面向团队场景的无证书云数据完整性授权审计方案研究

刘易齐[1)] 徐 贤[1)] 龙 宇[2)]

[1)](华东理工大学信息科学与工程学院 上海 200237)

[2)](上海交通大学计算机科学与工程系 上海 200240)

摘 要 在云存储技术飞速发展的环境下,大量用户可以将自己的数据上传至云服务器,从而节省本地存储空间。云服务器可能存在数据丢损风险,因此引入第三方审计员进行云数据完整性审计是很有必要的措施。在团队场景的审计任务中,现有研究方案主要集中于支持完整性审计和用户撤销操作。然而,审计员的审计操作可能会泄露用户的数据隐私信息,并非所有用户都希望任意审计员审计其数据。为此,我们提出了一个面向团队场景的无证书云数据完整性授权审计方案。该方案通过共用陷门实现了审计员的授权及变更,采用随机盲化因子对审计证明进行盲化处理,保护了用户的数据隐私。在安全性分析中,所提方案满足正确性、隐私保护性、数据标签的不可伪造性等特性。实验分析表明,我们的方案支持多用户高效批量验证,与传统的本地用户更新标签方案相比分别将更换授权审计者和处理用户撤销的时间降低至0.1%和46%左右,有效节省了计算耗费。

关键词 云存储;数据完整性审计;授权审计;团队场景;数据隐私

中图法分类号 TP309 **DOI号** 10.11897/SP.J.1016.2025.02948

Certificateless Cloud Data Integrity Authorization Audit Scheme for Team Scenario

LIU Yi-Qi[1)] XU Xian[1)] LONG Yu[2)]

[1)](School of Information Science and Engineering, East China University of Science and Technology, Shanghai 200237)

[2)](Department of Computer Science and Engineering, Shanghai Jiaotong University, Shanghai 200240)

Abstract With the rapid advancement of cloud storage technology, an increasing number of users are leveraging cloud servers to store massive volumes of data, thereby significantly reducing the demand for local storage resources. While cloud storage offers substantial benefits in terms of scalability and accessibility, it also introduces potential risks such as data corruption or loss due to server failures, malicious attacks, or operational errors. To mitigate these risks, the introduction of third-party auditors has been widely recognized as an effective mechanism to verify the integrity of remotely stored data. In team scenarios, where multiple users collaboratively store and manage shared data, the need for reliable integrity auditing becomes even more critical. Existing research in the field has predominantly focused on designing schemes that support public integrity auditing and efficient user revocation. These approaches allow an external auditor to periodically check whether the cloud server correctly retains the users' data without requiring local data retrieval. However, a significant drawback of such methods is that the auditing process may potentially expose sensitive information related to the users' data. Since the audit proofs often involve

收稿日期:2025-05-03;在线发布日期:2025-10-15。本课题得到上海市2024年度"科技创新行动计划"(Nos. 24BC3200500,24BC3200300)资助。**刘易齐**,硕士,主要研究领域为密码学与信息安全。E-mail:16638997897@163.com。**徐 贤**(通信作者),博士,副教授,主要研究领域为密码学、区块链与信息安全。E-mail:xuxian@ecust.edu.cn。**龙 宇**,博士,副教授,主要研究领域为区块链和密码学。

metadata generated from the original data, malicious or curious auditors could infer private information through repeated audit interactions. Moreover, in a collaborative team setting, not all users may be willing to allow arbitrary auditors to access their data. To address these challenges, we develop a novel certificateless cloud data integrity authorization audit scheme that is specifically designed for team scenarios. This approach differs from traditional methods that rely on public key infrastructure, as our certificateless framework removes the necessity for complex certificate management and simultaneously avoids key escrow problems. The proposed scheme is built around two key technical contributions. Firstly, it incorporates a shared trapdoor mechanism that enables dynamic authorization and revocation of auditors. This mechanism ensures that changes in auditor assignments can be handled efficiently. Secondly, the scheme utilizes random blinding factors to obscure the audit proof during the verification phase. This design effectively prevents auditors from gaining the actual content of the stored data, thereby safeguarding user privacy. In the security analysis part of our work, we provide proofs that the proposed scheme meets several critical security requirements, including correctness and privacy. Correctness guarantees that all properly generated proofs will be validated successfully. Privacy ensures that any leakage of sensitive information to auditors during the auditing process is prevented. Furthermore, it offers unforgeability of data tags, meaning that even a malicious cloud server cannot create valid authentication tags for data blocks that have been altered or compromised. Experimental evaluations indicate that our scheme supports efficient batch verification for multiple users. Compared to conventional approaches where users must locally update metadata during user revocation or auditor changes, our method significantly reduces computational overhead. Specifically, the time required for changing authorized auditors is reduced to approximately 0.1% of that required by traditional local user update methods, while the time for processing user revocation is cut down to about 46%. These improvements make the scheme highly practical for team-based cloud storage systems.

Keywords　cloud storage; data integrity audit; authorization audit; team scenario; data privacy

1　引　言

近年来，随着大数据和互联网技术的相互融合与飞速发展，数据信息已逐渐演变为一种新型生产要素，成了各行业的核心资源与重要管控对象，海量数据对存储与计算能力提出了更高要求。作为数字化转型的核心支撑技术，云计算[1]通过整合分布式计算资源，为用户提供弹性化的数据处理与存储解决方案，有效降低了本地设备的资源消耗压力。云计算的核心功能在于为用户提供数据计算和数据存储服务。依托云存储解决方案，用户既能以经济高效的方式获取优质服务，又可便捷实现数据共享与协同作业。尽管云存储为大规模数据处理与保存提供了灵活高效的解决途径，但该技术仍面临诸多安全挑战[2]。

当用户将数据上传至云服务提供商(Cloud Service Provider，CSP)后，用户将难以直接验证数据是否完整地存储在云端。在用户与CSP的交互过程中，CSP并非是完全可信的实体[3]。一方面出于维护商业声誉的考虑，CSP可能会刻意隐瞒数据丢失或损坏等异常情况[4]；另一方面由于外部恶意攻击、硬件设备故障或系统漏洞[5]等因素，CSP托管的数据可能面临恶意篡改或意外丢失的风险。这些潜在问题不仅会损害用户的数据安全，还可能引发严重的经济损失和法律纠纷。因此，在云存储环境中建立可靠的数据完整性审计机制就显得格外重要。数据完整性审计是一种辅助用户检查存储在云服务器中数据是否完整的技术，该技术可以帮助用户及时了解数据的存储状况[6]。一种较为普遍的云数据完整性审计方案是公共审计，它能通过引入第三方审计员(Third Party Auditor，TPA)代替用户执行数据完整性审计任务。公共审计方案的优势在于解除了用户必须持续在线的限制，显著降低了用户

的计算开销和网络通信负担，已成为当前数据完整性审计的主要范式。

目前，团队用户进行云数据共享的应用场景日益普遍，仅支持个人用户的数据完整性审计方案难以满足团队协作需求。然而，现有云数据完整性审计方案[7-8]在设计时往往未能充分考虑团队协作场景下的实用性及安全性需求。考虑到团队场景中用户加入与退出的实用性需求，审计方案需要支持用户的变更操作。同时，考虑到数据隐私保护的安全性需求，并非所有用户都愿意接受任意第三方审计员（Third party auditor，TPA）对其数据进行验证。因此，研究具有指定验证者功能的授权审计及用户变更机制具有重要意义。这种授权审计的细粒度访问控制策略既确保了验证的灵活性，又维护了数据所有者的隐私权益，同时用户变更机制的引入提升了方案在团队协作场景中的实用价值。

当前大部分云数据完整性审计方案主要依托公钥基础设施（Public Key Infrastructure，PKI）和基于身份的加密（Identity-Based Encryption，IBE）技术，然而这两类方案在实际部署中仍面临一定挑战，PKI体系下的证书管理复杂度较高，而IBE方案则存在密钥托管问题，这些挑战在一定程度上制约了审计方案的适用性和安全性[9]。相比之下，无证书密码体系[10]提供了一种新思路，它既消除了传统PKI体系中复杂的证书管理负担，又避免了IBE方案中的密钥托管问题，为云数据完整性审计方案提供了新的实现路径。

