计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2020,29(2):94−100 [doi: 10.15888/j.cnki.csa.007295]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

# 基于模糊身份的动态数据审计方案①

赵陈斌, 许　力, 王　峰

(福建师范大学 数学与信息学院, 福州 350117)
(福建师范大学 福建省网络安全与密码技术重点实验室, 福州 350007)
通讯作者: 许　力, E-mail: xuli@fjnu.edu.cn

**摘　要**: 云存储服务的快速发展, 也带来众多安全挑战. 针对云存储数据的完整性, 已有的基于模糊身份的审计方案仅仅支持静态数据, 因此很多情况并不适用. 本文提出了一种基于模糊身份的动态数据完整性审计方案, 结合默克哈希树的动态数据结构, 实现用户对云端数据的完全动态操作. 该方案采用基于模糊身份的密码体制, 与基于公钥基础设施的数据完整性审计方案相比, 避免了对公钥证书颁发、管理、吊销的过程, 降低了通信代价. 并且该方案能够支持批量验证, 提高认证效率. 最后, 本文从安全性和功能上对新方案进行分析, 能够抵抗伪造攻击, 也保护了数据隐私安全, 并且在功能上较其他方案也有一定的优势.

**关键词**: 云存储安全; 动态操作; 基于模糊身份; 默克哈希树; 数据完整性审计

引用格式: 赵陈斌,许力,王峰.基于模糊身份的动态数据审计方案.计算机系统应用,2020,29(2):94–100. http://www.c-s-a.org.cn/1003-3254/7295.html

## Fuzzy Identity-Based Dynamic Data Auditing Scheme

ZHAO Chen-Bin, XU Li, WANG Feng

(College of Mathematics and Informatics, Fujian Normal University, Fuzhou 350117, China)
(Fujian Provincial Key Lab of Network Security & Cryptology, Fujian Normal University, Fuzhou 350007, China)

**Abstract**: The rapid development of cloud storage services also brings many security challenges. The existing fuzzy identity-based data integrity auditing scheme only focuses on static data, which is obviously not suitable for many practical applications. This study proposes a fuzzy identity-based dynamic data integrity auditing scheme, which combines the dynamic data structure of Merkle hash tree to realize the complete dynamic operations of cloud data. Compared with data integrity auditing schemes based on the public key infrastructure, the scheme avoids the processes of issuing, managing, and revoking public key certificates by using fuzzy identity-based cryptosystem, and reduces the communication cost. Furthermore, the proposed scheme supports batch verification and improves authentication efficiency. Finally, the new scheme is analyzed in terms of security and function, which resists forgery attack and preserves data privacy, and has certain advantages over other schemes in terms of function.

**Key words**: cloud storage security; dynamic operation; fuzzy identity-based; Merkle hash tree; data integrity auditing

① 基金项目: 国家自然科学基金 (U1905211, 61771140); 福建省科技厅高校产学研项目 (2017H6005); 福州市科技局科技重大项目 (榕科 (2017)325 号); 企事业合作项目 (DH-1307)

Foundation item: National Natural Science Foundation of China (U1905211, 61771140); Higher Educations' Industry-University-Research Cooperation Project of Science and Technology Bureau, Fujian Province (2017H6005); Major Project of Science and Technology Bureau of Fuzhou Municipality (RongKe(2017)325); Enterprises-Institution Cooperation Project (DH-1307)

收稿时间: 2019-07-12; 修改时间: 2019-08-20; 采用时间: 2019-08-30; csa 在线出版时间: 2020-01-16

随着互联网不断发展, 网络中传输信息不断增多, 数据不断增长, 大数据时代给社会存储数据及消耗的成本带来了巨大压力[1]. 云存储技术随之产生, 云存储服务基于不同技术结合相关设备, 为广大用户提供云端数据的访问功能及数据存储服务[2]. 云存储服务具有价格低廉、访问自由方便、管理便捷等优势[3], 已经成为云计算中应用最广泛的服务, 成为研究人员关注的焦点. 但是在不断发展过程中, 云存储服务也面对一些严重的安全问题及数据可能会丢失等风险. 早在 2003 年 Deswarte 等[4]提出首个基于 RSA 的远程数据审计方案来保障数据安全, 但计算开销很大.

现有的数据审计方案中, 分成数据持有性证明 (Provable Data Possession, PDP)[5]和可恢复性证明 (Proof of Retrieveability, PoR)[6]两种. 其中 PDP 机制可以不用下载数据文件而直接进行数据的完整性审计. PoR 机制当数据丢失率在一定的范围内, 可以进行数据恢复. 两者的主要区别在于 PoR 机制使用纠错码来进行原始数据的恢复. 但是都只适用于静态数据. 为了解决这个问题, 学者们提出动态的审计方案[7], 但是不能支持完整的动态操作 (例如: 修改、插入、删除). 为了解决这一问题, Erway 等[8]结合认证跳表的数据结构, 首次提出可执行完整动态操作的 PDP 方案. 但是在前期的方案构造中都没有使用第三方可信中心, 因此用户与云服务器之间的审计无法保证公平公正. 随后, Wang 等[9]在默克哈希树 (Merkle Hash Tree, MHT) 动态结构的基础上, 提出支持公开审计和完整动态操作的审计方案. 随着对云存储安全更加广泛的研究, 也不断的衍生出具有更多特性的数据完整性审计方案 [10–12].

