDH 假设.

本文定义了以下 2 种类型的攻击者.

1) 攻击者 $A$ 是一个未被撤销的普通用户. 他的属性集合无法满足挑战访问策略 $T^*$, 但可以执行密文更新询问;
2) 攻击者 $A$ 是一个已被撤销的用户. 他的属性集合可以满足挑战访问策略 $T^*$, 但密钥询问后必须执行撤销询问和密文更新询问.

令 $G$, $G_T$ 为 2 个阶为质数 $p$ 的循环群, $g$ 是群 $G$ 的一个生成元. 双线性映射 $e$:$G\times G\to G_T$. $UT$ 为用户二叉树, $L$ 是系统属性空间, $S$ 是用户属性集合.

- 初始化阶段: 攻击者 $A$ 将挑战访问策略 $T^*$和一个随机选择的挑战用户身份 $u^*$发送给挑战者 $C$;
- 系统建立阶段: $C$ 随机选择 $rev\in\{0,1\}$和一个叶子结点$\theta^*$, 将 $u^*$与叶子结点$\theta^*$相关联. 如果 $rev$=0, 则攻击者 $A$ 是一个非撤销用户; 如果 $rev$=1, 则攻击者 $A$ 是一个撤销用户. $C$ 运行 $Setup(\cdot)$算法, 随机选择 $a,b,\beta\in Z_p^*$, 对于二叉树中的结点$\forall i\in UT$, 算法选择 $x_i\in Z_p^*$, 计算 $y_i=g^{x_i}$, 并将 $y_i$ 存储在结点 $i$ 中. 对于任意属性 $j\in L$, 算法随机选择 $v_j\in Z_p^*$, 并计算 $A_j=g^{v_j}$, 设置哈希函数 $H_0:\{0,1\}^*\to Z_p^*$, $H_1$: $\{0,1\}^*\to G$, 令 $hash=(H_0,H_1)$. 将系统最大用户数量 $n$ 作为用户二叉树 $UT$ 叶子结点的个数. 算法输出系统公钥 $PK$、用户二叉树 $UT$ 以及系统主私钥 $MSK$:

$$PK=\{g,g^a,g^b,g^\beta,\{A_j\}_{j\in L},hash\}$$
$$MSK=\{a,b,\beta,\{x_i\}_{i\in UT},\{v_j\}_{j\in L}\}$$

  挑战者 $C$ 设置撤销列表 $R=\varnothing$, 将系统公钥 $PK$ 和用户二叉树 $UT$ 发送给攻击者 $A$, 自己保存系统主私钥 $MSK$;
- 询问阶段 1: 攻击者 $A$ 自适应性地发起以下询问.
  - 密钥询问: 攻击者 $A$ 选择一个用户身份 $u$ 和一个属性集合 $S$ 提交给 $C$. 根据攻击者 $A$ 身份的不同, $C$ 的响应信息也有所不同:
    1) 如果 $rev$=0 且属性集合 $S$ 满足挑战访问策略 $T^*$, 则终止询问. 由于攻击者 $A$ 是未撤销用户, 因此禁止查询满足挑战访问策略 $T^*$的密钥;
    2) 如果 $rev$=0 且属性集合 $S$ 不满足挑战访问策略 $T^*$. $C$ 随机选择 $r,t,\delta\in Z_p^*$, 计算

$D=g^{(ab+\beta t)/\delta}H_1(u)^r$, 对任意$\tau\in S$, 计算 $D_{\tau,1}=H_1(u)^{\delta r}g^{v_\tau r}$, $D_2=g^r$. $C$ 为用户 $u$ 选择一个未分配的结点$\theta$并获取与该结点相关的 $x_{i_u}$. $C$ 计算与用户相关的 $K_u=g^{\beta t/x_{i_u}}$, 生成用户密钥 $SK_u=\{D,\{D_{\tau,1}\}_{\tau\in S},D_2,K_u,\{x_i\}_{i\in path(i_u)},\delta\}$. $C$ 将用户密钥 $SK_u$ 发送给攻击者 $A$;

3) 如果 $rev$=1 且属性集合 $S$ 满足挑战访问策略 $T^*$. $C$ 将用户身份 $u$ 分配给叶子结点$\theta^*$. $C$ 随机选择 $r,t,\delta\in Z_p^*$, 计算与用户 $u$ 相关叶子结点的密钥为 $K_u=g^{\beta t/x_{\theta^*}}$. 计算 $D=g^{(ab+\beta t)/\delta}H_1(u)^r$, 对任意$\tau\in S$, 计算 $D_{\tau,1}=H_1(u)^{\delta r}g^{v_\tau r}$, $D_2=g^r$. 最终生成的用户密钥为 $SK_u=\{D,\{D_{\tau,1}\}_{\tau\in S},D_2,K_u,\{x_i\}_{i\in path(i_u)},\delta\}$. $C$ 将用户密钥 $SK_u$ 发送给攻击者 $A$;

4) 如果 $rev$=1 且属性集合 $S$ 不满足挑战访问策略 $T^*$. $C$ 为用户身份 $u$ 选择一个未分配的叶子结点$\theta$. $C$ 随机选择 $r,t,\delta\in Z_p^*$, 计算 $D=g^{(ab+\beta t)/\delta}H_1(u)^r$, 对任意$\tau\in S$, 计算 $D_{\tau,1}=H_1(u)^{\delta r}g^{v_\tau r}$, $D_2=g^r$. 对$\forall i\in path(\theta)$, 如果 $i\in(path(\theta)\backslash path(\theta^*))$, 选择一个随机数 $z_i\in Z_p^*$, 令 $x_i=z_i$. 令 $x_\theta=z_\theta$, 计算与用户 $u$ 相关的叶子结点的密钥为 $K_u=g^{\beta t/x_\theta}$. 如果 $i\in(path(\theta)\cap path(\theta^*))$, 则 $x_i$ 不变. 最终生成的用户密钥为 $SK_u=\{D,\{D_{\tau,1}\}_{\tau\in S},D_2,K_u,\{x_i\}_{i\in path(i_u)},\delta\}$. $C$ 将用户密钥 $SK_u$ 发送给攻击者 $A$;