针对上述内容，本文提出了面向团队场景的无证书云数据完整性授权审计方案。主要贡献如下：

（1）提出了面向团队场景的无证书云数据完整性授权审计方案，方案采用无证书密码学并支持用户变更、授权审计、多用户批量审计等功能，具有更加强大的实用性。

（2）通过共用陷门进行标签计算，实现了对审计员的授权及高效变更。采用盲化因子对审计证明进行盲化处理，保护了用户的数据隐私。

（3）对本文方案进行了严格的安全性分析，分析表明方案满足数据标签、证明信息和陷门的抗伪造性、正确性和隐私保护特性。

（4）性能分析与对比实验表明本文方案支持多用户高效批量验证，与传统的本地用户更新标签方案相比将更换授权审计者和处理用户撤销的时间分别减少至0.1%和46%左右，验证了方案的有效性和可行性。

2 相关工作

在应用场景方面，云数据的应用场景已从单一用户扩展到多用户协同共享[11]。为了让审计方案更好地满足团队场景的需求，许多学者进行了相应研究。Yan等人[12]提出了一种支持隐私保护的数据完整性审计方案，适用于多用户共享数据的场景，然而该研究未能有效解决成员动态撤销的关键问题。针对此问题，Tian等人[13]设计了支持成员动态撤销和数据块更新的团队用户数据完整性验证方案。Li等人[14]基于区块链技术，构建了一个支持多用户场景下的共享审计机制，该机制能实现跨用户数据审计功能。为了提高审计效率，Liu等人[15]提出了一个具有文件预测功能的数据共享审计方案，该方案通过预测文件访问模式提升了审计执行效率。针对第三方审计人员的信任问题，Tian等人[16]提出了一种基于区块链的信誉奖惩审计机制，通过动态信用计算算法有效缓解了第三方审计员的信任缺陷问题。Yi等人[17]结合物联网中的车辆用户场景，设计了一种支持数据共享和车辆用户撤销功能的审计方案。Li等人[7]提出了一个支持用户撤销的数据完整审计方案，同时该方案采用了无证书密码体系，但该方案在数据隐私保护方面存在缺陷，可能导致审计员通过分析证明信息获取用户的隐私数据。

尽管这些方案在实现团队场景的数据完整性审计方面取得了一定进展，但其在审计权限方面仍存在安全缺陷，特别是在大规模团队用户的数据完整性审计下的应用。在团队场景的审计方案中，多个审计员参与数据完整性验证将会增加数据隐私泄露的风险，并非所有用户都希望任意审计员能审计其数据。因此，本研究旨在提出一种新型审计方案，以解决现有方法在审计权限方面的局限性。

大部分传统数据完整性审计方案[18-20]采用的是PKI的签名方式，但该方式会带来复杂的证书管理问题。Wang等人[21]首次提出一种基于身份的审计协议，有效解决了传统PKI方案下的证书管理难题。该方案不仅省去了证书验证，还在计算开销和通信效率方面展现出一定优势。Yu等人[22]提出了一种支持隐私保护的基于身份数据完整性验证方案。针对不可信第三方环境，研究人员采用了零知识证明技术[23]，有效保护了用户的隐私信息。随着基于身份的数据完整性审计方案的发展，密钥托管问题也逐渐显现出来，研究者们开始探索更灵活的无证书

密码体系以进一步降低系统开销。Wang等人[24]首次构建了一个完整的无证书公共审计机制，用于在不可信云环境中验证数据的完整性。Miao等人[25]提出了一种基于区块链的无证书透明化云存储数据完整性审计方案，该方案将无证书密码学和区块链技术结合，通过智能合约和承诺技术实现了透明且不可篡改的审计过程。无证书密码体系既消除了传统PKI体系中复杂的证书管理负担，又避免了IBE方案中的密钥托管问题，为云数据完整性审计提供了新的解决方案。

在数据完整性审计过程中，出于隐私安全需求，并非所有用户都希望任意第三方审计员都能审计其数据。针对这一需求，Worku等人[26]提出了一个具有指定验证者功能的数据完整性审计方案，该方案可以授权第三方审计员进行私密验证。为了提高审计效率，该方案还实现了批量审计功能。Huang等人[27]提出了一个支持指定验证者变更的数据完整性审计方案，同时该方案还采用了无证书密码体系。Yan等人[28]提出了一个抗重放攻击的指定验证者审计方案，该方案能有效防范恶意云服务器发起的重放攻击。Shao等人[29]提出了一种团队场景下具有激励机制的指定验证者审计方案，该方案采用阈值签名确保团队成员能在数据共享中公平参与，有效消除管理者的单一权利。考虑到指定验证者的变更问题，Liu等人[30]提出了一种支持指定验证者变更的隐私保护审计方案。该方案能支持指定验证者的变更，同时也支持数据的动态变化。Li等人[8]提出了一个具有批量审计功能的指定验证者数据完整性审计方案，该方案也支持在数据动态变化的情况下对其进行审计，但是无法实现审计员的动态更新需求。尽管现有的方法采用授权审计技术，但其难以支持团队场景下用户加入和退出的动态调整，因此实现支持用户变更的需求仍然存在。

本研究基于共用陷门技术实现审计员授权，通过用户与CSP的轻量级交互即可由CSP端完成审计员的高效替换。该设计面向团队协作场景，在确保数据完整性审计功能的同时，降低了网络带宽消耗与用户端计算开销。本文还对审计证明进行了盲化处理，在保证方案正确性的前提下有效保护用户数据隐私。这些技术创新为团队协作环境下的数据完整性审计提供了兼具实用性与安全性的解决方案，为构建更完善的云端数据安全保障体系提供了新的技术支持。

3　相关知识

3.1　双线性映射

定义1.　假设群G_1与G_2是阶数为素数q的两个乘法循环群，Z_q^*是除零元之外模q的剩余类群。通常称映射$e: G_1 \times G_1 \to G_2$为一个双线性映射，$e$满足如下性质：

（1）双线性：对于$\forall g_1, g_2 \in G_1$和$a, b \in Z_q^*$，有$e\left(g_1^a, g_2^b\right) = e\left(g_1, g_2\right)^{ab}$；

（2）非退化性：对于$\exists g_1, g_2 \in G_1$，有$e\left(g_1, g_2\right) \neq 1_{G_2}$；

（3）可计算性：对于$\forall g_1, g_2 \in G_1$，存在算法能计算出$e\left(g_1, g_2\right)$。

3.2　DL难题假设

定义2.　假设群G_1是阶数为素数q的乘法循环群，群G_1的生成元是g，$a \in Z_q^*$是未知元素，DL问题就是在给定元组$\left(g, g^a\right)$的情况下计算出a。

定义3.　对于任意概率多项式时间（Probabilistic Polynomial Time，PPT）算法A，算法A求解G_1中DL问题的优势是可忽略不计的。假设Adv是算法A求解G_1中DL问题的优势，具体表示如下：

$$Adv_{G_1 A}^{DL} = \Pr\left[A\left(g, g^a\right) = a : a \leftarrow Z_q^*\right] \leqslant \varepsilon.$$

其中，ε表示可忽略不计的优势。

3.3　CDH难题假设

定义4.　假设群G_1是阶数为素数q的乘法循环群，群G_1的生成元是g，$a, b \in Z_q^*$是两个未知元素，CDH问题就是在给定元组$\left(g, g^a, g^b\right)$的情况下计算出$g^{ab} \in G_1$。

定义5.　对于任意PPT算法A，算法A求解CDH问题的优势是可忽略不计的。假设Adv是算法A求解G_1中CDH问题的优势，具体表示如下：

$$Adv_{G_1 A}^{CDH} = \Pr\left[A\left(g, g^a, g^b\right) = g^{ab} : a, b \leftarrow Z_q^*\right] \leqslant \varepsilon.$$

其中，ε表示可忽略不计的优势。

3.4　可证明安全性理论

可证明安全性理论是一种形式化的安全分析方法，其核心思想是通过归约论证来验证协议的安全性。具体方法是定义一个敌手和挑战者，二者会按照指定的规则进行交互，假定敌手能在预先设定好的安全模型中攻破密码协议，此时挑战者就能利用敌手的输出信息在概率多项式时间内攻破数学难