在上述方案中, 大多是基于公钥基础设施. 众所周知, 从数字证书的颁发到吊销等一系列过程, 会带来不菲的代价和管理负担. 为解决这一问题, 2015 年, Yu 等[13]和 Zhang 等[14]提出基于身份的审计方案, 支持公开审计, 但两者都不支持动态操作. 2017 年, Wang 等[15]基于索引逻辑表动态数据结构, 提出基于身份的动态审计方案. 同年, Li 等[16]提出基于模糊身份的审计方案, 虽解决了基于 PKI 系统所带来的问题, 但此方案仅支持静态数据, 不支持数据动态操作.

针对已有方案的缺陷, 避免了基于 PKI 系统对数字证书管理过程所带来的负担, 我们采用基于模糊身份的密码体制. 在文献[16]基础上, 结合 MHT 的动态数据结构, 设计新的基于模糊身份的动态数据完整性审计方案, 实现了完整动态操作, 包括修改、插入和删除操作. 最后也从文章的功能和计算代价上与方案[9,13,16]进行比较, 来进一步说明我们的优势.

## 1 预备知识

### 1.1 双线性对

$G_1$和$G_2$是两个给定的素数 $q$ 阶乘法循环群, 其中 $g$ 是$G_1$的生成元. 双线性对运算定义为$e: G_1 \times G_1 \to G_2$. 线性对 $e$ 具有以下性质[17]:

(1) 双线性: $e(g^a, g^b) = e(g,g)^{ab}$, 其中$a, b \in Z_q$.

(2) 可交换性: $e(g^a, g^b) = e(g^b, g^a)$, 其中$a, b \in Z_q$.

(3) 非退化性: $e(g,g) \neq 1$.

### 1.2 默克哈希树 (MHT)

默克哈希树本质上是一棵二叉树[9], 其叶子节点存储数据信息, 用来验证所存储的数据是否完整. 我们通过链接孩子节点的值, 再进行哈希运算进而得到非叶子节点的值. 验证者依次从最底层开始逐层运算, 最终计算出根节点 $Root$ 的值. 如图 1 所示, 验证者拥有根节点值$h_R$, 并对叶子节点$x_2$进行验证. 首先证明者提供验证辅助信息$\theta_2 =< h(x_1), h_b >$, 然后验证者分别计算$h(x_2)$, $h_a = h(h(x_1) \| h(x_2))$和$h_R = h(h_a \| h_b)$, 最终通过对比$h_R$的值是否相等来判断$x_2$有没有被修改.

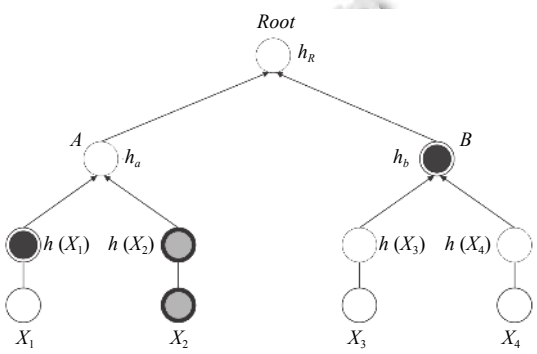


图 1 默克哈希树认证结构

### 1.3 基于模糊身份的签名[18]

具有模糊身份集合$\omega$的用户分发数据标签, 另一用户拥有模糊身份集合$\omega'$, 当且仅当满足$|\omega \cap \omega'| \geq t$时, 验证者才可以进行验证. 具体方案包括以下 4 个算法:

1) 初始化阶段: 系统输入安全参数 $1^k$ 以及门限值 $t$, 概率型算法产生公开参数$\widehat{pp}$以及主密钥$mk$.

2) 密钥生成阶段: 输入模糊身份集合$\omega$, 主密钥$mk$, 则产生用户私钥$pk_\omega$.

3) 标签生成阶段: 输入用户私钥$pk_\omega$, 公开参数$\widehat{pp}$, 以及明文消息$Message$, 输出相应的标签$SIG$.

4) 验证阶段: 输入验证者模糊身份集合$\omega'$, 公开参数$\widehat{pp}$, 明文消息$Message$, 以及标签$SIG$. 如果输出结果$b=1$, 则说明验证通过.