- 撤销询问: 攻击者 $A$ 选择一个用户身份 $u$ 并提交给挑战者 $C$. 挑战者 $C$ 运行撤销算法 $Revocation(\cdot)$, 选择随机数 $x\in Z_p^*$ 替换主密钥 $MSK$ 中相应的 $x_u$, 计算 $y=g^x$ 替换 $UT$ 中相应的 $y_u$, 最后将用户身份 $u$ 加入撤销列表 $R$, 并公布新的用户二叉树 $UT$;
- 密文更新询问: 攻击者 $A$ 选择一个撤销列表 $R$ 和密文 $CT$ 提交给挑战者 $C$ 进行密文更新询问, $R'$ 为系统中旧的撤销列表, 密文为 $CT=\{T,C,C_0,\{C_{i,1},C_{i,2}\}_{i\in Y},\{T_j\}_{j\in cover(R)},Ind_{kw}\}$. 令 $cover(R)$表示与攻击者 $A$ 提交的撤销列表相关的最小覆盖集合, $cover(R')$表示与系统中旧撤销列表相关的最小覆盖集合. 对于 $cover(R)$中的结点 $j$:

1) 如果存在一个结点 $j'\in cover(R')$使得 $j=j'$, 则令 $T_j=T_{j'}$;

2) 如果存在一个结点 $j'\in cover(R')$使得 $j'$是 $j$ 的一个祖先结点, 令 $path(j)=path(j')\cup\{i_{temp(j')+1},\ldots,i_{temp(j)}\}$, 其中, $i_{temp(j)}=j$, $i_{temp(j')}=j'$. 令 $Y_{j'}=T_{j'}$, 并依次计算 $Y_{i_{k+1}}=(Y_{i_k})^{x_{i_{k+1}}/x_{i_k}}=y_{i_{k+1}}^s$, 其中, $k=temp(j'),\ldots,temp(j)$, 再设置 $T_j=Y_j$. 更新后的密文为

$$CT=\{T,C,C_0,\{C_{i,1},C_{i,2}\}_{i\in Y},\{T_j\}_{j\in cover(R)},Ind_{kw}\};$$

- 挑战阶段: 攻击者 $A$ 向 $C$ 提交 2 个等长的消息 $M_0$ 和 $M_1$.

1) 如果 $rev$=0, $C$ 随机选择 $c\in\{0,1\}$, 确定需要加密的消息 $M_c$. 选择随机值 $s,z\in Z_p^*$, $\mu\in\{0,1\}$. 如果 $\mu$=1, 计算 $C=M_c\cdot(g,g)^{sab}$, $C_0=g^s$; 否则, 计算 $C=M_c\cdot(g,g)^z$, $C_0=g^s$. 按照第 4.1 节中描述的步骤, 生成与访问策略相关的密文 $C_{i,1}=g^{q_i(0)}$, $C_{i,2}=g^{v_{attr(i)}q_i(0)}$, 其中, $i\in Y$. 对任意 $j\in cover(R)$, 计算 $T_j=y_j^s$. 最终生成密文 $CT_c=\{T,C,C_0,\{C_{i,1},C_{i,2}\}_{i\in Y},\{T_j\}_{j\in cover(R)},Ind_{kw}\}$;

2) 如果 $rev$=1 且 $u^*\notin R$, 则算法终止. 因为当 $rev$=1 时, 攻击者为撤销用户, 挑战身份必须位于撤销列表中;

3) 如果 $rev$=1 且 $u^*\in R$, $C$ 随机选择 $c\in\{0,1\}$, 确定需要加密的消息 $M_c$. 选择随机值 $s,z\in Z_p^*$, $\mu\in\{0,1\}$. 如果$\mu$=1, 计算 $C=M_c\cdot(g,g)^{sab}$, $C_0=g^s$; 否则, 计算 $C=M_c\cdot(g,g)^z$, $C_0=g^s$. 按照第 4.1 节中描述的步骤, 生成与访问策略相关的密文 $C_{i,1}=g^{q_i(0)}$, $C_{i,2}=g^{v_{attr(i)}q_i(0)}$, 其中, $i\in Y$. 对任意 $j\in cover(R)$, 计算 $T_j=y_j^s$. 最终生成挑战密文 $CT_c=\{T,C,C_0,\{C_{i,1},C_{i,2}\}_{i\in Y},\{T_j\}_{j\in cover(R)},Ind_{kw}\}$;

- 询问阶段 2: 攻击者 $A$ 执行询问阶段 1 的询问;
- 猜测阶段: 攻击者 $A$ 将对 $c$ 的猜测 $c'$提交给 $C$. 如果 $CT_c$ 为无效密文, 此时攻击者 $A$ 仅能随机猜测, 其猜对的概率为 $\Pr[B(g^a,g^b,g^s,e(g,g)^z)=0]=1/2$. 如果 $CT_c$ 为有效密文, 此时攻击者 $A$ 猜对的概率为

$\Pr[B(g^a,g^b,g^s,e(g,g)^{abs})=1]=1/2+\varepsilon$.综上所述, 攻击者 $A$ 获胜的优势为

$$1/2\Pr[B(g^a,g^b,g^s,e(g,g)^{abs})=1]+1/2\Pr[B(g^a,g^b,g^s,e(g,g)^z)=0]-1/2=\varepsilon/2.$$