题，这与目前的数学难题假设定义相矛盾。由此可证，敌手攻破协议的假设不成立，从而确保协议的安全性。下面介绍可证明安全性理论中三个常见的概念。

（1）安全模型：安全模型是一种形式化框架，常用于定义密码协议所需要满足的安全目标、攻击者的能力以及安全性的量化标准。通过形式化定义敌手和挑战者的交互规则与胜利条件，将现实世界的安全需求转化为可数学证明的命题。

（2）数学难题：离散对数、大整数分解、CDH和WDH等数学困难问题，这些困难问题常用于密码协议的安全性证明中。

（3）安全性规约：规约证明的本质是构造一个命题，如果数学难题假设成立，则密码协议满足特定安全模型下的安全性。在安全性规约过程中，如果敌手能攻破密码协议，则数学假设不成立，这与数学难题假设成立相矛盾，从而完成证明。

4 系统模型和定义

本节将对系统模型、算法定义和安全模型这三个部分进行详细介绍。

4.1 系统模型

本节所提方案的系统模型如图1所示，该图展示的系统模型包含了四个组成部分，分别是密钥生成中心（Key Generate Center，KGC）、用户团队、第三方审计机构TPA和云存储服务器CSP。

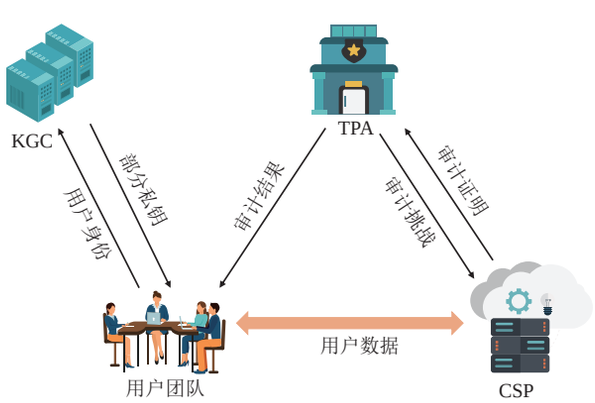

图1 系统模型图

（1）KGC：密钥生成中心是一个受信任的机构，负责整个系统的初始化操作，同时还负责接收用户的身份信息并为用户生成部分私钥。

（2）用户团队：用户团队由多个共享数据的成员组成，这些成员同时作为云存储服务的数据提供方。在团队内部，负责向其他成员共享数据的用户被视为数据所有者。当数据所有者退出团队时，可将其数据所有权转移给团队内的其他指定成员。

（3）TPA：TPA是第三方审计人员，专门负责协助用户执行云端数据完整性的验证工作。

（4）CSP：云存储服务器作为半可信实体，承担着用户数据的存储管理、访问控制以及完整性保障等核心功能，同时需要为TPA的审计请求生成有效的完整性证明。

上述四个参与方会按照本文所提方案进行交互。首先，KGC将执行系统初始化操作。接着，用户团队中的用户成员会向KGC提交自己的用户身份并获取部分私钥，用户结合获取的部分私钥信息完成公钥和私钥的生成。当用户获得自己的公私钥对后，用户就可以将自己的文件生成相应的数据块标签，并将文件数据及标签发送给CSP，完成数据的上传任务。

在审计阶段，TPA会选取部分随机数作为挑战信息发送给CSP；CSP收到挑战信息后，会根据挑战信息中的数据，结合伪随机置换和伪随机函数，选取出相应的数据块和标签，生成本次挑战信息的证明并发送给TPA；TPA会将证明信息与挑战信息结合，验证证明信息是否有效。最终，TPA将审计结果反馈给用户。

4.2 算法定义

本方案由以下十个算法构成：

（1）$Setup(1^n)$：该算法用于初始化系统，KGC执行生成系统参数$params$和密钥生成中心的主密钥sk_{KGC}。

（2）$SetSecretValue$：用户使用该算法为自己生成相应的私密值s_{ID}和部分公钥ppk_{ID}。

（3）$ExtractPartialKey$：KGC执行该算法为身份为ID的每一个用户生成一个部分私钥psk_{ID}。

（4）$SetPrivateKey$：该算法让身份为ID的用户生成自己的私钥sk_{ID}。

（5）$SetPublicKey$：该算法让身份为ID的用户生成公钥pk_{ID}。

（6）$TrapdoorGen$：该算法用于生成数据拥有者DO和数据验证者DV的共用陷门δ。

（7）$TagGen$：数据拥有者可以执行该算法生成数据文件F对应的标签T。

（8）$Challenge$：第三方审计员也即是数据验证者，向云服务器发送挑战信息$chal$。

（9）*ProofGen*：云服务器根据挑战*chal*选择对应的数据块和标签，生成证明Φ,η并返回给数据验证者。

（10）*Verify*：数据验证者执行该算法用于检查云服务器是否给出有效的证明，若通过验证输出1，否则输出0。

4.3　安全模型

安全模型考虑了三个敌手，分别是A_{I}，A_{II}，A_{III}。敌手A_{I}和A_{II}都想伪造数据块的标签，不同之处在于，A_{I}扮演的是外部用户，A_{I}不能访问密钥生成中心的主密钥，但是能替换用户的公钥；A_{II}扮演的是密钥生成中心，能够获取KGC的主密钥，但是无法替换用户的公钥。敌手A_{III}扮演的是云存储服务器提供者，A_{III}采用伪造数据完整性证明的方式来欺骗第三方审计者，以期能通过验证。

通过挑战者C与三类敌手在安全实验中的交互行为来形式化定义所提出方案的安全性，三个游戏的定义如下：

Game Ⅰ：该游戏由挑战者C和敌手A_{I}进行交互。A_{I}是一个恶意的外部攻击者，尝试通过替换公钥来伪造数据块的标签。

初始化：C执行*Setup*算法进行初始化获取主密钥和公共参数，C会独自保存主密钥并把公共参数发送给A_{I}。

查询：A_{I}能自适应地向C进行下列查询。

（1）*SecretValueQuery*：A_{I}提交一个身份*ID*给C，C会执行*SetSecretValue*算法并把s_{ID}返回给A_{I}。

（2）*PartialPublicKeyQuery*：A_{I}提交一个身份*ID*给C，C会返回ppk_{ID}给A_{I}。

（3）*HashQuery*：敌手A_{I}会适应性地向C进行哈希查询，C会返回相应的哈希值给A_{I}。

（4）*PartialPrivateKeyQuery*：A_{I}会提交一个身份*ID*和部分公钥ppk_{ID}，C会执行*ExtractPartialKey*算法并把psk_{ID}返回给A_{I}。

（5）*PrivateKeyQuery*：A_{I}会提交一个有关身份*ID*的私钥查询，C会执行*SetPrivateKey*算法并把sk_{ID}返回给A_{I}。

（6）*PublicKeyQuery*：A_{I}会提交一个有关身份*ID*的公钥查询，C会执行*SetPublicKey*算法并把pk_{ID}返回给A_{I}。

（7）*PartialPublicKeyReplacement*：A_{I}可以替换用户*ID*的部分公钥。

（8）*PublicKeyReplacement*：A_{I}可以替换用户*ID*的公钥。

（9）*TagQuery*：A_{I}会提交(ID,ω,δ,m)给C，C会通过*TagGen*算法生成标签并返回给A_{I}。

伪造：最终A_{I}使用身份ID'、部分公钥$ppk_{ID'}$和公钥$pk_{ID'}$对数据块m'生成伪造的T'。如果下列条件满足，则A_{I}获胜：

（1）伪造的标签T'在使用身份ID'、部分公钥$ppk_{ID'}$和公钥$pk_{ID'}$的条件下对于数据块m'是有效的。

（2）A_{I}没有查询过用户身份ID'的私钥。

（3）A_{I}没有同时查询过用户身份ID'的部分私钥并替换公钥。

（4）A_{I}没有使用身份ID'、部分公钥$ppk_{ID'}$和公钥$pk_{ID'}$对于数据块m'进行标签生成查询。

Game Ⅱ：该游戏由挑战者C和敌手A_{II}进行交互。A_{II}是一个恶意的密钥生成中心，并尝试伪造数据块的标签。

初始化：C执行*Setup*算法进行初始化获取主密钥和公共参数。C会把主密钥和公共参数发送给A_{II}。

查询：A_{II}能自适应地向C进行下列查询。

（1）*SecretValueQuery*：A_{II}提交一个身份*ID*给C，C会执行*SetSecretValue*算法并把s_{ID}返回给A_{II}。