## 2 系统模型

我们的方案包括 4 个实体, 如图 2 所示, 分别为: 密钥生成中心 (Key Generation Centre, KGC), 云服务器 (Cloud Server, CS), 云用户 (Cloud User, CU), 第三方审计者 (Third-Party Auditor, TPA). 其中, KGC 根据云用户的模糊身份集合生成相应的用户私钥. TPA 根据云用户的请求执行数据审计操作. 具体方案由以下 6 个算法组成:

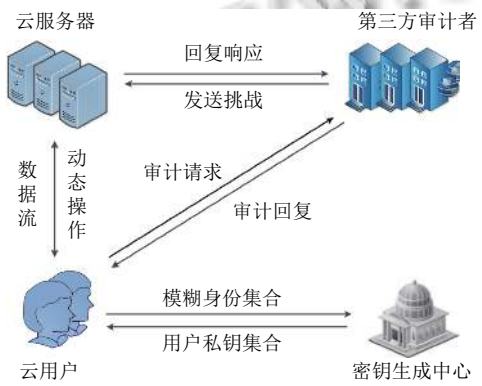


图 2 数据完整性审计系统架构

1) 初始化阶段: KGC 执行该算法, 输入安全参数$1^k$, 门限值$t$, 以及用户模糊身份集合中元素个数最大值$m$. 生成公开参数$\widehat{pp}$和主密钥$mk$.

2) 密钥生成阶段: KGC 执行该算法, 输入用户模糊身份集合$\omega$, 公开参数$\widehat{pp}$和主密钥$mk$, 生成云用户的私钥$pk_\omega$.

3) 标签生成阶段: CU 执行该算法, 输入公开参数$\widehat{pp}$, 用户的私钥$pk_\omega$, 以及文件$File$, 生成数据标签$SIG$.

4) 挑战-响应阶段: TPA 执行此算法, 输入公开参数$\widehat{pp}$, 文件$File$, 以及用户模糊身份集合$\omega$, 输出挑战$CHAL$, 云服务器根据挑战信息进行响应回复.

5) 验证阶段: TPA 接收到 CS 返回的响应 $RESP$, 输入公开参数$\widehat{pp}$, 用户模糊身份集合$\omega'$, 挑战 $CHAL$ 和响应 $RESP$, 进行验证. 如果输出结果为 1, 说明数据未被改变.

6) 动态操作阶段: 当符合模糊身份集合的 CU 执行动态操作, 输入新的文件$File$, 以及用户私钥$pk_\omega$, 生成对应的数据标签上传至 CS.

## 3 安全需求

在云存储系统中, 我们假设 KGC 是可信的, 执行密钥生成操作. TPA 是半可信的, 它会诚实的执行操作, 但是也会对用户的信息产生好奇. 并且 CS 可能会由于某些利益因素, 恶意的篡改或者删除数据. 因此用户数据存储在云端可能会面临以下风险:

1) 正确性: 在审计验证过程中, TPA 首先向 CS 发送挑战, 然后 CS 根据挑战信息进行响应回复. 最终, TPA 验证响应信息的正确性.

2) 隐私性: 受用户委托, TPA 完成云端数据审计. 但在整个审计协议实施过程中, 禁止泄露任何的用户数据信息给 TPA. 即 TPA 会接收到 CS 发送的响应信息, 但 TPA 不能得到用户具体的数据块信息.

3) 伪造攻击: CS 在提供数据存储服务时, 为了维护自身的利益, 可能故意删除普通云用户很少访问的数据文件. 或者由于其他利益因素, CS 会发生篡改数据等恶意行为. 因此 CS 会通过某些方式对数据块和认证标签进行伪造, 以至于在整个审计过程中, 审计者不能发现数据被损坏.

## 4 具体方案

### 4.1 初始化阶段

该算法由 KGC 执行, 进行系统初始化. 其中$G_1$, $G_2$分别是阶为$q$的两个乘法循环群, $q$为大素数. $g$是群$G_1$的生成元, 假定双线性对运算$e: G_1 \times G_1 \to G_2$, 哈希函数$H: \{0,1\}^* \to G_1$. 算法首先随机选择$g_1 = g^y$, $g_2 \in G_1$以及随机数$t_1, t_1, \cdots, t_{m+1} \in G_1, z' \in Z_q$, 并计算$v' = g^{z'}$. 假定$\widehat{M}$为集合$\{1,2,\cdots,m+1\}$, 其中$m$表示模糊身份集合中元素个数的最大值. 定义一个函数$T(x)$为:

$$T(x) = {g_2}^{x^m} \cdot \prod_{i=1}^{m+1} t_i^{\Delta_{i,m}(x)} \tag{1}$$

其中,$\Delta_{i,m}(x)$为拉格朗日插值公式. 最终算法输出系统主密钥$mk = y$以及公开参数:

$$\widehat{pp} = \{g_1, g_2, t_1, \cdots, t_{m+1}, v', \Lambda = e(g_1, g_2)\} \tag{2}$$

### 4.2 密钥生成阶段

用户向 KGC 请求对应私钥, 其拥有模糊身份集合

$\omega$. 首先 KGC 选择$t-1$阶多项式$\hat{q}$, 假设$\hat{q}(0)=y$, 然后选择随机数$r_k \in Z_q$, 进行计算:

$$SK_k = g_2^{\hat{q}(k)} \cdot T(x)^{r_k}, sk_k = g^{-r_k} \tag{3}$$

最终输出用户的私钥$pk_\omega = \{(SK_k)_{k\in\omega}, (sk_k)_{k\in\omega}\}$.

