基于改进 Merkle-Tree 认证方法的可验证多关键词搜索方案

田有亮[1,2,3]，骆琴[1,3,4]

（1. 公共大数据国家重点实验室（贵州大学），贵州 贵阳 550025；2. 贵州大学计算机科学与技术学院，贵州 贵阳 550025；
3. 贵州大学密码学与数据安全研究所，贵州 贵阳 550025；4. 贵州大学数学与统计学院，贵州 贵阳 550025）

摘 要：针对可搜索加密方案中的结果验证方法复杂，搜索成本高且效率低，难以满足多关键词搜索结果高效验证和安全性需求的问题，提出了基于改进的 Merkle-Tree 认证方法的多关键词搜索方案。首先，利用双线性映射构造多关键词的可搜索算法，实现高效精准的多关键词搜索；其次，基于 Bawa 改进的 Merkle-Tree 认证方法构造搜索方案的验证及动态更新算法，将计算成本从经典的 MHT 的 $O(n)$ 降低到 $O(\log n)$，防止数据篡改、删除和伪造等不法操作的高效验证。在决策线性假设和 CDH 假设下，所提方案满足密文不可区分性和签名不可伪造性。

关键词：云计算；搜索隐私性；有效验证；动态更新

中图分类号：TP309

文献标识码：A

doi: 10.11959/j.issn.1000−436x.2020146

Verifiable multi-keyword search scheme based on improved Merkle-Tree authentication method

TIAN Youliang[1,2,3], LUO Qin[1,3,4]

1. State Key Laboratory of Public Big Data (Guizhou University), Guiyang 550025, China
2. College of Computer Science and Technology, Guizhou University, Guiyang 550025, China
3. Institute of Cryptography & Data Security, Guizhou University, Guiyang 550025, China
4. College of Mathematics and Statistics, Guizhou University, Guiyang 550025, China

Abstract: Aiming at the problem that the result verification method in the searchable encryption scheme was complicated, the search cost was high and the efficiency was low, it was difficult to meet the requirements of efficient verification and security of multi-keyword search results, a multi-keyword search scheme based on the improved Merkle-Tree authentication method was proposed. Firstly, a multi-keyword searchable algorithm was built to achieve efficient and accurate multi-keyword search by the bilinear mapping. Secondly, based on the improved Merkle-Tree authentication method of Bawa, the verification and dynamic update algorithm of the search scheme was constructed to reduce the calculation cost of the classic MHT to prevent the efficient verification and update of illegal operations such as data tampering, deletion, and forgery. Under the decision-type linear hypothesis and CDH hypothesis, the proof scheme meets the indistinguishability and unforgeability of ciphertext.

Key words: cloud computing, search privacy, valid verification, dynamic update

---

收稿日期：2020–04–14；**修回日期**：2020–06–19

基金项目：教育部—中国移动科研基金资助项目（No.MCM20170401）；国家自然科学基金资助项目（No.61772008, No.U1836205）；贵州省科技重大专项计划基金资助项目（No.20183001）；贵州省科技计划基金资助项目（黔科合基础[2019]1098, 黔科合平台人才[2017]5788）；贵州省科学技术基金资助项目（黔科合 J 字[2008]2121）

Foundation Items: Ministry of Education-China Mobile Research Fund Project (No.MCM20170401), The National Natural Science Foundation of China (No.61772008, No.U1836205), Science and Technology Major Support Program of Guizhou Province (No.20183001), Guizhou Provincial Science and Technology Plan Project (No.[2019]1098, No.[2017]5788), Science and Technology Foundation of Guizhou Province (No.[2008]2121)

1　引言

随着大数据的开放与共享，数据搜索结果的可验证性显得尤其重要。搜索结果的可验证性是指用户能够高效地对服务器返回的搜索结果进行认证。现有可搜索加密方案中的结果验证方法普遍存在成本高、效率低等问题，为实现多关键词搜索结果高效验证和安全性需求带来了巨大挑战。

可搜索加密的目的是解决云存储[1-3]条件下密文的高效搜索问题，一个安全的可搜索加密方案可以使云服务器在不知道用户明文数据信息的情况下，实现对密文的搜索。最早由 Song 等[4]提出的方案使用对称加密算法对文档中的每个单词进行加密，搜索时对关键词也同样加密，然后通过全文扫描，将加密的搜索关键词与加密的文档进行比对，匹配到具有相同密文单词的文件则是用户想要的文件。但是这种方法检索的效率较低，耗时较多。Goh[5]基于“文件–关键词”的思想，通过利用布隆过滤器来建立文件索引，对每个文件包含的关键词使用伪随机函数映射到该文件的布隆过滤器索引中，搜索时提交查询陷门，根据每个文件的索引判断该文件是否包含特定的查询关键词。这种建立安全索引的方法使搜索效率得到了显著提高，但云服务器需要与用户进行两次交互才能完成搜索，增加了用户的通信开销。Chang 等[6]利用随机比特建立关键词的索引，并在搜索时通过恢复索引的部分信息来匹配关键词。Curtmola 等[7]提出了在整个文件集合上建立加密关键词的 Hash 索引，搜索令牌由关键词陷门和文件拥有者的身份信息组成，可以通过关键词陷门与关键词密文的匹配来产生搜索结果，最初的研究者只关注单关键词搜索方案的实现。

考虑到单关键词不能精准定位到用户想要的文件，Golle 等[8]提出 2 个连接关键词搜索方案。该方案假设对每次搜索的每个文件都有固定数量的关键词域，第一个方案需要为每个文件进行两次模幂运算，其陷门的大小与加密文件的总数成线性关系。第二个方案虽然使陷门的大小固定，但是却与关键词域的数量成正比。而且第二个方案的存储开销是第一个方案的两倍。Zheng 等[9]提出加密数据的无证书关键词搜索（CLKS, certificateless keyword search on encrypted data）方案，该方案不仅支持对加密数据进行多关键词搜索，而且由于无证书加密技术避免了证书管理问题，但通信开销比较大。Zhang 等[10]提出了一种排序的多关键词搜索方案，该方案支持在多所有者模型中进行结果相关性排序搜索，降低了返回所有搜索结果的需要，但不能对恶意云服务器返回的虚假搜索结果进行验证。

因此，应使用某种验证方法来确保用户搜索结果的正确性[11-12]。Sun 等[13]提出了一种有效的基于树的索引结构来实现真实性验证。Wang 等[14]提出了一个完全实现查询完整性验证的方案，该方案利用布隆过滤器[15]、MHT（Merkle Hash Tree）等数据结构，借助伪随机置换、哈希函数等工具来验证查询完整性。但该方案只适用于静态数据库，不支持数据库的动态更新。文献[16]提出了一种可验证的多关键词搜索方案，该方案借助私人审计服务器来验证搜索结果的正确性，但也不能实现动态性。考虑到用户需要经常更新存储到云端的数据，支持动态操作的可搜索加密方案[17-20]得到研究者们的广泛研究。Kurosawa[21]演示了如何以验证的方式支持文档的更新操作，以及检测恶意云服务器的攻击行为。

在传统的有效验证方案中，用户需要委托第三方来验证搜索结果，此方法虽然保证了搜索结果的正确性，但是增加了计算和通信开销。针对上述问题，本文采用文献[16]所提的验证思想，基于改进的 Merkle-Tree 认证方法[22]提出搜索结果高效验证的多关键词搜索方案，主要贡献如下。

1) 在搜索阶段，基于 Miao 等[16]提出的（VMKS, verifiable multiple keywords search）方案构造多关键词的可搜索算法，实现高效精准的多关键词搜索，并且通过方案分析，证明其可行性。

2) 在认证阶段，利用改进的 Merkle-Tree 认证方法构造搜索方案的验证及动态更新算法，实现高效认证和动态更新；与经典的 MHT 相比，计算成本从 $O(n)$ 降低到 $O(\log n)$ 。

3) 对所提方案的正确性、安全性和性能进行了详细证明与分析，并在决策线性假设和 CDH（computational Diffie-Hellman）假设下证明所提方案满足密文不可区分性和不可伪造性。

2　预备知识

BLS（Boneh-Lynn-Shacham）短签名由文献[23]定义，它是用于消息签名的身份验证，使用双线性对[16]进行签名身份认证。与本文方案安全性相关的

安全假设描述如下。

定义 1 CDH 假设。假设 G 是阶为 p 的乘法循环群，g 是 G 的生成元，给定三元组 $g, g^a, g^b \in G$，且随机选择 2 个元素 $a, b \in Z_p^*$，对于任何概率多项式时间的敌手 A，以不可忽略的优势 ε，计算 $g^{ab} \in G$ 是不可行的，其中 A 的优势满足 $\Pr[A_{\mathrm{CDH}}(g, g^a, g^b) = g^{ab}] < \varepsilon$。

