软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2020,31(2):578−596 [doi: 10.13328/j.cnki.jos.005633]

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

用户可动态撤销及数据可实时更新的云审计方案*

韩 静[1], 李艳平[1], 禹 勇[2], 丁 勇[3]

[1](陕西师范大学 数学与信息科学学院,陕西 西安 710119)

[2](陕西师范大学 计算机科学学院,陕西 西安 710119)

[3](广西密码学与信息安全重点实验室(桂林电子科技大学),广西 桂林 541004)

通讯作者: 李艳平, Email: lyp@snnu.edu.cn

摘 要: 随着云存储的出现,越来越多的用户选择将大量数据存储在远程云服务器上,以节约本地存储资源.如何验证用户远程存储在云端数据的完整性,成为近年来学术界的一个研究热点.虽然现已提出了很多云审计方案,但大多数方案都假设个人和企业在使用云存储系统的整个过程中,用户及其公私钥始终不变,且不能高效地对数据进行实时动态更新.为此,提出一种轻量级的支持用户可动态撤销及存储数据可动态更新的云审计方案.首先,该方案允许用户可高效地动态撤销(包括更换公私钥),在用户撤销阶段,采用了多重单向代理重签名技术,新用户只需计算重签名密钥,而无需从云端下载数据再重新签名后上传到云端;其次,该方案能够保证数据可实时动态更新(插入、删除、修改),通过在数据块的身份识别码中引入虚拟索引,数据动态更新时,只有被更新数据块的身份识别码发生变化,其余数据块的身份识别码保持不变;最后,在重签名阶段,云服务器代替新用户进行签名,在审计阶段,第三方审计者代表当前用户对存储在远程云服务器上的数据进行完整性验证,减轻了终端用户的计算开销及系统的通信开销(轻量级).安全性分析和性能分析进一步说明,该方案是安全的和高效的.

关键词: 云审计;数据完整性;用户动态可撤销;轻量级;虚拟索引;隐私保护

中图法分类号: TP333

中文引用格式: 韩静,李艳平,禹勇,丁勇.用户可动态撤销及数据可实时更新的云审计方案.软件学报,2020,31(2):578−596. http://www.jos.org.cn/1000-9825/5633.htm

英文引用格式: Han J, Li YP, Yu Y, Ding Y. Cloud auditing scheme with dynamic revocation of users and real-time updates of data. Ruan Jian Xue Bao/Journal of Software, 2020,31(2):578−596 (in Chinese). http://www.jos.org.cn/1000-9825/5633.htm

Cloud Auditing Scheme with Dynamic Revocation of Users and Real-time Updates of Data

HAN Jing[1], LI Yan-Ping[1], YU Yong[2], DING Yong[3]

[1](School of Mathematics and Information Science, Shaanxi Normal University, Xi'an 710119, China)

[2](School of Computer Science, Shaanxi Normal University, Xi'an 710119, China)

[3](Guangxi Key Laboratory of Cryptography and Information Security (Guilin University of Electronic Technology), Guilin 541004, China)

Abstract: With the advent of cloud storage, more and more users choose to store large amounts of data on the remote cloud server in order to save local storage resources. In recent years, how to verify the integrity of remote stored data in the cloud has been become a hotspot in academia. Although many cloud auditing protocols have been put forward, most of them are based on the assumption that users (individuals or enterprises) and their public/private keys remain constant in the whole process of using cloud storage system, and these

* 基金项目: 国家自然科学基金(61802243, 61872229, 61772150); 陕西省工业领域重点研发项目(2019GY-013); 中央高校基本科研业务费专项资金(2018CSLY002, GK201803005)

Foundation item: National Natural Science Foundation of China (61802243, 61872229, 61772150); Key R&D Program in Industry Field of Shaanxi Province (2019GY-013); Fundamental Research Funds for the Central Universities (2018CSLY002, GK201803005)

收稿时间: 2017-09-25; 修改时间: 2017-12-28, 2018-06-06; 采用时间: 2018-07-23

schemes cannot dynamically update data in real time. Therefore, this study proposes a lightweight cloud auditing scheme which supports dynamic revocation of users and real-time updating of data. First of all, this scheme allows users to revoke dynamically and efficiently (including the updating of public private keys), multi-use unidirectional proxy re-signature technology is adopted in the stage of revocation, that is, a new user simply needs to calculate the re-signature key instead of downloading data from the cloud to re-sign and then uploading it to the cloud. Secondly, this scheme can realize the data dynamic updating (inserting, deleting, and modifying) in real time by introducing the virtual index into the identification code of data block. Consequently, only the identification code of updated data block changes while the other's remain unchanged when dynamically updating data. Finally, in the stage of re-signature, the cloud server is able to represent a new user to re-sign, and in the stage of auditing, third party audit center can represent the current user to verify the integrity of data in the cloud, which greatly reduce the computational overhead of user and communication overhead of system (lightweight). The security and performance analyses of this study further show that the proposed scheme is secure and efficient.

Key words: cloud auditing; data integrity; dynamic revocation of users; lightweight; virtual index; privacy protection

云存储是一个由网络设备、存储设备、服务器、应用软件、公用访问接口、接入网和客户端等多个部分组成的系统.它可以使用户以较低廉的价格获取海量的存储能力,但高度集中的计算资源使云存储面临着严重的安全挑战.最近几年,各大云运营商各自暴露的安全存储问题,引起了人们的广泛关注与担忧.如 2011 年 3 月,谷歌 Gmail 邮箱出现故障,造成大约 15 万用户的数据丢失.2013 年 8 月,国内云提供商盛大云因机房一台物理服务器磁盘发生故障,导致客户部分数据丢失.由此可见,远程云存储中数据安全问题的研究是非常有意义的.

通常,用户将数据原文直接存储至远程云服务器,其为了节约本地存储资源,可能并没有保存数据副本,此时用户可能面临以下 3 种损坏数据的行为[1−5]:① 云服务器的软件失效或硬件损坏;② 云服务器可能遭到其他用户的恶意攻击,如文献[6]以亚马逊弹性计算云 Amazon EC2(Amazon elastic compute cloud)存储服务为例,指出恶意的用户可以对云服务器中同一宿主机上的其他虚拟机发起攻击,损坏其他用户的数据;③ 云服务提供商 CSP(cloud service provider)可能没有遵守服务等级协议(service level agreement,简称 SLA),其为了经济利益,擅自删除一些用户不常访问的数据,或采取离线存储的方式,因此用户无法保证存储在云中数据的完整性.

近年来,Wang 等人实现了一种支持全动态操作的 PDP(provable data possession)机制[7].该机制采用 Merkle 哈希树来确保数据块存储在正确的位置上,而数据块值则通过 BLS 签名机制来确保.为了减轻用户的负担,该机制还引入独立的第三方审计者 TPA(third party auditor)来代替用户验证云中数据的完整性,但采用这种方式存在用户隐私泄漏的风险.针对这一缺陷,Wang 等人提出了另一种保护隐私的数据完整性验证机制[8].该机制通过随机掩码技术,有效地隐藏了云服务器返回证据中的数据信息,使得 TPA 无法获取数据真实内容.

文献[7−10]提出了一系列经典的具有数据隐私保护功能的云数据公开审计方案,但这些方案都局限于用户(企业和个人)在使用云存储系统过程中,用户(及其公私钥)始终不变,且不能高效地对数据进行实时动态更新.

- 首先,用户始终不变的原因是云存储服务器中数据认证标签与用户的私钥密切相关.若用户(用户的公私钥对)已更换,且云服务器中依然保存着原私钥签名的数据认证标签,则 TPA 无法完成审计任务.而在实际应用中,显然存在用户随时撤销的需求,比如:(i) 在一个云存储系统中,经过一段时间,用户的公私密钥可能会因为某些原因而进行更新;(ii) 用户可能是一个公司数据的管理者,他可能因为某些原因而离职,例如任期已满或跳槽等.
- 其次,这些方案中,数据块的认证标签中均包含了数据块的真实索引,在这种情况下,云中数据的动态更新效率不高,如果插入或删除一个数据块,则该数据块之后所有数据块的索引都会发生变化,即使这些数据块的内容并没有改变,用户依然必须对改变索引的数据块重新计算其认证标签,这样的云审计方案并不能够高效地实现数据动态更新.

因此,一个支持用户高效动态可撤销、存储数据可高效实时动态更新的云存储数据审计方案更满足于实际应用.

Wang 等人首先引入共享云存储审计问题[11],提出了一个基于群签名的用户可撤销的自我审计方案,以及基于动态广播重签名方案和双向代理签名的共享云用户可撤销公开审计方案[12,13].随后,Yuan 等人使用了一个类似的群签名技术提出了一个公开方案版本[14].由于上述方案都涉及到群签名和广播加密技术,导致用户可撤

销审计方案的效率太低,难以满足实际需求.2013 年,Wang 等人提出了一个高效的用户可撤销公开审计方案[15].此方案借助代理重签名技术,将已撤销用户的数据块认证标签转换为当前用户签名形式,很好地满足了用户可撤销的需求.2015 年,Wang 等人在原方案[15]的基础上提出了 Panda[16].该方案减少了重签名密钥计算的次数,并支持多任务同时审计,极大地提高了公共审计效率,是当前此类问题非常优秀的解决方案.然而该方案具有一定的局限性:(1) 云服务器与已撤销用户合谋可能会造成当前用户私钥的泄露;(2) 第三方审计者 TPA 与已撤销用户合谋可能会造成 TPA 窃取用户的数据隐私.2016 年,张新鹏等人提出了基于代理重签名的支持用户可撤销的云存储数据公共审计方案[17],但该方案并不能满足用户动态可撤销和云中数据可实时动态更新.2017 年,Yang 等人提出了用户可撤销和数据动态更新的公共云审计方案[18],但该方案并未解决用户可随时动态地撤销、已撤销用户与 TPA 合谋及已撤销用户的可追踪问题;其次,该方案采用了传统的数据动态更新方法(见本文第 3.2 节第 2 段解释),数据动态更新效率不高.随后,徐云云等人提出了云存储中基于虚拟用户的数据完整性验证方案[19],Samundiswary 等人提出了安全的用户可撤销的公共云审计方案[20].然而,这些方案[19,20]均未解决已撤销用户与 TPA 合谋、已撤销用户可追踪、用户数据隐私保护及数据动态可动态更新问题.2018 年,Liu 等人提出了基于数据备份的用户可撤销共享数据的公共云审计方案[21].该方案允许云服务器对用户存储的数据文件进行临时签名,避免了生成重签名密钥的过程,有效地防止了云服务器与已撤销用户合谋造成的当前用户私钥泄露的问题.但该方案备份存储需云服务器提供巨大的存储空间,进而云用户须为之付出昂贵的存储代价;其次,该方案并未解决已撤销用户与 TPA 合谋攻击、已撤销用户的可追踪问题.

为了更好地解决上述存在问题,本文提出了一种轻量级的支持用户可动态撤销、数据可实时更新(主要指插入、删除、修改操作,后不再赘述)的云审计方案,具有以下特点.

(1) 将用户任期的哈希值与选取的随机数作乘积,作为用户的私钥,用于计算数据块的签名认证标签,使 CSP 代替现任用户重签名时不受上任用户任期的影响,上任用户可以做到随时动态可撤销.

