doi:10.11835/j.issn.1000.582X.2024.02.008

面向云审计的轻量级隐私保护方案

张晓琴[1],姚 远[2],王 颖[3]
(1. 重庆市信息通信咨询设计院有限公司,重庆 400041; 2. 中国中医科学院 西苑医院信息中心,北京 100091; 3. 重庆大学 大数据与软件学院,重庆 401331)

摘要:在大数据爆发式增长背景下,云存储服务的发展为用户数据存储带来了极大的方便,按需服务特性使其备受青睐。但由于失去了对云服务器中数据的直接控制,不确定因素可能导致用户的数据损坏,这给云存储发展带来极大挑战。笔者提出轻量级计算和验证的数据审计方案,简化了用户在上传数据之前的标签计算操作,保证用户上传数据的安全性。此外,云服务器和审计者的计算任务也得到减轻,进一步降低计算开销。为保护用户数据隐私,借鉴了图像加密中的置乱加密,让用户使用随机函数对数据块的位置进行置乱,同时让审计者计算出数据块的真实位置,完成审计操作。实验结果表明,该方案有效节省了审计流程中用户和服务器、审计者3方的计算资源,提升流程效率。

关键词:数据审计;云服务器;隐私保护;置乱加密;随机函数
中图分类号:TP333 **文献标志码:**A **文章编号:**1000-582X(2024)02-075-09

Lightweight privacy protection scheme for cloud audit

ZHANG Xiaoqin[1], *YAO Yuan*[2], *WANG Ying*[3]
(1. Chongqing Communication Design Institute Company Ltd., Chongqing 400041, P. R. China; 2. Information Center of Xiyuan Hospital, China Academy of Chinese Medical Sciences, Beijing 100091, P. R. China; 3. College of Big Data and Software, Chongqing University, Chongqing 401331, P. R. China)

Abstract: In the context of the explosive growth of big data, the emergence of cloud storage services has significantly facilitated user data storage. The on-demand nature of cloud servers further contributes to their widespread popularity. However, this convenience comes at the expense of direct user control over data stored in the cloud, exposing it to potential damage from various uncertain factors. This brings great challenges to the advancement of cloud storage. To address these challenges, a data auditing scheme is proposed, emphasizing lightweight calculation and verification. This solution streamlines the user's label calculation operation before

收稿日期:2022-10-13
基金项目:国家重点研发计划资助项目(2022YFB3402003);重庆市技术创新与应用发展专项重点资助项目(CSTB2022TIAD-KPX0054);贵州省交通运输厅科技资助项目(2015121024)。
Supported by National Key Research and Development Program of China (2022YFB3402003), Special Key Program for Technological Innovation and Application Development of Chongqing (CSTB2022TIAD-KPX0054) and the Scientific Project of Guizhou Transportation Department (2015121024).
作者简介:张晓琴(1984—),女,博士研究生,正高级工程师,主要从事网络安全、数据安全、人工智能方向研究,(E-mail) zhangxq.cq@chinaccs.cn。
通信作者:姚远,男, (E-mail)xyyyao@sohu.com。

uploading data, ensuring data security during the upload process. This approach concurrently reduces the calculation tasks of both the cloud server and the auditor, minimizing overall calculation overhead. To protect user data privacy, the scheme incorporates scrambling encryption inspired by image encryption. This enables users to use random functions to scramble the data block's location, while still allowing the auditor to calculate the actual data block location for successful auditing. The results show that the proposed solution effectively saves computing resources for users, servers, and auditors during the audit process, thereby improving overall process efficiency.

Keywords: data audit; cloud server; privacy protection; scrambling encryption; random function

云计算服务具有按需服务,接入网络便捷,存储资源丰富,灵活性高等特性,已吸引越来越多的组织和个人用户关注[1-2]。通过将数据存储迁移到云服务器或将计算任务交由云服务代理完成,用户可节省大量计算资源。云计算在数据量巨大、计算任务繁重领域如电商、车联网[3]、医疗护理[4]中得到广泛应用。但随着云存储大范围使用,数据安全问题也日益凸显[5-7],数据一旦被迁移到云服务器后,用户就不能够对其进行直接操作控制,在此情况下传统方法验证数据的完整性和正确性不再适用。云服务代理可能在未经用户许可情况下使用用户数据,侵犯用户隐私。除云服务代理之外,网络中还存在着外部攻击者有意监听或篡改用户数据的风险。因此,对于使用云服务的用户而言,能够验证被迁移数据完整性和正确性的数据审计方案具有重要研究意义。