（2）*PartialPublicKeyQuery*：A_{II}提交一个身份*ID*给C，C会返回ppk_{ID}给A_{II}。

（3）*HashQuery*：敌手A_{II}会适应性地向C进行哈希查询，C会返回相应的哈希值给A_{II}。

（4）*PublicKeyQuery*：A_{II}会提交一个有关身份*ID*的私钥查询，C会执行*SetPublicKey*算法并把pk_{ID}返回给A_{II}。

（5）*TagQuery*：A_{II}会提交(ID,ω,δ,m)给C，C会通过*TagGen*算法生成标签并返回给A_{II}。

伪造：最终A_{II}使用身份ID'对数据块m'生成伪造标签T'。如果下列条件满足，则A_{II}获胜：

（1）伪造标签T'对身份ID'的数据块m'是有效的。

（2）A_{II}没有查询身份ID'的私密值$s_{ID'}$。

（3）A_{II}没有查询身份ID'的数据块m'的标签T'。

定义6. 对于两个任意概率多项式时间的敌手A_{I}和A_{II}，如果敌手A_{I}和A_{II}分别在游戏Game Ⅰ和游戏Game Ⅱ中获胜的概率是可忽略不计的，那么本文所提方案中数据块的标签是不可伪造的。

Game III：该游戏由挑战者C和敌手A_{III}进行交互。A_{III}是一个非诚实的云存储服务方，尝试伪造证明以此通过审计验证。

初始化：C生成了公共参数，主密钥和所有用户的部分私钥及私钥。C仅把公共参数发送给敌手A_{III}。

查询：

（1）*TagQuery*：A_{III}会适应性地选择(ID, m)，并将其发送给C，C会执行*TagGen*算法生成标签并返回给敌手A_{III}。

（2）*Challenge*：C会生成一个随机挑战，并把该挑战发送给A_{III}，敌手应该给出对应的有效数据证明。

伪造：对于该挑战，A_{III}生成对应的证明并将其发送给C，如果在数据块错误的情况下生成了能通过验证的证明，A_{III}获胜。

Game IV：该游戏与Game III基本一致，区别在于伪造阶段，A_{III}会生成对应的证明并将其发送给C，如果在使用非挑战数据块的情况下生成有效证明，则A_{III}获胜。

定义7. 对于任意概率多项式时间的敌手A_{III}，如果敌手A_{III}在游戏Game III和Game IV中获胜的概率是可忽略不计的，那么本文所提方案中的证明信息是不可伪造的，满足方案的可靠性。

5 方案构造

本节对面向团队场景的无证书云数据完整性授权审计方案进行了详细介绍，分别为具体审计方案、更换审计员方案以及用户变更方案。

5.1 具体审计方案

假定一个团队中有λ个用户，ID为用户u的身份标识，用ID_i表示用户$u_i(i\in[1,\lambda])$的唯一身份标识。设置u_1为团队管理员，团队管理员负责指定第三方审计机构TPA中的验证者，维护一份日志文件并管理组员的加入与退出。假定文件F被分成n块，可以表示为$F=(m_1, m_2, \cdots, m_n)$，其中$m_i\in Z_q^*(i\in[1,n])$。

系统成员公私钥生成及陷门生成流程如图2所示。图3展示了标签生成及完整审计流程。具体的方案构造细节如下：

（1）初始化$(1^n)\rightarrow(sk_{KGC}, params)$：给出安全参数$1^n$，该算法能生成系统参数$params$和密钥生成中心的主密钥$sk_{KGC}$。密钥生成中心会选择两个阶数为$q$的循环群$G_1$，$G_2$，一个安全的双线性映射$e$：$G_1\times G_1\rightarrow G_2$。此外，KGC会定义一个伪随机置换$\mu$：$Z_q^*\times\{1,2,\cdots,n\}\rightarrow\{1,2,\cdots,n\}$和一个伪随机函数$\pi$：$Z_q^*\times Z_q^*\rightarrow Z_q^*$。假定$g$是$G_1$的一个生成元，$H_1$、$H_2$和$H_3$是三个安全哈希函数，有$H_1, H_2, H_3$：$\{0,1\}^*\rightarrow G_1$。KGC可以从$Z_q^*$中随机选取一个$x$设置为主密钥$sk_{KGC}$，并计算公钥$pk_{KGC}=g^x$。

最后，密钥生成中心设置系统参数$params=(G_1, G_2, e, q, g, H_1, H_2, H_3, pk_{KGC}, \mu, \pi)$并发送给云服务器、用户和第三方审计员。

（2）设置私密值$\rightarrow(s_{ID}, ppk_{ID})$：该算法能够为身份为ID的用户生成自己的私密值。

用户随机地选取一个私密值$s_{ID}\in Z_q^*$并计算出部分公钥$ppk_{ID}=g^{s_{ID}}$。此外，通过运行该算法，身份

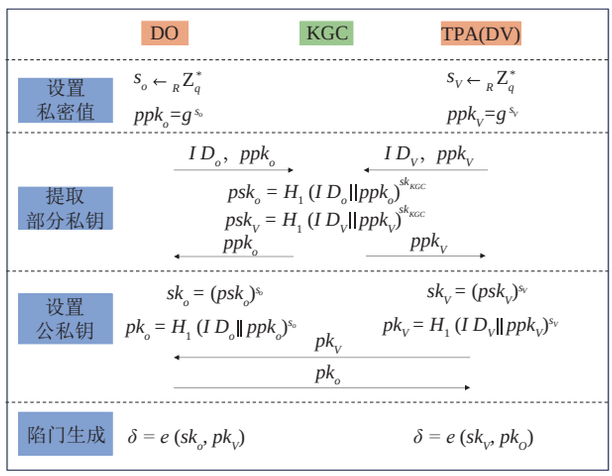

图2 系统成员公私钥生成及陷门生成流程

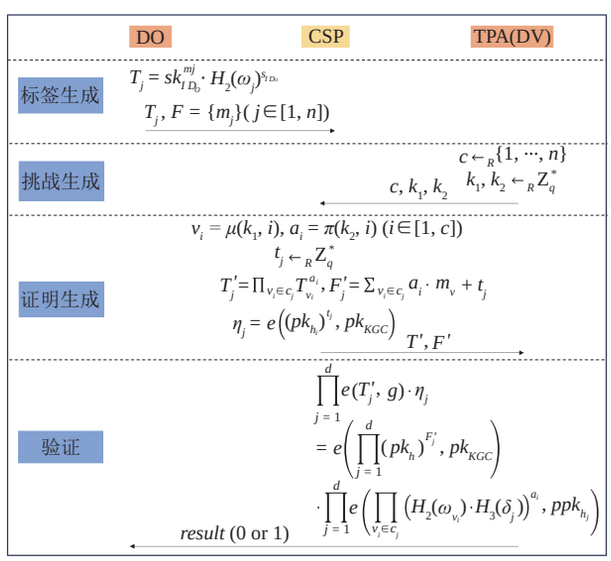

图3 标签生成及审计流程

为ID_o的数据拥有者DO能通过选取私密值s_O获取自己的部分公钥值$ppk_O=g^{s_O}$，身份为ID_V的数据验证者DV也能通过选取私密值s_V获取自己的部分公钥值$ppk_V=g^{s_V}$。

(3) 提取部分私钥$(ID,ppk_{ID},sk_{KGC})\rightarrow psk_{ID}$：密钥生成中心执行该算法为每一个用户生成一个部分私钥。计算出$k_{ID}=H_1(ID\|ppk_{ID})$，$psk=(k_{ID})^x$并把它们返回给用户。同样，DO和DV也可以通过密钥生成中心获取对应的部分私钥$psk_O=(k_O)^x$和$psk_V=(k_V)^x$。

(4) 设置私钥$(s_{ID},psk_{ID})\rightarrow sk_{ID}$：该算法能让身份为ID的用户生成自己的私钥。通过输入部分私钥psk_{ID}和秘密值s_{ID}计算出$sk_{ID}=(psk_{ID})^{s_{ID}}$。同样，$DO$和$DV$也可以通过执行该算法获取对应的私钥$sk_O=psk_O^{s_O}$和$sk_V=psk_V^{s_V}$。