在以上安全证明中, $C$ 模拟的系统参数、用户密钥和密文的分布都与本文方案完全一致. 定理 1 证毕. □

(2) 关键词不可区分安全

定理 2 保证了本文方案在选择关键词攻击下满足不可区分安全性, 其证明过程通过模拟攻击者 $A$ 与挑战者 $C$ 之间的安全游戏来实现. 游戏共分为 5 个阶段. 首先, 在系统建立阶段, 为了保证系统的运行, 需要挑战者 $C$ 执行 *Setup*(·)算法, 生成必要的系统公钥、用户二叉树以及系统主私钥. 挑战者 $C$ 公开系统公钥与用户二叉树, 秘密保存系统主私钥. 在询问阶段 1, 攻击者 $A$ 按照其身份向挑战者 $C$ 发起询问请求来模拟其在攻击之前的信息收集行为. 在关键词不可区分安全的证明中, 攻击者 $A$ 可以向挑战者 $C$ 询问不同关键词所对应的陷门. 在挑战阶段, 攻击者 $A$ 向挑战者 $C$ 提交 2 个等长的关键词, 挑战者 $C$ 随机选择一个并生成对应的关键词陷门作为挑战目标, 我们要求攻击者 $A$ 提交的关键词不处于询问阶段 1 中询问的范围之内. 随后, 攻击者 $A$ 在询问阶段 2 中继续向挑战者 $C$ 进行询问, 此时要求攻击者 $A$ 的询问内容不与挑战阶段提交的关键词相同. 最后, 在猜测阶段, 攻击者 $A$ 输出对挑战目标的猜测结果, 如果与挑战者 $C$ 选择的目标相一致, 则攻击者 $A$ 赢得这个游戏. 令攻击者 $A$ 赢得该游戏的概率为 $P$, 考虑到攻击者 $A$ 有 1/2 的概率可以猜到正确的结果, 将攻击者 $A$ 赢得该游戏的优势定义为 $P-1/2$. 由于关键词陷门的构造是基于 DL 问题完成的, 如果攻击者 $A$ 在多次游戏中获胜的优势是不可忽略的, 那么就可以同样以这个优势破解 DL 问题.

**定理 2**. 如果 DL 假设成立, 那么本文方案在选择关键词攻击下满足不可区分安全性.

证明: 假设存在一个概率多项式时间攻击者 $A$ 能以不可忽略的优势$\varepsilon$破解本文方案, 那么就可以构造一个概率多项式时间算法以$\varepsilon$的优势破解 DL 假设.

- 系统建立阶段: $C$ 运行 *Setup*(·)算法, 随机选择 $a,b,\beta\in Z_p^*$, 对于二叉树中的结点$\forall i\in UT$, 算法选择 $x_i\in Z_p^*$, 计算 $y_i=g^{x_i}$, 并将 $y_i$ 存储在结点 $i$ 中. 对于任意属性 $j\in L$, 算法随机选择 $v_j\in Z_p^*$, 并计算 $A_j=g^{v_j}$, 设置哈希函数: $H_0:\{0,1\}^*\to Z_p^*$, $H_1:\{0,1\}^*\to G$, 令 $hash=(H_0,H_1)$. 将系统最大用户数量 $n$ 作为用户二叉树 $UT$ 叶子结点的个数. 算法输出系统公钥 $PK$、用户二叉树 $UT$ 以及系统主私钥 $MSK$:

$$PK=\{g,g^a,g^b,g^\beta,\{A_j\}_{j\in L},hash\}$$
$$MSK=\{a,b,\beta,\{x_i\}_{i\in UT},\{v_j\}_{j\in L}\}$$

  $C$ 将系统公钥 $PK$ 和用户二叉树 $UT$ 发送给攻击者 $A$, $C$ 保存系统主私钥 $MSK$;
- 询问阶段 1: 攻击者 $A$ 适应性地发起关键词询问, 其定义如下.
  - 关键词询问: 攻击者 $A$ 选择一个用户身份 $u$、属性集合 $S$ 以及关键词 $SKW$ 提交给挑战者 $C$. $C$ 随机选择 $r,t,\delta\in Z_p^*$, 计算 $D=g^{(ab+\beta t)/\delta}H_1(u)^r$, 对于$\forall\tau\in S$, 计算 $D_{\tau,1}=H_1(u)^{\delta r}g^{v_\tau r}$, $D_2=g^r$. $C$ 计算与用户 $u$ 相关的用户密钥 $K_u=g^{\beta t/x_{i_u}}$. 生成用户密钥 $SK_u=\{D,\{D_{\tau,1}\}_{\tau\in S},D_2,K_u,\{x_i\}_{i\in path(i_u)},\delta\}$. $C$ 选择随机数 $d\in Z_p^*$, 计算关键词陷门为 $trapdoor=\{td_1,\{td_{2,i}\}_{i\in path(i_u)}\}$, 其中, $td_1=g^{dH_0(SKW)/\delta}$, $td_{2,i}=\{g^{dx_i/\delta}\}_{i\in path(i_u)}$. $C$ 将陷门发送给攻击者 $A$;
- 挑战阶段: 攻击者 $A$ 向 $C$ 提交 2 个等长的关键词 $SKW_1$ 和 $SKW_2$. $C$ 随机选择 $c\in\{0,1\}$并运行密钥生成算法 *KeyGen*(·)和陷门生成算法 *Trapdoor*(·), 生成挑战陷门 $trapdoor_c=\{g^{dH_0(SKW_a)/\delta},\{g^{dx_i/\delta}\}_{i\in path(i_u)}\}$. 要求 $SKW_1$ 和 $SKW_2$ 不能是在询问阶段 1 中出现过的关键词;
- 询问阶段 2: 攻击者 $A$ 执行询问阶段 1 的关键词询问, 其中的限制条件为: 不能询问关键词 $SKW_1$ 和 $SKW_2$;
- 猜测阶段: 当攻击者 $A$ 结束询问阶段 2 之后, 输出一个猜测结果 $c'\in\{0,1\}$. 如果 $c=c'$, 那么称攻击者 $A$ 赢得了这个游戏. 由于攻击者 $A$ 获胜的概率来源于随机数 $c$、哈希函数 $H_0(\cdot)$在 $Z_p^*$ 上的随机选取以及 DL 问题的破解难度, 因此攻击者 $A$ 无法以不可忽略的优势赢得该游戏.

定理 2 证毕. □

(3) 前后向安全

前向安全是指: 当某个用户被撤销之后, 将无法访问撤销前的加密数据. 后向安全是指: 当某个用户被撤销之后, 将无法访问撤销后的加密数据. 由于在解密过程中需要用户二叉树中存在一个满足 $cover(R)\cap path(i_u)$的结点 $j$, 获取其中的 $x_j$, 从而计算 $B=e(K_u,T_j)^{x_{i_u}/x_j}$ (见第 4.1 节算法(6)), 因此, 本文的前后向安全都是由用户二叉树中与用户相关的随机值来保证的. 在撤销时, 可信中心会选择一个新的随机数 $x$ 来替换主密钥中以及用户二叉树中与撤销用户相关的值, 同时将撤销用户加入撤销列表(见第 4.1 节算法(7)), 即同时使用一个随机值替换主密钥、用户二叉树中与被撤销用户相关的结点值. 因此, 该撤销用户无法在用户二叉树中找到满足 $cover(R)\cap path(i_u)$的结点也就无法完成解密, 从而保证本文方案的后向安全性. 在前向安全性方面, 每当用户撤销之后, 系统都会执行一次密文更新算法. 通过对比与新旧撤销列表相关的最小覆盖集合, 更新密文中的$\{T_j\}_{j\in cover(R)}$(见第 4.1 节算法(8)), 通过保证撤销用户无法计算 $B=e(K_u,T_j)^{x_{i_u}/x_j}$, 来保证本文方案的前向安全性.