近年来,国内外对审计的研究成果相当丰富。在审计架构方面主要有两种方案:①基于私有审计架构[8-9]。Juels等[10]首次提出了可恢复性证明模型(proofs of retrievability,PoR),它可使用户远程验证半可信服务器中的数据完整性。但PoR是一个私有审计方案,不支持对数据的动态更新;②公有审计架构[11-12]。Ateniese等[13]提出了可证明数据所有模型(provable data posession,PDP),首次提出了公开审计概念。Shacham等[14]提出了新颖的公开审计方案,该方案使用BLS签名。与传统基于RSA签名相比,BLS签名的长度更短,对于降低通信成本有显著表现。此方案提出后,许多方案也采用BLS签名来节省通信计算量和实现批量审计功能。Curtmola等[15]提出了多副本PDP模型方案,使用该方案的用户需要在服务器上存储一个文件的多个副本。当某些文件被破坏时,可利用其他副本对文件进行快速恢复。私有审计方案虽然能很好满足当时用户需求,但随着信息基础设施快速升级,审计结果可信度也易受质疑。私有审计模型由于需要用户承担大量通信和计算开销,目前也无法满足用户的全部需求。相比之下,公有审计模型加入了一个可信第三方审计员,用户可将数据审计的任务委派给第三方完成,节约自己的资源。这一概念最早由Wang等提出,第三方审计者可代表用户进行数据验证,帮助用户节省因审计产生的开销。尽管第三方审计者在之后的审计方案中得到了广泛应用,但仍存在一些缺点,如缺少对审计者的有效监管。

综上所述,私有审计方案架构简单、安全性较高,且能够最大限度保证审计结果的真实性。但是,该方案要求用户定期进行数据审计,频繁与云服务器进行交互,严重消耗用户自身的网络和计算资源,与用户使用云服务器目的背道而驰。对于公开审计而言,虽然引入了第三方审计者,用户可以对第三方审计者进行授权,委托其执行数据审计工作[16-18]。第三方审计者通常都被假设为是完全可信的,实际上第三方审计者仍可能会与云服务提供商共同向用户隐藏数据损坏或丢失的事实。而且第三方审计者作为中心化的一方,一旦受到外部攻击或发生内部故障,审计过程都会被影响。为解决上述审计中存在的资源消耗严重且用户隐私得不到保证的问题,研究以最小化计算开销和保护用户数据隐私为重心,设计了轻量级计算方法和审计流程,其主要贡献在于:

1)借鉴了图像加密中的置乱加密,使用伪随机函数对文件内数据块的顺序进行置乱,不增加用户额外开销;

2)在数据存储阶段,伪随机函数的种子使用双线性映射结合用户私钥计算,保证只有用户和第三方审计者才能计算出数据块正确顺序;

3)用户在计算数据块签名时也采用了伪随机函数种子,具体可验证标签则由云服务器生成,可有效节省审计流程中用户和服务器、审计者三方计算资源。

1　系统模型及设计目标

1.1　建立系统模型

方案采用了传统的云数据公开审计模型,系统模型中包含3个角色:用户、云服务提供商和可信第三方审计者。不同角色在方案中的定位和功能如下所述:

用户(data owner,DO):用户拥有一定容量数据,但自身的存储和计算能力有一定限制,选择将数据外包给云服务器。用户需要确保自己的数据完整保存在云服务器上,并且授权可信第三方审计者对自己的数据进行定期审查。在审计方案中,用户需承担的计算和通信开销应尽可能最小。

云服务提供商(cloud service provider,CSP):云服务提供商即管理着足量的云服务器提供大量存储和计算资源角色。用户将自己的数据交由云服务提供商存储到云服务器中,同时要求云服务提供商保证数据完整性。

可信第三方审计者(trusted third party authority,TPA):审计者的作用是获得用户的委托和授权,对云服务器中对应用户的数据进行定期审计,将审计结果发送给用户,确保在数据受损时用户能及时发现采取后续措施。在这个过程中,假设第三方审计者是半可信的,即审计者会诚实完成审计,但会对用户的数据保持好奇,试图通过审计数据恢复出用户原始数据。因此,审计流程需要在保证审计效果前提下,保护用户数据的安全性。

本方案提出了一种轻量级的计算方法和审计方法,其简要步骤为:首先,用户对需要上传到云服务器的数据进行预处理,生成可验证数据标签和其他信息,以便审计者进行审计;其次,将处理后的数据及数据标签上传到云服务器,授权第三方审计者对自己的数据进行审计。审计者将定期对云服务器发起审计挑战,云服务器计算出相应的数据拥有证明并将其发送给对应审计者。审计者收到数据拥有证明后进行验证,并将验证结果发送给用户。如果验证成功,则说明数据仍然保持完整;如果验证失败,则说明数据的完整性受损,用户需要及时采取对应措施。