(5) 设置公钥$(s_{ID},k_{ID})\rightarrow pk_{ID}$：该算法能让一个身份为$ID$的用户生成公钥。通过输入私密值$s_{ID}$和参数$k_{ID}$，能够计算出公钥$pk_{ID}=(k_{ID})^{s_{ID}}$。同样，$DO$和$DV$也可以通过执行该算法获取对应的公钥$pk_O=(k_O)^{s_O}$和$pk_V=(k_V)^{s_O}$。

(6) 陷门生成$(sk_O,pk_V)\rightarrow\delta$或者陷门生成$(sk_V,pk_O)\rightarrow\delta$：该算法用于生成$DO$和$DV$共用的陷门$\delta$。陷门的生成需要$DO$的私钥和$DV$的公钥，或者$DV$的私钥和$DO$的公钥。如果算法的输入为$(sk_O,pk_V)$，则有陷门$\delta$计算过程如下：

$$\begin{aligned}\delta:&=e(sk_O,pk_V)=\\&e(H_1(ID_O\|ppk_O)^{x\cdot s_O},H_1(ID_V\|ppk_V)^{s_V})=\\&e(H_1(ID_O\|ppk_O),H_1(ID_V\|ppk_V))^{x\cdot s_O\cdot s_V}\end{aligned}\tag{1}$$

同样，如果算法输入为(sk_V,pk_O)，该陷门δ计算过程如下：

$$\begin{aligned}\delta:&=e(sk_V,pk_O)=\\&e\left(H_1(ID_V\|ppk_V)^{x\cdot s_V},H_1(ID_O\|ppk_O)^{s_O}\right)=\\&e(H_1(ID_V\|ppk_V),H_1(ID_O\|ppk_O))^{x\cdot s_V\cdot s_O}\end{aligned}\tag{2}$$

DO和DV通过该算法生成了相同的共用陷门。

(7) 标签生成$(F,sk_D)\rightarrow T$：DO可以执行该算法生成数据文件F对应的标签T。

团队中的用户可以访问文件的数据块并用自己的私钥生成数据块所对应的标签。假定用户$u_i(i\in[1,z])$想生成块$m_j(j\in[1,n])$的标签，标签的计算等式为$T_j=sk_i^{m_j}\cdot H_2(\omega_j)^{s_i}\cdot H_3(\delta_i)^{s_i}$，其中，$\omega_j=F_{id}\|n\|j$，$F_{id}$表示文件的唯一身份。组管理员维护一份日志文件，该文件存储着$\omega_j(j\in[1,n])$，$H_3(\delta_i)$以及对应的标签生成者的身份。在更新或者修改数据块之后，用户应该添加或者更新日志文件中相关的$\omega_j(j\in[1,n])$，$H_3(\delta_i)$和标签生成者的身份。用户上传数据块和标签给云服务器之后，云服务器可以通过如下等式(3)验证标签的有效性：

$$\begin{aligned}e(T_j,g)=&e(pk_i^{m_j},pk_{KGC})\cdot\\&e(H_2(\omega_j)\cdot H_3(\delta_i),ppk_i)\end{aligned}\tag{3}$$

(8) 挑战生成$\rightarrow chal$：DV执行该算法向云服务器发起挑战。

DV随机选择一个挑战块数量$c\in[1,n]$和两个随机值k_1和k_2，其中，k_1和k_2分别作为伪随机置换μ和伪随机函数π的随机种子。验证者DV把$chal=(c,k_1,k_2)$作为挑战信息发送给云服务器。

(9) 证明生成$(chal,T,F)\rightarrow\Phi,\eta$：云服务器根据挑战$chal$选择对应的数据块和标签生成证明并返回给$DV$。

当云服务器收到挑战信息$chal=(c,k_1,k_2)$之后，计算出集合$C=\{(v_i,a_i)\}$，其中，$v_i=\mu(k_1,i)$，$a_i=\pi(k_2,i)$，$(i\in[1,c])$。云服务器按照每个挑战块对应标签的生成者把集合C划分成d个子集$C=\{C_1,C_2,\cdots,C_d\}(d\leqslant z)$，其中，每个子集都是由用户$u_{h_j}(j\in[1,d],h_j\in[1,\lambda])$生成的标签的集合。用$c_j$表示每个子集$C_j$的数量，则有$c=\sum_{j=1}^{d}c_j$，$C=C_1\cup C_2\cdots\cup C_d$，$C_j\cap C_{j'}=\varnothing$，$(j\neq j')$。云服务器可以计算出每个子集$C_j$的证明$(T_j',F_j')$和辅助参数$\eta_j$，其中$T_j'=\prod_{v_i\in C_j}T_{v_i}^{a_i}$，$F_j'=\sum_{v_i\in C_j}a_i\cdot m_{v_i}+t_j$，$\eta_j=e\left((pk_{h_i})^{t_j},pk_{KGC}\right)$，$(t_j\in Z_q^*)$。最终，云服务器把证明$\Phi=(T',F')$和$\eta$返回给$DV$，其中$T'=\{T_1',\cdots,T_d'\}$，$F'=\{F_1',\cdots,F_d'\}$，$\eta=\{\eta_1,\cdots,\eta_d\}$。

(10) 验证$(\delta,chal,\Phi,\eta)\rightarrow 0/1$：$DV$执行该算法用于检查云服务器返回证明$\Phi$的正确性。

从云服务器处接收证明Φ后，DV计算出集合$C=\{(v_i,a_i)\}$，其中，$v_i=\mu(k_1,i)$，$a_i=\pi(k_2,i)$，$(i\in[1,c])$。然后检查下列等式：

$$\prod_{j=1}^{d} e(T_j', g) \cdot \eta_j = e\left(\prod_{j=1}^{d} (pk_{h_j})^{F_j'}, pk_{KGC}\right) \cdot \prod_{j=1}^{d} e\left(\prod_{v_i \in C_j} (H_2(\omega_{v_i}) \cdot H_3(\delta_j))^{a_i}, ppk_{h_j}\right) \quad (4)$$

如果等式(4)成立，则输出1，否则输出0。

5.2 更换审计员方案

当用户需要更换审计员时，数据块对应标签中的陷门信息需要更新，否则，这些标签的有效性和数据块的完整性将无法被更换后的审计员验证。该方案将标签的更新操作外包给云服务器，能实现在不泄露双方陷门和私钥的情况下完成标签更新。假定用户 $u_j(1 \leqslant j \leqslant \lambda)$ 需要更新审计员，通过使用审计员信息变更算法来更新标签，并替换为新的审计员，整个交互过程使用安全通道进行数据传输。图4展示了更换审计员流程。

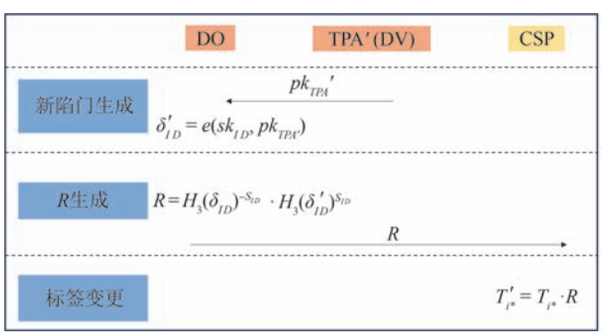

图4 更换审计员流程

审计员信息变更：该算法需要 $u_j(1 \leqslant j \leqslant \lambda)$ 与云服务器进行交互。

(1) 用户 u_j 重新选取审计员 TPA'，并计算 $\delta_j' = e(sk_j, pk_{TPA'})$；

(2) 用户 u_j 计算 $R = H_3(\delta_j)^{-s_j} \cdot H_3(\delta_j')^{s_j}$，然后把 R 发送给云服务器；

云服务器将对标签 T_{i^*} 进行更新，过程如下：

$$T_{i^*}' = T_{i^*} \cdot R = T_{i^*} \cdot H_3(\delta_j)^{-s_j} \cdot H_3(\delta_j')^{s_j} = sk_j^{m_{i^*}} \cdot H_2(\omega_i)^{s_j} \cdot H_3(\delta_j)^{s_j} \cdot H_3(\delta_j)^{-s_j} \cdot H_3(\delta_j')^{s_j} = sk_j^{m_{i^*}} \cdot H_2(\omega_i)^{s_j} \cdot H_3(\delta_j')^{s_j}。$$