(2) 在数据动态更新的阶段引入了虚拟索引,确保所有的数据块按正确的顺序排序,在动态更新的过程中只改变被更新数据块的虚拟索引,其他数据块的虚拟索引不变,也就是其相对应的认证标签也不用改变,提高了动态更新的效率.

(3) 重签名密钥的计算方式防止了已撤销用户与 CSP 的合谋攻击,解决了用户私钥泄露问题;证据μ'的盲化采用了随机掩码技术和抗已撤销用户与 TPA 的合谋攻击,使 TPA 无法获得用户的数据隐私信息.

(4) 本文从代理重签名安全、审计安全、隐私保护这 3 个方面进行了方案的安全性分析;在通信开销、计算开销及性能方面与文献[15,17–21]进行了分析比较,说明本文方案具有一定的轻量级优势和实用价值.

1 系统模型与设计目标

1.1 系统基本模型

本文方案的系统模型由 3 个部分构成,如图 1 所示.

- 用户(U_j):他们拥有大量的数据需要存储到云上.
- 云服务提供商:提供数据存储服务、大量的存储空间和计算资源.
- 第三方审计者:他们拥有用户 U_j 没有的审计专业技能和能力,能够代表用户 U_j 对存储在云服务器上的数据进行完整性检验.

用户 U_j 依靠 CSP 对大量的数据进行存储和维护,他们之间根据实际需要进行动态交互.因为用户 U_j 不再拥有本地的数据记录,所以确保存储在云中数据的完整性是至关重要的.为了节约计算资源和减轻用户 U_j 的负担,用户 U_j 依靠 TPA 去验证其外包数据的完整性;同时,在审计过程中,要防止数据隐私被 TPA 窃取.

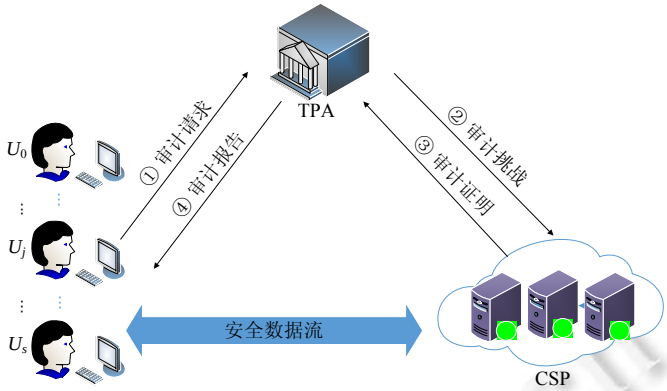


Fig.1 System model

图 1 系统模型

假设某一时段内,只有 1 个用户对数据进行管理.当该用户的任期结束,就更换新用户继续对数据进行管理(同一用户的动态撤销可视作该用户不同时期公私钥对的更改).按照时间先后顺序将不同的用户依次记为 $U_0,U_1,\ldots,U_s$,相应任期依次记为 $T_0,T_1,\ldots,T_s$.最初,初始用户 U_0 对文件 F 进行分块,并利用自己的私钥计算所有数据块的认证标签 $\sigma_i^{(0)}$ (它表示初始用户 U_0 对数据块 m_i 的认证标签).当 U_0 任期结束后,U_1 将会取代 U_0 继续对数据进行管理,依次类推,当 U_{j-1} 被 U_j 取代后,U_j 将计算重签名密钥,并将其发送给 CSP,由 CSP 代替新上任的用户 U_j 实施代理重签名.在每一位用户的任期内,他都可以对存储在云端的数据进行实时动态更新(插入、删除、修改等操作).当用户需要验证存储在云中数据的完整性时,U_j 发送审计请求给 TPA.TPA 将会代表 U_j 对存储在 CSP 上相应的数据进行完整性验证.

1.2 设计目标

本文方案设计实现以下 6 个安全高效的目标.

1. 用户动态可撤销:每任用户都可以随时高效地动态撤销,且已撤销用户不会对云存储系统中的 TPA、CSP、现任用户增加负担.
2. 存储数据动态更新:每任用户在任期内都可对存储在 CSP 上的数据进行合理的动态更新(插入、删除和修改),且对整个云存储系统产生的计算和通信开销非常小.
3. 公共审计:TPA 可以代表用户验证存储在云中数据的完整性,且不会对用户增加额外的负担.
4. 存储正确性:当且仅当 CSP 完整地保存了用户的真实数据,CSP 生成的审计证明 P 才能成功通过 TPA 的审计.
5. 隐私保护:i) 抗已撤销用户与 CSP 合谋攻击,从而避免用户私钥泄露给 CSP;ii) 在审计过程中,抗已撤销用户与 TPA 合谋及采用随机掩码技术盲化证据 μ',使得 TPA 不能获得用户的任何数据信息,从而保护了用户的数据隐私.
6. 轻量级:在重签名阶段,CSP 代理用户进行重签名;在审计阶段,TPA 代替用户进行数据的完整性验证,减轻了用户的计算开销和云存储系统的通信开销,此时,用户可以是计算能力受限的移动终端.此外,CSP、TPA 拥有远远高于普通用户强大的计算能力和专业技能,所以整个云存储系统的运行耗时将会有效降低.

2 定义与框架

2.1 基础知识

2.1.1 双线性映射

G_1,G_2 是阶为素数 p 的循环乘群,g 是群 G_1 的生成元,双线性映射 $e:G_1\times G_1\to G_2$,满足如下的性质.

(1) 双线性性:给定任意元素 $u,v\in G_1$,对任意的 $a,b\in Z_p$,有 $e(u^a,v^b)e(u,v)^{ab}$.

(2) 非退化性:$e(g,g)\neq 1$.

(3) 可计算性:存在有效的算法,对任何合法输入都能有效地进行计算 e.

(4) 可交换性:$e(u_1\cdot u_2,v)=e(u_1,v)\cdot e(u_2,v)$,其中,$u_1,u_2,v\in G_1$.

2.1.2 代理重签名算法

为了实现第 1.2 节所描述的目标,本方案在代理重签名方案[22]中做了进一步的修改.原代理重签名方案具体算法如下.

- *Global_setup*(λ):输入安全参数λ,G_1 与 G_2 为素数阶 p 的循环群,g 为群 G_1 的生成元,双线性映射:$e:G_1\times G_1\to G_2$,哈希函数:$H:\{0,1\}^*\to G_1$.
- *Keygen*(λ):用户 i 的公钥为 $Y_i=g^{x_i},x_i\in_R Z_p^*$.
- *Rekeygen*(x_j,Y_i):计算重签名密钥: $k_{i\to j}=Y_i^{(1/x_j)}=g^{x_i/x_j}$.
- *Sign*($1,x_i,m$):计算第 1 层签名: $\sigma^{(1)}=H(m)^{x_i}\in G_1$.
- *Sign*($2,x_i,m$):计算第 2 层签名: $\sigma^{(2)}=(\sigma_0,\sigma_1,\sigma_2)=(H(m)^{x_i t},Y_i^t,g^t),t\in_R Z_p^*$.
- *Re-Sign*($1,m,\sigma^{(1)},k_{i\to j},Y_i,Y_j$):检验$\sigma^{(1)}$是否有效:若有效,则计算:

$$\sigma^{(2)}=(\sigma_0',\sigma_1',\sigma_2')=((\sigma^{(1)})^t,Y_i^t,k_{i\to j}^t)=(H(m)^{x_i t},Y_i^t,g^{tx_i/x_j}).$$

令 $\tilde{t}=tx_i/x_j$, 则 $\sigma^{(2)}=(H(m)^{x_i\tilde{t}},Y_j^{\tilde{t}},g^{\tilde{t}})$.

- *Verify*($1,m,\sigma^{(1)},Y_i$): $e(\sigma^{(1)},g)\stackrel{?}{=}e(H(m),Y_i)$.
- *Verify*($2,m,\sigma^{(2)},Y_i$): $e(\sigma_0,g)\stackrel{?}{=}e(\sigma_1,H(m)),e(\sigma_1,g)\stackrel{?}{=}e(Y_i,\sigma_2)$.

2.2 困难性假设

1. Discrete Logarithm problem(DLP):设 G 为一个素数 p 阶的循环乘群,g 是群 G 的生成元,给定元素 $g^c\in G$ 和 g,计算 c 是困难的.
2. L-flexible Diffie Hellman problem(L-FlexDH)[23]:设 G 为一个素数 p 阶的循环乘群,g 是群 G 的生成元,给定$(g,A=g^a,B=g^b)\in G^3$,输出一个 $2L+1$ 重数组 $(C_1,C_2,...,C_\ell,D_1^a,D_2^a,...,D_L^a,D_L^{ab})\in G^{2L+1}$ 是困难的,其中,$C_L=g^{t_L},D_L=g^{t_1t_2\cdots t_L}$ 满足如下关系:

$$\log_g(D_j)=\prod_{i=1}^{j}\log_g(C_i)\neq 0,j\in\{1,2,...,L\}.$$

3 本文方案

该方案包含 8 个多项式时间算法:Setup、SigGen、ReKeygen、ReSiggen、DynUpdate、Challenge、GenProof、VerifyProof;涉及 3 方:云服务提供商 CSP、第三方审计者 TPA、用户 U(负责管理数据并将其上传至 CSP).考虑到现实情况中数据管理者 U 可能因为某些原因而需随时更换,故本文用 $U_0,U_1,\ldots,U_s$ 来表示时间顺序上动态更换的用户,$T_0,T_1,\ldots,T_s$ 表示每任用户的任期.

3.1 主方案

Setup:输入安全参数λ,云存储系统输出公开参数$\{G_1,G_2,p,g,e,H_1,H_2,h,u,\rho\}$,其中,$G_1$ 与 G_2 是素数 p 阶循环乘群,g 为 G_1 的生成元,$e:G_1\times G_1\to G_2$ 为双线性对,$H_1:\{0,1\}^*\to G_1,H_2:\{0,1\}^*\to Z_p$, $h:\{0,1\}^*\to Z_p^*$ 为从安全的哈希函数族中随机选择的 3 个哈希函数,$u\in G_1$ 为全局常量,$\rho=2^\delta$为步长($\delta\in N^*$由存储文件的类型及用户的数量决定,其具体量化关系见本文第 3.2 节数据动态更新部分).

系统中每一个用户 U_j 选取一个随机数 $x_j\in Z_p^*$,计算其私钥 $X_j=sk_j=x_jh(T_j)$、公钥 $Y_j=pk_j=g^{x_jh(T_j)}=g^{X_j}$ 、任期的认证标签 $t^{(j)}=T_j\|Sig_{x_i}(T_j)$,其中,公开 $pk_j,j\in Q=\{0,1,2,\ldots,s\}$.

SigGen:初始用户 U_0 对文件 F 进行分块 $F=\{m_1,m_2,\ldots,m_n\}$,每一个数据块 $m_i\in Z_p$,其中,$i\in I=\{1,2,\ldots,n\}$,每一个文件 F 有一个 Tag_F,Tag_F 中包含文件的名字或文件的其他属性特征.