为更好保护用户隐私,本方案在数据预处理阶段引入图像加密中的置乱加密操作,即用户在对数据块生成可验证标签后,使用伪随机函数对同一文件中数据块的顺序置乱。然后,用户将打乱数据块顺序的文件发送给云服务器,防止数据隐私泄露。伪随机函数的随机种子采取密钥交换原理,由用户的私钥和第三方审计者私钥共同生成,保证对云服务器的保密性。对于用户对第三方审计者进行授权操作,用户可为审计者生成相应证书,以证明审计者获得了用户授予的审计权限。证书可以使用签名、加密算法等方式生成,并用于验证审计者的身份。

1.2　设计系统目标

笔者提出的方案主要目标是实现可靠的审计结果。同时减少方案参与者处理数据所需的计算资源,减轻用户的资源消耗,加强对用户数据的隐私保护。本方案有如下设计目标:

1)用户原始数据隐私保护:在用户预处理过程中,采用图像加密中的置乱加密对用户文件中的数据块进行乱序操作。该操作不需要额外存储空间,能有效保护用户原始数据的隐私。

2)减轻参与者的计算消耗:在能够保证审计效果以及审计流程安全性前提下,简化用户预处理步骤,修改数据标签计算方法。此外,减少计算时间,将部分数据标签的生成操作转移到云服务器进行,减少审计过程中的计算资源消耗。

2　方案详细设计

本方案体包括5个阶段:

1)初始化:用户确定进行数据上传所需的各类参数和计算函数,云服务提供商以及第三方审计者也需要生成自己的公私钥对。

2)数据存储:参数生成完毕后,用户开始处理将外包给云服务器的数据。用户需要对文件进行分块、生成可验证数据标签、数据块乱序以及发送等操作。

3)数据审计:第三方审计者接受用户的授权和委托,对云服务器发起审计挑战。

4)证明生成:云服务器根据收到的挑战信息计算数据拥有证明,并将其发送给第三方审计者。

5)数据验证:第三方审计者验证数据拥有证明是否正确。

以上5个阶段即为本方案主要流程,同时还支持用户对云服务器中数据动态操作。本方案中使用的符号如表1所示。

表1 方案符号说明

Table 1 Scheme symbol description

符号	说明
G,G_T	循环群
p	循环群的大素数阶
e	双线性映射
F,F_{ID}	用户的文件和文件的标识号
m_i	文件中的数据块
H	映射$\{0,1\}^* \rightarrow G$的哈希函数
α	用户的私钥
β,g,u	用户的公钥
y,Y	云服务器的私钥和公钥
z,Z	第三方审计者的私钥和公钥
ϕ	数据块的状态信息集合
σ_i,σ	用户生成的数据块签名和签名集合
θ_i,θ	云服务器生成的数据块标签和标签集合
T	用户生成的文件标签
(sk,pk)	用户的签名密钥对
$\tau_{key}(\cdot)$	伪随机函数

2.1 初始化阶段

在初始化阶段,用户、云服务器以及第三方审计者都需要生成自己的参数以及一些函数。对于用户,首先用户需要确定一个大素数p。接着,选择2个以大素数p为阶的循环群G,G_T,设定双线性映射$e:G \times G \rightarrow G_T$。设$H$为哈希函数$\{0,1\}^* \rightarrow G$上的映射,$\{0,1\}^*$表示由0和1组成的任意长度字符串;其次生成用户的公钥和私钥:用户随机选择元素$\alpha \in Z_p$以及$g,u \in G$,设$\beta = g^{\alpha}$。然后用户选择随机签名密钥对(sk,pk),用于对数据进行签名;最终,本方案中用户的参数为私钥$SK = (sk,\alpha)$,公钥$PK = (pk,\beta,g,u)$。用户的参数确定后,云服务器随机选择元素$y \in Z_p$作为自己的私钥,对应的公钥为$Y = g^y$,第三方审计者从Z_p中随机选择元素z作为私钥,并令公钥$Z = g^z$。以上参数中私钥均由生成者保存并保证其私密性,公钥保持公开,即任何第三方都可以获得。

2.2 数据存储阶段

参数初始化完毕后,用户对需要上传到云服务器的数据进行预处理。首先对文件进行分块,一个文件会被分为若干个大小相同的数据块,假设一个文件可被分为n块,记为$F = \{m_1,m_2,\cdots,m_n\}$。同时用户为每个文件分配唯一标识号,令文件$F$的标识号为$F_{ID}$。文件分割完成后,用户开始为数据块生成签名,首先用户计算$t = e(Y,Z)^{\alpha}$,然后计算数据块的签名