### 4.3 标签生成阶段

CU 首先对文件进行分块得到$F=(m_1, m_2, \cdots, m_n)$, 并对每个数据块$m_i$进行签名. 首先选择随机数$u \in G_1$以及$s_k \in Z_q$, 计算$\Gamma = \mathbb{N} \| u \| Sig(\mathbb{N} \| u)$, 其中$\mathbb{N}$为每个文件对应的文件名. 接下来, 对每个数据块$m_i$生成对应的数据块标签:

$$\begin{cases} SIG_{1,i}^k = \{SK_k \cdot (H(m_i) \cdot \upsilon' \cdot u^{m_i})^{s_k}\} \\ SIG_{2,i}^k = \{g^{-s_k}\} \\ SIG_{3,i}^k = \{g^{-r_k}\} \end{cases} \tag{4}$$

其中, 用$\Phi = \{SIG_{1,i}^k, SIG_{2,i}^k, SIG_{3,i}^k\}_{(1\le i\le n)}$来表示数据块标签集合. 然后, CU 基于 MHT 的动态结构生成根节点 $R$ 的值, 其中叶子节点分别为$H(m_i)(i=1,2,\cdots,n)$, 并对根节点的哈希值进行签名[18], 生成$Sig(H(R))$. 最终 CU 将元组数据$\{F, \Gamma, \Phi, Sig(H(R))\}$上传至 CS 中, 并且删除本地记录.

### 4.4 挑战-响应阶段

拥有模糊身份集合$\omega'$的用户将集合发送给 TPA、发起对云端数据审计的请求. 首先, TPA 判断是否满足条件$|\omega \cap \omega'| \ge t$. 如果满足, TPA 将向 CS 发起挑战. 具体如下:

1) 挑战阶段: TPA 从$[1,n]$中随机选择子集 $L=\{1,2,\cdots,c\}$, 对于每个$i \in L$, 选择相应的随机数$\mu_i \subseteq Z_q$. 然后将挑战信息 $CHAL=\{(i,\mu_i)\}_{(1\le i\le c)}$发送给 CS.

2) 响应阶段: 当 CS 接收到 TPA 的挑战信息, 它首先计算:

$$\begin{cases} \mu = \sum_{i=1}^{c} \mu_i \cdot m_i \in Z_q \\ SIG_1^k = \prod_{i=1}^{c} (SIG_{1,i}^k)^{\mu_i} \\ SIG_2^k = SIG_{2,i}^k \\ SIG_3^k = SIG_{3,i}^k \end{cases} \tag{5}$$

其次, CS 需要给验证者提供部分辅助信息$\{\theta_i\}_{(1\le i\le c)}$. 最终, CS 发送响应回复 $RESP=\left\{\mu, SIG_1^k, SIG_2^k, SIG_3^k, \{H(m_i), \theta_i\}_{(1\le i\le c)}, Sig(H(R))\right\}$给 TPA 进行审计.

### 4.5 验证阶段

当 TPA 接收到 CS 发回的响应, 首先利用$\{H(m_i), \theta_i\}_{(1\le i\le c)}$产生根节点 $R$ 的值, 然后验证根节点哈希值的签名$Sig(H(R))$. 如果验证失败, TPA 直接返回 FALSE. 否则的话, TPA 继续验证:

$$\prod_{(i,\mu_i)\in CHAL} \Lambda^{\mu_i} \stackrel{?}{=} \prod_{k\in s} \left\{ e(SIG_1^k, g) \cdot \prod_{(i,\mu_i)\in CHAL} e(T(k), \{SIG_3^k\}^{\mu_i}) \cdot e((H(m_i)\cdot\upsilon')^{\mu_i} \cdot u^{\mu}, SIG_2^k) \right\}^{\Delta_{i,m}(0)} \tag{6}$$

### 4.6 动态操作阶段

动态操作阶段包括 3 种情况: 数据的修改操作 (M), 插入操作 (I) 以及删除操作 (D).

(1) 修改操作: 当用户把第$i$个数据块 $m_i$ 修改成$m_i'$ (例如: $i=3$, 如图 3). 首先, 在新的数据块$m_i'$的基础上计算其相应的标签$SIG_{1,i}^k{}' = \{SK_k \cdot (H(m_i') \cdot \upsilon' \cdot u^{m_i'})^{s_k}\}$, $SIG_{2,i}^k{}' = \{g^{-s_k}\}$, $SIG_{3,i}^k{}' = \{g^{-r_k}\}$, 因此产生新的标签集合$\Phi' = \{SIG_{1,i}^k{}', SIG_{2,i}^k{}', SIG_{3,i}^k{}'\}$. 然后用户发送更新$update=(M,i,m_i',\Phi')$给 CS. 接收到请求后, CS 执行更新操作: ① 将数据块$m_i$修改成$m_i'$, 并且输出新的文件$F'$; ② 将相应数据块标签进行修改, 输出新的数据标签集合$\Phi'$; ③ 在 MHT 中, 将$H(m_i)$修改成$H(m_i')$, 并且产生新的根节点 $R'$. 更新操作完成后, CS 发送更新执行证明$Proof_{\text{update}} = \{\theta_i, H(m_i), R', Sig(H(R))\}$给用户. 当用户接收到证明信息, 根据$\{H(m_i), \theta_i\}$的值计算出根节点 $R$, 随后验证根节点哈希值的签名$Sig(H(R))$. 如果验证不通过, 输出 FALSE. 否则的话, 用户继续验证 CS 是否执行修改操作, 使用$\{H(m_i'), \theta_i\}$的值来计算新的根节点并与 $R'$进行比较. 如果不等, 输出 FALSE, 否则的话输出 TRUE. 最后, 用户对新产生的根节点 $R'$进行签名, 计算$Sig(H(R'))$并发送到 CS.