初始用户 U_0 输入其公私密钥对(X_0,Y_0).对每一个数据块 $m_{i\in I}$,U_0 计算数据块 m_i 的认证标签:

$$\sigma_i^{(0)}=(H_1(id_i)u^{m_i})^{X_0},$$

其中,$id_i=\{name\|\eta_i\|\xi_i\}$ 为数据块 m_i 的身份识别码,$name$ 为文件 F 的名字,$\eta_i=i\cdot\rho$ 为数据块 m_i 的虚拟索引,$\xi_i=H_2(m_i\|\eta_i)$.认证标签集合记为 $\Phi^{(0)}=\{\sigma_i^{(0)}\}_{i\in I}$ .U_0 发送 $\{F,Tag_F,\Phi^{(0)},t^{(0)}\}$ 给 CSP,随后,U_0 删除本地数据记录 F.

ReKeygen:如果用户 U_{j-1} 任期已满,则新的用户 U_j 将取代 U_{j-1},此时,U_j 需要计算重签名密钥 $k_{j-1\to j}$ 并将其发送给 CSP:

$$k_{j-1\to j}=(Y_{j-1})^{1/X_j}=g^{X_{j-1}/X_j}.$$

ReSiggen:CSP 收到新上任用户 U_j 的重签名密钥 $k_{j-1\to j}$ 后,代表用户 U_j 对数据块 m_i 进行代理重签名.已撤销用户的任期是公开的,为了后继表示简洁,令

$$\gamma_j=h(T_0)h(T_1)...h(T_{j-1})=\prod_{\ell=0}^{j-1}h(T_\ell),\tau_j=h(T_j)(X_{j-1}/X_j)=h(T_{j-1})(x_{j-1}/x_j),\tilde{\gamma}_j=\prod_{\ell=1}^{j}\tau_\ell.$$

(1) 计算第 $j-1$ 层~第 j 层的新标签 $\phi_i^{(j)}$:

$$\phi_i^{(j)}=(\phi_i^{(j-1)})^{h(T_j)}=(\sigma_i^{(0)})^{h(T_1)h(T_2)...h(T_j)}=(\sigma_i^{(j)})^{\tilde{\gamma}_j}.$$

(2) CSP 对已撤销用户的公钥和重签名密钥进行如下处理:

$$\alpha_\ell=(Y_{\ell-1})^{\prod_{\theta=\ell}^{j}h(T_\theta)}=(g^{x_{\ell-1}h(T_{\ell-1})})^{h(T_\ell)h(T_{\ell+1})...h(T_j)}=Y_j^{\prod_{\theta=\ell}^{j}\tau_\theta},$$

$$\beta_\ell=(k_{\ell-1\to\ell})^{h(T_\ell)}=(g^{X_{\ell-1}/X_\ell})^{h(T_\ell)}=(g^{x_{\ell-1}h(T_{\ell-1})/x_\ell h(T_\ell)})^{h(T_\ell)}=g^{\tau_\ell}.$$

(3) CSP 将 $S_i^{(j)}=(\phi_i^{(j)},\{\alpha_\ell\},\{\beta_\ell\})$ 作为对数据块 m_i 的第 j 次代理重签名.

Challenge:当现任用户 U_j 想要验证存储在 CSP 上数据的完整性时:

(1) U_j 发送审计请求、通信时间上限θ和数据动态更新后的数据块的块索引集合 $\tilde{I}$ (在动态更新算法 *DynUpdate* 中将会介绍)给 TPA.

(2) TPA 收到 U_j 的上述信息后,从块索引集合中 $\tilde{I}$ 随机选取 c 个块索引 $\bar{I}=\{s_1,s_2,...,s_c\}$,并对每一个块索引 $i\in\bar{I}$,选取随机数 $v_i\in Z_p^*$ (其比特长度应小于$|p|$,文献[24]已给出详细解释),组成审计挑战 $chal=\{i,v_i\}_{i\in\bar{I}}$ .TPA 将挑战请求 *chal* 发送给 CSP,并记录此时的时间 CT_1.

GenProof:当 CSP 接收到来自 TPA 的挑战请求 *chal* 后:

(1) CSP 首先计算数据块 $m_i(i\in\bar{I})$ 的线性组合 $\mu'=\sum_{i=1}^{c}m_iv_i$. 为了防止用户数据隐私泄露,CSP 选取一个随机数 $r\in Z_p^*$ 盲化μ',其计算:

$$R=u^r\in G_1,\psi=h(R),\mu=\mu'+r\psi\in Z_p^*,\sigma=\left(\prod_{i=1}^{c}\sigma_i^{(j)v_i}\right)^{\tilde{\gamma}_j}=\prod_{i=1}^{c}\phi_i^{(j)v_i}.$$

(2) CSP 返回审计证明 $P=\{\sigma,\mu,R,\{\alpha_\ell\}_{\ell\in[1,j]},\{\beta_\ell\}_{\ell\in[1,j]}\}$ 给 TPA.

VerifyProof:当 TPA 接收到 CSP 返回的证明 P 时,记录此刻的时间 CT_2:

(1) TPA 先计算$\Delta t=CT_2-CT_1$:若$\Delta t\leqslant\theta$,TPA 执行审计并输出“Valid”;否则,TPA 中止审计任务并输出“Invalid”.

(2) TPA 验证等式:

$$e(\sigma,g)=e\left(u^{\mu}\cdot R^{-\psi}\cdot\prod_{i\in\bar{I}}H_1(id_i)^{v_i},\alpha_1\right),$$

$$e(\alpha_\ell,g)=e(\alpha_{\ell+1},\beta_\ell),$$

$$e(\alpha_j,g)=e(Y_j,\beta_j),\ \ell\in[0,j-1].$$

如果上述等式成立,则说明存储在云服务器中的数据是完整的,TPA 输出“Success”;否则,TPA 输出“Failure”.

算法的流程如图 2 所示.

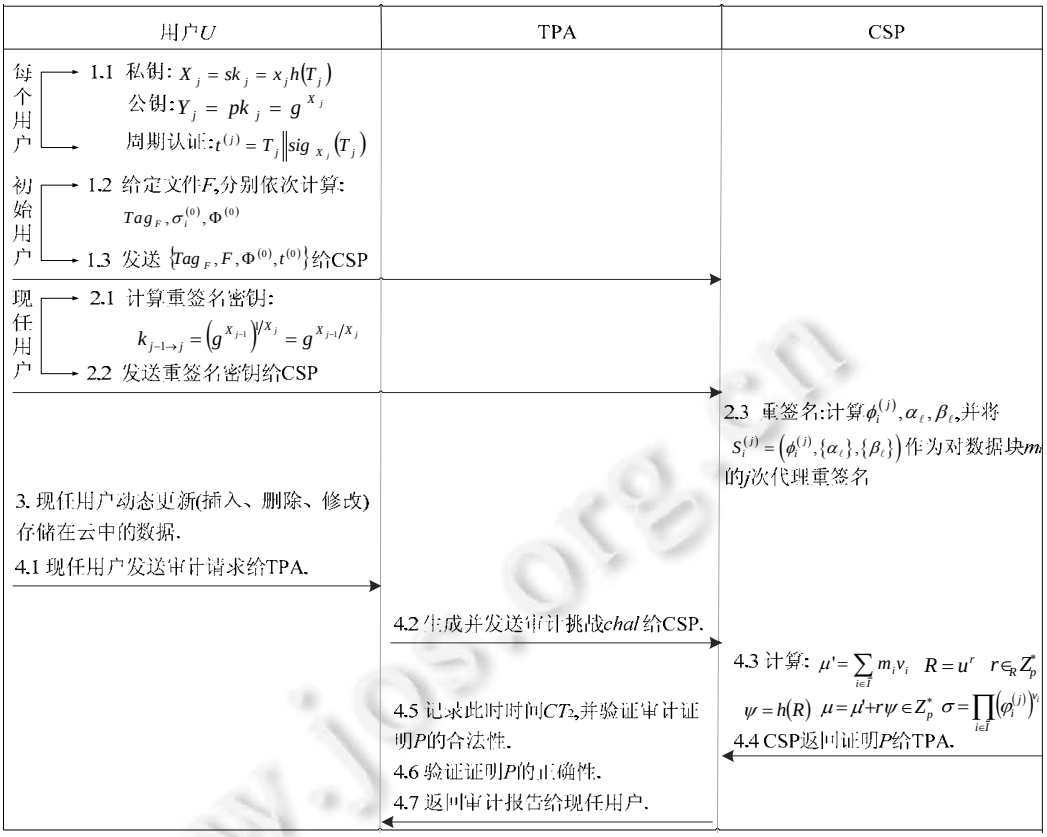


Fig.2 Algorithm flowchart

图 2 算法流程图

3.2 数据动态更新

为了建立更加完善的云存储服务系统体制,本文重点考虑了如何高效地实现云中数据的动态更新.

在传统方法中,数据块的认证标签中包含了它的真实索引,在此情况下,云端数据的动态更新效率不高.如图 3 所示,如果插入图 3(a)或删除图 3(b)一个数据块,则该数据块之后所有数据块的索引都会发生变化,即使这些数据块的内容并没有改变,用户依然必须对改变索引的数据块重新计算其认证标签.

序号	数据块	认证元
1	f_1	$\sigma_1^{(l)}$
2	f_2	$\sigma_2^{(l)}$
…	…	…
i	f_i	$\sigma_i^{(l)}$
i+1	f_{i+1}	$\sigma_{i+1}^{(l)}$
…	…	…
η	f_η	$\sigma_\eta^{(l)}$

插入

序号	数据块	认证元
1	f_1	$\sigma_1^{(l)}$
2	f_2	$\sigma_2^{(l)}$
…	…	…
i	f_i	$\sigma_i^{(l)}$
i+1	f_i'	$\sigma_i'^{(l)}$
i+2	f_{i+1}	$\sigma_{i+1}^{(l)}$
…	…	…
η	f_η	$\sigma_\eta^{(l)}$

(a) 插入数据块 m_i' 后,其之后数据块的身份识别码均会被改变

序号

序号	数据块	认证元
1	f_1	$\sigma_1^{(l)}$
2	f_2	$\sigma_2^{(l)}$
…	…	…
i–1	f_{i-1}	$\sigma_{i-1}^{(l)}$
i	f_i	$\sigma_i^{(l)}$
i+1	f_{i+1}	$\sigma_{i+1}^{(l)}$
…	…	…
η	f_η	$\sigma_\eta^{(l)}$

删除

序号	数据块	认证元
1	f_1	$\sigma_1^{(l)}$
2	f_2	$\sigma_2^{(l)}$
…	…	…
i–1	f_{i-1}	$\sigma_{i-1}^{(l)}$
i	f_{i+1}	$\tilde{\sigma}_{i+1}^{(l)}$
…	…	…
η–1	f_η	$\tilde{\sigma}_\eta^{(l)}$

(b) 删除数据块 m_i 后,其之后数据块的身份识别码均会被改变

Fig.3 Recalculations of block identifiers caused by data additions and deletions in traditional methods

图 3 传统方法中数据增加和删除引起数据块身份识别码的重新计算

为了减轻系统的通信开销、计算开销和用户的负担,本文引入了虚拟索引,它能够确保所有的数据块是按正确的顺序进行排序的,例如,如果$\eta_i<\eta_j$,则数据块 m_j 排在数据块 m_i 之后.定义数据块 m_i 最初的虚拟索引为$\eta_i=i\cdot\rho$,其中,$\rho=2^{\delta}(\delta\in N^*)$表示步长.由于$\delta$一旦选定,则相邻两个数据块之间至多可插入 $2^{\delta}-1$ 个数据块.故在实际中,通常根据存储数据文件的类型与内容、用户的数量选取合适的δ.以下为虚拟索引的计算方式.