(4) 抗关键词猜测攻击

考虑到恶意的云服务器可能会为了获取用户搜索的内容遍历生成关键词空间的所有索引, 再与用户提交的搜索陷门(关键词)进行匹配. 在本文方案中, 用户提交的搜索陷门为 $trapdoor=\{td_1,\{td_{2,i}\}_{i\in path(i_u)}\}$, 其中, $td_1=g^{dH_0(SKW)/\delta}$, $td_{2,i}=\{g^{dx_i/\delta}\}_{i\in path(i_u)}$. 由于 $d$ 和$\delta$均为系统选择的随机数, 且没有指定关键词空间, 关键词的选择和加密完全依赖于用户选择关键词的随机性和哈希函数 $H_0(\cdot)$以及 $d$, $\delta$在 $Z_p^*$ 上的随机选择, 因此, 恶意的云服务器并不能完成遍历生成关键词空间的所有索引的操作. 此外, 由于本文方案可以实现关键词不可区分安全, 云服务器也无法猜测关键词的具体内容, 因此, 本文方案可以抵抗关键词猜测攻击.

(5) 抗共谋攻击

本文方案中, 用户密钥 $SK$ 分为 2 个部分: 一部分与用户属性集合相关, 另一部分与撤销列表相关.因此在抵抗共谋攻击方面, 需要考虑非撤销用户之间的共谋和撤销用户与非撤销用户之间的共谋这两种情况.

当共谋用户是非撤销用户时, 这些用户所拥有的属性往往并不能够为他们提供足够的访问权限来解密数据, 因此, 需要将他们的密钥合并起来尝试解密. 由于用户密钥中与属性相关的部分为 $D_{\tau,1}=H_1(u)^{\delta r}g^{v_\tau r}$, $D_2=g^r$, 其中, $r$ 为随机值, $v_\tau$与用户属性相关, $u'$与用户身份相关, 因此, 不同用户之间即使属性相同, 这些属性所生成的密钥也互不相同, 无法合并密钥进行解密.

当共谋用户同时包含撤销用户与非撤销用户时, 往往撤销用户拥有足够的属性而非撤销用户拥有解密资格. 因此, 可能会出现撤销用户提供与属性相关的密钥, 非撤销用户提供与身份(撤销列表)相关的密钥的情况. 在本文方案中, 与属性相关的密钥为 $D=g^{(ab+\beta t)/\delta}H_1(u)^r$, $D_{\tau,1}=H_1(u)^{\delta r}g^{v_\tau r}$, $D_2=g^r$, 与撤销列表相关的密钥为 $K_u=g^{\beta t/x_{i_u}}$, 其中, $r$, $t$, $\beta$为随机值, $x_{i_u}$ 与用户身份相关, $\tau\in S$. 由于随机值$\beta$的存在, 为了保证随机值的一致, $D$ 与 $K_u$ 必须由同一个用户提供. 而在与属性相关的密钥 $D_{\tau,1}$ 中包含用户的身份信息 $H_1(u)$, 如果与 $K_u$ 中的用户信息不一致, 则无法完成解密. 因此, 本文方案可以抵抗撤销用户与非撤销用户之间的共谋攻击.

## 5 性能分析

### 5.1 理论分析

在本节中, 将本文的方案与文献[13]、文献[14]以及文献[33]中的方案从计算开销的角度进行对比, 其中, $t$ 代表访问策略中属性的数量, $k$ 代表用户密钥中属性的数量, $L$ 代表系统中所有的属性数量, $r$ 代表 $cover(R)$中结点的数量, $U$ 代表系统中用户的数量, $n$ 代表系统中用户数量的上限, $G$, $G_T$ 分别代表群 $G$ 和 $G_T$ 上的一次运算, $C_e$ 代表一次双线性配对计算. 表 3 分别对比了系统建立、加密、密钥生成、解密、搜索以及密文更新 6 个算法的计算开销.

表 3 计算开销

| 方案 | 系统建立 | 加密 | 密钥生成 | 解密 | 搜索 | 密文更新 |
|---|---|---|---|---|---|---|
| 文献[13] | $(L+2)G+G_T$ | $(2k+2)G+G_T$ | $(4k+8)G$ | $(2t+1)C_e+3G+(t+3)G_T$ | $2C_e$ | $G_T+C_e$ |
| 文献[14] | $2nG+G_T$ | $(5k+r+2)G+2G_T$ | $(3k+6)G$ | $(3t+2)C_e+(2t+7)G+(2t+1)G_T$ | – | 0 or $((1+\log_2 n)\times\log_2 n)/2$ |
| 文献[33] | $(2L+4)G$ | $(k+5)G+C_e$ | $(2k+L+3)G$ | – | $2C_e+(k+L)G$ | – |
| 本文方案 | $(2n+L)G+G_T$ | $(2k+r+1)G+2G_T$ | $(3k+5)G$ | $(2t+2)C_e+(t+5)G_T$ | $2C_e$ | 0 or $((1+\log_2 n)\times\log_2 n)/2$ |