(2) 插入操作: 在用户第 $i$ 个数据块 $m_i$ 后插入数据块$m_i^*$ (例如: $i=3$, 如图 4 所示). 具体操作为: 首先, 在新的数据块$m_i^*$的基础上计算其相应的标签$SIG_{1,i}^k{}^* = \{SK_k \cdot (H(m_i^*) \cdot \upsilon' \cdot u^{m_i^*})^{s_k}\}, SIG_{2,i}^k{}^* = \{g^{-s_k}\}, SIG_{3,i}^k{}^* = \{g^{-r_k}\}$. 因此, 产生新的标签集合$\Phi^* = \{SIG_{1,i}^k{}^*, SIG_{2,i}^k{}^*, SIG_{3,i}^k{}^*\}$. 然后, 用户发送更新请求$update=(I,i,m_i^*,\Phi^*)$给 CS. 接收到请求后, CS 执行更新操作: ① 保存数据块$m_i^*$, 并在 MHT 中叶子节点$h(H(m_i))$之后插入新的叶子节点$h(H(m_i^*))$, 输出新的文件 $F^*$; ② 增加新产生的标签, 输

出新的数据标签集合$\Phi^*$; ③ 基于更新之后的 MHT 产生新的根节点 $R^*$. 更新操作完成后, CS 发送更新执行证明$Proof_{\text{update}}=\{\theta_i, H(m_i), R^*, Sig(H(R))\}$给用户. 例如: 我们假设在$h(H(m_3))$之后插入$h(H(m_3{}^*))$, 仅仅增加叶子节点$h(H(m_3{}^*))$和内部节点 $C$, 其中$h_c = h(h(H(m_3)) \| h(H(m_3{}^*)))$. 当用户接收到证明信息, 首先根据$\{H(m_i), \theta_i\}$的值计算出根节点 $R$, 随后验证根节点哈希值的签名$Sig(H(R))$. 如果验证不通过, 输出 FALSE. 否则的话, 用户继续验证 CS 是否执行插入操作, 使用$\{H(m_i{}^*), H(m_i), \theta_i\}$的值来计算新的根节点并与 $R^*$进行比较. 如果不等, 输出 FALSE, 否则, 输出 TRUE. 最后, 用户对新产生的根节点 $R^*$进行签名, 计算$Sig(H(R^*))$并发送到 CS.

3) 删除操作: 删除操作与插入操作刚好相对立. 假设 CS 接收到删除数据块$m_i$的请求 (例如: $i=4$, 如图 5 所示), 则 CS 在存储空间中删除数据块$m_i$以及 MHT 中相应的叶子节点$h(H(m_i))$, 然后重新计算新的根节点 $R''$, 具体细节与插入操作相类似.