- 如果一个新的数据块 m_i' 被插入(介于数据块 m_i 和 m_{i+1} 之间),则它的虚拟索引的计算方式为

$$\eta_i'=(\eta_i+\eta_{i+1})/2;$$

- 如果一个数据块 m_i 被删除,则直接将其虚拟索引一并删除,其余数据块的虚拟索引保持不变;
- 如果一个数据块 m_i 被修改为 m_i',则 m_i' 的虚拟索引依然为原始数据块 m_i 的虚拟索引.

具体动态更新过程如下.

DynUpdate:现任用户 U_j 可以对已存储云服务器上的数据进行动态的更新(插入、删除、修改).

(1) 插入:现任用户 U_j 在数据块 m_i 和 m_{i+1} 之间插入一个新的数据块 m_i'(如图 4 所示).

序号	数据块	虚拟索引	认证元
1	f_1	1ρ	$\sigma_1^{(l)}$
2	f_2	2ρ	$\sigma_2^{(l)}$
…	…	…	…
i	f_i	$i\cdot\rho$	$\sigma_i^{(l)}$
i+1	f_{i+1}	$(i+1)\cdot\rho$	$\sigma_{i+1}^{(l)}$
…	…	…	…
η	f_η	$\eta\cdot\rho$	$\sigma_\eta^{(l)}$

插入

序号	数据块	虚拟索引	认证元
1	f_1	1ρ	$\sigma_1^{(l)}$
2	f_2	2ρ	$\sigma_2^{(l)}$
…	…	…	…
i	f_i	$i\cdot\rho$	$\sigma_i^{(l)}$
i+1	…	$(i\cdot\rho+(i+1)\cdot\rho)/2$	$\sigma_i'^{(l)}$
i+2	f_{i+1}	$(i+1)\cdot\rho$	$\sigma_{i+1}^{(l)}$
…	…	…	…
η+1	f_η	$\eta\cdot\rho$	$\sigma_\eta^{(l)}$

Fig.4 Inserting data block m_i' under a virtual index

图 4 在虚拟索引下插入数据块 m_i'

① 首先,U_j 计算新数据块 m_i' 的虚拟索引 $\eta_i'=(\eta_i+\eta_{i+1})/2$;其次,计算新数据块 m_i' 的认证标签:

$$\sigma_i'^{(j)}=(H_1(id')u^{m_i'})^{X_j},\text{其中},id_i'=\{name\|\eta_i'\|\xi_i'\},\xi_i'=H_2(m_i'\|\eta_i').$$

② U_j 发送插入请求和 m_i' 的信息 $Tag_F,id_i',m_i',\sigma_i'^{(j)},t^{(j)}$ 给 CSP,并删除本地记录.U_j 及时更新文件 F 的数据块的块索引集合,记为 $\tilde{I}$.

③ CSP 接收到验证信息后,根据新数据块 m_i' 的身份识别码 id_i' 找到 m_i' 的存储位置,存储新插入的数据块 m_i' 及其认证标签 $\sigma_i'^{(j)}$.

(2) 删除:现任用户 U_j 删除数据块 m_i(如图 5 所示).

① U_j 发送删除请求和数据块 m_i 的身份信息 $Tag_F,id_i,t^{(j)}$给 CSP,U_j 及时更新文件 F 的数据块的块索引集合,记为 $\tilde{I}=\{1,2,...,\tilde{n}\}$.

② CSP 根据数据块 m_i 的身份识别码 id_i,寻找数据块 m_i 及其认证标签 $\sigma_i^{(j)}$ 一并删除.

(3) 修改:现任用户 U_j 将数据块 m_i 修改为数据块 m_i'(如图 6 所示).

① U_j 发送修改请求和 m_i 的身份信息 $Tag_F,id_i,t^{(j)}$给 CSP,CSP 接收到信息后,根据其身份识别码 id_i 寻找到数据块 m_i,将其返回给 U_j.

② U_j 对数据块 m_i 进行合理的修改,修改后的数据块记为 m_i'.计算数据块 m_i' 的认证签:

$$\sigma_i'^{(j)}=(H_1(id_i')u^{m_i'})^{X_j},\text{其中},id_i'=\{name\|\eta_i\|\xi_i'\},\xi_i'=H_2(m_i'\|\eta_i).$$

③ U_j 发送数据块 m_i' 的验证信息 $\{Tag_F,id_i',m_i',\sigma_i'^{(j)},t^{(j)}\}$ 给 CSP,U_j 及时更新文件 F 的数据块的块索引集合,记为 $\tilde{I}=\{1,2,...,\tilde{n}\}$ (此时 $\tilde{n}=n$),CSP 根据数据块 m_i 的身份识别码 id_i 寻找到 m_i,将数据块 m_i' 及其认证标签 $\sigma_i'^{(j)}$ 覆盖原数据块 m_i 及其认证标签 $\sigma_i^{(j)}$.

序号	数据块	虚拟索引	认证元
1	f_1	1ρ	$\sigma_1^{(l)}$
2	f_2	2ρ	$\sigma_2^{(l)}$
…	…	…	…
i−1	f_{i-1}	$(i-1)\cdot\rho$	$\sigma_{i-1}^{(l)}$
i	f_i	$i\cdot\rho$	$\sigma_i^{(l)}$
i+1	f_{i+1}	$(i+1)\cdot\rho$	$\sigma_{i+1}^{(l)}$
…	…	…	…
η	f_η	$\eta\cdot\rho$	$\sigma_\eta^{(l)}$

删除

序号	数据块	虚拟索引	认证元
1	f_1	1ρ	$\sigma_1^{(l)}$
2	f_2	2ρ	$\sigma_2^{(l)}$
…	…	…	…
i−1	f_{i-1}	$(i-1)\cdot\rho$	$\sigma_{i-1}^{(l)}$
i	f_{i+1}	$(i+1)\cdot\rho$	$\sigma_{i+1}^{(l)}$
…	…	…	…
η−1	f_η	$\eta\cdot\rho$	$\sigma_\eta^{(l)}$

Fig.5 Deleting data block m_i under a virtual index

图 5 在虚拟索引下删除数据块 m_i

序号	数据块	虚拟索引	认证元
1	f_1	1ρ	$\sigma_1^{(l)}$
2	f_2	2ρ	$\sigma_2^{(l)}$
…	…	…	…
i−1	f_{i-1}	$(i-1)\cdot\rho$	$\sigma_{i-1}^{(l)}$
i	f_i	$i\cdot\rho$	$\sigma_i^{(l)}$
i+1	f_{i+1}	$(i+1)\cdot\rho$	$\sigma_{i+1}^{(l)}$
…	…	…	…
η	f_η	$\eta\cdot\rho$	$\sigma_\eta^{(l)}$

修改

序号	数据块	虚拟索引	认证元
1	f_1	1ρ	$\sigma_1^{(l)}$
2	f_2	2ρ	$\sigma_2^{(l)}$
…	…	…	…
i−1	f_{i-1}	$(i-1)\cdot\rho$	$\sigma_{i-1}^{(l)}$
i	…	$i\cdot\rho$	$\sigma_i'^{(l)}$
i+1	f_{i+1}	$(i+1)\cdot\rho$	$\sigma_{i+1}^{(l)}$
…	…	…	…
η	f_η	$\eta\cdot\rho$	$\sigma_\eta^{(l)}$

Fig.6 Modifying data block m_i under a virtual index

图 6 在虚拟索引下修改数据块 m_i

4 安全性分析

4.1 正确性分析

证明:方案的正确性分析如下:

$$
\begin{aligned}
e(\sigma,g) &= e\Big(\prod_{i\in\bar{I}}\sigma_i^{(j)v_i\tilde{\gamma}_j},g\Big)\\
&= e\Big(\prod_{i\in\bar{I}}(H_1(id_i)u^{m_i})^{X_jv_i\tilde{\gamma}_j},g\Big)\\
&= e\Big(\prod_{i\in\bar{I}}(H_1(id_i)u^{m_i})^{v_i},g^{X_j\tilde{\gamma}_j}\Big)\\
&= e\Big(\prod_{i\in\bar{I}}H_1(id_i)^{v_i}\cdot\prod_{i\in\bar{I}}u^{m_iv_i},g^{x_jh(T_j)\tilde{\gamma}_j}\Big)\\
&= e\Big(\prod_{i\in\bar{I}}H_1(id_i)^{v_i}\cdot u^{\sum_{i\in\bar{I}}m_iv_i},g^{x_jh(T_j)h(T_0)(x_0/x_1)h(T_1)(x_1/x_2)\ldots h(T_{j-1})(x_{j-1}/x_j)}\Big)\\
&= e\Big(\prod_{i\in\bar{I}}H_1(id_i)^{v_i}\cdot u^{\mu'},g^{x_0h(T_0)h(T_1)\ldots h(T_{j-1})h(T_j)}\Big)\\
&= e\Big(u^{\mu-r\psi}\cdot\prod_{i\in\bar{I}}H_1(id_i)^{v_i},\alpha_1\Big)\\
&= e\Big(u^{\mu}\cdot R^{-\psi}\cdot\prod_{i\in\bar{I}}H_1(id_i)^{v_i},\alpha_1\Big).
\end{aligned}
$$

对所有的$\ell\in\{1,2,\ldots,j\}$,有:

$$
e(\alpha_\ell,g)=e(Y_j^{\tau_\ell\tau_{\ell+1}\ldots\tau_j},g)=e(Y_j^{\tau_{\ell+1}\ldots\tau_j},g^{\tau_\ell})=e(\alpha_{\ell+1},\beta_\ell),
$$

$$
e(\alpha_j,g)=e(Y_j^{\tau_j},g)=e(Y_j,g^{\tau_j})=e(Y_j,\beta_j).
$$

综上所述,本文方案是正确的. □

4.2 代理重签名安全

定理 1. 在随机预言机模型中,若敌手 A_1 至多经过 q_H 次哈希问询、q_S 次签名问询,最后在时间 t 内,能够以一个不可忽略的概率优势ε在游戏 1 中获胜,那么算法 C_1 就能以概率 $\varepsilon' \geqslant \frac{\varepsilon}{e(q_s+1)}$ 在时间 $t' \leqslant t+q_H cG_1+q_S cG_1$ 内解决 L-FlexDH 困难性问题,其中,cG_1 表示在 G_1 中求幂运算所花时间.

证明:首先,我们进行内部安全的证明,即保护用户抵抗不诚实的代理.

输入$(g,A=g^a,B=g^b)$,算法 C_1 的模拟的游戏如下.