$$\sigma_i = t \cdot u^{m_i}, 1 \leqslant i \leqslant n \ , \tag{1}$$

接着,用户开始对文件F中的数据块进行置乱操作。假设t为伪随机函数种子,计算数据块新的排布位置,计算方式为$k_i = \tau_t(i), 1 \leqslant i \leqslant n$。计算完成后,用户根据$k_i$对文件重新进行排布,得到$F' = \{\cdots, m_{k_i}, \cdots\}_{1 \leqslant i \leqslant n}$。为保证文件唯一标识号完整性,用户需要为文件生成文件标签$T = F_{\mathrm{ID}} \| \mathrm{sig}_{sk}(F_{\mathrm{ID}})$,其中$\mathrm{sig}_{sk}(F_{\mathrm{ID}})$表示使用私钥$sk$为文件标识号生成的签名,并令所有数据块签名的集合为$\sigma = \{\sigma_i\}_{1 \leqslant i \leqslant n}$。最终用户将$\{F', \sigma, T\}$发送给云服务器保存,并删除本地存储数据。

收到用户发送数据后,云服务器还需要使用双线性映射e进一步计算其可验证标签。云服务器根据收到的数据块签名计算可验证标签方式为:$\theta_i = e(\sigma_i, \beta)$;计算完所有数据块的可验证标签后,云服务器将它们组成的集合记为$\theta = \{\theta_i\}_{1 \leqslant i \leqslant n}$。同时,云服务器将用于数据验证的$\theta$和$T$与文件$F'$一同保存。

2.3　数据审计阶段

用户将数据上传至云服务器后,授权并委托第三方审计者代表自己对云服务器发起审计挑战。在发起挑战之前,审计者需先通过验证文件标签是否正确来确认该文件是否属于该用户。审计者从云服务器处获得T,并通过用户的签名公钥pk验证其是否正确。验证失败即中止本次审计并向用户提交错误报告;验证成功则审计者获取到F_{ID}并可继续生成挑战信息。审计者构造挑战信息为$\mathrm{chal} = \{F_{\mathrm{ID}}, i, s_i, l,\}_{i \in I, l \in L}$。由于存储在云服务器中的数据块顺序和可验证签名顺序不对应,在挑战信息中审计者需要为受到挑战的数据块生成数据块序号集合和签名序号集合。挑战信息中L为此次受到挑战的数据块可验证标签的索引集合,包含从$\{1,2,\cdots n\}$中挑选的c个元素。I为与可验证标签对应的数据块索引集合。为得到集合I,审计者需要先使用自己的私钥以及用户和云服务器的公钥计算随机函数种子,即$t = h(e(\beta, Y)^Z)$,$I = \{i = \tau_t(l) | l \in L\}$。$s_i$则与$i$一一对应,从$Z_{\mathrm{p}}$中选取的$c$个元素。为防止云服务器的被攻击,每一次挑战信息中的集合L和s_i都是随机选择,保证每次挑战的信息都不同。挑战信息生成完毕后,审计者将chal发送给云服务器,并等待其回复。

2.4　证明生成阶段

在收到审计者发来的挑战信息后,云服务器需要计算对应的数据拥有证明,以证明自己所保存的该用户数据完整性。为此,云服务器首先生成可验证标签证明

$$\Theta = \prod\nolimits_{i \in I, l \in L} \theta_l^{s_i}, \tag{2}$$

基于可验证数据标签的同态性,多个数据块的标签可被聚合为一个,在此用Θ表示。然后,云服务器计算数据证明

$$M = \sum\nolimits_{i \in I} s_i \cdot m_i, \tag{3}$$

计算完成后,云服务器将数据拥有证明$\{\Theta, M\}$发送给第三方审计者。

2.5　数据验证阶段

收到数据拥有证明后,审计者使用如下表达式对其进行验证

$$\Theta = e\left(\prod\nolimits_{i \in I} t^{s_i} \cdot u^M, \beta\right), \tag{4}$$

若审计者验证成功,则说明云服务器完整保存了用户数据,否则数据的完整性受到了破坏,需要用户立即采取相应措施。

2.6　数据批量审计

在实际应用中,如果审计者能一次性批量审计多个用户数据,将极大提高审计者的工作效率,最大程度确保每个用户数据都按时定期进行完整性检验。

假设有γ个用户数据需要进行审计,且将用户集合记为$U = \{1,2,\cdots\gamma\}$,其中的某个用户用U_ω表示,$\omega \in U$。用户U_ω的私钥用$\alpha_\omega \in Z_{\mathrm{p}}$表示,公钥用$\beta_\omega = g^{\alpha_\omega}$表示。用户仍然随机选择$u_\omega \in G$以及随机签名密钥对$(sk_\omega, pk_\omega)$系统的全局参数与单用户数据审计相同,有双线性映射$e: G \times G \rightarrow G_{\mathrm{T}}$以及$H$和$h2$个哈希函数,其中$H$为$\{0,1\}^* \rightarrow G$上的映射,$h$为$\{0,1\}^* \rightarrow Z_{\mathrm{p}}$上的映射。云服务器的私钥为从$Z_{\mathrm{p}}$中随机选取的元素$y$,公钥为$Y = g^y$,第三方审计者的私钥为从$Z_{\mathrm{p}}$中随机选取的元素$z$,公钥为$Z = g^z$。接下来的步骤与单用户数据审计相似。