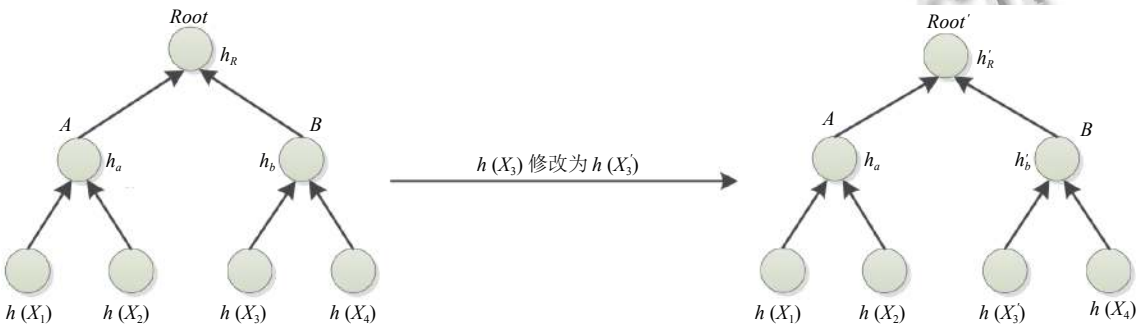


图 3 表示数据块的修改操作, 其中$x_i$和$x_i'$分别指$H(m_i)$和$H(m_i')$

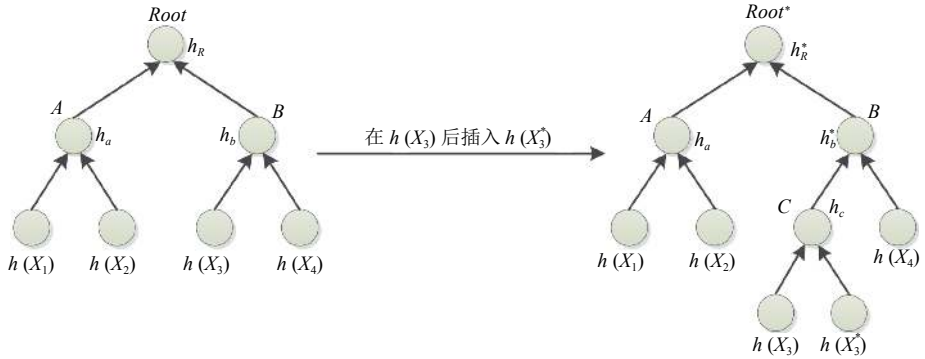


图 4 表示数据块的插入操作, 其中$x_i$和$x_i{}^*$分别指$H(m_i)$和$H(m_i{}^*)$

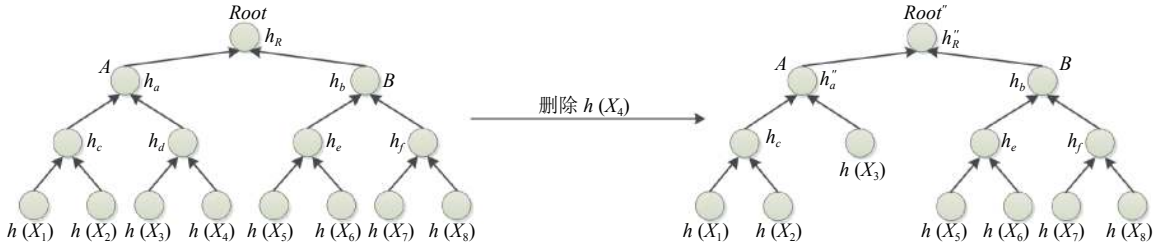


图 5 表示数据块的删除操作

## 5 正确性和安全性分析

在 4.3 节标签生成阶段, 数据标签$SIG_{1,i}^k$中包含对数据块$m_i$的哈希值, 以此作为唯一标识符, 动态操作时数据块信息与相应的标签信息都随之变化, 因此我们的方案支持完整动态操作. 当拥有模糊身份集合$\omega'$的用户想要对云端数据进行审计, 将模糊身份集合发送给 TPA. 首先, TPA 判断是否满足条件$|\omega \cap \omega'| \geq t$. 如果满足, TPA 向 CS 发送挑战信息, TPA 根据响应信息对

云数据进行审计. 若 CS 发送的响应信息正确, 则相应的数据块和标签信息可以通过正确性验证, 具体如下:

$$e(SIG_1^k,g)=e\left(\prod_{i=1}^{c}(SIG_{1,i}^k)^{\mu_i},g\right)=e\left(\prod_{i=1}^{c}(SK_k\cdot(H(m_i)\cdot\upsilon'\cdot u^{m_i})^{s_k})^{\mu_i},g\right)$$
$$=\prod_{i=1}^{c}e(g_2^{\widehat{q}(k)},g)^{\mu_i}\cdot e(T(x)^{r_k},g)^{\mu_i}\cdot e((H(m_i)\cdot\upsilon'\cdot u^{m_i})^{\mu_i},g^{s_k}) \tag{7}$$

然后, 我们计算:

$$\prod_{k\in s}\left\{e\left(SIG_1^k,g\right)\cdot\prod_{(i,\mu_i)\in CHAL}e\left(T(k),\left\{SIG_3^k\right\}^{\mu_i}\right)\cdot e\left((H(m_i)\cdot\upsilon')^{\mu_i}\cdot u^{\mu},SIG_2^k\right)\right\}^{\Delta_{i,m}(0)}$$
$$=\prod_{k\in s}\prod_{(i,\mu_i)\in CHAL}\left\{e(g_2^{\widehat{q}(k)},g)^{\mu_i}\right\}^{\Delta_{i,m}(0)}=\prod_{(i,\mu_i)\in CHAL}e(g_2,g_1)^{\mu_i}=\prod_{(i,\mu_i)\in CHAL}\Lambda^{\mu_i} \tag{8}$$

因此, 式 (6) 中的等式成立, 验证通过.

2) 隐私保护: 在数据审计过程中, TPA 向 CS 发送挑战信息 CHAL, 并且收到响应回复 RESP. 数据信息主要在响应回复中, 但是从式 (5) 中可以看出, 所有的响应回复都是数据块以及认证标签与随机数聚合的结果. 在整个过程中, TPA 只能得到聚合值$\mu$和$SIG_1^k$, 根本得不到具体的数据块$m_i$的信息. 因此用户的数据隐私得到了很好的保护.