游戏 1:系统参数:算法 C_1 选择系统参数$\{G_1,G_2,p,g,e,H_1,h,u\}$并发送给 A_1.

公钥生成:当A_1 询问用户 $j,j\in\{1,2,\ldots,N\}$的公钥时,C_1回应一个新生成的公钥 $Y_j = A^{X_j} = g^{aX_j}$,其中,$X_j=x_j h(T_j)$, $x_j \in_R Z_p^*$.实际上,用户 j 的私钥为 aX_j.对于每一对(i,j),重签名密钥 $k_{i\to j} = g^{aX_i/aX_j}$.

询问阶段:在此阶段敌手A_1适应性地进行以下预言机询问,挑战者 C_1 在此过程中维护 H-list,列表初始为空.

H_1 问询:当 A_1 输入(id_i,m_i)进行问询时,C_1 维护列表 H-list-$(id_i,m_i,H^{(i)},\omega_i,c)$:

1. 如果(id_i,m_i)出现在列表中,则 C_1 返回 $H^{(i)}$;
2. 如果(id_i,m_i)不在列表中,则 C_1 生成一个随机比特值 c,使得 $\Pr(c=0)=\zeta$,并且选取 $\omega_i \in_R Z_p^*$:

(1) 如果 c=0,则 C_1 计算 $H^{(i)} = B^{\omega_i} = g^{b\omega_i}$;

(2) 如果 c=1,则 C_1 计算 $H^{(i)} = g^{\omega_i}$.

C_1 将新的数组$(id_i,m_i,H^{(i)},\omega_i,c)$添加至列表 H-list,并返回 $H^{(i)}$作为随机预言机询问的输出.

签名问询:当签名者 j 关于消息(id_i,m_i)的签名被询问时,C_1 运行列表 H-list.

1. 如果 c=0,则 C_1 输出"⊥";
2. 如果 c=1,则 C_1 返回 $H^{(i)X_j a} = g^{\omega_i aX_j} = A^{\omega_i X_j}$ 作为对消息(id_i,m_i)的合法签名.

伪造阶段:经过至多 q_H 次哈希询问、q_S 次签名询问后,A_1 提出了一个从来没有被任何签名者签过名的消息(id^*,m^*),签名者索引 $i^*\in\{1,2,\ldots,N\}$的一个 L^{th} 重伪造签名 $\sigma^{*(L)} = (\sigma_0^*,\sigma_1^*,\ldots,\sigma_{2L}^*) \in G^{2L+1}$,此时,$C_1$ 运行列表 H-list,如果 c=1,则 C_1 失败;如果 c=0,则签名可以写成如下形式:

$$\sigma^{*(L)} = (\sigma_0^*,\sigma_1^*,\ldots,\sigma_{2L}^*) = \left(B^{\omega^* X_i^* a t_1 t_2 \ldots t_L}, A^{t_1 t_2 \ldots t_L}, \ldots, A^{t_1}, g^{t_1}, g^{t_2}, \ldots, g^{t_L}\right).$$

这为挑战者 C_1 提供了一个有效的数组:

$$(C_1,C_2,\ldots,C_L,D_1^a,D_2^a,\ldots,D_L^a,D_L^{ab}),$$

其中, $D_L^{ab} = \sigma_0^{*(1/\omega^* X_i^*)}$, $\log_g(D_j) = \prod_{i=1}^{j}\log_g(C_i)$, $j\in\{1,2,\ldots,L\}$.

此时,C_1 成功的概率为 $\varepsilon' = \varepsilon\cdot(1-\zeta)^{q_S}\zeta$,当ζ=1/q_S+1 时, ε' 达到最大值:

$$\varepsilon'_{\max} = \varepsilon\cdot\left(\frac{q_S}{q_S+1}\right)^{q_S}\cdot\frac{1}{q_S+1}.$$

当 q_S 足够大时, $\left(\frac{q_S}{q_S+1}\right)^{q_S}$ 近似于 e^{-1}.所以,C_1 成功的概率 $\varepsilon' \geqslant \frac{\varepsilon}{e(q_s+1)}$.

接下来,我们计算 C_1 总共需要花费的时间.其中包括:

(1) A_1 问询且 C_1 做出回答所需的时间:$q_H cG_1+q_S cG_1$,其中,cG_1 表示在 G_1 中求幂运算所花时间;

(2) A_1 伪造签名的时间 t.

所以,C_1 所需花费的时间为 $t' \leqslant t+q_H cG_1+q_S cG_1$. □

下面进行外部安全的证明,即证明本方案能抵抗系统外部的敌手.

定理 2. 在随机预言机模型中,如果敌手 A_2 至多经过 q_H 次哈希问询、q_S 次签名问询、q_{RS} 次重签名问询,最后在时间 t 内,以一个不可忽略的概率优势ε在游戏 2 中获胜,那么算法 C_2 将调用 A_2 为子程序,在时间 $t' \leqslant$

$t+q_H cG_1+q_S cG_1+q_{RS} cRS$ 内以概率$\varepsilon' \geqslant \varepsilon/Nq_H$解决 L-FlexDH 问题.

证明:游戏 2:给定 C_2一个 L-FlexDH 实例,参数$(g,A=g^a,B=g^b)$,C_2将先建立系统,并将系统参数$\{G_1,G_2,p,g,e,H_1,h,u\}$发送给 A_2.

公钥生成:选取一个目标 $i^* \in_R \{1,2,\ldots,N\}$,目标公钥为 $Y_{i^*}=A=g^a$,其他签名者的 $Y_i=g^{X_i}$.对于每一对(i,j),重签名密钥 $k_{i\to j}=g^{X_i/X_j}$.

询问阶段:在此阶段,敌手 A_2适应性地进行以下预言机询问,挑战者 C_2在此过程中维护列表 H-list,列表初始为空.

H_1 问询:当 A_2 输入(id_i,m_i)进行问询时,C_2 维护列表 H-list-$(id_i,m_i,H^{(i)},y_i)$.

1. 如果(id_i,m_i)出现在列表中,则 C_2 返回相应的 $H^{(i)}$.
2. 如果(id_i,m_i)不在列表中,则 C_2 猜测(id_i,m_i)是否为 A_2 将要伪造的目标数据块(id^*,m^*):
   (1) 如果 $id_i=id^*$且 $m_i=m^*$,则 C_2 计算 $H^{(i)}=g^b$;
   (2) 如果 $id_i\neq id^*$或 $m_i\neq m^*$,C_2 计算 $H^{(i)}=g^{y_i}, y_i \in_R Z_p^*$.

C_2 将新的数组$(id_i,m_i,H^{(i)},y_i)$添加至列表 H-list,并返回 $H^{(i)}$作为随机预言机询问的输出.

签名问询:当 A_2 询问关于签名者 j 对消息(id_i,m_i)的签名时,C_2 运行列表 H-list.

1. 若 $j=i^*$:若 $id_i=id^*$且 $m_i=m^*$,则 C_2 输出"⊥";若 $id_i\neq id^*$或 $m_i\neq m^*$,则 C_2 输出 $\sigma^{(j)}=H^{(i)a}=g^{y_i a}$.
2. 若 $j\neq i^*$:若 $id_i=id^*$且 $m_i=m^*$,则 C_2 输出 $\sigma^{(j)}=H^{(i)X_j}=g^{bX_j}$;若 $id_i\neq id^*$或 $m_i\neq m^*$,则 C_2 输出 $\sigma^{(j)}=H^{(i)X_j}=g^{y_i X_j}$.

重签名问询:当 A_2 询问关于合法签名 $\sigma^{\ell-th}$ 从 i 到 j 的重签名时,C_2 生成签名者 j 的第 1 层重签名 $\sigma^{(\ell+1)-th}$,其中,$\ell \in \{1,2,\ldots,L\}$.

伪造阶段:经过至多 q_H 次哈希询问、q_S 次签名询问、q_{RS} 次重签名询问后,A_2 提出了一个从来没有被任何签名者签过名的消息(id^*,m^*),签名者索引 $i^* \in \{1,2,\ldots,N\}$的一个 L^{th} 层伪造重签名 $\sigma'^{(L)}=(\sigma'_0,\sigma'_1,\ldots,\sigma'_{2L}) \in G^{2L+1}$,重签名可以表示成如下形式:

$$\sigma'^{(L)}=(\sigma'_0,\sigma'_1,\ldots,\sigma'_{2L})=(B^{a\tilde{\gamma}_1\tilde{\gamma}_2\cdots\tilde{\gamma}_L},A^{\tilde{\gamma}_1\tilde{\gamma}_2\cdots\tilde{\gamma}_L},A^{\tilde{\gamma}_1\tilde{\gamma}_2\cdots\tilde{\gamma}_{L-1}},\ldots,A^{\tilde{\gamma}_1},g^{\tilde{\gamma}_1},g^{\tilde{\gamma}_2},\ldots,g^{\tilde{\gamma}_L}).$$

此时,C_2 能够成功地提取到$(2L+1)$重数组(与内部安全分析中 C_1 相同).在游戏 2 中,C_2 猜对目标公钥的概率为 $1/N$,猜对伪造数据块的概率为 $1/q_H$,因此,如果 A_2 能够以一个不可忽略的概率ε成功伪造一个重签名,那么 C_2 就有概率$\varepsilon' \geqslant \varepsilon/Nq_H$解决 L-FlexDH 问题.

接下来,我们计算 C_2 总共需要花费的时间,其中包括:

(1) A_2 问询且 C_2 作出回答所需的时间:$q_H cG_1+q_S cG_1+q_{RS} cRS$,其中,$cG_1$ 表示在 G_1 中求幂运算所花时间,cRS 表示 C_2 计算重签名的时间;

(2) A_2 成功伪造重签名的时间 t.

所以,C_2 所需花费的时间为 $t' \leqslant t+q_H cG_1+q_S cG_1+q_{RS} cRS$. □

综合定理 1 和定理 2,无论是内部敌手还是外部敌手,只要其能够伪造一个合法签名,就存在一个算法 C 能够以不可忽略的概率解决 L-FlexDH 问题,而 L-FlexDH 问题在当前假设是计算困难性问题.所以,伪造本文方案的一个合法签名是不可能的.

4.3 审计安全

定理 3. 如果在数据不正确的情况下(如数据被 CSP 私自篡改或被外部攻击损坏),CSP 生成一个伪造的审计证明 $\tilde{P}$,并能赢得游戏 3,那么在 G_1 中解决 DL 困难性问题的概率是 $1-1/p$(p 是一个大素数).

证明:游戏 3 定义如下.

游戏 3:TPA 发送挑战请求$chal=\{i,v_i\}_{i\in I}$给 CSP,在正确数据 F 下的审计证明 $P=\{\sigma,\mu,R,\{\alpha_\ell\}_{\ell\in[1,j]},\{\beta_\ell\}_{\ell\in[1,j]}\}$,并且 P 能通过 TPA 的验证.