定义 2 决策线性假设。假设 G 是阶为 p 的群，g 是 G 的生成元，给定元组 $(g,u,v,g^{r_1},u^{r_2},v^{r_1+r_2})$ 和 $(g,u,v,g^{r_1},u^{r_2},l)$，其中 $u,v,l \in G$，概率多项式时间敌手 A 的主要目标是区分 $v^{r_1+r_2}$ 与在群 G 里的随机元素 l，如果 A 在打破 DL（decisional linear）问题上的优势 $\mathrm{Adv}_{\mathrm{A}}^{\mathrm{DL}}(\lambda)$ 可以忽略不计，其优势定义为 $\mathrm{Adv}_{\mathrm{A}}^{\mathrm{DL}}(\lambda) = |\Pr[A(g,u,v,g^{r_1},u^{r_2},v^{r_1+r_2})=1] - \Pr[A(g,u,v,g^{r_1},u^{r_2},l)=1]|$。

2.1 经典的 MHT

MHT 是著名的二叉树数据结构，其中除叶子节点外的每个节点都是其叶子节点的串联，顶部的节点称为根节点。MHT 中的叶子节点表示数据块的哈希值，根节点的身份验证可确保所有节点的完整性。图 1 描述了具有 8 个叶子节点的 MHT，如果要对 $h(f[3])$处的数据节点进行完整性验证，则需要知道辅助信息（AI, auxiliary information）和 $\mathrm{AI}(f[3]):\{(h_C,L),h(f[3]),\ (h(f[4]),R),(h_B,R)\}$，为了验证数据块 $f[3]$的完整性，认证者执行以下操作。

1) 计算 $h_D \leftarrow (h(f[3]) \| h(f[4]))$。

2) 计算 $h_A \leftarrow (h_C \| h_D)$。

3) 计算根 $h_R \leftarrow (h_A, h_B)$。

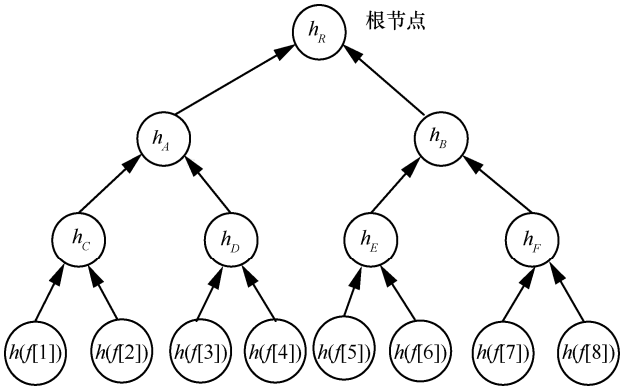

图 1 经典的 MHT

2.2 改进的 Merkle-Tree 认证方法

为了降低在运用 MHT 验证过程中节点搜索的计算成本，本文结合文献[22]改进的 Merkle-Tree 认证方法，将计算成本从经典的 MHT 的 $O(n)$ 降低到 $O(\log n)$，树的每个节点都有 2 个信息：一个是数据块的哈希值；另一个是相对索引。与节点 T 关联的相对索引是指属于 T 的子树的叶子节点的数量，叶子节点的相对索引值设置为 1。例如，如果 L 和 R 是左右子节点，其哈希值分别为 h_a 和 h_b，相对索引字段分别为父节点 T 的 r_a 和 r_b，则节点 T 的哈希值为 $(h_a \| h_b)$ 且相对索引字段值为 $(r_a + r_b)$。时间戳字段与树的根节点串联，该时间戳是与根哈希值串联的树创建的日期和时间。改进的 Merkle-Tree 如图 2 所示，有 $h_R:(h_A \| h_B \| d_t)$，其中 d_t 是 Merkle-Tree 创建的日期和时间。由于 h_R 总是针对树中任何数据块做任何修改而更新，因此最后修改的日期和时间会反映在 h_R，从而确保了数据的新鲜度。

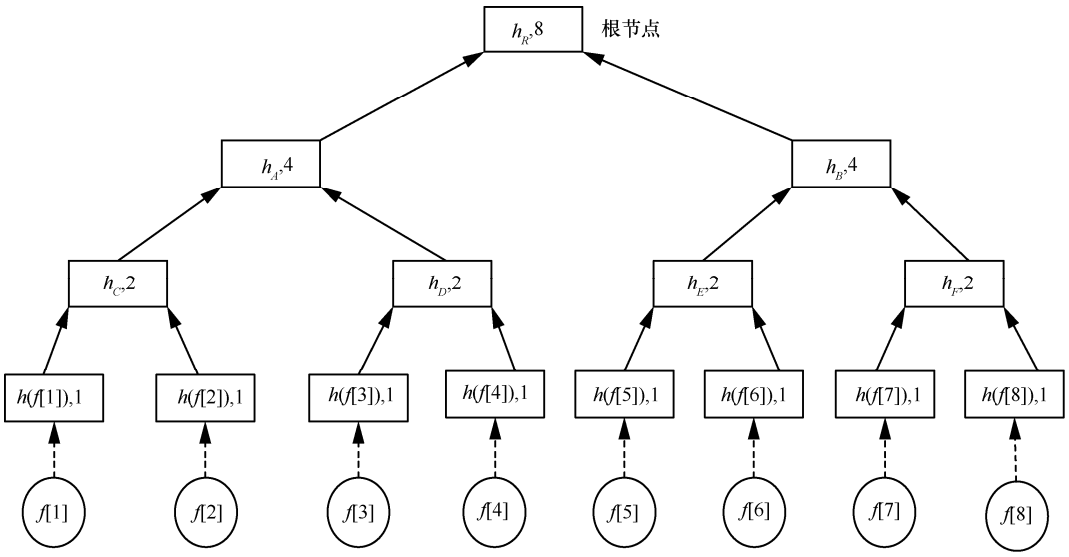

图 2 改进的 Merkle-Tree

3　问题描述

3.1　方案模型

如图 3 所示，模型中有 4 个实体，即密钥生成中心（KGC, key generation center）、数据拥有者（DO, data owner）、用户（DU, data user）以及云服务器（CSP, cloud service provider）。首先，KGC 根据用户相应标识生成部分私钥，其余部分私钥由用户自己生成。其次，DO 先对文件加密并为文件创建索引，将文件 F 分割成 n 个数据块 $(f[1],f[2],\cdots,f[n])$，数据块 $f[i]$ 作为 Merkle-Tree 的叶子节点生成完整的树并对根进行签名，然后将这些发送给 CSP 存储。DU 按照自己的需求向服务器提交相应的查询陷门，CSP 根据每个文件的索引判断该文件是否包含特定的查询关键词搜索匹配的结果。当用户收到服务器返回的结果时，向服务器提出质询消息验证结果的完整性。

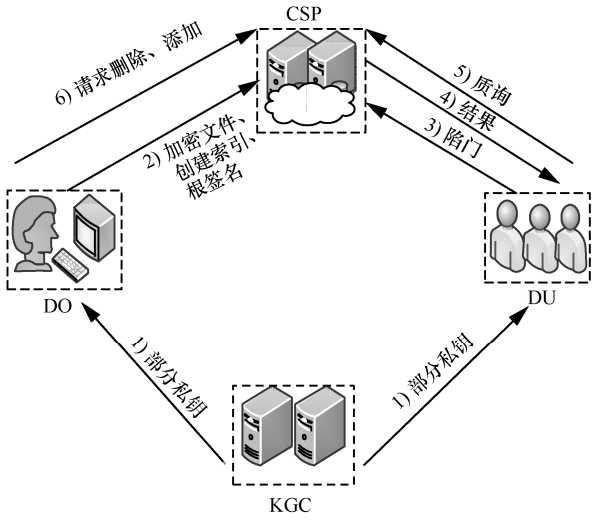

图 3　方案模型

3.2　方案定义

本文方案由以下 7 个算法组成。

1) GlobalSetup(λ)。输入安全参数 λ，发布公共参数，保存主密钥，输出系统公共参数 param 和主密钥 msk。

2) ParKeyGen(msk,ID)。给定用户 ID，KGC 运行该算法并为用户 ID 返回相应的部分私钥 $\text{psk}_{\text{ID}}=(\text{psk}_1,\text{psk}_2)$。

3) KeyGen(param,psk_{ID},ID,msk)。输入 msk、ID、param、psk_{ID}，最后为用户输出公私钥(pk,sk)。

4) Encrypt(F,W,pk,psk_{ID},ID)。给出数据文件集 F 和关键词集 W，DO 执行该算法，输出密文集 CT、索引集 I、文件标签 τ 和签名集 δ，DO 将这些元组 (CT,I,τ,δ) 上传至 CSP 中存储。

5) Trapdoor(sk,pk_{ID},W',pk,ID)。输入搜索查询 W'，用户 DO 执行该算法，生成搜索令牌 $T_{W'}$ 并发送给 CSP。

6) Search(pk,$T_{W'}$,I,CT)。CSP 收到用户的搜索请求后，搜索与关键词陷门 $T_{W'}$ 匹配的索引 I 的相关文件 CT' 及其相应的身份标识 ID'，然后将 (CT',ID') 发送给用户。