与文献[13]相比, 本文方案在系统建立算法中需要额外为用户二叉树中每个节点进行一次群 $G$ 上的指数运算, 但由于该算法只需要由可信中心执行一次, 对数据用户的影响较小. 在加密算法部分, 本文的计算开销为$(2k+r+1)G+2G_T$, 其中, $r$ 代表 $cover(R)$中结点的数量. 由第 2.6 节的定义可知: 无论撤销列表 $R$ 中包含多少撤销用户, $r\in[1,U/2]$(仅在$|R|=U/2$ 且处于最糟情况下才会取到 $U/2$). 而文献[13]由于采用“版本控制”的方法来实现用户撤销, 因此加密算法的计算开销与撤销用户的数量无关. 在密钥生成算法中, 本文方案的计算开销为$(3k+5)G$, 比文献[13]更优. 在解密算法部分, 本文方案的计算开销为$(2t+2)C_e+(t+5)G_T$, 与文献[13]基本一致. 类似的, 在密文搜索算法中, 本文方案的计算开销与文献[13]同样相差不大. 在密文更新算法中, 文献[13]的计算开销固定为 $G_T+C_e$, 而在本文方案中, 由于密文更新的计算开销会随着撤销列表中用户在用户二叉树中位置的不同而变化, 因此我们分别给出了最优以及最差情况下的计算开销: 当处于最优情况时, 算法只需从原密文中删去部分元素, 计算开销为 0; 处于最差情况时, 密文更新的计算开销为$((1+\log_2 n)\times\log_2 n)/2$. 虽然本文方案在加密算法中的计算开销较大, 但在密文更新时, 本文方案仅需更新密文中的一小部分, 并且无需更新用户密钥. 而在文献[13]中, 当撤销发生之后, 除了要更新密文, 还需要为系统中未撤销的用户分发更新密钥.

与文献[14]相比, 本文方案在系统建立算法中需要额外为系统预设的所有属性进行一次群 $G$ 上的指数运算. 同样, 由于该算法只需要由可信中心执行一次, 对数据用户的影响较小. 在加密、密钥生成以及解密算法部分, 本文方案的计算开销分别为$(2k+r+1)G+2G_T$, $(3k+5)G$, $(2t+2)C_e+(t+5)G_T$, 与文献[14]相比有明显的优势. 在密文更新算法方面, 本文方案的计算开销与文献[14]相一致. 在功能上, 本文方案支持重复使用用户二叉树中撤销用户的节点, 并且以较低的计算开销($2C_e$)提供了文献[14]所不具备的密文搜索功能.

与文献[33]相比, 本文方案在系统建立算法中需要额外为用户二叉树中每个节点进行一次群 $G$ 上的指数运算, 但由于该算法只需要由可信中心执行一次, 因此对数据用户的影响较小. 在加密算法阶段, 虽然本文方案的计算开销较大, 但这是由于文献[33]的加密算法仅包括生成关键词密文的计算, 并没有涉及消息加密的计算过程. 也是基于同样的原因, 文献[33]中由于没有消息密文, 没有设计相应的解密与密文更新算法, 因此表 3 中无法统计文献[33]的解密以及密文更新计算开销. 在密钥生成阶段, 本文方案的计算开销为$(3k+5)G$, 文献[33]的计算开销为$(2k+L+3)G$. 由于 $L$ 代表系统中所有的属性数量, $k$ 代表用户密钥中属性的数量, 因此本文方案的计算开销更优. 在密文更新算法中, 本文方案的计算开销为 $2C_e$, 比文献[33]更优.

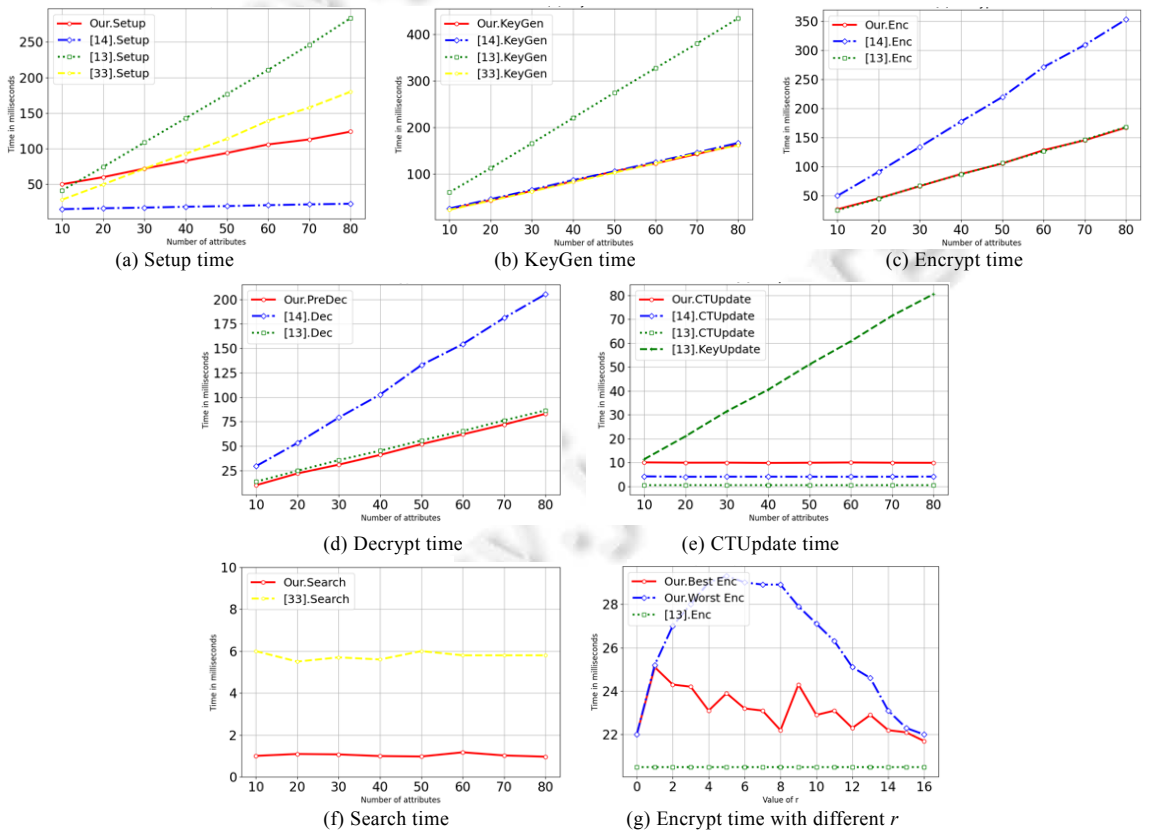
在使用二叉树实现用户撤销的属性基加密方案中, 用户数量受用户二叉树叶子节点数量的限制存在上限. 目前常见的解决方法是生成一个新的根节点, 将当前的用户二叉树作为新的根节点的一个子树, 同时为新的根节点生成一个与当前用户二叉树一样大小的空白子树(如当前用户二叉树的根节点作为新生成根节点的左孩子, 新生成的空白子树的根节点作为新生成根节点的右孩子). 按这样的方法, 每次可以将系统中用户的数量增加一倍. 然而, 无限地扩展用户二叉树, 却也会导致系统计算开销的增长. 此外, 在现存的方案中, 用户撤销算法同样存在问题. 在撤销算法中, 可信中心需要将撤销用户加入撤销列表更新 $cover(R)$, 使得 $path(u)$与 $cover(R)$不存在交集, 其中, $u$ 代表撤销用户. 一旦后续将与该用户相关联的节点分配给其他新用户 $u'$, 更新后的 $cover(R)$必然与 $path(u')$存在交集, 此时, 如果撤销用户 $u$ 仍然持有曾经的密钥, 就可以继续访问系统中的数据. 这样一来, 当一个用户数量为 $U$ 的系统执行过 $U$ 次用户撤销后, 将不得不选择前文所述的方法扩充系统用户数量. 考虑到在诸如公司内部数据共享的应用场景中, 数据用户的数量相对固定的情况, 我们将本文的目标定为如何重复利用用户二叉树中与被撤销用户相关联的节点, 缓解该类方案中撤销用户数量存在上限的问题. 为了达成这一目标, 本文方案在执行撤销时, 除了将撤销用户加入撤销列表, 还会同时更新