然而,CSP 在不正确数据 $\tilde{F}$ 的情况下,生成一个伪造的审计证明:

$$\tilde{P}=\{\sigma,\tilde{\mu},R,\{\alpha_\ell\}_{\ell\in[1,j]},\{\beta_\ell\}_{\ell\in[1,j]}\},$$

其中, $\tilde{\mu}=\tilde{\mu}'+r\psi\in Z_p^*,\tilde{\mu}'=\sum_{i\in\bar{I}}\tilde{m}_i v_i$. 定义:

$$\Delta\mu=\tilde{\mu}-\mu=\tilde{\mu}'+r\psi-(\mu'+r\psi)=\tilde{\mu}'-\mu'=\sum_{i\in\bar{I}}\tilde{m}_i v_i-\sum_{i\in\bar{I}}m_i v_i=\sum_{i\in\bar{I}}(\tilde{m}_i-m_i)v_i=\sum_{i\in\bar{I}}\Delta m_i v_i.$$

因为 $\tilde{F}$ 是不正确的数据,所以数据 $\tilde{F}$ 与数据 F 的内容必然是存在差异的.即至少存在 1 个 $\Delta m_i\neq 0(i\in\bar{I})$, 如果伪造的审计证明 $\tilde{P}$ 能够通过 TPA 的审计,则 CSP 赢得游戏的胜利.

假设:CSP 可以赢得游戏胜利,即伪造的审计证明 $\tilde{P}$ 可以通过 TPA 的验证,即有:

$$e(\tilde{\sigma},g)=e\left(u^{\tilde{\mu}}\cdot R^{-\psi}\cdot\prod_{i\in\bar{I}}H_1(id_i)^{v_i},\alpha_1\right) \tag{1}$$

因为正确的审计证明 P 可以通过 TPA 的审计,所以有:

$$e(\sigma,g)=e\left(u^{\mu}\cdot R^{-\psi}\cdot\prod_{i\in\bar{I}}H_1(id_i)^{v_i},\alpha_1\right) \tag{2}$$

由公式(1)、公式(2)可得:

$$e\left(u^{\tilde{\mu}}\cdot R^{-\psi}\cdot\prod_{i\in\bar{I}}H_1(id_i)^{v_i},\alpha_1\right)=e\left(u^{\mu}\cdot R^{-\psi}\cdot\prod_{i\in\bar{I}}H_1(id_i)^{v_i},\alpha_1\right),\ u^{\tilde{\mu}'}=u^{\mu'},\ u^{\Delta\mu}=1.$$

因为 G_1 是一个循环群,所以对于$\forall a,b\in G_1,\exists x\in Z_p$,使得 $b=a^x$.

不失一般性,给定 a 和 b,则 u 可以写成 $u=a^{y_1}\cdot b^{y_2}\in G_1$ 的形式,其中,$y_1,y_2\in Z_p$.所以有:

$$1=u^{\Delta\mu}=(a^{y_1}b^{y_2})^{\Delta\mu}=a^{y_1\Delta\mu}b^{y_2\Delta\mu}.$$

进一步化简得:

$$b=a^{-y_1\Delta\mu/y_2\Delta\mu} \tag{3}$$

要使得公式(3)不成立,当且仅当分母 $y_2=0$,此时公式(3)没有意义.而 $y_1,y_2\in Z_p$,所以 $P[y_2=0]=1/p$,即公式(3)不成立的概率是 $1/p$,公式(3)成立的概率为 $1-1/p$(p 为大素数).

因此,在数据不正确的情况下,CSP 伪造一个通过 TPA 审计的证明是困难的.即 CSP 生成一个通过 TPA 审计的证明 P,当且仅当 CSP 完整的保存了用户的真实数据. □

4.4 隐私保护

本方案中,隐私保护主要体现在两个方面:(1) 在审计阶段,采用了随机掩饰码技术盲化证据μ',并满足抗已撤销用户与 TPA 合谋攻击,防止云存储中 TPA 在审计时获取用户数据隐私(定理 4);(2) 抗已撤销用户与 CSP 合谋,防止当前用户私钥泄露(定理 5).具体分析如下.

定理 4. 在审计阶段,TPA 不能从 CSP 返回的审计证明 P 中获得用户存储真实数据的任何内容.

证明:在本方案的审计阶段,TPA 可能通过 CSP 返回的证明 P 中的μ或σ提取到证据μ',并从中窃取用户的数据隐私,具体分析如下.

(1) 如果 TPA 想通过 CSP 返回的证明 P 中的μ中提取到证据μ',具体分析如下.

传统方案中, $\mu=\mu'=\sum_{i\in\bar{I}}m_i v_i$ 是用户存储数据的线性组合,TPA 可重复的对相同的数据块进行完整性检验,即每次都发送审计挑战 $chal=\{i,v_i\}_{i\in\bar{I}}$ 给 CSP,经过 c 次挑战后,TPA 就可以得到一个以数据块 m_i 为未知数的线性方程组,通过求解线性方程组的解可以获得用户的隐私.

为了避免该攻击,本方案采用随机掩码技术隐藏证据μ'(本文第 3.1 节已详细介绍其隐藏过程).因为在 G_1 中,DL 问题是计算困难性问题,故由 $u,R\in G_1$ 计算 r 是困难的.所以,TPA 欲通过求解 r 来获得证据μ'是困难的,从而防止了数据泄露.

(2) 如果 TPA 想通过 CSP 返回的证明 P 中的σ中提取到证据μ',σ计算如下.

$$\begin{aligned}\sigma &= \left(\prod_{i\in\tilde{I}}(\phi_i^{(j)})^{v_i}\right)\\ &= \left(\prod_{i\in\tilde{I}}(\sigma_i^{(j)})^{v_i\tilde{\gamma}_j}\right)\\ &= \prod_{i\in\tilde{I}}(H_1(id_i)u^{m_i})^{X_j v_i\tilde{\gamma}_j}\\ &= u^{\mu' X_j\tilde{\gamma}_j}\cdot\prod_{i\in\tilde{I}}H_1(id_i)^{v_i X_j\tilde{\gamma}_j}\\ &= \left(u^{\mu'}\cdot\prod_{i\in\tilde{I}}H_1(id_i)^{v_i}\right)^{X_0 h(T_1)h(T_2)\ldots h(T_j)}.\end{aligned}$$

因为 TPA 无法获取历任用户和现任用户的私钥,所以 TPA 不能从上式中获取 $u^{\mu'}$的信息及证据μ'.

假设已撤销用户,不失一般性,假设 U_0 与 TPA 合谋,把其私钥 X_0 发送给 TPA,TPA 仍无法从中提取到证据μ'的信息(其他已撤销用户与 TPA 合谋同样成立).给定 $u^{\mu'}=\varphi\in G,u\in G$,基于 DL 困难性,已知 u 和φ,求解μ'是困难的.

综上所述,TPA 从 CSP 返回的证明 P 中是无法提取到证据μ'.即本文方案在 TPA 审计阶段可以很好地保护用户的数据隐私. □

定理 5. 本方案可抵抗已撤销用户与 CSP 合谋攻击,防止当前用户的私钥被 CSP 窃取.

证明:本方案的初始阶段,定义每一位用户 U_j 的私钥 $X_j=x_j h(T_j), x_j\in_R Z_p^*$,公钥为 $Y_j=g^{X_j}$;在代理重签名阶段,重签名密钥是由上一任用户的公钥与新上任用户私钥计算按此方式 $k_{j-1\to j}=(Y_{j-1})^{1/X_j}=g^{X_{j-1}/X_j}$ 计算.CSP 接收到新任用户发送的重签名密钥后,不能从中获取新任用户的私钥 X_j.这是因为 DL 是计算困难性问题,给定 $k_{j-1\to j}\in G_1$ 和 $g\in G_1$,求解 X_{j-1}/X_j 是计算困难的.即使已撤销用户 U_{j-1} 与 CSP 合谋,将其私钥 X_{j-1} 发送给 CSP,CSP 仍然无法获得新任用户的私钥 X_j.所以本方案中重签名密钥的构造方式,可抵抗已撤销用户与 CSP 合谋,很好地解决了用户私钥泄露问题. □

5 性能分析

本节先讨论本文方案的通信开销、计算开销,然后与文献[15,17−21]在通信开销、计算开销及性能方面进行了比较分析.因为在类似方案[11−21]中,文献[15−21]所给方案在用户可撤销阶段都使用了代理重签名技术,是此类问题较优的解决方案,其中,文献[16]是在文献[15]的基础上做了进一步的优化,但并未对方案进行实质性的修改,因此选取文献[15,17−21]与本文方案进行了详细的分析比较.

表 1 为通信开销与计算开销分析中需使用的符号及含义.

Table 1 Notations and their meanings

表 1 符号及其含义

符号	含义	符号	含义
$Pair$	双线性对运算	$ExpG_1$	G_1 上的指数运算
$HashG_1$	$H:\{0,1\}^*\to G_1$	$\|v\|$	$v_i\in Z_p^*$ 的比特长度,$\|v\|<\|p\|$
$\|p\|$	Z_p^* 中元素的比特长度	$\|G_1\|$	G_1 中元素的比特长度
$\|n\|$	$\tilde{I}=\{1,2,\ldots,\tilde{n}\}$ 中元素的比特长度	d	已撤销用户人数
n	数据块的总数	c	审计挑战中数据块的个数

从实验结果看,本文方案在通信开销及计算开销方面具有一定的优势,在性能方面具有一定的实用价值.

5.1 通信开销

本方案中,通信开销主要集中在发送重签名密钥与审计阶段(发送审计挑战和返回审计证明).当前用户 U_j 发送重签名密钥 $k_{j-1\to j}$ 给 CSP,通信量为$|G_1|$,TPA 发送审计挑战 $chal$ 到 CSP,通信量为 $c(|n|+|v|)$.CSP 返回审计证明 P 给 TPA 的通信量为 $2|G_1|+|p|+2d|G_1|$.

所以,本文方案的通信开销为$(3+2d)|G_1|+c(|n|+|v|)+|p|$.

5.2 计算开销

本方案中,计算开销主要集中在重签名阶段和审计阶段.

- 重签名阶段:计算重签名密钥产生的计算量为 $1ExpG_1$,计算新签名产生计算量为$(3ExpG_1+2dPair)n$;
- 审计阶段:CSP 计算证明 P 产生的计算量为$(1+c)ExpG_1$,TPA 审计时产生的计算量为 $2Pair$.

本文方案的计算开销为$(3n+c+2)ExpG_1+2(nd+1)Pair$.

5.3 与其他方案的比较

本方案与文献[15,17–21]中的方案在通信、计算开销及性能方面进行了分析比较.为使比较结果更直观,采用常规数据长度$|n|$=32bit,$|p|$=160bit,$|G_1|$=160bit,$|v|$=160bit.为使计算开销比较更简洁,本文主要考虑对运算 $Pair$,G_1 中的幂运算 $ExpG_1$、点乘运算 $MulG_1$ 及哈希运算 $HashG_1$,而忽略一些计算代价极低的运算,如普通加法、乘法和 Hash 到 Z_p^* 上的计算(需毫秒级计算时间).选取了实验参数设定相同的文献[25,26].从文献[25]可知,计算一个对运算花费 1.9s,一个 G_1 中的幂运算花费 0.9s;从文献[26]可知,计算一个点乘运算花费 0.81s.表 2 是本文方案与文献[15,17–21]中方案在通信、计算及存储开销方面的比较,因为 $c<n$,所以本文方案在通信开销及计算开销方面具有一定的优势.