7) Verify(CT',pk,ρ)。用户收到搜索结果 CT'，向服务器提交质询消息，服务器返回相应的辅助消息，验证结果的完整性，如果 CT' 通过认证算法，用户输出 1；否则，输出 0。

3.3　安全模型

与方案[16,24]相似，本文首先假设 CSP 是半诚实且好奇的，这意味着它将诚实地执行算法，但会尝试学习尽可能多的私有信息。其次，假设不能完全信任 KGC，那么 KGC 诚实地遵循算法规定，但可能会尝试使用其获得的信息来渗透更多的信息。本文考虑 2 种类型的敌手，即 A_1 和 A_2。A_1 表示外部攻击者，除了主密钥 msk 以外允许控制用户 ID 的公钥；A_2 表示内部攻击者，可以访问主密钥 KGC，而不需要控制用户 ID 的公钥。

本文通过下面的 2 个安全游戏来说明本文方案对以上 2 种不利因素都是安全的。设 A_1 为 Game1 中的敌手，A_2 为 Game2 中的敌手，C 是维护下面 2 个列表的挑战者。

Tlist：用于存储元组 (ID,W)，这就意味着与用户 ID 的关键词 W 相关的搜索令牌已被 (A_1,A_2) 查询。

Ulist：用于存储元组 (ID,psk,sk,pk,$\varpi_1,\varpi_2,\varpi_3$)，$\varpi_1$=1 表示 psk 被 (A_1,A_2) 查询；否则，表示未被查询。同样地，ϖ_2=1 和 ϖ_3=1 表示 sk,pk 被 (A_1,A_2) 查询；否则，表示敌手不会查询 sk,pk。具体来说，本文的安全游戏是在敌手 A_1 和挑战者 C 之间进行的。

初始化。挑战者运行 GlobalSetup，输出公共参数 param 和主密钥 msk，把公共参数 param 发送给敌手 A_1，并且设置列表为空。

阶段 1。假设 A_1 可以在多项式时间查询以下预言机，则 C 可以执行以下算法。

ParKeyGen。给定来自 A_1 的用户标识 ID，在 Ulist 中的 psk 不为空，C 返回 psk；否则，挑战者

利用主密钥 msk 执行 ParKeyGen 算法以获得 psk，将元组 (ID,psk,*,*,1,0,0) 添加到列表 Ulist 中，并将 psk 发送给 A_1，其中符号“*”表示为空。

KeyGen。给定 A_1 的 ID，如果 sk 不为空，则挑战者从 Ulist 中选择 sk；如果 psk 在 Ulist 中不为空，则挑战者调用 ParKeyGen 算法利用 psk 生成 sk，然后挑战者在 Ulist 中添加 sk。假设上面 2 种情况都不成立，则 C 执行 ParKeyGen 和 KeyGen 算法获得 psk 和 sk。然后，C 在 Ulist 中添加元组 (ID,psk,sk)。最后，C 更新与用户 ID 关联的 Ulist 中的 ϖ_1=1，ϖ_2=1 并将 sk 发送给 A_1。

Ulist 中的 pk 不为空，那么挑战者检索 pk；如果 Ulist 中的 sk 不为空，则检索 sk 且挑战者运行 KeyGen 以输出 pk 并将其添加到与 ID 有关的 Ulist 中。

如果 Ulist 中的 psk 不为空，则 C 将检索 psk，运行 KeyGen 生成 sk 和 pk，并将 sk、pk 添加到 Ulist 中。

否则，C 将同时运行 ParKeyGen 和 KeyGen 算法以生成 psk、sk 和 pk，并将元组 (ID,psk, sk,pk,0,0,0) 添加到 Ulist 中，C 将 pk 发送给 A_1。

Rplace-pk。给定用户 ID 和来自 A_1 的替换公钥 pk，C 将 Ulist 中的 pk 更新为 pk'，并针对 ID 设置 ϖ_3=1。

Trapdoor。给定用户 ID 和 A_1 的关键词 W，挑战者在 Ulist 中检索 sk，执行 Trapdoor 算法，将元组 (ID,W) 添加到 Tlist 中，然后将陷门发送给 A_1。

挑战阶段。A_1 提交 2 个相同长度的不同关键词 W_0，W_1 以及用户 ID。给定 Ulist 中的 $(\mathrm{ID}^*,\mathrm{psk}^*,\mathrm{sk}^*,\varpi_1,\varpi_2,\varpi_3)$，如果 ϖ_1=0，ϖ_2=0，仍然要求 psk^* 和 sk^* 不能被 A_1 查询。此外，(ID^*,W_0) 和 (ID^*,W_1) 都没有存储在 Tlist 中。如果 ϖ_3=0 则可以替换 pk^*，否则无法替换。C 选择一个随机位 $\lambda\in(0,1)$，通过运行 Encrypt 算法对 W_λ 进行加密并将密文 I^* 返回给 A_1。

阶段 2。该阶段与阶段 1 相似，唯一的限制是不能进行以下查询：A_1 不能查询 ParKeyGen(ID^*)、KeyGen(ID^*)、Trapdoor(ID^*,W_0) 或者 Trapdoor(ID^*,W_1)。

猜测。A_1 输出为 λ^*，A_1 赢得游戏的条件是 $\lambda^*=\lambda$。

定义 3 对于任何概率多项式时间的算法 A_1，能以不可忽略的优势 $|\Pr[\lambda=\lambda^*]-\frac{1}{2}|$ 赢得 Game1，则实现了针对 A_1 的密文不可区分性。

4 方案构造

本节介绍方案的具体构造。首先，基于 Miao 等[16]提出的 VMKS 方案构造多关键词的可搜索算法，实现高效精准的多关键词搜索。其次，利用改进的 Merkle-Tree 认证方法构造搜索方案的验证及动态更新算法，防止数据篡改、删除和伪造等不法操作的高效验证，且时间戳与根节点的连接确保了数据的新鲜度，这是文献[16]没有的特点，具体算法如下。

4.1 初始化

GlobalSetup(λ)。输入安全参数 λ，假设 $e:G\times G\to G_T$ 是双线性映射，G 是生成元为 g、阶数为 p 的一个群，KGC 选择随机元素 $x,y\in Z_p^*$，$u,u_1,\cdots,u_n\in G$，并且计算 g^x,g^y，最后，选择 2 个哈希函数 $h:\{0,1\}^*\to G$，$H_0:\{0,1\}^*\to Z_p^*$，并且定义一个哈希函数 $H_1(\mathrm{ID})=u\prod_{i=1}^{n}u_i^{\mathrm{ID}_i}$，其中 ID={$\mathrm{ID}_1$, $\mathrm{ID}_2,\cdots,\mathrm{ID}_n$}，此外，KGC 生成公私钥对 (pk,sk)。系统公共参数 $\mathrm{param}=\{g,h,H_0,H_1,g_0,u,u_1,\cdots,u_n,g^x,g^y\}$，主密钥 $\mathrm{msk}=\{x,y\}$，将系统参数 param 发布出来，msk 由 KGC 自己保存。

4.2 密钥生成

密钥生成分为 2 个步骤。首先，KGC 在 ParKeyGen 算法中为用户 ID 生成部分私钥；其次，用户 ID 在 KeyGen 算法中为自己生成最终的私钥。

1) ParKeyGen(msk,ID)。给出特定用户 DU 的身份 ID，KGC 选取随机元素 $t_1\in Z_p^*$，计算 $\mathrm{psk}_1=g^{t_1}$，$\mathrm{psk}_2=g^xH_1(\mathrm{ID})^{t_1}$，并将其通过安全通道发送给用户 DU，则它的部分私钥为 $\mathrm{psk}_{\mathrm{ID}}$=($\mathrm{psk}_1$,$\mathrm{psk}_2$)。

2) KeyGen(param,$\mathrm{psk}_{\mathrm{ID}}$,ID,msk)。DU 随机选取 2 个元素 $x',y'\in Z_p^*$ 并且计算 $g^{x'},g^{y'}$，DU 的公私钥对为 $\mathrm{pk}=\{g^{x'},g^{y'}\}$，$\mathrm{sk}=\{\mathrm{psk}_{\mathrm{ID}},x',y'\}$。

4.3 密文生成

DO 将存储在服务器中的数据文件 F 分割成 n 个数据块 $(f[1],f[2],\cdots,f[n])$，并在上传之前对其加密，具体算法如下。

1) FileTagGen(fname,t,n,d_t)。由 DO 执行该算

法，为文件 F 生成标签，选择随机元素 $\mu \in G$，$t \in Z_p^*$ 生成系统日期和时间，记为 d_t，系统日期和时间连接在文件标签 τ 后，确保文件的新鲜度，使 $\pi = (\text{fname} \| n \| \mu \| d_t)$，其中 $\pi \in G$ 且 τ=sig$_t(\pi)$ 是文件 F 的标签，连接的字符串 π 被存储在本地是为了以后对文件标签的验证。