3) 抵抗伪造攻击: 假设用户想要对数据块$m_i$进行验证, CS 发现该数据存在丢失或者损坏等情况, 于是通过某种手段伪造一个数据块信息$\widehat{m_i}$, 并且明显可知$\widehat{m_i}\neq m_i$. 当 CS 接收到 TPA 发起的挑战时, 返回的响应信息$RESP$中包含有数据块$\widehat{m_i}$对应的哈希值$H(\widehat{m_i})$以及相应的辅助信息$\widehat{\theta_i}$和根节点哈希值的签名$Sig(H(R))$. 当 TPA 接收到响应信息时, 首先利用$\{H(\widehat{m_i}),\widehat{\theta_i}\}$产生根节点$\widehat{R}$的值, 然后验证根节点哈希值的签名$Sig(H(R))$. 如果通过验证, 则说明 CS 伪造的数据块信息$\widehat{m_i}$以及相应的哈希值$H(\widehat{m_i})$可以通过验证. 则说明 CS 有能力找到$\widehat{m_i}\neq m_i$, 使得$H(\widehat{m_i})=H(m_i)$. 根据哈希函数抗碰撞性: 找出任意两个不同的$x'$和$x$, 使得$H(x')=H(x)$是困难的. 则上述过程与哈希函数的抗碰撞性相矛盾, CS 伪造数据块信息$\widehat{m_i}$不成功. 因此 CS 不可能以不可忽略的概率通过 TPA 的验证. 即本方案能够抵抗伪造攻击.

## 6 功能对比和性能分析

1) 功能对比: 我们在功能上分别与文献[9,13,16]进行对比, 见表 1. 在文献[9]的审计方案中, 支持完整的数据动态操作, 但是没有采用基于模糊身份的密码体制. 在文献[13]的方案是基于身份的密码体制, 文献[16]采用的是基于模糊身份的密码体制, 但是两者都只针对静态数据, 不能支持数据动态操作. 而我们的方案采用基于模糊身份的密码体制, 保证数据完整性的同时, 可以支持完整的动态操作.

表 1 本文方案与已有方案功能对比

| 方案 | 文献[9] | 文献[13] | 文献[16] | 本文方案 |
|---|---|---|---|---|
| 支持动态操作 | √ | × | × | √ |
| 基于模糊身份 | × | × | √ | √ |
| 支持批量验证 | √ | √ | √ | √ |

2) 性能分析: 分别从标签生成阶段, 挑战-响应阶段, 验证阶段, 以及动态操作阶段的计算代价进行分析. 其中$E_{G_1}$和$E_{G_2}$分别表示在群$G_1$和$G_2$中的指数运算, $P$ 表示双线性对运算, $M_{G_1}$和$M_{G_2}$分别表示在群$G_1$和$G_2$中的乘法运算. 从表 2 中可以看出, 相对于文献[9], 我们的计算代价稍有逊色, 但是在可以接受的范围. 但是文献[9]是基于 PKI 密码体制, 数据完整性审计协议需要对公钥证书认证管理. 如证书生成、交付、撤销、更新等. 改进方案使用基于模糊身份的密码体制, 在数据标签生成阶段, 用户将原始数据上传至云端时, 简化了云端对用户的证书认证管理. 在审计操作过程中, TPA 只需要验证用户的模糊身份集合, 简化了云端对用户证书的认证操作. 在云计算中, 大多数验证人员的计算能力有限, 为了提高效率, 基于模糊身份的数据完整性验证更有优势[19]. 并且, 我们的方案在执行动态操作权限管理上也更加灵活. 相对于文献[16]来说, 我们方案相当于该文献方案中 $s$ 取 1 的情况. 因此, 在各个阶段的计算代价相当, 最关键的是本文方案能够支持数据的动态操作, 在云存储的环境下也更加适用.

## 7 结语

本文采用基于模糊身份的密码体制, 提出支持数据动态操作的审计方案. 在完成数据标签生成及审计操作过程中, 本方案简化了基于 PKI 审计方案所带来的证书管理问题. 更进一步结合 MHT 的动态数据结构, 使方案支持完整的动态操作, 且仍然保证数据完整性, 更加适用于云存储服务的需求. 通过进行安全性分析, 可以看出本方案能够抵抗伪造攻击, 且能够保护用户数据在审计方的隐私安全. 最后从功能对比和性能分析上也可以看出本方案的优势.

表 2 本文方案与已有方案计算代价定性分析

| 方案 | 标签生成阶段 | 挑战-响应阶段 | 验证阶段 | 动态操作阶段[2] |
|---|---|---|---|---|
| 文献[9] | $2nE_{G_1}+nM_{G_1}$ | $cE_{G_1}+(c-1)M_{G_1}$ | $4P+(c+1)E_{G_1}+cM_{G_1}$ | $2P+2E_{G_1}+1M_{G_1}$ |
| 文献[13] | $(s+1)nE_{G_1}+snM_{G_1}$ | $cE_{G_1}+(c-1)M_{G_1}$ | $2P+(s+c)E_{G_1}+(s+c-1)M_{G_1}$ | 不支持动态操作 |
| 文献[16] | $(s+3)nE_{G_1}+(s+2)nM_{G_1}$ | $cE_{G_1}+(c-1)M_{G_1}$ | $(2c+1)P+(s+2)cdE_{G_1}+(d+c)E_{G_2}+(s+1)cdM_{G_1}+(3cd+d+c-2)M_{G_2}$ | 不支持动态操作 |
| 本文方案 | $4nE_{G_1}+3nM_{G_1}$ | $cE_{G_1}+(c-1)M_{G_1}$ | $(2c+1)dP+3cdE_{G_1}+(d+c)E_{G_2}+3cdM_{G_1}+(3cd+d+c-2)M_{G_2}$ | $2P+2E_{G_1}+3M_{G_1}$ |