存储在用户二叉树中与撤销用户相关联的随机值. 这样, 即使后续将该节点分配给了其他新用户, 由于节点中存储的随机值并不一致, 撤销用户也无法完成解密, 因此可以安全地实现节点的再使用.

### 5.2 模拟结果及分析

为了更全面地展示本文方案在实际运行中的性能, 本节通过模拟实验的方法分别对本文方案与文献[13]、文献[14]各个算法的运行时间进行统计. 实验基于 Charm 框架[34], 使用 Python3 编程实现所有的算法, 其中, 双线性映射采用 Charm 框架中的默认曲线“SS512”. 实验的平台是一台 Macbook Pro 笔记本电脑, 处理器为 Inter Core i7 (2.7 GHz), 内存 16 GB. 实验中的访问策略均以 AND 互相连接, 如: $A_1$ AND $A_2$ AND … AND $A_n$, 其中, $A_i$ 代表一个属性.在密钥生成阶段(如图 3(b)所示)和加密阶段(如图 3(c)所示), 用户属性集中属性的数量和访问策略中属性的数量按照 10,20,...,80 的规律递增. 系统中用户数量为 10, 且每次撤销 1 个用户. 此外, 为了更全面地展示本文方案加密算法的执行时间与撤销用户数量之间的关系, 我们在图 3(g)中设置访问策略中的属性数量为 10, 用户数量为 16, 且按照 0,1,...,16 的规律依次撤销. 考虑到本文方案中 $r$ 的值会随着待撤销用户在用户二叉树中位置的不同而变化, 我们分别统计了最理想以及最差情况下加密算法的时间开销, 与文献[13]进行比较. 为了保证统计结果的可信度, 每个算法都重复运行了 20 次, 以统计平均运行时间.

图 3 展示了 *Setup*(·), *KeyGen*(·), *Encryption*(·), *CTUpdate*(·), *Decryption*(·)以及 *Search*(·)这 6 个算法的运行时间对比情况, 其中, 图 3(a)是系统建立运行时间对比, 图 3(b)是密钥生成运行时间对比, 图 3(c)是加密运行时间对比, 图 3(d)是解密运行时间对比, 图 3(e)是密文更新运行时间对比, 图 3(f)是搜索运行时间对比, 图 3(g)是加密算法时间开销随撤销用户数量变化情况的对比.




(a) Setup time (b) KeyGen time (c) Encrypt time

(d) Decrypt time (e) CTUpdate time

(f) Search time (g) Encrypt time with different $r$

图 3 运行时间对比

相比于文献[13], 本文方案在 *Setup*(·)和 *KeyGen*(·)中的时间开销有了明显的改善. 本文方案与文献[14]在

*KeyGen*(·)中的时间开销基本一致, 但在*Setup*(·)中, 本文方案的时间开销却偏大. 这是由于本文方案在*Setup*(·)过程中, 为属性空间中的所有属性进行了初始化操作. 因此, 随着属性数量的增加, 时间开销的增长相比于文献[14]会更快. 虽然文献[14]并没有对属性空间初始化的操作, 但在后续的加密算法中, 为了提供访问策略隐藏的功能, 需要对属性名和属性值分别进行处理. 也就是说, 在使用之前仍然需要对系统中的属性进行初始化操作, 因此, 时间开销存在进一步增加的可能. 相比于文献[33], 本文方案虽然在系统中属性数量较少时, *Setup*(·)算法的时间开销较大, 但随着系统中属性数量的增加, *Setup*(·)算法的时间开销逐渐优于文献[33]. 在*Encryption*(·)算法中, 本文方案与文献[13]基本一致, 和文献[14]相比, 虽然都需要生成与用户二叉树相关密文的部分, 但在生成与访问策略相关密文的部分时, 本文方案的计算量更低, 因此时间开销更低. 本文方案在*KeyGen*(·)中的时间开销与文献[33]接近, 但在 *Encryption*(·)算法以及 *Decryption*(·)算法中, 由于文献[33]并不涉及消息加解密的计算过程, 因此我们没有统计其时间消耗情况. 本文方案的 *Decryption*(·)时间开销优于文献[14], 这是由于在解密与访问策略相关的密文时, 需要更少的双线性映射运算. 在 *CTUpdate*(·)中, 虽然本文方案计算开销高于文献[13]和文献[14], 但在撤销时, 本文方案并不需要更新所有未撤销用户的密钥. 而由图3(e)可知: 文献[13]在更新用户私钥时的计算开销随着用户属性的数量线性增长, 总体时间开销大大超过了本文方案和文献[14]. 本文方案比文献[14]在密文更新时需要多花费大约 5ms 的计算时间, 但由于本文方案可以有效地利用用户二叉树中的叶子结点, 在功能上更加实用. 在 *Search*(·)中, 本文方案与文献[33]的计算开销都与系统中属性的数量无关, 虽然本文方案的时间开销更低, 但文献[33]可以将单关键词的搜索算法扩展为多关键词搜索, 功能上更加灵活. 综合图 3(g)中本文方案加密时间随撤销用户数量变化最好以及最坏的情况来看, 本文方案的加密时间开销略大于文献[13]的方案, 而文献[13]方案的加密时间与撤销用户的数量无关. 总的来说, 本文方案有效地丰富了系统功能, 提升了系统运行效率, 提高了可撤销属性基加密方案中用户二叉树的实用性.