2) BlockSigGen(t,$h(f[i])$,d_t,μ)。生成文件标签之后，DO 每个文件块 $f[i]$ 生成的 BLS 签名为 $\varphi_i = (h(f[i])\mu^{f[i]})^t$，$i \in \{1,2,\cdots,n\}$，签名集为 $\delta = \{\varphi_i\}, 1 \leqslant i \leqslant n$，DO 用 $h(f[i])$ 作为叶子节点生成树，其中，根节点 h_R 是与系统日期和时间连接的，即 $h_R = h_R \| d_t$，根的签名为 $\rho \leftarrow (h_R)^t$。

3) Encrypt(F,W,pk,psk$_{\text{ID}}$,ID)。DO 从文件 $f[i]$ 中提取关键词集 $W_i \subseteq W = \{w_1,\cdots,w_n\}$ 并为这些文件创建索引 I_i。DO 选择 2 个随机数 $r_1, r_2 \in Z_p^*$ 并且计算 $I_1 = g^{r_1}$，$I_2 = g^{(x+x')(r_1+r_2)} g^{yH_0(W)r_1}$，$I_3 = g^{r_2}$，$I_4 = H_1(\text{ID})^{r_2}$，最后，DO 将 $\{I, C, \tau, \delta, \rho\}$ 和相应的身份标识 ID=$\{\text{ID}_1,\cdots,\text{ID}_d\}$ 上传给 CSP。

4.4　陷门生成

用户想要搜索感兴趣的文件，需要将关键词加密向 CSP 请求查询，CSP 收到相应的查询陷门执行搜索算法。

Trapdoor(sk,pk$_{\text{ID}}$,W',pk,ID)。给定查询的关键词集 $W' = \{w_1,\cdots,w_l'\} \subseteq W$，用户 DU 选择元素 $s \in Z_p$，然后计算陷门 $T_{W'} = \{T_1, T_2, T_3, T_4\}$，其中，$T_1 = g^{t_1 s}$，$T_2 = g^{(x+x')s} H_1(\text{ID})^{t_1 s}$，$T_3 = g^s$，$T_4 = g^{(x+x')} \cdot g^{yH_0(w_i')s}$。

4.5　密文搜索

根据用户提交的相应查询陷门，CSP 执行搜索算法，搜索与陷门匹配的结果。

Search(pk,$T_{W'}$,I,C)。收到陷门 $T_{W'}$ 之后，CSP 计算三元组 $(\sigma_1, \sigma_2, \sigma_3)$，其中 $\sigma_1 = e(I_2, T_3)$，$\sigma_2 = e(I_1, T_4)$，$\sigma_3 = \dfrac{e(I_3, T_2)}{e(I_4, T_1)}$，CSP 通过判断式(1)是否成立，来匹配陷门 $T_{W'}$ 和索引 I，最后，返回相关的密文 $C' = \{c_1', \cdots, c_q'\}$。

$$\sigma_1 = \sigma_2 \sigma_3 \tag{1}$$

4.6　结果认证

Verify(C',pk,ρ)。用户收到结果 C' 时，向 CSP 提出质询消息返回相应辅助消息来验证 C' 的正确性，用户首先选择 k 个元素集 $Q = \{\varsigma_1,\cdots,\varsigma_k\}, k \in [1,n]$ 且 $\forall i \in Q$，其次选择一个随机元素 $b_i \in Z_p$，最后，发送质询消息 $M \leftarrow (i, b_i)_{i \in Q}$ 给 CSP。

1) CSP 计算 $\phi = \prod_{i=\varsigma_1}^{\varsigma_k} \phi_i^{b_i} \in G$ 和 $\psi = \sum_{i=\varsigma_1}^{\varsigma_k} b_i f[i] \in Z_p$ 之后，向用户回应 $P_f = \{\psi, \varphi, (h(f[i]), \text{AI}(i))_{i \in Q}, \text{SIB}(i), \rho\}$ 作为持有证明，其中 AI(i) 是节点 i 的辅助信息。

2) 用户对文件标签及根签名进行验证，当用户收到 CSP 回应的证明信息时，则需要计算以下 3 个式子来验证结果的正确性。

$$e(\pi, g^t) \stackrel{?}{=} e(\tau, g) \tag{2}$$

$$e(h(R), g^t) \stackrel{?}{=} e(\rho, g) \tag{3}$$

$$e\left(\prod_{i=\varsigma_1}^{\varsigma_k} h(f[i])^{b_i} \mu^{\psi}, g^t\right) \stackrel{?}{=} e(\varphi, g) \tag{4}$$

5　动态更新

允许数据动态过程是可搜索加密方案中数据完整性查询的一个重要特性。但是，现有的大多数方案都没有这个特性，本文将介绍利用改进的 Merkle-Tree 认证方法构造搜索方案的动态更新算法。

5.1　数据修改

DO 向服务器发出请求，并按以下方式对数据进行修改。

首先，DO 为新的文件生成标签 $\varphi' = (h(f[i]')\mu^{f[i]'})^t$。

接下来，DO 生成新的文件标签 $\tau' \leftarrow \text{sig}_k(\text{fname} \| n \| \mu \| d_t)$，来验证修改的日期和时间，以确保数据的新鲜度。

数据修改的框架为 $(X, i, f[i]', h(f[i]'), \varphi', \tau')$，将这些信息传送给 CSP，其中 X 表示修改操作，i 表示要修改的数据块。收到以上消息后，CSP 将执行以下替换：首先，用 f_i' 替换 f_i；其次，分别用 ϕ' 和 τ' 替换 ϕ 和 τ；最后，用 $h(f[i]')$ 替换 $h(f[i])$。最终，CSP 生成新的根哈希值 R' 并用 $h(f[i]')$ 进行 Merkle-Tree 重建，CSP 将修改过程的证明提供给数据拥有者进行验证，即 $P_f = \{\varphi, (h(f[i]), \text{AI}(i)), \rho, \text{SIB}(i), R'\}$，数据拥有者从 CSP 收到数据修改证明后，首先验证 τ，其次通过式(3)使用 $\{h(f[i]'), \text{AI}(i))_i, \rho\}$ 验证根。如果认证成功，数据拥有者使用

$\{h(f[i]'),\mathrm{AI}(i))_i\}$ 计算新生成的根并且将其与 R' 进行比较，若比较成功，数据拥有者通过签名私钥 t 认证 R'，因此生成根的签名 $\rho'=h(R')^t$，并把它发送到 CSP 存储。最后，DO 为新的数据块运行验证算法，如果结果为真，DO 可以从本地删除 $\{f[i]',\tau',\rho'\}$。

5.2 数据添加

若 DO 想在一个特殊的位置添加一个数据块 f^*，则按以下操作执行。

首先，为新的数据块生成签名 $\varphi^*=(h(f^*)\mu^{f^*})^t$。

其次，生成新的文件标签 $\tau^*\leftarrow \mathrm{sig}_t(\mathrm{fname}\|n\|\mu\|d_t)$。若 DO 插入（如图 4 所示）的数据消息为 $(I,V,i,f^*,\varphi',\tau')$，其中，$I$ 表示数据插入，字段 V 表示要插入的新块的位置，$V\leftarrow A$ 表示第 i 个位置之后的插入，$V\leftarrow B$ 表示第 i 个位置之前的插入，并且把这些消息发送到CSP，收到消息后，CSP 保存 f^* 和对应的叶子节点 $h(f^*)$，CSP 在 Merkle-Tree 中找到 $h(f[i])$ 并保留 $\mathrm{AI}(i)$ 插入叶子节点 $h(f^*)$，如果将字段 V 设置为 A，则具有哈希值 $(h(f[i])\|h(f^*))$ 的内部节点将被连接到原始树中；如果将字段 V 设置为 B，将具有哈希值 $(h(f^*)\|h(f[i]))$ 的内部节点添加到索引设置为 2 的原始树中。

最后，CSP 修改从第 i 个节点到最高节点（根节点）的路径上每个节点的详细信息。由于重新生成了 Merkle-Tree，CSP 产生了一个新的根 R 并且为 DO 提供了插入操作的证明消息，表示为 $P_f=\{\varphi,(h(f[i]),\mathrm{AI}(i)),\mathrm{SIB}(i),\rho,R'\}$，其中 $\mathrm{AI}(i)$ 表示先前树中 $f[i]$ 的辅助信息。在收到插入过程的证明时，DO 首先验证 τ，验证成功后，使用 $\{h(f[i]),\mathrm{AI}(i)\}$ 产生根，然后通过式(3)验证这个新生成的根，如果式(3)成立，那么 DO 可以验证CSP 是否正确地完成了文件插入过程，从而使用 $\{h(f[i]),\mathrm{AI}(i),,h(f^*)\}$ 生成新的根并与 R' 进行比较，若结果成功，DO 将 R' 的认证 $(h(R'))^t$ 发送给CSP 更新。最后，DO 为新的数据块运行认证算法，如果结果为真，数据拥有者就可以从本地存储中移除 $\{\rho',f^*,\tau'\}$。