注: 其中$n$表示文件被划分的块数, $s$表示每个数据块划分的扇区数, $c$表示审计方挑战的数据块个数, $d$表示模糊身份集合的身份信息个数. 动态操作阶段我们只以修改操作为例, 插入操作与其一致, 删除操作计算量几乎忽略不计.

## 参考文献


1 Siddiqa A, Karim A, Gani A. Big data storage technologies: A survey. Frontiers of Information Technology & Electronic Engineering, 2017, 18(8): 1040–1070.

2 黄宇, 吴维刚, 赵军平. 分布式云存储: 理论、技术、系统专题前言. 软件学报, 2017, 28(8): 1927–1928. [doi: 10.13328/j.cnki.jos.005205]

3 邓晓鹏, 马自堂, 高敏霞. 一种基于双线性对的云数据完整性验证算法. 计算机应用研究, 2013, 30(7): 2124–2127. [doi: 10.3969/j.issn.1001-3695.2013.07.051]

4 Deswarte Y, Quisquater JJ, Saidane A. Remote integrity checking: How to trust files stored on untrusted servers. Proceedings of the 6th Working Conference on Integrity and Internal Control in Information Systems. Lausanne, Switzerland. 2004. 1−11.

5 Ateniese G, Burns R, Curtmola R, *et al*. Provable data possession at untrusted stores. Proceedings of the 2007 ACM Conference on Computer and Communications Security. Alexandria, VA, USA. 2007. 598−609.

6 Juels A, Kaliski Jr BS. PORs: Proofs of retrievability for large files. Proceedings of the 2007 ACM Conference on Computer and Communications Security. Alexandria, VA, USA. 2007. 584−597.

7 Ateniese G, Di Pietro R, Mancini LV, *et al*. Scalable and efficient provable data possession. Proceedings of the 4th International Conference on Security and Privacy in Communication Netowrks. Istanbul, Turkey, 2008: 9.

8 Erway CC, Küpçü A, Papamanthou C, *et al*. Dynamic provable data possession. Proceedings of the 16th ACM Conference on Computer and Communications Security. Chicago, IL, USA. 2009. 213–222.

9 Wang Q, Wang C, Li J, *et al*. Enabling public verifiability and data dynamics for storage security in cloud computing. Proceedings of the 14th European Symposium on Research in Computer Security. Saint-Malo, France. 2009. 355–370.

10 Fu AM, Yu S, Zhang YQ, *et al*. NPP: A new privacy-aware public auditing scheme for cloud data sharing with group users. IEEE Transactions on Big Data, 2017. [doi: 10.1109/TBDATA.2017.2701347]

11 Yan H, Li JG, Han JG, *et al*. A novel efficient remote data possession checking protocol in cloud storage. IEEE Transactions on Information Forensics and Security, 2017, 12(1): 78–88. [doi: 10.1109/TIFS.2016.2601070]

12 Li JG, Yan H, Zhang YC. Certificateless public integrity checking of group shared data on cloud storage. IEEE Transactions on Services Computing, 2018: 1. [doi: 10.1109/TSC.2018.2789893]

13 Yu Y, Zhang YF, Mu Y, *et al*. Provably secure identity based provable data possession. In: Au MH, Miyaji A, eds. Provable Security. Cham: Springer, 2015. 310–325.

14 Zhang JH, Dong QC. Efficient ID-based public auditing for the outsourced data in cloud storage. Information Sciences, 2016, 343–344: 1–14. [doi: 10.1016/j.ins.2015.12.043]

15 Wang F, Xu L, Wang HQ, *et al*. Identity-based non-repudiable dynamic provable data possession in cloud storage. Computers & Electrical Engineering, 2018, 69: 521–533.

16 李艳楠. 基于属性的云数据审计协议研究[硕士学位论文]. 成都: 电子科技大学, 2017.

17 Boneh D, Franklin M. Identity-based encryption from the Weil pairing. SIAM Journal on Computing, 2003, 32(3): 586–615. [doi: 10.1137/S0097539701398521]

18 Yang PY, Cao ZF, Dong XL. Fuzzy identity based signature with applications to biometric authentication. Computers & Electrical Engineering, 2011, 37(4): 532–540.

19 Wang HQ. Identity-based distributed provable data possession in multicloud storage. IEEE Transactions on Services Computing, 2015, 8(2): 328–340. [doi: 10.1109/TSC.2014.1]