1)数据存储

设用户 U_ω 需要上传的文件为 F_ω,需要将其切分为同等大小的 n 个数据块,表示为 $F_\omega=\{m_{\omega,1},\ m_{\omega,2},\ \cdots,\ m_{\omega,n}\}$,文件的标识符为 $F_{\omega,\ \mathrm{ID}}$。切分完成后,用户首先计算 $t_\omega=h(e(Y,Z)^{\alpha^\omega})$,然后为每个数据块生成签名

$$\sigma_{\omega,i}=h(t_\omega)\cdot u_\omega^{m_i},1\leqslant i\leqslant n\ , \tag{5}$$

签名计算后,用户接着对文件进行置乱,打乱数据块顺序,用户计算 $k_{\omega,i}=\tau_{t_\omega}(i),1\leqslant i\leqslant n$,然后按照 $k_{\omega,i}$ 来对数据块进行重新排序,记为 $F'_\omega=\{\cdots,m_{k_{\omega,i}},\cdots\}_{1\leqslant i\leqslant n}$。然后用户基于 $F_{\omega,\ \mathrm{ID}}$ 为文件生成标签 $T_\omega=F_{\omega,\ \mathrm{ID}}\left\|\mathrm{sig}m_{sk_\omega}(F_{\omega,\ \mathrm{ID}})\right.$,将对数据块的签名集合记为 $\theta_\omega=\{\theta_{\omega,i}\}_{1\leqslant i\leqslant n}$。再将 $\{F'_\omega,\ \theta_\omega,T_\omega\}$ 上传至云服务器进行保存,删除本地的数据文件,完成数据存储操作。

云服务器收到用户 U_ω 发来的数据后,再基于用户的数据块签名为其计算出可验证标签 $\theta_{\omega,i}=e(\theta_{\omega,i},\beta_\omega)$,其合集记为 $\theta_\omega=\{\theta_{\omega,i}\}\ U_{1\leqslant i\leqslant n}$。然后云服务器存储 $\{F'_\omega,\ \theta_\omega,T_\omega\}$。

2)发起挑战

对于 $U_\omega\in U$,审计者生成挑战信息 $\mathrm{chal}_\omega=\{F_{\omega,\mathrm{ID}},i,s_i,l\}_{i\in I_\omega,l\in L}$。挑战信息内各参数的意义与单用户审计相同。为便于计算,假设审计者为所有用户选取的数据标签索引集合 L 均为同一个,使用不同为随机函数种子生成对应用户的数据块索引集合 I_ω,S_i 仍为随机选取的 Z_p 中的元素。

3)生成数据证明

接收到来自审计者挑战信息后,云服务器需要为每个用户分别生成可验证标签证明 $\Gamma_\vartheta=\prod_{\omega\in U}\Gamma_\omega$ 和数据证明 M_ω,然后将标签证明聚合到一起,聚合公式为:$\Gamma_\vartheta=\prod_{\omega\in U}\Gamma_\omega$。聚合完成后,云服务器将 $\{\Gamma_\vartheta,M_\vartheta|\omega\in U|\}$ 发送给审计者。

4)数据验证

对于聚合证明,审计者通过如下表达式进行数据验证

$$\Gamma_\vartheta=\prod_{\omega\in U}e(\prod_{i\in I_\omega}h(t_\omega)^{S_i}\cdot u^{M_\omega},\beta), \tag{6}$$

若上述等式成立,表示所有受到挑战的文件数据都完整无损,否则这些文件中有一些受到了损坏或篡改。

3 方案安全性分析

对用户而言,其上传的数据需要能够抵抗恶意攻击者的伪造攻击,同时云服务器无法使用伪造的数据拥有证明通过审计。具体安全性分析如下:

1)恶意攻击者无法对用户上传到云服务器的数据进行伪造攻击。

在用户上传数据过程中,可能会遭遇恶意攻击者的数据伪造攻击。攻击者会截获用户发送的文件并将其替换为自己的数据,然后使用自己生成的参数计算出其他信息后再将其发送给云服务器。但在用户发送给云服务器的数据 $\{F',\sigma,T\}$ 中包含了用户生成的签名 $\sigma_i=h(t)\cdot u^{m_i}$,其中 t 由用户的私钥计算而成,即攻击者无法在有限时间内推测出 $h(t)$ 的值,也就无法伪造出正确的用户签名。