6 方案分析

6.1 正确性分析

为了实现方案的安全性，本文必须保证CSP 能够诚实地执行密文检索操作，通过验证式(1)和式(4)的成立，确定没有篡改结果。

对于式(1)，有

$$\sigma_1=e(I_2,T_3)=e(g^{(x+x')(r_1+r_2)}g^{bH_0(w')r_1},g^s)=$$
$$e(g,g)^{(x+x')(r_1+r_2)s}e(g,g)^{bH_0(w')r_1s}$$
$$\sigma_2=e(I_1,T_4)=e(g^{r_1},g^{(x+x')s}g^{bH_0(w')s})=$$
$$e(g,g)^{(x+x')r_1s}e(g,g)^{(x+x')bH_0(w')s}$$

$$\sigma_3=\frac{e(I_3,T_2)}{e(I_4,T_1)}=\frac{e(g^{r_2},g^{(x+x')s}H_1(\mathrm{ID})^{t_1s})}{e(H_1(\mathrm{ID})^{r_2},g^{t_1s})}=e(g,g)^{(x+x')sr_2}$$

如果 $W'\subseteq W$，有 $\sigma_1=\sigma_2\sigma_3$，因此式(1)成立，对搜索结果的完整性验证则需要验证式(4)的正确性。

$$左边=e\left(\prod_{i=\varsigma_1}^{\varsigma_k}h(f[i])^{b_i}\mu^{w'},g^t\right)=e\left(\prod_{i=\varsigma_1}^{\varsigma_k}h(f[i])^{b_i}\mu^{b_id[i]},g^t\right)=$$
$$e\left(\prod_{i=\varsigma_1}^{\varsigma_k}(h(f[i])\mu^{f[i]})^{b_i},g^t\right)=e\left(\prod_{i=\varsigma_1}^{\varsigma_k}((h(f[i])\mu^{f[i]}))^{b_it},g\right)=$$
$$e\left(\prod_{i=\varsigma_1}^{\varsigma_k}((h(f[i])^{b_i}u^{f[i]})^t)^{b_i},g\right)=e\left(\prod_{i=\varsigma_1}^{\varsigma_k}\varphi_i^{b_i},g\right)=e(\varphi,g)=右边$$

则有式(4)成立，因此本文方案是正确的。

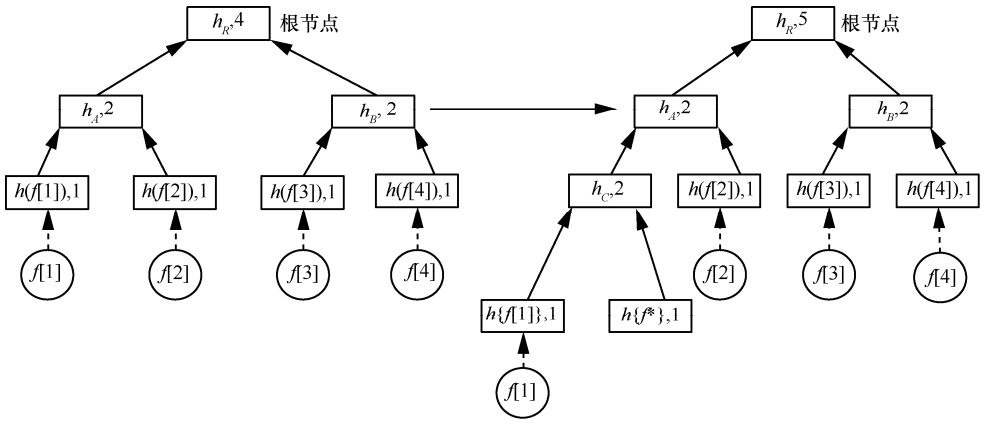

图 4 数据添加

6.2 安全性分析

在决策线性假设和CDH 假设下，实现了密文不可区分性和签名的不可伪造性，半诚实且好奇的CSP无法伪造有效的证明信息。通过利用与文献[22]方案相同的安全游戏实现安全目标。本文通过以下定理来保证本文方案的安全性。

定理 1　假设第 1 型敌手最多 ξ_1 次查询预言机 ParKeyGen，ξ_2 次查询预言机 KeyGen，Adv_{H_0} 表示敌手 A 打破哈希函数 H_0 的优势，$\mathrm{Adv}_{\mathrm{DL}}(\lambda)$ 表示敌手 A 打破 DL 假设的优势，则第 1 型敌手打破密文不可区分性的优势为

$$\mathrm{Adv}_{\mathrm{Type1}} \leqslant \mathrm{Adv}_{H_0} + 2(\xi_1+\xi_2)(n+1)\mathrm{Adv}_{\mathrm{DL}}(\lambda)$$

证明　设 S_i 表示敌手 A 想要赢得游戏 i 的事件，Adv_i 表示敌手 A 的优势。假设除了预定义的事件 E_P 发生，$i+1$ 轮游戏终止并输出一个随机位外，i+1 轮游戏与 i 轮游戏执行相同的操作。如果 E_P 是不可忽略的且它独立于 S_i，则有

$$\Pr[S_{i+1}]=\Pr[S_{i+1}\mid E_P]\Pr[E_P]+\Pr[S_{i+1}\mid \neg E_P]\Pr[\neg E_P]=$$
$$\frac{1}{2}\Pr[E_P]+\Pr[S_{i+1}\mid \neg E_P]\Pr[\neg E_P]=$$
$$\frac{1}{2}(1-\Pr[\neg E_P])+\Pr[S_i]\Pr[\neg E_P]$$

因此，有

$$\left|\Pr[S_{i+1}]-\frac{1}{2}\right|=\Pr[\neg E_P]\left|\Pr[S_i]-\frac{1}{2}\right|$$

$$\mathrm{Adv}_{i+1}=\Pr[\neg E_P]\mathrm{Adv}_i$$

接下来，本文将展示一系列安全游戏模拟。

Game1。在该轮游戏中，敌手 A 按照 Game1 中定义的步骤执行，即挑战者 C 生成主密钥、公共参数和用户部分私钥。令 $(\mathrm{ID}^*,\mathrm{pk}_{\mathrm{ID}}{}^*)$ 是挑战阶段用户的身份和公钥，$C^*=(I_1^*,I_2^*,I_3^*,I_4^*)$ 是返回给敌手 A 的密文。

Game2。在该轮游戏中，A 继续执行 Game1 中定义的步骤，除了 H_0 是一个抗碰撞的哈希函数外，有 $\Pr[S_2]=\Pr[S_1]-\mathrm{Adv}_{H_0}$ 且 $\mathrm{Adv}_2=\mathrm{Adv}_1-\mathrm{Adv}_{H_0}$。

Game3。在该轮游戏中，除了生成的公共参数外，C 执行的游戏与 Game2 相同。

1) C 选择 $x,y\in Z_p{}^*$、$\gamma_u\in\{0,\cdots,n\}$ 和 $\eta_u\in\{0,\cdots,p\}$，使 $\eta_u(n+1)<p$。

2) C 选择 $K_u'\in Z_{\eta_u}$ 和 $(K_{u1},\cdots,K_{un})$，其中 $K_{uj}\in Z_{\eta u},1\leqslant j\leqslant n$。

3) C 选择 $T_u'\in Z_p$ 和 $(T_{u1},\cdots,T_{un})$，其中 $T_{uj}\in Z_{\eta u},1\leqslant j\leqslant n$。

公共参数设置为

$$u=(g^x)^{K_u'-\gamma_u\eta_u}g^{T_u'}$$

$$u_j=(g^x)^{K_{uj}}g^{T_{uj}},1\leqslant j\leqslant n$$

可以看到，生成过程中并没有改变公共参数，因此，有 $\Pr[S_3]=\Pr[S_2]$ 且 $\mathrm{Adv}_3=\mathrm{Adv}_2$。

Game4。在该轮游戏中，除了猜测阶段，C 执行的游戏与 Game3 相同，其中 C 为输入 $\mathrm{ID}=\mathrm{ID}_1,\cdots,\mathrm{ID}_n$。定义 2 个函数分别为

$$A_u(\mathrm{ID})=K_u'-\gamma_u\eta_u+\sum_{j=1}^{n}\mathrm{ID}_jK_{uj}$$

$$B_u(\mathrm{ID})=T_u'+\sum_{j=1}^{n}\mathrm{ID}_jT_{uj}$$

在猜测阶段，C 检查 $A_u(\mathrm{ID}^*)$ 是否等于零，如果为零，则 C 终止并且输出 $b'\in\{0,1\}$ 作为 A 的猜测；否则，C 执行与 Game3 相同的步骤。