## 6 总 结

为了降低属性基加密系统在用户撤销以及密文更新时的计算开销, 提高利用二叉树实现用户撤销时的实用性, 本文提出了一种适用于用户动态变化场景的可搜索属性基加密方案. 通过更新用户二叉树中结点随机值的方式, 使得新加入的用户可以重新使用被撤销后(用户使用过)的结点, 且之前被撤销的用户仍无法访问系统中的数据, 保障了系统中数据的安全. 在密文搜索方面, 本文方案提供了一种高效的密文匹配机制, 可以防止撤销用户通过搜索功能获取存储在系统中的密文, 提高了方案的实用性和安全性. 本文方案在随机谕言模型下满足选择明文不可区分安全性以及关键词不可区分安全性. 性能分析和实验数据表明: 本文方案与同类方案相比, 在计算效率上具有一定优势.

**References**:

[1] Reinsel D, Gantz J, Rydning J. Data age 2025: The evolution of data to life-critical. 2017. https://www.import.io/wp-content/uploads/2017/04/Seagate-WP-DataAge2025-March-2017.pdf

[2] Yan XX, Yuan XH, Yang YL, *et al*. Verifiable attribute-based searchable encryption scheme based on blockchain. Journal on Communications, 2020, 41(2): 187−198 (in Chinese with English abstract). [doi: 10.11959/j.issn.1000-436x.2020011]

[3] Ma HY, Wang ZJ, Guan ZJ. Efficient ciphertext-policy attribute-based online/offline encryption with user revocation. Security and Communication Networks, 2019, 2019: 8093578:1−8093578:11. [doi: 10.1155/2019/8093578]

[4] Sahai A, Waters B. Fuzzy identity-based encryption. In: Ronald C, ed. Proc. of the 24th Annual Int'l Conf. on the Theory and Applications of Cryptographic Techniques. Aarhus: Springer, 2005. 457−473. [doi: 10.1007/11426639_27]

[5] Waters B. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In: Dario C, Nelly F, Rosario G, Antonio N, eds. Proc. of the 14th Int'l Conf. on Practice and Theory in Public Key Cryptography. Taormina: Springer, 2011. 53−70. [doi: 10.1007/978-3-642-19379-8_4]

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

[7] Sun L, Zhao ZY, Wang JH, *et al*. Attribute-based encryption scheme supporting attribute revocation in cloud storage environment. Journal on Communications, 2019, 40(5): 47−56 (in Chinese with English abstract). [doi: 10.11959/j.issn.1000−436x. 2019116]

[8] Wang T, Ma H, Zhou YB, *et al*. Fully accountable data sharing for pay-as-you-go cloud scenes. IEEE Trans. on Dependable and Secure Computing, 2019, 18(4): 2005−2016. [doi: 10.1109/TDSC.2019.2947579]

[9] Goyal V, Pandey O, Sahai A, *et al*. Attribute-Based encryption for fine-grained access control of encrypted data. In: Juels A, Wright RN, eds. Proc. of the 13th ACM Conf. on Computer and Communications Security. Alexandria: ACM, 2006. 89−98. [doi: 10.1145/1180405.1180418]

[10] Ostrovsky R, Sahai A, Waters B. Attribute-based encryption with non-monotonic access structures. In: Ning P, Vimercati SDC, Syverson PF, eds. Proc. of the 14th ACM Conf. on Computer and Communications Security. Alexandria: ACM, 2007. 195−203. [doi: 10.1145/1315245.1315270]

[11] Zhang MQ, Du WD, Yang XY, *et al*. A full secure KP-ABE scheme in the standard model. Journal of Computer Research and Development, 2015, 52(8): 1893−1901 (in Chinese with English abstract). [doi: doi:0.7544/issn1000-1239.2015.20140605]

[12] Liu JH, Ma JH, Xiang Y, *et al*. Authenticated medical documents releasing with privacy protection and release control. IEEE Trans. on Dependable and Secure Computing, 2019, 18(1): 448−459. [doi: 10.1109/TDSC.2019.2892446]

[13] Wang JW, Yin XC, Ning JT, *et al*. Attribute-based encryption with efficient keyword search and user revocation. In: Guo FC, Huang XY, Yung M, eds. Proc. of the 14th Int'l Conf. on Information Security and Cryptology. Fuzhou: Springer, 2018. 490−509. [doi: 10.1007/978-3-030-14234-6_26]

[14] Han DZ, Pan NN, Li KC. A traceable and revocable ciphertext policy attribute-based encryption scheme based on privacy protection. IEEE Trans. on Dependable and Secure Computing, 2020. [doi: 10.1109/TDSC.2020.2977646]

[15] Pirretti M, Traynor P, Mcdaniel P. Secure attribute-based systems. In: Proc. of the 13th Conf. on Computer and Communications Security. Alexandria: ACM, 2006. 99−112. [doi: 10.1145/1180405.1180419]

[16] Yang K, Jia XH. Expressive, efficient, and revocable data access control for multi-authority cloud storage. IEEE Trans. on Parallel and Distributed System, 2014, 25(7): 1735−1744. [doi: 10.1109/TPDS.2013.253]

[17] Varri US, Kasani S, Pasupuleti SK, *et al*. Felt-ABKS: Fog-enabled lightweight traceable attribute-based keyword search over encrypted data. IEEE Internet of Things, 2021. [doi: 10.1109/JIOT.2021.3139148]

[18] Hoang V, Lehtihet E, Doudane YG. Forward-secure data outsourcing based on revocable attribute-based encryption. In: Proc. of the 15th Int'l Wireless Communications & Mobile Computing Conf. Tangier: IEEE, 2019. 1836−1846. [doi: 10.1109/IWCMC.2019. 8766674]