2)云服务器无法使用假冒的数据拥有证明通过审计。

当云服务器中存储的用户数据因某些原因受到损坏时,云服务器可能会试图伪造数据拥有证明应对审计者的审计挑战。但审计者每次发起审计时都会选取随机生成的数据作为挑战值,且每次受到挑战的数据块都不同,云服务器需要根据审计者发送的信息计算数据拥有证明。用户对文件内的数据块排列顺序进行了置乱。如果云服务器想事先计算出所有可能受到挑战的数据块组合对应的数据拥有证明,云服务器会消耗更多存储空间保存数据拥有证明。本方案假设云服务器中存储了 n 个数据块-可验证标签对,设挑战信息为 $\{F_{\mathrm{ID}},i,s_i,l.\}_{i\in I,l\in L}$,有 c 个数据块将被挑战,其中 $1\leqslant c\leqslant n$,即存在 C_n^c 个有可能数据块的排列组合。由于 $c\in\{1,2,\cdots,n\}$,那么所有可能受到挑战的数据块集个数为 $C_n^1+C_n^2+\cdots+C_n^n=2^n-1$。如果云服务器要实现计算出数据拥有证明然后再删除数据就需要存储 2^n-1 个聚合数据证明和标签证明。当 $n\geqslant 2$ 时 $2^n-1>$

n,在本文的方案中数据块和数据标签的顺序并不是按照存储顺序一一对应,云服务器如果进行计算出所有可能的数据拥有证明,所需的存储量将远大于 $2^n - 1$。且所有可能的数据拥有证明数量会随着原数据的增加而呈现指数级增加,因此,这种行为对于云服务器是不可取的。即使云服务器提前计算出了所有可能数据拥有证明,用户对文件中数据块的乱序操作也会使有 c 个数据块受到挑战时,正确数据证明和标签证明对应方式只有一种,即云服务器只有 $1/C_n^{c^c}$ 的概率通过这次审计,在数据数量足够大时,这几乎是不可能的。

4　方案性能分析

对所提方案的性能进行实验,与现有审计方案进行对比分析。首先选择了提出 DHT 这一数据结构来辅助审计的 Tian 等人[16]提出的 DHT-PA 方案,接着对比经典的 Wang 等人[17-18]提出的 W-PoR 方案。性能指标主要包括:所要评估的环节包括用户生成验证标签、云服务器生成数据拥有证明以及第三方审计者进行数据验证的计算开销。所有的实验使用配备了 Windows7 操作系统,2.4GHz Intel Core i7-4500U CPU,以及 4GBRAM 的计算机进行,实验代码使用 C++语言编写,密码学算法则是基于 0.5.14 版本的配对基础密码(PBC)库,使用 A 类配对参数模拟运行,其中循环群的阶长度为 160 bit。

表 2 列出了方案中使用的一些基础运算操作,表 3 总结了在方案不同阶段所需的计算量,并与其他审计方案进行对比,表中:n 代表数据块的总量;c 代表受到审计挑战的数据块个数。

表 2　计算符号说明

Table 2　Notation of calculation sign

计算符号	符号含义
M_G	群上的乘法运算
E_G	群上的指数运算
H_G	群上的哈希运算
BP	双线性运算

表 3　各阶段计算开销

Table 3　computing costs in each phase

方案	生成验证标签	生成数据拥有证明	数据验证
DHT-PA	$(M_G + E_G + H_G) \cdot n$	$cM_G + cE_G + BP$	$2cM_G + (c+2)E_G + 2cH_G + BP$
W-PoR	$(M_G + 2E_G + H_G) \cdot n$	$(c-1)M_G + cE_G$	$cM_G + (c+1)E_G + cH_G + 2BP$
提出方案	$E_G + BP + n(M_G + E_G)$	$(c-1)M_G + cE_G + cM_{Z_p}$	$cM_G + (c+1)E_G + BP$

由于将计算可验证标签的任务交由云服务器来完成,且云服务器计算可验证数据标签采用了不同计算方式,使整个计算开销得到显著降低。虽然用户部分计算也外包给了云服务器,但数据签名的计算中加入了由用户私钥计算出的参数,并且利用双线性映射性质,第三方审计者也可以计算出该参数,该参数被看做是用户对于审计者的委托授权,只有得到授权的审计者才能正确审计用户的数据,用户不担心受到伪造数据攻击。云服务器虽然承担了生成可验证标签名的计算任务,但生成过程只需要在用户最初上传数据时进行一次即可,之后需要生成数据拥有证明时可直接使用之前生成的可验证标签,这部分开销对于云服务器而言是恒定的。相同地,用户在生成数据标签后对文件中数据块进行置乱操作需要使用伪随机函数,该操作只需要在数据上传阶段进行,且伪随机函数的计算消耗与群上的哈希运算相近,带来的计算开销也是用户可以承受的。在发出审计挑战阶段,生成可验证标签索引集合后,审计者还需要使用伪随机函数生成对应数据块索引集,这会消耗一部分计算资源,但保证安全性。由于伪随机函数种子在计算过程中用到了第三方审计者私钥,能防止审计者和云服务器共谋,使用之前数据拥有证明通过审计。