其中，T_{i^*}' 是用户 u_j 对数据块 m_{i^*} 更换审计员后的有效标签。

5.3 用户变更方案

如果任意用户 $u_i(2 \leqslant i \leqslant \lambda)$ 离开该用户团队，则用户 u_i 的密钥信息就会失效。因此，用户 u_i 的密钥信息应该从数据块对应的标签中被移除，否则，这些标签的有效性和数据的完整性将无法被验证。传统的标签更新操作会将数据块下载后重新生成标签，再进行上传，这显著增加了通信耗费和用户端的计算资源和时间耗费。为了减少这些开销，该方案设计将标签的更新操作外包给云服务器，在不泄露私钥的情况下完成标签的更新，从而大幅降低通信消耗、用户端计算资源和时间的浪费。通过使用用户信息变更算法来更新离开团队的用户数据标签，假定 u_i 是离开的用户，$u_j(1 \leqslant j \leqslant \lambda, j \neq i)$ 是团队中另外一个有效的用户。考虑到 u_1 是用户组管理员，u_1 不会离开该用户团队。图5展示了用户撤销流程。

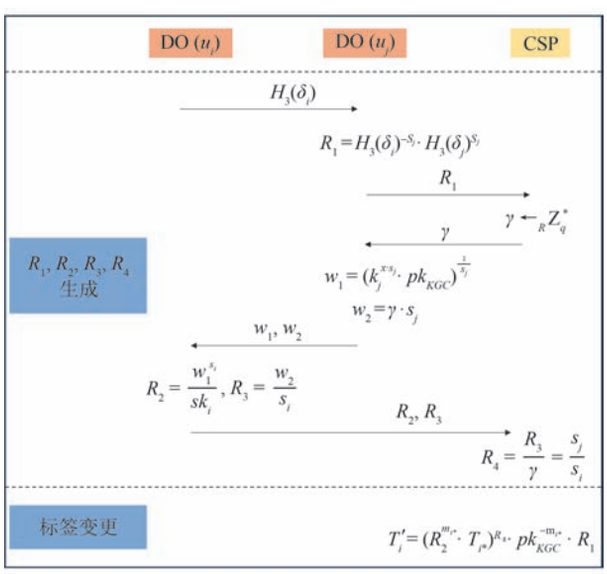

图5 用户撤销流程

用户信息变更：该算法需要 u_i，u_j 和云服务器这三者进行交互，假定交互中都使用了安全通道。

(1) u_i 将 $H_3(\delta_i)$ 发送给 u_j；

(2) u_j 计算 $R_1 = H_3(\delta_i)^{-s_j} \cdot H_3(\delta_j)^{s_j}$，并将 R_1 发送给云服务器；

(3) 云服务器随机选择一个值 $\gamma \in Z_q^*$，并把 γ 发送给 u_j；

(4) u_j 计算 $w_1 = \left(k_j^{x \cdot s_j} \cdot pk_{KGC}\right)^{\frac{1}{s_j}}$，$w_2 = \gamma \cdot s_j$，接着把 (w_1, w_2) 发送给 u_i；

(5) u_i 计算 $R_2 = \frac{w_1^{s_i}}{sk_i}$，$R_3 = \frac{w_2}{s_i}$，接着把 R_2、R_3 发送给云服务器；

(6) 当云服务器收到 (R_2, R_3) 后，计算 $R_4 = \frac{R_3}{\gamma} = \frac{s_j}{s_i}$；

此时，用户u_i，u_j可以离线。云服务器将对标签T_{i^*}进行更新，过程如下：

$$T'_{i^*}=\left(R_2^{m_{i^*}}\cdot T_{i^*}\right)^{R_4}\cdot pk_{KGC}^{-m_{i^*}}\cdot R_1=$$

$$\left(\left(\frac{\left(k_j^{x\cdot s_j}\cdot pk_{KGC}\right)^{\frac{s_i}{s_j}}}{k_i^{x\cdot s_i}}\right)^{m_{i^*}}\cdot k_i^{x\cdot s_i\cdot m_{i^*}}\cdot H_2\left(\omega_{i^*}\right)^{s_i}\cdot H_3\left(\delta_i\right)^{s_i}\right)^{\frac{s_j}{s_i}}\cdot$$

$$pk_{KGC}^{-m_{i^*}}\cdot R_1=\left(\left(k_j^{x\cdot s_j}\cdot pk_{KGC}\right)^{m_{i^*}}\cdot H_2\left(\omega_{i^*}\right)^{s_j}\cdot H_3\left(\delta_i\right)^{s_j}\right)\cdot$$

$$pk_{KGC}^{-m_{i^*}}\cdot R_1=k_j^{x\cdot s_j\cdot m_{i^*}}\cdot H_2\left(\omega_{i^*}\right)^{s_j}\cdot H_3\left(\delta_i\right)^{s_j}\cdot R_1=$$

$$k_j^{x\cdot s_j\cdot m_{i^*}}\cdot H_2\left(\omega_{i^*}\right)^{s_j}\cdot H_3\left(\delta_i\right)^{s_j}\cdot H_3\left(\delta_i\right)^{-s_j}\cdot H_3\left(\delta_j\right)^{s_j}=$$

$$sk_j^{m_{i^*}}\cdot H_2\left(\omega_{i^*}\right)^{s_j}\cdot H_3\left(\delta_j\right)^{s_j}。$$

其中，T'_{i^*}是用户u_j的数据块m_{i^*}的有效标签。

6　安全性分析

本节对所提审计方案进行了相应的安全性分析，分别为正确性、数据标签的抗伪造性、证明信息的抗伪造性、陷门的抗伪造性和隐私保护性五个部分。

6.1　正确性

定理1.　本文方案应满足以下正确性：

(1)标签的正确性：如果用户生成的数据块标签是有效的，则标签可以通过CSP的验证；

(2)审计的正确性：如果CSP生成的证明是有效的，则证明可以通过TPA的验证。

证明.

(1)标签的正确性。在*TagGen*算法中的验证等式(3)证明如下：

$$e\left(T_j,g\right)=e\left(sk_i^{m_j}\cdot H_2\left(\omega_j\right)^{s_i}\cdot H_3\left(\delta_i\right)^{s_i},g\right)=$$

$$e\left(sk_i^{m_j},g\right)\cdot e\left(H_2\left(\omega_j\right)^{s_i}\cdot H_3\left(\delta_i\right)^{s_i},g\right)=$$

$$e\left(pk_i^{m_j},g^x\right)\cdot e\left(H_2\left(\omega_j\right)\cdot H_3\left(\delta_i\right),g^{s_i}\right)=$$

$$e\left(pk_i^{m_j},pk_{KGC}\right)\cdot e\left(H_2\left(\omega_j\right)\cdot H_3\left(\delta_i\right),ppk_i\right)。$$

(2)审计的正确性。在*Verify*算法中的验证等式(4)证明如下：

$$\prod_{j=1}^{d}e\left(T'_j,g\right)\cdot\eta_j=\prod_{j=1}^{d}e\left(\prod_{v_i\in C_j}T_{v_i}^{a_i},g\right)\cdot\eta_j=$$

$$\prod_{j=1}^{d}e\left(\prod_{v_i\in C_j}\left(H_1\left(ID_{h_j}\|ppk_{h_j}\right)^{x\cdot s_{h_j}\cdot m_{v_i}\cdot a_i}\cdot\left(H_2\left(\omega_{v_i}\right)\cdot H_3\left(\delta_j\right)\right)^{s_{h_j}\cdot a_i}\right),g\right)\cdot\eta_j=$$

$$\prod_{j=1}^{d}\left(e\left(\prod_{v_i\in C_j}H_1\left(ID_{h_j}\|ppk_{h_j}\right)^{s_{h_j}\cdot m_{v_i}\cdot a_i},g^x\right)\cdot e\left(\prod_{v_i\in C_j}\left(H_2\left(\omega_{v_i}\right)\cdot H_3\left(\delta_j\right)\right)^{a_i},g^{s_{h_j}}\right)\right)\cdot\eta_j=$$