由于 $(K_u',K_{u1},\cdots,K_{un})$ 对 A 是不知道的，即 $\Pr[A_u(\mathrm{ID}^*)=0\bmod p]=\dfrac{1}{\eta_u(n+1)}$，因此，$\mathrm{Adv}_4=\dfrac{\mathrm{Adv}_3}{\eta_u(n+1)}$。

Game5。在该轮游戏中，除了猜测阶段，C 执行的游戏与 Game4 相同，C 检查以下 2 种情况是否发生。

1) 对于身份 ID，$A_u(\mathrm{ID})=0$ 且对 ParKeyGen 预言机进行查询。

2) 对于身份 ID，$A_u(\mathrm{ID})=0$ 且对 KeyGen 预言机进行查询。

如果以上 2 种情况发生了，则 C 终止并且输出 $b'\in\{0,1\}$ 作为 A 的猜测，由于上述情况的发生不独立于 Game4，则本文将估计 $\Pr[\neg E_5]$ 的下确界，即

$$\Pr[\neg E_5]=\Pr[(\cap_{\mathrm{ID}\in\Omega}A_u(\mathrm{ID})\neq 0)\cap(\cap_{\mathrm{ID}\in\Omega'}A_u(\mathrm{ID})\neq 0)]=$$
$$1-\Pr[(\cup_{\mathrm{ID}\in\Omega}A_u(\mathrm{ID})\neq 0)\cup(\cup_{\mathrm{ID}\in\Omega'}A_u(\mathrm{ID})\neq 0)]\geqslant$$
$$1-\sum_{\mathrm{ID}\in\Omega}\Pr[A_u(\mathrm{ID})\neq 0]-\sum_{\mathrm{ID}\in\Omega'}\Pr[A_u(\mathrm{ID})\neq 0]=1-\frac{\xi_1+\xi_2}{\eta_u}$$

其中，Ω 表示用户 ID 查询预言机 ParKeyGen 的集合，Ω' 表示用户 ID 查询预言机 KeyGen 的集合，ξ_1 表示查询预言机 ParKeyGen 的次数，ξ_2 表示查询预言机 KeyGen 的次数。

如果 $\eta_u=2(\xi_1+\xi_2)$，则 $\Pr[\neg E_5]\geqslant\dfrac{1}{2}$，因此，

$\mathrm{Adv}_5 \geqslant \frac{1}{2}\mathrm{Adv}_4$。

Game6。在该轮游戏中，除了使用 g^x, g^y 作为公钥外，其中 x, y 是 C 不知道的，C 执行的游戏与 Game5 相同；除了预言机 ParKeyGen，C 根据算法规范处理预言机。

给定用户 ID，如果 $A_u(\mathrm{ID})=0$，则 C 终止；否则它选择 $t_1' \in Z_p^*$ 并设置 $\mathrm{psk}_1 = (g^y)^{\frac{-1}{A_u(\mathrm{ID})}} g^{t_1}$，$\mathrm{psk}_2 = (g^y)^{\frac{-\gamma_u(\mathrm{ID})}{A_u(\mathrm{ID})}} H_1(\mathrm{ID})^{t_1}$。

注意，psk_1 和 psk_2 是部分私钥的有效组成部分，因为 $H_1(\mathrm{ID}) = g^{xA_u(\mathrm{ID})} g^{\gamma_u(\mathrm{ID})}$，所以 $\mathrm{psk}_1 = g^{t_1' - \frac{1}{A_u(\mathrm{ID})}}$，$\mathrm{psk}_2 = g^x H_1(\mathrm{ID})^{t_1' - \frac{1}{A_u(\mathrm{ID})}}$，其中 $t_1 = t_1' - \frac{1}{A_u(\mathrm{ID})}$。可以看到，主密钥、公钥和部分私钥的分布与 Game5 相同，因此 Adv_6=Adv_5。

Game7。在该轮游戏中，除了挑战阶段，C 执行的游戏与 Game6 相同，给出来自 A 的 $(\mathrm{psk}_{\mathrm{ID}^*}=(\mathrm{psk}_1^*, \mathrm{psk}_2^*), \mathrm{ID}^*, W_0, W_1)$；如果 $A_u(\mathrm{ID}^*) \neq 0$，则 C 终止；否则选择 $b \in \{0,1\}$、$r_1, r_2 \in Z_p^*$ 和 $C^* = (I_1^*, I_2^*, I_3^*, I_4^*)$。其中 $I_1^* = (g\mathrm{psk}_1^*)^{r_1}$，$I_2^* = (g^x \mathrm{psk}_2^*)^{r_1+r_2} g^{yH_0(w)r_1}$，$I_3^* = g^{r_2}$，$I_4^* = g^{\gamma_u(\mathrm{ID}^*)r_2}$。

因此有 Adv_7=Adv_6，因为挑战密文的分布没有改变。

Game8。在该轮游戏中，除了挑战密文的生成，C 执行的游戏与 Game7 相同，给定一个 DL 实例 $(g, u, v, g^{r_1}, u^{r_2}, l)$，$C^* = (I_1^*, I_2^*, I_3^*, I_4^*)$，其中，$I_1^* = u^{r_1}$，$I_2^* = lu^{H_0(w_\lambda)r_1}$，$I_3^* = g^{r_2}$，$I_4^* = g^{\gamma_u(\mathrm{ID}^*)r_2}$。

在该轮游戏中，C 不使用 x, y, r_1, r_2，Game7 和 Game8 的区分度与 DL 问题有关。$\mathrm{Adv}_{\mathrm{DL}}$ 表示敌手 A 区分 DL 问题的优势，则 $|\Pr[S_7]-\Pr[S_8]| \leqslant \mathrm{Adv}_{\mathrm{DL}}(\lambda)$；而且，在 Game8 中 w_λ 被完美地隐藏了，因此 $\Pr[S_8]=\frac{1}{2}$，到此完成了模拟，且有以下不等式成立

$$\mathrm{Adv}_1 = \mathrm{Adv}_2 + \mathrm{Adv}_{H_0}$$

$$\mathrm{Adv}_2 = \mathrm{Adv}_3$$

$$\mathrm{Adv}_3 = \eta_u(n+1)\mathrm{Adv}_4$$

$$\mathrm{Adv}_4 \leqslant 2\mathrm{Adv}_5$$

$$\mathrm{Adv}_5 = \mathrm{Adv}_6 = \mathrm{Adv}_7 \leqslant \mathrm{Adv}_{\mathrm{DL}}(\lambda)$$

因此，有

$$\mathrm{Adv}_{\mathrm{Type1}} \leqslant \mathrm{Adv}_{H_0} + 2(\xi_1+\xi_2)(n+1)\mathrm{Adv}_{\mathrm{DL}}(\lambda)$$

定理 1 证毕。

定理 2 假设第 2 型敌手最多 ξ_2 次查询预言机 KeyGen，ξ_3 次查询预言机 Trapdoor，Adv_{H_0} 表示敌手 A 打破哈希函数 H_0 的优势，$\mathrm{Adv}_{\mathrm{DL}}(\lambda)$ 表示敌手 A 打破 DL 假设的优势，则第 2 型敌手打破密文不可区分性的优势为 $\mathrm{Adv}_{\mathrm{Type2}} \leqslant \mathrm{Adv}_{H_0} + 2\xi_2\xi_3(n+1)\mathrm{Adv}_{\mathrm{DL}}(\lambda)$。

定理 2 的证明过程与定理 1 相似，在此省略。

定理 3 如果敌手 A_I 以 ε 的优势在数据块 $f[i]$ 上产生时间跨度为 t_1 的伪造，通过模拟 Game$I$，并对哈希查询进行 q_H 次查询、签名查询进行 q_S 次查询和系统参数查询进行 q_{param} 次查询，则用以下的概率和多项式时间解决 CDH 问题

$$\frac{\varepsilon}{(q_S+1)\mathrm{e}} \leqslant \varepsilon'$$

$$t_1 + t_{\mathrm{sm}}(q_H + 2q_s) \leqslant t_c$$

其中，t_{sm} 是 G 中的一个标量乘法的时间，e 是自然对数的底数。

证明 A 为攻击者，A 通过使用算法 B 模拟 Game$I$ 与 A_I 交互以解决 CDH 问题，这里哈希函数 H 被视为随机预言机，列表 L_H 由 B 维护，该列表最初为空。

A_I 在 Game$I$ 的各个阶段进行查询，而 B 在整个游戏中对 A_I 的所有查询做出相应答复，B 由 A 维护。

1) 系统参数查询。首先 A_I 请求 B 进行系统初始化，B 回答此查询如下。B 选择一个随机数 $t \in Z_p$ 作为密钥，生成公钥 $g^t \in G$，并将 (g, g^t, G) 发送给 A_I，t 保密。