本方案与其他方案对比了用户生成数据标签所消耗的时间及云服务器生成数据拥有证明时间,如图 1-2 所示。在 W-PoR 方案中,验证标签计算和拥有证明生成所需时间较长,且随数据块数量增长增长较快,而

DHT-PA方案则与本方案耗时相近。在数据上传阶段,W-PoR方案中采取比本方案更耗时的指数运算来计算可验证标签。本方案中标签计算方式虽然与DHT-PA方案相同,但提出的方案在标签计算中加入了由用户私钥计算出的参数,这可以防止数据上传过程中恶意攻击者的数据伪造攻击,DHT-PA方案无法抵抗这种攻击。在拥有证明生成阶段,由于标签的计算方式不同,云服务器所进行的标签聚合操作也存在类似趋势的时间消耗。

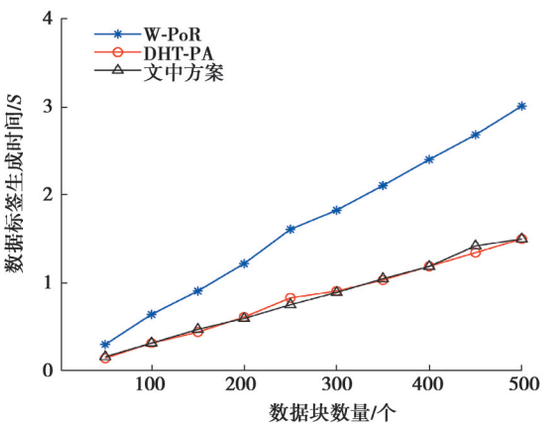

图1 数据标签生成时间对比

Fig. 1 Data tag generation time comparison

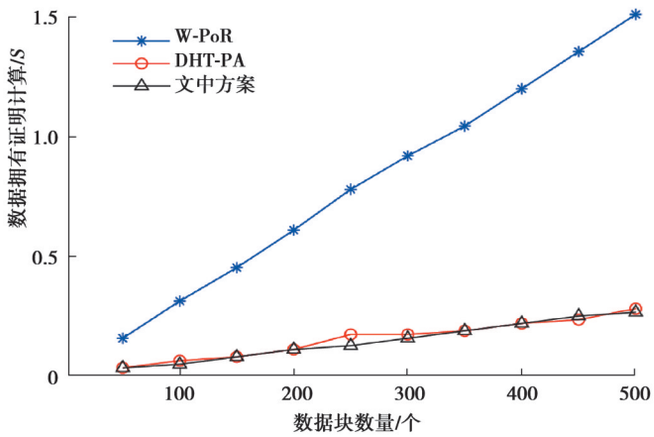

图2 数据拥有证明生成时间对比

Fig. 2 Proof time comparison

在图3中各个方案在数据验证阶段所需的时间消耗进行了比较,本方案中数据验证所需时间花费短,且随着数据增多也能基本保持稳定。这对第三方审计者非常重要,可进一步支持审计者进行批量审计,为更多用户提供更好审计服务。

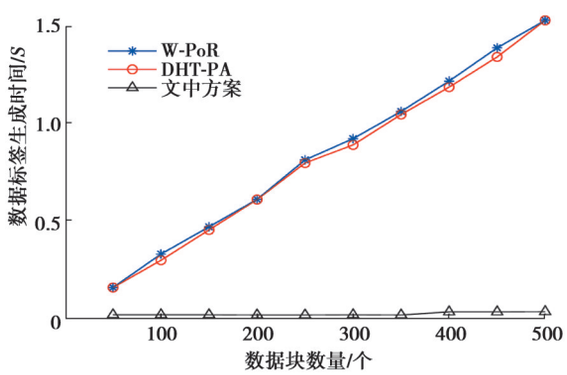

图3 数据验证时间对比

Figure 3 Data verification time comparison

5 结束语

笔者提出一种旨在加强用户隐私保护和降低用户计算消耗的云数据审计方案。方案主要目的是在实现公开审计方案基本功能前提下,进一步加强对于用户原数据的隐私保护,减少审计方案各阶段计算开销。在与其他方案进行对比时,一些方案通过在用户文件数据块中随机插入混淆数据块以保护用户隐私,这会增加用户通信开销和云服务器存储开销。研究提出的方案则以图像加密中置乱加密为启发点,使用伪随机函数对文件内数据块的顺序进行置乱,同时不增加额外开销。此外,伪随机函数种子使用双线性映射结合用户的私钥计算,保证只有用户和第三方审计者才能计算出数据块正确顺序。最后,通过实验对比,本方案有效节省审计流程中用户和服务器、审计者三方的计算资源,提升了整个流程效率。

参考文献

[1] Armbrust M, Fox A, Griffith R, et al. A view of cloud computing[J]. Communications of the ACM, 2010, 53(4): 50-58.