[19] Chen JW, Ma HD. Efficient decentralized attribute-based access control for cloud storage with user revocation. In: Proc. of the IEEE Int'l Conf. on Communications. Sydney: IEEE, 2014. 3782−3787. [doi: 10.1109/ICC.2014.6883910]

[20] Guo R, Yang G, Shi HX, *et al*. $O^3$-r-cp-abe: An efficient and revocable attribute-based encryption scheme in the cloud-assisted IoMT system. IEEE Internet of Things Journal, 2021, 8(11): 8949−8963. [doi: 10.1109/JIOT.2021.3055541]

[21] Ge CP, Susilo W, Baek J, *et al*. Revocable attribute-based encryption with data integrity in clouds. IEEE Trans. on Dependable and Secure Computing, 2021. [doi: 10.1109/TDSC.2021.3065999]

[22] Wei JH, Chen XF, Huang XY, *et al*. Rs-habe: Revocable-storage and hierarchical attribute-based access scheme for secure sharing of e-health records in public cloud. IEEE Trans. on Dependable and Secure Computing, 2019: 18(5): 2301−2315. [doi: 10.1109/TDSC.2019.2947920]

[23] Zhang JW, Ma JF, Ma Z, *et al*. Time-based and privacy protection revocable and traceable data sharing scheme in cloud computing. Journal on Communications, 2021, 42(10): 81−94 (in Chinese with English abstract). [doi: 10.11959/j.issn.1000−436x.2021206]

[24] Yang Y, Liu XM, Zheng XH, *et al*. Efficient traceable authorization search system for secure cloud storage. IEEE Trans. on Cloud Computing, 2020, 8(3): 819−832. [doi: 10.1109/TCC.2018.2820714]

[25] Boldyreva A, Goyal V, Kumar V. Identity-based encryption with efficient revocation. In: Ning P, Syverson PF, Jha S, eds. Proc. of the 15th ACM Conf. on Computer and Communications Security. Alexandria: ACM, 2008. 417−426. [doi: 10.1145/1455770. 1455823]

[26] 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]

[27] Curtmola R, Garay JA, Kamara S, *et al*. Searchable symmetric encryption: Improved definitions and efficient constructions. In: Juels A, Wright RN, Vimercati SDC, eds. Proc. of the 13th Conf. on Computer and Communications Security. Alexandria: ACM, 2006. 79–88. [doi: 10.1145/1180405.1180417]

[28] Li HW, Yang Y, Dai YS, *et al*. Achieving secure and efficient dynamic searchable symmetric encryption over medical cloud data. IEEE Trans. on Cloud Computing, 2020, 8(2): 484–494. [doi: 10.1109/TCC.2017.2769645]

[29] Awad A, Matthews A, Qiao Y, *et al*. Chaotic searchable encryption for mobile cloud storage. IEEE Trans. on Cloud Computing, 2018, 6(2): 440–452. [doi: 10.1109/TCC.2015.2511747]

[30] Liu SH, Yu JG, Xiao YH, *et al*. Bc-Sabe: Blockchain-aided searchable attribute-based encryption for cloud-IoT. IEEE Internet of Things Journal, 2020, 7(9): 7851–7867. [doi: 10.1109/JIOT.2020.2993231]

[31] He K, Guo J, Weng J, *et al*. Attribute-based hybrid Boolean keyword search over outsourced encrypted data. IEEE Trans. on Dependable and Secure Computing, 2020, 17(6): 1207–1217. [doi: 10.1109/TDSC.2018.2864186]

[32] Miao YB, Ma JF, Liu XM, *et al*. Attribute-based keyword search over hierarchical data in cloud computing. IEEE Trans. on Service Computing, 2020, 13(6): 985–998. [doi: 10.1109/TSC.2017.2757467]

[33] Sultan NH, Kaaniche N, Laurent M, *et al*. Authorized keyword search over outsourced encrypted data in cloud environment. IEEE Trans. on Cloud Computing, 2019. [doi: 10.1109/TCC.2019.2931896]

[34] Akinyele JA, Green M, Rubin AD. Charm: A framework for rapidly prototyping cryptosystems. In: Proc. of the 19th Annual Network and Distributed System Security Symp. San Diego: The Internet Society, 2012. 111–128. [doi: 10.1007/s13389-013-0057-3]

**附中文参考文献**:

[2] 闫玺玺, 原笑含, 汤永利, 等. 基于区块链且支持验证的属性基搜索加密方案. 通信学报, 2020, 41(2): 187–198. [doi: 10.11959/j.issn.1000−436x.2020011]

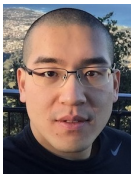
[7] 孙磊, 赵志远, 王建华, 等. 云存储环境下支持属性撤销的属性基加密方案. 通信学报, 2019, 40(5): 47–56. [doi: 10.11959/j.issn.1000−436x.2019116]

[11] 张敏情, 杜卫东, 杨晓元, 等. 标准模型下全安全的密钥策略属性基加密方案. 计算机研究与发展, 2015, 52(8): 1893–1901. [doi: 0.7544/issn1000-1239.2015.20140605]

[23] 张嘉伟, 马建峰, 马卓, 等. 云计算中基于时间和隐私保护的可撤销可追踪的数据共享方案. 通信学报, 2021, 42(10): 81–94. [doi: 10.11959/j.issn.1000−436x.2021206]

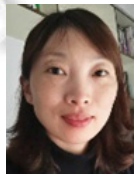
**王经纬**(1993－), 男, 博士生, CCF 学生会员, 主要研究领域为属性基加密.

**宁建廷**(1988－), 男, 博士, 教授, 博士生导师, 主要研究领域为密码学与数据安全, 区块链安全, 隐私保护技术.



**许胜民**(1989－), 男, 博士, 主要研究领域为密码学与数据安全.

**殷新春**(1962－), 男, 博士, 教授, 博士生导师, CCF 高级会员, 主要研究领域为密码学, 高性能计算.



**陈海霞**(1982－), 女, 博士生, CCF 学生会员, 主要研究领域为图像信息隐藏, 图像数据认证, 密码与信息安全.