2) 哈希查询。A_I 可以在任何时间进行哈希查询，为了跟踪 A_I 所查询过的每个数据块 $f[i]$，B 生成一个列表 L_H : $\{f[i], w_i, k_i, c_i\}$，该列表最初是空的。要回答此查询 B 需执行以下操作。B 检查列表 L_H 对数据块 $f[i]$ 之前是否查询过，如果找到，则 B 响应 $H(f[i])=w_i$ 作为答复；否则，B 随机选择 $c_i \in \{0,1\}$ 使 $c_i=0$ 的概率为 $P_{c_i}=\delta$。B 选择一个随机数 $k_i \in Z_p$，并计算 $w_i = (g^b)^{(1-c_i)} \varphi(g)^{k_i} \in G$，然后将元组 $\{f[i], w_i, k_i, c_i\}$ 添加到列表 L_H 中，并响应 $H(f[i])=w_i$ 作为 A_I 回答。

3) 签名查询。A_I 为数据块 $f[i]$ 制作签名查询，B 对此查询进行以下回答。B 首先发出哈希查询以

获得列表 L_H，然后检查 c_l，如果 $c_l=0$，则 B 报告失败并停止；否则，如果 $c_l=1$，$w_i=\varphi(g)^{k_i}$，并设置 $\sigma_i=(\varphi(g^a)^{k_i})(\varphi(g)^{tk_i})$，观察 $\sigma_i=w_i^a w_i^t=w_i^{(a+t)}$；$B$ 对 A_I 的伪造响应为 σ_i，最终 A_I 生成 $<f[k],\sigma_k^*>$，假设 A_I 为数据块 $f[k]$ 生成一个伪造的签名 σ_k^*，且没有为 $f[k]$ 发出签名查询，现在 B 执行以下步骤生成一个伪造签名。

① B 为 $f[k]$ 发出哈希查询来获取列表 L_H，B 检查 c_l 的值，如果 $c_l=1$，B 停止操作。

② 如果 $c_l=0$ 且 $h(f[k])=w_i=g^b\varphi(g)^k$，则有

$$\sigma_k^*=(g^b\varphi(g)^k)^{(a+t)}=g^{b(a+t)}(\varphi(g)^k)^{(a+t)}=$$

$$g^{ab}g^{tb}\varphi(g^{ak})\varphi(g^{tk})\Rightarrow g^{ab}=\frac{\sigma_k^*}{(g^{tb}\varphi(g^{ak})\varphi(g^{tk}))}$$

B 知道 σ_k^*,g^b,t,k 的值，因此 B 可以计算 g^{ab} 或者可以在多项式时间内解决 G 中的 CDH 问题，这与 CDH 问题困难性的假设相矛盾。

接下来的证明表明 CDH 问题可以由以下概率解决：$\left(\frac{1}{(q_S+1)e}\right)\varepsilon\leqslant\varepsilon'$。

B 要成功需要 3 个事件。

E_1：A_I 的任何签名查询都不会终止。

E_2：A_I 为数据块 $f[k]$ 生成真实且不重复的签名 σ_k。

E_3：A_I 会产生有效且不重要的伪造签名，B 不终止是可能的。

在 3 个事件发生之后，B 成功的概率为 $P(E_1\cap E_2\cap E_3)=P(E_1)P(E_2\mid E_1)P(E_3\mid(E_1\cap E_2))$。

推论 1　A_I 的任何签名查询都不会终止 A_I 的概率至少为 $(1-\delta^{q_S})$。

证明　如上所述，$P[c_l=1]=(1-\delta)$，因此 B 不会终止的概率为 $(1-\delta)$，因为它至少进行 q_S 次签名查询，所以 A_I 查询后不会终止的概率至少为 $(1-\delta)^{q_S}$。

推论 1 证毕。

推论 2　A_I 签名查询后并为 $f[k]$ 生成一个有效的签名且没有终止的概率为 $P(E_2\mid E_1)\geqslant\varepsilon$。

推论 3　在事件 E_1 和 E_2 发生并且 B 没有终止的情况下，A_I 产生有效且不重复伪造的概率为 $P(E_3\mid(E_1\cap E_2))\geqslant\delta$。

证明　$P(E_1\cap E_2\supset E_3)=P(E_1)P(E_2\mid E_1)P[E_3\mid(E_1\cap E_2)]=(1-\delta)^{q_S}\varepsilon\delta$

因此，有

$$\varepsilon'=\delta(1-\delta)^{q_S}\varepsilon \tag{5}$$

为了找到 ε' 的最小值，对式(5)求偏微分

$$\frac{\mathrm{d}\varepsilon'}{\mathrm{d}\delta}=\frac{\mathrm{d}(\delta(1-\delta)^{q_S}\varepsilon)}{\mathrm{d}\varepsilon}=$$

$$((1-\delta)^{q_S})\varepsilon+(\delta q_S(1-\delta)^{(q_S-1)}(-1)\varepsilon)=$$

$$(1-\delta)^{(q_S-1)}\varepsilon((1-\delta)-\delta q_S)=(1-\delta)^{(q_S-1)}\varepsilon(1-\delta(1+q_S))$$

令 $\frac{\mathrm{d}\varepsilon'}{\mathrm{d}\delta}=0$，即 $(1-\delta)^{(q_S-1)}\varepsilon(1-\delta(1+q_S))=0$；因此有 $\delta_{\mathrm{opt}}=\frac{1}{1+q_S}$，因此，式 (5) 变为 $\varepsilon'\geqslant\delta(1-\delta)^{q_S}\varepsilon\geqslant\frac{1}{q_S+1}\left[1-\frac{1}{q_S+1}\right]^{q_S}\varepsilon$，对于较大的 q_S 值 $[1-\frac{1}{(q_S+1)}]\to\frac{1}{\mathrm{e}}$，则概率为 $\varepsilon'\geqslant\left[\frac{1}{(q_S+1)\mathrm{e}}\right]\varepsilon$。

推论 3 证毕。

接下来，进一步证明 B 在多项式时间内解决 CDH 问题。

1) 算法 B 的运行时间等于敌手 A_I 的运行时间与对哈希查询 (q_H+q_S) 的响应时间以及签名查询 (q_S) 的回应时间之和。

2) 每个查询都需要计算群 G 中等于 t_{sm} 的求幂运算。

根据 1)和 2)，算法 B 解决 CDH 问题的总时间为 $t_1+t_{\mathrm{sm}}(q_H+2q_S)\leqslant t_c$。

定理 3 证毕。

7　性能分析

7.1　计算量比较

在此，对文献[9, 16, 25]方案以及本文方案进行计算量比较。符号注释如表 1 所示。假设 CSP 能够诚实地执行操作，返回的结果为非空集合，本文对各个方案算法的计算量进行比较，如表 2 所示。

表 1　**符号注释**

符号	含义
$\lvert u\rvert$	数据用户的大小
E	幂指数操作
n	数据块的大小
m	每条记录的关键词字段的大小
P	配对操作
l	关键词数量
d	搜索结果的数量

表 2 计算量比较

方案	KeyGen	Enc	Trap	Search	Verify
文献[9]方案	$2\|u\|E$	$(8+m)E$	$(9+l)E$	$lP+3P$	—
文献[16]方案	$2\|u\|E+2E$	$(10+m)E+H_o$	$10E$	$lP+3P$	$lP+3P$
文献[25]方案	$(\|\mathrm{DO}\|+1)E$	$(m+5+2n)E+nH_1+3P$	$2E$	$2P+3E$	$(d+1)E+dH_1+2P$
本文方案	$\|n\|E+3E$	$10E+nE$	$(9+l)E$	$4P$	$nE+4P$

通过表 2 可知，本文方案在搜索算法中的效率比文献[9]方案、文献[16]方案和文献[25]方案高。

7.2 功能比较

对文献[9]方案、文献[16]方案、文献[25]方案以及本文方案进行功能比较，如表 3 所示。

表 3 功能比较

方案	数据保密性	可验证查询完整性	支持多关键词搜索	搜索隐私性	支持动态更新
文献[9]方案	是	否	是	是	否
文献[16]方案	是	是	是	是	否
文献[25]方案	是	是	是	是	是
本文方案	是	是	是	是	是

由表 3 可知，文献[9]方案和文献[16]方案不支持动态更新，虽然文献[25]方案和本文方案能够实现同样的功能，但是文献[25]方案是利用代理重加密技术更新数据拥有者以及部分密文实现方案的动态更新，大大增加了开销。而本文方案通过对 Merkle-Tree 的操作实现数据的动态更新（数据的添加和删除操作）。

7.3 仿真分析

在两台机器上进行了实验，其中一台 CPU 为 Intel Core i5 3230M，内存为 4 GB，操作系统为 Windows10；另一台 CPU 为 Intel Core i5 10210U，内存为 8 GB。选用 C++作为编程语言，分别来模拟运行 Encrypt 算法和 Search 算法。如图 5 所示，本文使用从 0 到 1 000 的不同数量的关键词运行了 6 个实验，比较了加密所有关键词和查找具有给定搜索令牌的匹配关键词密文的执行时间。结果显示，加密整个关键词与查找匹配的关键词密文相比，加密整个关键词的成本更高。但加密整个关键词只执行一次，而找到匹配的密文则需要对每个搜索请求执行一次。