[2] Butoi A, Tomai N. Secret sharing scheme for data confidentiality preserving in a public-private hybrid cloud storage approach [C]//2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing. December 8-11, 2014. London, UK: IEEE, 2015: 992-997.

[3] Naseer Qureshi K, Bashir F, Iqbal S. Cloud computing model for vehicular ad hoc networks[C]//2018 IEEE 7th International Conference on Cloud Networking (CloudNet). October 22-24, 2018. Tokyo, Japan: IEEE, 2018: 1-3.

[4] Cao S, Zhang G X, Liu P F, et al. Cloud-assisted secure eHealth systems for tamper-proofing EHR via blockchain[J]. Information Sciences, 2019, 485: 427-440.

[5] Diao Z, Wang Q H, Su N Z, et al. Study on data security policy based on cloud storage[C]//2017 IEEE 3rd International Conference on Big Data Security on Cloud (Big Data Security), IEEE International Conference on High Performance and Smart Computing (Hpsc), and Ieee International Conference on Intelligent Data and Security (IDS), May 26-28, 2017, Beijing, China, IEEE, 2017: 145-149.

[6] Markandey A, Dhamdhere P, Gajmal Y. Data access security in cloud computing: a review[C]//2018 International Conference on Computing, Power and Communication Technologies (GUCON). September 28-29, 2018, Greater Noida, India: IEEE, 2019: 633-636.

[7] Kumar R, Goyal R. On cloud security requirements, threats, vulnerabilities and countermeasures: a survey[J]. Computer Science Review, 2019, 33: 1-48.

[8] 王靖, 傅剑峰. 关于政务私有云日志审计系统设计与实现[J]. 长江信息通信, 2021, 34(4): 185-187.
Wang J, Fu J F. Design and implementation of government private cloud log audit system[J]. Changjiang Information & Communications, 2021, 34(4): 185-187.(in Chinese)

[9] 饶水林. 云计算技术的审计私有云模式建设与优化探讨[J]. 中国内部审计, 2017(10): 92-95.
Rao S L. Discussion on the construction and optimization of audit private cloud mode of cloud computing technology[J]. Internal Auditing in China, 2017(10): 92-95.(in Chinese)

[10] Juels A, Kaliski B S. Pors: proofs of retrievability for large files[C]//Proceedings of the 14th ACM conference on Computer and Communications Security, 2 November 2007, Alexandria, Virginia, USA. New York: ACM, 2007: 584-597.

[11] Wang Q, Wang C, Li J, et al. Enabling public verifiability and data dynamics for storage security in cloud computing[C]// European Symposium on Research in Computer Security. Berlin, Heidelberg: Springer, 2009: 355-370.

[12] Wang C, Chow S S M, Wang Q, et al. Privacy-preserving public auditing for secure cloud storage[J]. IEEE Transactions on Computers, 2013, 62(2): 362-375.

[13] Ateniese G, Burns R, Curtmola R, et al. Provable data possession at untrusted stores[C]//Proceedings of the 14th ACM conference on Computer and Communications Security, 2 November 2007, Alexandria, Virginia, USA. New York: ACM, 2007: 598-609.

[14] Shacham H, Waters B. Compact proofs of retrievability[C]//International Conference on the Theory and Application of Cryptology and Information Security. Berlin, Heidelberg: Springer, 2008: 90-107.

[15] Curtmola R, Khan O, Burns R, et al. MR-PDP: multiple-replica provable data possession[C]//2008 the 28th International Conference on Distributed Computing Systems, June 17-20, 2008, Beijing, China. IEEE, 2008: 411-420.

[16] Tian H, Nan F L, Chang C C, et al. Privacy-preserving public auditing for secure data storage in fog-to-cloud computing[J]. Journal of Network and Computer Applications, 2019, 127: 59-69.

[17] Wang H Q. Proxy provable data possession in public clouds[J]. IEEE Transactions on Services Computing, 2013, 6(4): 551-559.

[18] Wang C, Wang Q, Ren K, et al. Ensuring data storage security in cloud computing[C]//2009 17th International Workshop on Quality of Service, July 13-15, 2009, Charleston, SC. IEEE, 2009: 1-9.

（编辑　侯　湘）