Table 2 Comparisons of communication, computation, and storage cost

表 2 通信、计算及存储开销比较

方案	通信开销(bytes)			计算开销(s)			存储开销
	发送重签名密钥	重签名	云审计	重签名密钥	重签名	云审计	
文献[15]	60	0	$74c$	2.43	$4.7nd$	$2.8c+1.9$	1×
文献[17]	20	0	$24n+60$	0.9	$0.8nd+5.6d$	$0.9n+4.7+HashG_1$	1×
文献[18]	20	0	$80c+60$	0.9	$0.8nd+5.6d$	$0.9c+8.5+HashG_1$	1×
文献[19]	60	0	$40c$	2.43	$4.7nd$	$3.6c+1.9$	1×
文献[20]	60	0	$40c+60$	2.43	$9.4nd$	$3.6c+3.6$	1×
文献[21]	0	$60nd$	$84c$	0	$6.6nd$	$2.8c+1.9$	2×
本文方案	20	0	$24c+60$	0.9	$0.8nd+5.6d$	$0.9c+4.7$	1×

表 3 是本文方案与文献[15,17–21]中方案在安全性能方面的比较,从中可看出,本文方案具有一定的实用价值和应用前景.

Table 3 Comparison of performance

表 3 性能比较

方案	是否抵抗已撤销用户与 CSP 合谋	是否抵抗已撤销用户与 TPA 合谋	是否支持已撤销用户的可追踪性	是否支持用户动态可撤销	是否支持数据隐私保护	是否支持数据实时动态更新	是否支持验证审计证明的合法性
文献[15]	×	×	×	√	×	√	×
文献[17]	√	×	√	×	√	×	×
文献[18]	√	×	×	×	√	√	×
文献[19]	√	×	×	√	×	×	×
文献[20]	×	×	×	√	×	×	×
文献[21]	√	×	×	√	√	√	×
本文方案	√	√	√	√	√	√	√

5.4 实验结果

本文方案所有的实验是在 Linux 操作系统,CPU 为 3.6GHz,RAM 为 8GB 的环境中运行的,所使用的网络为学校校园网,下载和上传的速度均为 1.2Mb/s,主要采用的编程语言是 C 语言,利用 PBC(pairing-based cryptography)程序库.G_1, Z_p^* 中的元素比特长度依次为$|G_1|$=160bit,$|p|$=160bit,文件 F 的标签的比特长度为$|Tag_F|$=32bit,每位用户的任期认证标签的比特长度为$|t^{(j)}|$=32bit,每个数据块的身份识别码的比特长度为$|id_i|$=80bit,用户撤销后需重签名的数据块的数量记为 k.

5.4.1 用户可动态撤销性能分析

本方案的主要特点之一是用户可动态高效的撤销.在传统方案中,旧用户撤销后,新用户须从云端重新下载被已撤销用户签过名的数据块,并需对全部数据块进行完整性的验证,在每个数据块都正确的情况下,新用户利用自己的私钥对该数据块重新签名并生产相应的认证信息,然后重新上传数据及认证信息至 CSP 进行存储.

传统的撤销方法使得系统的通信开销和计算开销非常大.

- 当需要签名的数据块数为 10 000 时,本文方案产生的通信开销为 390.40KB,传统方案产生的通信开销为 1212.20KB,约为本文方案所产生通信开销的 3.1 倍(如图 7 所示);
- 当需要签名的数据块数为 10 000 时,本文方案产生的计算开销为 9.00s,传统方案产生的计算开销为 13.60s,约为本文方案的 1.51 倍(如图 8 所示);
- 当需要签名的数据块数为 10 000 时,本文方案用户可撤销阶段的耗时为 9.20s,传统方案耗时为 14.6s,约为本文方案的 1.59 倍(如图 9 所示).

由此可知,本方案在用户可动态撤销方面具有明显的优势.

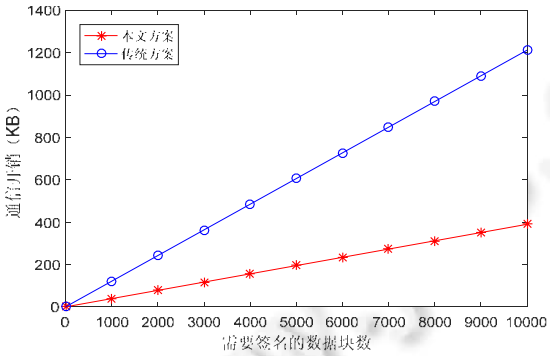


Fig.7 Comparison of user revocation communication cost between our scheme and traditional schemes

图 7 本文方案与传统方案用户撤销通信开销比较

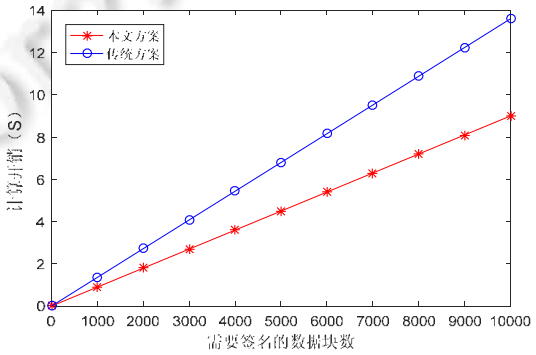


Fig.8 Comparison of user revocation computation cost between our scheme and traditional schemes

图 8 本文方案与传统方案用户撤销计算开销比较

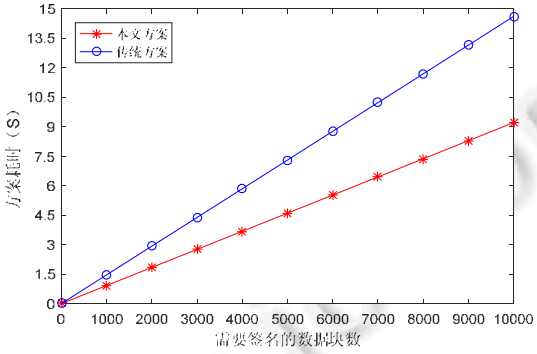


Fig.9 Comparison of user revocation time

图 9 本文方案与传统方案用户撤销时间比较

5.4.2 审计性能分析

由文献[27]可知,在挑战生成、证明生成与验证证明的计算开销和通信开销均与挑战请求中的 c 成线性关系,c 越大,数据的正确性越高.但 c 的增大必然引起云存储系统更大的计算开销与通信开销,所以 c 的选取需要权衡验证准确性与计算通信开销之间的关系.

在本文方案的审计阶段,并不需要 TPA 对所有的数据块进行验证,只是从所有 n 个数据块中随机选取 c 个

数据块进行完整性校验.假设所有数据块中有 t%的数据块已被损坏,随机选取的数据块个数为 c,那么这 c 个数据块中,含有损坏数据块的概率为 $1-(1-t)^c$.当 t=1 时,TPA 仅需要审计 c=300 或 460 个数据块(类似文献常设数据块数),就能依次以高概率 0.950 9 或 0.990 1 发现有损坏的数据块.

如图 10 所示,在审计阶段,分别选取需要挑战数据块数 c=300 和 c=460,随着已撤销用户数的增加,审计时间也在增加.这是因为 TPA 需要验证重签名的合法性.当已撤销用户数为 200 时,选取 300 个挑战数据块所需要的审计时间仅为 0.53s,选取 460 个挑战数据块所需要的审计时间仅为 0.61s.在假定有 1%损坏数据块的情况下,审计过程中选取 c=300 或 460 不仅有很高的概率能够发现有损坏的数据块,而且所需的审计时间是非常短的,审计时间易被大多数用户接受.故审计阶段随机选取 c 个数据块进行数据完整性验证,理论上是可行的.

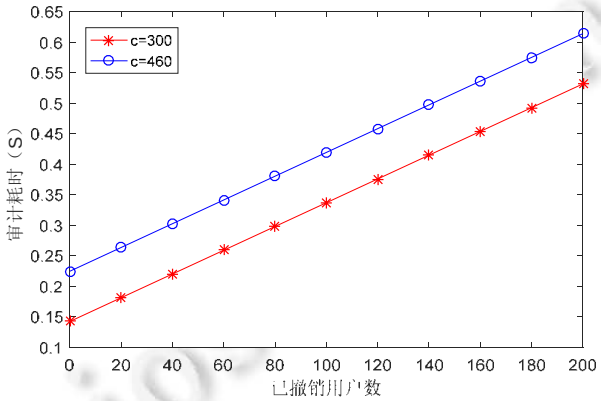


Fig.10 Comparison of auditing time

图 10 审计时间比较

5.4.3 数据动态更新性能分析

根据第 3.2 节,本方案可对存储在云端的数据进行实时的动态更新(插入、删除、修改等操作).为了证明数据动态更新过程并不会对系统造成过多额外的计算开销与通信开销,现定义方案 A:它与本文方案唯一区别是没有数据实时动态更新过程.在相同的操作环境下运行本文方案与方案 A,假设 n=10 000,c=300,在动态更新过程中,随机选取插入 30 个新的数据块、删除 30 个旧的数据块、修改 30 个数据块,假定修改 1 个数据块的内容为 3ms.

- 如图 11 所示,当已撤销用户数为 200 时,本方案所产生的通信开销为 216.562 5KB,方案 A 产生的通信开销为 210.234 4KB,相差 6.328 1KB;

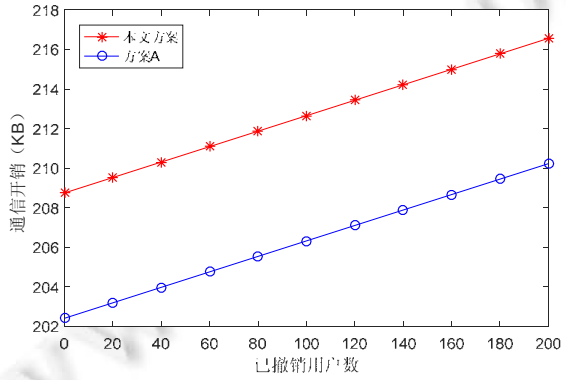


Fig.11 Comparison of communication costs between our scheme and a hypothetical scheme A

图 11 本文方案与假想方案 A 通信开销比较

- 如图 12 所示,当已撤销用户数为 200 时,本方案所产生的计算开销为 69.979 2s,方案 A 产生的计算开销为 69.751 2s,仅相差 0.228s.
- 如图 13 所示,当已撤销用户数为 200 时,运行本文方案所需时间为 70.155 2s,运行方案 A 所需时间为 69.922 3s,仅相差 0.232 9s.