由图 6 可知，在 KeyGen 算法中，文献[25]方案的计算成本几乎随着 DO 的数量线性增加，由于文献[25]方案利用代理重加密技术实现对数据拥有者和部分密文的更新，则对于不同的 DO 需要生成不同的密钥，因此增加了计算开销。而本文方案通过对 Merkle-Tree 的操作实现数据的动态更新，不需要对数据拥有者进行变更，所以本文方案在密钥生成算法中计算开销相对平稳，优于文献[25]方案。

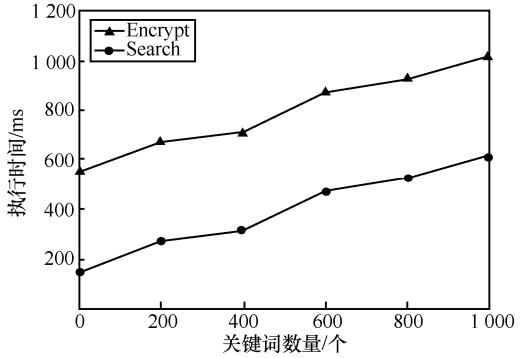

图 5 执行时间

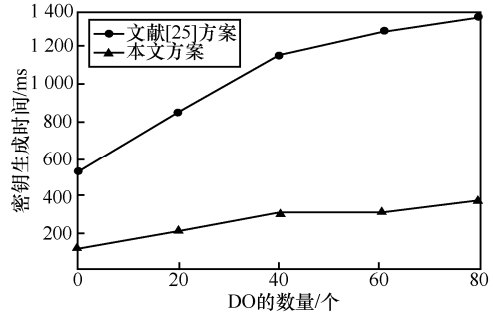

图 6 密钥生成时间

为了实现一个高效的动态方案，本文利用改进的 Merkle-Tree 对文献[16]方案进行拓展实现动态更新。图 7 为更新数据块的数目在 50 到 500 变化的情况下进行文件更新的模拟实验。本文模拟了 10 个实验。从图 7 中可以看出，随着修改数据块数目的增加，计算成本逐渐增加，最后趋于平稳。故本文方案在计算成本上有一定的优势。

8 结束语

本文提出了搜索结果高效验证的多关键词搜索

方案，实现了文献[16]方案不能实现的动态更新功能。首先，利用改进的 Merkle-Tree 认证方法构造搜索方案的验证及更新算法，不但防止了数据篡改、删除和伪造等不法操作的高效验证，而且时间戳字段与根节点的连接保证了数据的新鲜度；其次，对方案的正确性和安全性进行了详细的分析与证明，证明了本文方案满足密文不可区分性和不可伪造性；最后，性能分析表明，本文方案满足高效验证需求下具有低计算成本和更多安全功能的优势。

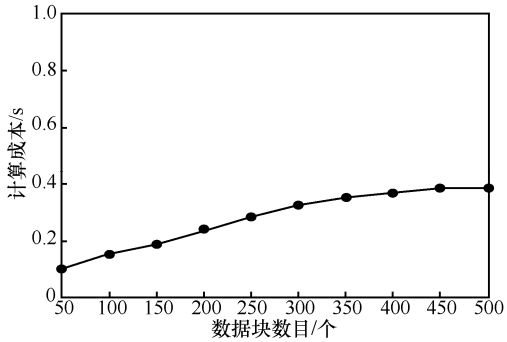

图 7　修改数据块的计算成本

参考文献：

[1] MELL P, GRANCE T. The NIST definition of cloud computing[J]. National Institute of Standards and Technology, 2009, 53(6): 50-53.

[2] ARMBRUST M, FOX A, GRIFFITH R, et al. A view of cloud computing[J]. Communications of the ACM, 2010, 53(4): 50-58.

[3] SAHAI A, WATERS B. Fuzzy identity-based encryption[C]//Annual International Conference on the Theory and Applications of Cryptographic Techniques. Berlin: Springer, 2005: 457-473.

[4] SONG D X, WAGNER D, PERRIG A. Practical techniques for searches on encrypted data[C]//Proceeding 2000 IEEE Symposium on Security and Privacy. Piscataway: IEEE Press, 2000: 44-55.

[5] GOH E J. Secure indexes[J]. IACR Cryptology ePrint Archive, 2003, 6(12): 12-22.

[6] CHANG Y C, MITZENMACHER M. Privacy preserving keyword searches on remote encrypted data[C]//Applied Cryptography and Network Security. Berlin: Springer, 2005: 442-455.

[7] CURTMOLA R, GARAY J A, KAMARA S, et al. Searchable symmetric encryption: improved definitions and efficient constructions[C]//ACM Conference Computer and Communication Security. New York: ACM Press. 2006: 376-379.

[8] GOLLE P, STADDON J, WATERS B. Secure conjunctive keyword search over encrypted data[C]//International Conference on Applied Cryptography and Network Security. Berlin: Springer, 2004: 31-45.

[9] ZHENG Q, LI X, AZGIN A. CLKS: certificateless keyword search on encrypted data[C]//International Conference on Network and System Security. Berlin: Springer, 2015: 239-253.

[10] ZHANG W, LIN Y, XIAO S, et al. Privacy preserving ranked multi-keyword search for multiple data owners in cloud computing[J]. IEEE Transactions on Computers, 2015, 65(5): 1566-1577.

[11] YANG Y, MA M. Conjunctive keyword search with designated tester and timing enabled proxy re-encryption function for e-health clouds[J]. IEEE Transactions on Information Forensics and Security, 2015, 11(4): 746-759.

[12] SHEN J, SHEN J, CHEN X, et al. An efficient public auditing protocol with novel dynamic structure for cloud data[J]. IEEE Transactions on Information Forensics and Security, 2017, 12(10): 2402-2415.

[13] SUN W, WANG B, CAO N, et al. Verifiable privacy-preserving multi-keyword text search in the cloud supporting similarity-based ranking[J]. IEEE Transactions on Parallel and Distributed Systems, 2013, 25(11): 3025-3035.

[14] WANG J, CHEN X, HUANG X, et al. Verifiable auditing for outsourced database in cloud computing[J]. IEEE Transactions on Computers, 2015, 64(11): 3293-3303.

[15] CHUM C S, ZHANG X. A new bloom filter structure for searchable encryption schemes[C]//Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy. New York: ACM Press, 2017: 143-145.

[16] MIAO Y, WENG J, LIU X, et al. Enabling verifiable multiple keywords search over encrypted cloud data[J]. Information Sciences, 2018, 465(10): 21-37.

[17] BONEH D, CRESCENZO G D, OSTROVSKY R, et al. Public key encryption with keyword search[C]//International Conference on the Theory and Applications of Cryptographic Techniques. Berlin: Springer, 2004: 506-522.

[18] GUO Y, LI J, ZHANG Y, et al. Hierarchical attribute-based encryption with continuous auxiliary inputs leakage[J]. Security & Communication Networks, 2016, 9(18): 4852-4862.

[19] MIAO Y, MA J, LIU X, et al. Practical attribute-based multi-keyword search scheme in mobile crowdsourcing[J]. IEEE Internet of Things Journal, 2017, 5(4): 3008-3018.

[20] KUROSAWA K, OHTAKI Y. How to update documents verifiably in searchable symmetric encryption[C]//International Conference on Cryptology and Network Security. Berlin: Springer, 2013: 309-328.

[21] KUROSAWA K. Garbled searchable symmetric encryption[C]//International Conference on Financial Cryptography and Data Security. Berlin: Springer, 2014: 234-251.

[22] GARG N, BAWA S. RITS-MHT: relative indexed and time stamped Merkle Hash tree based data auditing protocol for cloud computing[J]. Journal of Network and Computer Applications, 2017, 84: 1-13.

[23] 杨波. 密码学中的可证明安全性[M]. 北京: 清华大学出版社, 2017.
YANG B. Provable security in cryptography[M]. Beijing: Tsinghua University Press, 2017.

[24] ZHENG Q, LI X, AZGIN A. CLKS: certificateless keyword search on encrypted data[C]//International Conference on Network and System Security. Berlin: Springer, 2015: 239-253.

[25] MIAO Y, MA J, LIU X, et al. VMKDO: verifiable multi-keyword search over encrypted cloud data for dynamic data-owner[J]. Peer-to-peer Networking and Applications, 2018, 11(2): 287-297.

[作者简介]

田有亮（1982– ），男，贵州盘县人，博士，贵州大学教授、博士生导师，主要研究方向为算法博弈论、密码学与安全协议、大数据安全与隐私保护等。

骆琴（1994– ），女，贵州贵阳人，贵州大学硕士生，主要研究方向为可搜索加密协议。