$$\prod_{j=1}^{d}\left(e\left(\prod_{v_i\in C_j}pk_{h_j}^{m_{v_i}\cdot a_i},pk_{KGC}\right)\cdot e\left(\prod_{v_i\in C_j}\left(H_2\left(\omega_{v_i}\right)\cdot H_3\left(\delta_j\right)\right)^{a_i},g^{s_{h_j}}\right)\right)\cdot\prod_{j=1}^{d}e\left(pk_{h_j}^{t_j},pk_{KGC}\right)=$$

$$e\left(\prod_{j=1}^{d}pk_{h_j}^{\sum_{v_i\in C_j}m_{v_i}\cdot a_i},pk_{KGC}\right)\cdot\prod_{j=1}^{d}e\left(pk_{h_j}^{t_j},pk_{KGC}\right)\cdot\prod_{j=1}^{d}e\left(\prod_{v_i\in C_j}\left(H_2\left(\omega_{v_i}\right)\cdot H_3\left(\delta_j\right)\right)^{a_i},ppk_{h_j}\right)=$$

$$e\left(\prod_{j=1}^{d}pk_{h_j}^{\sum_{v_i\in C_j}m_{v_i}\cdot a_i+t_j},pk_{KGC}\right)\cdot\prod_{j=1}^{d}e\left(\prod_{v_i\in C_j}\left(H_2\left(\omega_{v_i}\right)\cdot H_3\left(\delta_j\right)\right)^{a_i},ppk_{h_j}\right)=$$

$$e\left(\prod_{j=1}^{d}pk_{h_j}^{F'_j},pk_{KGC}\right)\cdot\prod_{j=1}^{d}e\left(\prod_{v_i\in C_j}\left(H_2\left(\omega_{v_i}\right)\cdot H_3\left(\delta_j\right)\right)^{a_i},ppk_{h_j}\right)。$$

6.2　数据标签的抗伪造性

定理2.　如果一个概率多项式时间的敌手A_I对*SecretValueQuery*、*PartialPublicKeyQuery*、H_1*Query*、*PartialPrivateKeyQuery*、H_2*Query*、*PrivateKeyQuery*、*PublicKeyQuery*、H_3*Query*、*PartialPublicKeyReplacement*、*PublicKeyReplacement*和*TagQuery*分别进行至多q_{sv}、q_{ppk}、q_{h1}、q_{psk}、q_{h2}、q_{sk}、q_{pk}、q_{h3}、q_{ppkr}、q_{pkr}和q_{tag}次查询后，在时间t内能以不可忽略的概率ε赢得Game I，那么就能构造一个模拟器B以概率$\varepsilon'\geqslant\frac{\varepsilon}{\left(q_{psk}+q_{sk}+q_{pk}+q_{tag}\right)\cdot 2e}$在时间$t'\leqslant t+O(q_{sv}+q_{ppk}+q_{h1}+q_{psk}+q_{sk}+q_{pk}+q_{pkr}+q_{ppkr}+q_{h2}+q_{h3}+q_{tag})$内攻破CDH难题。

证明.

给出CDH的一个实例$\left(g,g^a,g^b,G_1\right)$。如果敌手$A_1$能以不可忽略的概率$\varepsilon$赢得Game I，模拟器B就能借助敌手$A_1$的能力以不可忽略的概率计算出$g^{ab}$。B和$A_1$的交互过程如下：

(1)**初始化**：模拟器B生成公共参数，并设置主公钥的值pk_{KGC}为g^a，主密钥被隐式地设置为未知值a。

(2)**查询阶段**：敌手会适应性地向模拟器进行查询，查询过程如下：

SecretValueQuery：模拟器B保存着一个列表

$L_1=\{(ID,ppk_{ID},s_{ID})\}$，B会检查$ID^*$是否位于$L_1$中。如果$ID^*$不在$L_1$中，B随机选取一个值$x\in Z_q^*$并设置$s_{ID^*}=x$，$ppk_{ID^*}=g^x$，最终，B会把新元组$(ID^*,ppk_{ID^*},s_{ID^*})$放入$L_1$中，返回$s_{ID^*}$给敌手$A_1$。如果$ID^*$在$L_1$中，B检查是否$s_{ID^*}=\perp$。如果$s_{ID^*}=\perp$，B随机选取一个值$x\in Z_q^*$并设置$s_{ID^*}=x$，$ppk_{ID^*}=g^x$，B将$s_{ID^*}$和$ppk_{ID^*}$的值更新到$L_1$中。如果$s_{ID^*}\neq\perp$，B会检索出$s_{ID^*}$并返回给敌手$A_1$。

PartialPublicKeyQuery：敌手A_I可以对身份为ID^*的用户进行部分公钥查询。如果ID^*在L_1中，B会检查对应元组中ppk_{ID^*}的值。如果$ppk_{ID^*}=\perp$，B会执行*SecretValueQuery*，接着从L_1中查询ppk_{ID^*}并返回给A_1。如果$ppk_{ID^*}\neq\perp$，B会把ppk_{ID^*}返回给A_1。如果ID^*不在L_1中，B会*SecretValueQuery*，接着从L_1中查询ppk_{ID^*}并返回给A_1。

H_1*Query*：B维护一个列表$L_{h1}=\{(ID,ppk_{ID},h_1,Q,Y)\}$，其中$Q\in G_1$为返回的哈希值，$Y\in\{0,1\}$为抛一枚硬币所得的随机数。如果$(ID^*,ppk_{ID^*})$存在于$L_{h1}$中，检索并返回$Q^*$给$A_1$。如果$(ID^*,ppk_{ID^*})$不存在于$L_{h1}$中，B随机选取一个$h_1^*\in Z_q^*$，并抛掷一枚硬币得到$Y^*\in\{0,1\}$，假定$Y^*=0$的概率是$\eta$，则$Y^*=1$的概率是$1-\eta$。如果$Y^*=0$，设置$Q^*=g^{h_1^*s_{ID}^{-1}}$；如果$Y^*=1$，设置$Q^*=(g^b)^{h_1^*s_{ID}^{-1}}$。B把元组$(ID^*,ppk_{ID^*},h_1^*,Q^*,Y^*)$放入$L_{h1}$中，返回$Q^*$给$A_1$。

PartialPrivateKeyQuery：模拟器B维护列表$L_2=\{(ID,pk_{ID},psk_{ID},sk_{ID})\}$，敌手$A_1$对任意$ID^*$进行部分私钥查询。模拟器B首先从$L_1$中获取$ID^*$对应的$ppk_{ID^*}$，若不存在则执行*PartialPublicKeyQuery*，然后从$L_{h1}=\{(ID,ppk_{ID},h_1,Q,Y)\}$中获取$(ID^*,ppk_{ID^*})$对应的$Q^*$。如果$L_{h1}$中不存在$(ID^*,ppk_{ID^*})$，B会执行$H_1$*Query*查询。如果$Y^*=1$，B停机；否则$Y^*=0$，则正常获取$Q^*=g^{h_1^*s_{ID}^{-1}}$。

模拟器B将会检查ID^*在列表$L_2=\{(ID,pk_{ID},psk_{ID},sk_{ID})\}$中是否存在。如果$ID^*$在$L_2$中存在，B会检查$psk_{ID^*}$的值。若$psk_{ID^*}=\perp$，B计算$psk_{ID^*}=(Q^*)^{sk_{KGC}}=(Q^*)^a=(g^a)^{h_1^*s_{ID^*}^{-1}}$，并将$psk_{ID^*}$更新于$L_2$中；若$psk_{ID^*}\neq\perp$，B直接返回$psk_{ID^*}$给敌手$A_1$。如果$ID^*$不在$L_2$中，B会计算$psk_{ID^*}=(Q^*)^{sk_{KGC}}=(Q^*)^a=(g^a)^{h_1^*s_{ID^*}^{-1}}$，并将元组$(ID^*,\perp,psk_{ID^*},\perp)$放入$L_2$中。