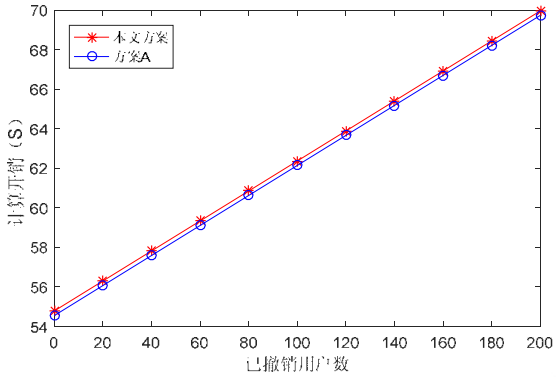


Fig.12 Comparison of computation cost between our scheme and a hypothetical scheme A

图 12 本文方案与假想方案 A 计算开销比较

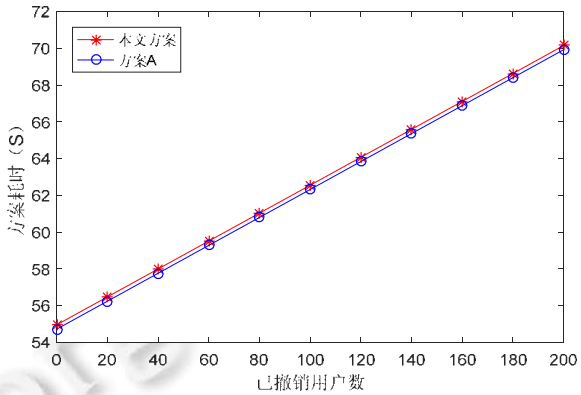


Fig.13 Comparison of running time cost between our scheme and a hypothetical scheme A

图 13 本文方案与假想方案 A 运行时间开销比较

由图 11~图 13 可看出,本文方案在实现数据实时动态更新时产生的通信开销与计算开销及方案整体运行的所需时间是非常小的,在用户可接受的合理范围之内.因此,数据实时动态更新有理论意义和实用价值.

6 结 语

本文提出了一个轻量级的支持用户可动态撤销及存储数据可实时动态更新的云存储审计方案.

- 该方案采用多重单向代理重签名技术,使用户可以高效地动态撤销,并且减轻了云存储系统的通信开销及用户计算开销.
- 在数据实时动态更新方面,在每个数据块的身份识别码中引入了虚拟索引.它在确保所有数据块按正确顺序排序的同时,能够保证数据高效地动态更新.
- 在安全方面,抗已撤销用户与 CSP 的合谋攻击,解决了用户私钥泄露问题.
- 抗已撤销用户与 TPA 的合谋攻击,用户的数据隐私得到了更好的保护,使得本文方案的目标得以高效地实现.

在后续工作中,拟考虑无双线性对的用户动态可撤销的云审计方案.

References:

[1] Li NH, Li TC, Venkatasubramanian S. *t*-closeness: Privacy beyond *k*-anonymity and *l*-diversity. In: Proc. of the IEEE 23rd Int'l Conf. on Data Engineering (ICDE 2007). IEEE, 2007. 106−115. [doi: 10.1109/ICDE.2007.367856]

[2] Machanavajjhala A, Gehrke J, Kifer D, Venkitasubramaniam M. *l*-diversity: Privacy beyond *k*-anonymity. In: Proc. of the IEEE 22nd Int'l Conf. on Data Engineering (ICDE 2006). IEEE, 2006. 34−36. [doi: 10.1109/ICDE.2006.1]

[3] Han J, Li YP, Chen WF. A lightweight and privacy-preserving public cloud auditing scheme without bilinear paring in smart cities. Computer Standards & Interfaces, 2019,62(1):84−97. [doi: 10.1016/j.csi.2018.08.004]

[4] Tan S, Jia Y, Han WH. Research and development of provable data integrity in cloud storage. Chinese Journal of Computers, 2015, 38(1):164−177 (in Chinese with English abstract). [doi: 10.3724/SP.J.1016.2015.00164]

[5] Ateniese G, Burns R, Curtmola R, Herring J, Kissner L, Peterson Z, Song D. Provable data possession at untrusted stores. In: Proc. of the 14th ACM Conf. on Computer and Communications Security (CCS 2007). ACM, 2007. 598−609. [doi: 10.1145/1315245.1315318]

[6] Ristenpart T, Tromer E, Shacham H, Savage S. Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. In: Proc. of the 16th ACM Conf. on Computer and Communications Security (CCS 2009). ACM, 2009. 199−212. [doi: 10.1145/1653662.1653687]

[7] Wang Q, Wang C, Ren K, Lou WJ, Li J. Enabling public verifiability and data dynamics for storage security in cloud computing. IEEE Trans. on Parallel and Distributed Systems, 2011,22(5):847−859. [doi: 10.1109/TPDS.2010.183]

[8] Wang C, Wang Q, Ren K, Lou WJ. Privacy-preserving public auditing for data storage security in cloud computing. In: Proc. of the IEEE INFOCOM 2010. IEEE, 2010. 1−9. [doi: 10.1109/INFCOM.2010.5462173]

[9] Wang C, Chow SSM, Wang Q, Ren K, Lou WJ. Privacy-preserving public auditing for secure cloud storage. IEEE Trans. on Computers, 2013,62(2):362−375. [doi: 10.1109/TC.2011.245]

[10] Zhu Y, Hu HX, Ahn GJ, Yu MY. Cooperative provable data possession for integrity verification in multicloud storage. IEEE Trans. on Parallel and Distributed Systems, 2012,23(12):2231−2244. [doi: 10.1109/TPDS.2012.66]

[11] Wang BY, Li BC, Li H. Knox: Privacy-preserving auditing for shared data with large groups in the cloud. In: Proc. of the Int'l Conf. on Applied Cryptography and Network Security. Springer-Verlag, 2012. 507−525. [doi: 10.1007/978-3-642-31284-7_30]

[12] Wang BY, Li H, Li M. Privacy-preserving public auditing for shared cloud data supporting group dynamics. In: Proc. of the IEEE Int'l Conf. on Communications. IEEE, 2013. 1946−1950. [doi: 10.1109/ICC.2013.6654808]

[13] Wang BY, Li BC, Li H. Oruta: Privacy-preserving public auditing for shared data in the cloud. IEEE Trans. on Cloud Computing, 2014,2(1):43−56. [doi: 10.1109/TCC.2014.2299807]

[14] Yuan JW, Yu SC. Efficient public integrity checking for cloud data sharing with multi-user modification. In: Proc. of the IEEE INFOCOM 2014. IEEE, 2014. 2121−2129. [doi: 10.1109/INFOCOM.2014.6848154]

[15] Wang BY, Li BC, Li H. Public auditing for shared data with efficient user revocation in the cloud. In: Proc. of the IEEE INFOCOM 2013. IEEE, 2013. 2904−2912. [doi: 10.1109/INFCOM.2013.6567101]

[16] Wang BY, Li BC, Li H. Panda: Public auditing for shared data with efficient user revocation in the cloud. IEEE Trans. on Services Computing, 2015,8(1):92−106. [doi: 10.1109/TSC.2013.2295611]

[17] Zhang XP, Xu CX, Zhang XY, Sai W, Han XY, Liu GP. Efficient public auditing scheme for cloud storage supporting user revocability with proxy re-signature scheme. Journal of Computer Applications, 2016,36(7):1816−1821 (in Chinese with English abstract). [doi: 10.11772/j.issn.1001-9081.2016.07.1816]

[18] Yang XD, Liu TT, Yang P, An FY, Yang MM, Xiao LK. Public auditing scheme for cloud data with user revocation and data dynamics. In: Proc. of the IEEE Info. Technology, Networking, Electronic and Automation Control Conf. IEEE, 2017. 813−817. [doi: 10.1109/ITNEC.2017.8284847]

[19] Xu YY, Bai GW, Shen H, Huang ZP. Virtual-user-based public auditing integrity in cloud storage. Computer Science, 2017,44(5):95−99 (in Chinese with English abstract). [doi: 10.11896/j.issn.1002-137X.2017.05.017]

[20] Samundiswary S, Dongre NM. Public auditing for shared data in cloud with safe user revocation. In: Proc. of the Int'l Conf. of Electronics, Communication and Aerospace Technology. 2017. 53−57. [doi: 10.1109/ICECA.2017.8203603]

[21] Liu HQ, Wang BC, Lu K, Gao ZY, Zhan Y. Public auditing for shared data utilizing backups with user revocation in the cloud. Wuhan University Journal of Natural Sciences, 2018,23(2):129−138. [doi: 10.1007/s11859-018-1303-4]

[22] Libert B, Vergnaud D. Multi-use unidirectional proxy re-signatures. In: Proc. of the 15th ACM Conf. on Computer and Communications Security. ACM, 2008. 511−520. [doi: 10.1145/1455770.1455835]

[23] Naor M. On cryptographic assumptions and challenges. In: Boneh D, ed. Proc. of the Advances in Cryptology—CRYPTO 2003. LNCS, Berlin, Heidelberg: Springer-Verlag, 2003. 96−109. [doi: 10.1007/978-3-540-45146-4_6]

[24] Shacham H, Waters B. Compact proofs of retrievability. In: Pieprzyk J, ed. Proc. of the Advances in Cryptology—ASIACRYPT 2008. LNCS, Berlin, Heidelberg: Springer-Verlag, 2008. 90−107. [doi: 10.1007/978-3-540-89255-7_7]

[25] Shim KA, Lee YR, Park CM. EIBAS: An efficient identity-based broadcast authentication scheme in wireless sensor networks. Ad Hoc Networks, 2013,11(1):182−189. [doi: 10.1016/j.adhoc.2012.04.015]

[26] Gura N, Patel A, Wander A, Eberle H, Shantz SC. Comparing elliptic curve cryptography and RSA on 8-bit CPUs. In: Joye M, Quisquater JJ, eds. Proc. of the Cryptographic Hardware and Embedded Systems—CHES 2004. LNCS, Berlin, Heidelberg: Springer-Verlag, 2004. 119−132. [doi: 10.1007/978-3-540-28632-5_9]

[27] Shen WT, Yu J, Xia H, Zhang HL, Lu XQ, Hao R. Light-weight and privacy-preserving secure cloud auditing scheme for group users via the third party medium. Journal of Network and Computer Applications, 2017,82:56−64. [doi: 10.1016/j.jnca.2017.01.015]

附中文参考文献:

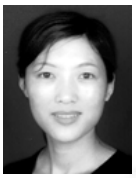
[4] 谭霜,贾焰,韩伟红.云存储中的数据完整性证明研究及进展.计算机学报,2015,38(1):164−177. [doi: 10.3724/SP.J.1016.2015.00164]

[17] 张新鹏,许春香,张新颜,赛伟,韩兴阳,刘国平.基于代理重签名的支持用户可撤销的云存储数据公共审计方案.计算机应用,2016,36(7):1816−1821. [doi: 10.11772/j.issn.1001-9081.2016.07.1816]

[19] 徐云云,白光伟,沈航,黄中平.云存储中基于虚拟用户的数据完整性验证.计算机科学,2017,44(5):95−99. [doi: 10.11896/j.issn.1002-137X.2017.05.017]

韩静(1992－),女,山西吕梁人,硕士生,主要研究领域为云存储数据完整性验证.

禹勇(1980－),男,博士,教授,博士生导师,CCF 专业会员,主要研究领域为区块链与密码货币,云计算安全,大数据安全与隐私保护.



李艳平(1978－),女,博士,副教授,主要研究领域为云存储数据完整性验证,可搜索加密.

丁勇(1975－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为无线网络安全,椭圆曲线密码体系,可证明安全性研究.