PrivateKeyQuery：A_1对任意ID^*进行私钥查询。B检查ID^*在$L_2=\{(ID,pk_{ID},psk_{ID},sk_{ID})\}$中是否存在。如果$ID^*$在$L_2$中存在，B会检查$sk_{ID^*}$的值。若$sk_{ID^*}=\perp$，B会执行*PartialPrivateKeyQuery*，如果B未停机，则检索$psk_{ID^*}=(Q^*)^{sk_{KGC}}$，计算$sk_{ID^*}=(Q^*)^{sk_{KGC}\cdot s_{ID^*}}=g^{ah_1^*}$，将$sk_{ID^*}$更新于$L_2$中，并把$sk_{ID^*}$返回给$A_1$；否则$sk_{ID^*}\neq\perp$，B返回$sk_{ID^*}$给敌手$A_1$。如果$ID^*$不在$L_2$中存在，B执行*PartialPrivateKeyQuery*，如果B未停机，则检索$psk_{ID^*}=(Q^*)^{sk_{KGC}}$，计算$sk_{ID^*}=(Q^*)^{sk_{KGC}\cdot s_{ID^*}}=g^{ah_1^*}$，将$sk_{ID^*}$更新于$L_2$中，并把$sk_{ID^*}$返回给$A_1$。

PublicKeyQuery：敌手A_1对任意ID^*进行公钥查询。B检查ID^*在$L_2=\{(ID,pk_{ID},psk_{ID},sk_{ID})\}$中是否存在。如果$ID^*$在$L_2$中存在，B会检查$pk_{ID^*}$的值。若$pk_{ID^*}=\perp$，B会执行*PartialPublicKeyQuery*获取ppk_{ID^*}。接着执行H_1*Query*算法，若$Y^*=1$，B会停机；否则$Y^*=0$，则正常获取$Q^*=g^{h_1^*s_{ID}^{-1}}$，模拟器B检索L_1获取s_{ID^*}，计算$pk_{ID^*}=(Q^*)^{s_{ID}}=g^{h_1}$并返回给$A_1$。若$pk_{ID^*}\neq\perp$，模拟器B返回$pk_{ID^*}$给敌手$A_1$。如果$ID^*$不在$L_2$中，B会执行*PartialPublicKeyQuery*获取ppk_{ID^*}，接着执行H_1*Query*算法，若$Y^*=1$，B会停机；否则$Y^*=0$，则正常获取$Q^*=g^{h_1^*s_{ID}^{-1}}$，模拟器B检索L_1获取s_{ID^*}，计算$pk_{ID^*}=(Q^*)^{s_{ID}}=g^{h_1}$并返回给$A_1$。

PartialPublicKeyReplacecment：A_I可以用(ID^*,ppk'_{ID^*})替换用户部分公钥。如果$(ID^*,ppk_{ID^*},s_{ID^*})$存在于$L_1$中，B更新元组为$(ID^*,ppk'_{ID^*},\perp)$。如果$(ID^*,ppk_{ID^*},s_{ID^*})$不在$L_1$中，B添加元组$(ID^*,ppk'_{ID^*},\perp)$到$L_1$中。如果$(ID^*,pk_{ID^*},psk_{ID^*},sk_{ID^*})$存在于$L_2$中，B更新元组为$(ID^*,\perp,\perp,\perp)$。

PublicKeyReplacement：A_1用(ID^*,pk'_{ID^*})替换用户公钥。如果$(ID^*,pk_{ID^*},psk_{ID^*},sk_{ID^*})$存在$L_2$中，B更新元组为$(ID^*,pk_{ID^*},psk_{ID^*},\perp)$。如果$(ID^*,pk_{ID^*},psk_{ID^*},sk_{ID^*})$不在$L_2$中，B添加元组$(ID^*,pk_{ID^*},\perp,\perp)$到$L_2$中。

H_2*Query*：A_1会用ω^*进行哈希查询，B保存列表$L_{h2}=\{(\omega,h_2)\}$。如果ω^*位于列表L_{h2}中，查询h_2^*并返回$g^{h_2^*}$给A_1。如果ω^*不在列表L_{h2}中，B会选取

一个随机值$h_2^*\in Z_q^*$，同时把元组(ω^*,h_2^*)记录到列表L_{h2}中，最终返回$g^{h_2^*}$给A_{I}。

H_3Query：A_{I}会用δ^*进行哈希查询，B保存列表$L_{h3}=\{(\delta,h_3)\}$。如果δ^*位于列表L_{h3}中，查询h_3^*并返回$g^{h_3^*}$给A_{I}。如果δ^*不在列表L_{h3}中，模拟器B会选取一个随机值$h_3^*\in Z_q^*$，同时把元组(δ^*,h_3^*)记录到列表L_{h3}中，最终返回$g^{h_3^*}$给A_{I}。

$TagQuery$：A_{I}使用$(ID^*,\omega^*,\delta^*,m^*)$进行适应性标签查询。如果$L_1$中发现$Y^*=1$，模拟器B停机。如果$L_1$中$Y^*=0$，B从列表$L_{h2}$中获取$h_2^*$，从列表$L_1$和$L_2$中分别获取$sk_{ID^*}$和$s_{ID^*}$，通过$TagGen$算法计算出标签并返回给$A_{\mathrm{I}}$。

(3)**伪造阶段**：敌手A_{I}输出$(Tag',\omega',\delta',m',ID',pk'_{ID},ppk'_{ID})$，其中，$Tag'$是对身份为$ID'$的数据块$m'$进行标签伪造的结果。

(4)**分析**：如果敌手A_{I}赢得了Game I，模拟器B能获得：

$$e(Tag',g)=e\left(\left(pk'_{ID}\right)^{m'},pk_{KGC}\right)\cdot e\left(H_2(\omega')\cdot H_3(\delta'),ppk'_{ID}\right)\tag{5}$$

模拟器B从L_{h1}中检索元组$(ID',ppk_{ID'},h'_1,Q',Y')$，如果$Y'=0$，模拟器B停机。否则，B从$L_{h1}$中检索获取$H_1(ID\|ppk_{ID})=g^{bh'_1s_{ID}^{-1}}$，从$L_{h2}$中检索获取$H_2(\omega')=g^{h'_2}$。

根据上述提到的验证等式(5)，B可以获得

$$e(Tag',g)=e\left(\left(g^{bh'_1}\right)^{m'},g^a\right)\cdot e\left(g^{h'_2}\cdot g^{h'_3},g^{s_{ID'}}\right)=e\left(g^{abh'_1m'}\cdot g^{(h'_2+h'_3)s_{ID'}},g\right)。$$

因此，可以计算出$g^{ab}=\left(\dfrac{Tag'}{\left(ppk'_{ID}\right)^{(h'_2+h'_3)}}\right)^{\frac{1}{h'_1\cdot m'}}$。

如果模拟器B在与敌手A_{I}的交互过程中没有终止运行，那么模拟器与敌手进行了完美的交互模拟过程。模拟器在$PartialPrivateKeyQuery$、$PrivateKeyQuery$、$PublicKeyQuery$和$TagQuery$这四个查询过程中可能会发生终止，而对于H_1Query、$SecretValueQuery$、$PartialPublicKeyQuery$、$PublicKeyReplacement$、H_2Query、H_3Query和$PartialPublicKeyReplacecment$这几个查询会正常执行。因此，B会和敌手$A_{\mathrm{I}}$进行完美交互模拟的概率大于$\eta^{q_{psk}+q_{sk}+q_{pk}+q_{tag}}$。B正确输出$g^{ab}$的概率为$\varepsilon'\geqslant\varepsilon\cdot(1-\eta)\cdot\eta^{q_{psk}+q_{sk}+q_{pk}+q_{tag}}\geqslant\dfrac{\varepsilon}{(q_{psk}+q_{sk}+q_{pk}+q_{tag})\cdot 2e}$，相应的执行时间为$t'\leqslant t+O(q_{sv}+q_{ppk}+q_{h1}+q_{psk}+q_{sk}+q_{pk}+q_{pkr}+q_{ppkr}+q_{h2}+q_{h3}+q_{tag})$。

定理3.　如果概率多项式时间的敌手A_{II}对H_1Query、$SecretValueQuery$、$PartialPublicKeyQuery$、$PublicKeyQuery$、H_2Query、H_3Query和$TagQuery$分别进行至多q_{h1}、q_{sv}、q_{ppk}、q_{pk}、q_{h2}、q_{h3}和q_{tag}次查询后，在时间t内以不可忽略的概率ε赢得Game II，那么就可以构造一个模拟器B以概率$\varepsilon'\geqslant\dfrac{\varepsilon}{(q_{sv}+q_{tag})\cdot 2e}$在时间$t'\leqslant t+O(q_{sv}+q_{ppk}+q_{h1}+q_{pk}+q_{h2}+q_{h3}+q_{tag})$内攻破CDH难题。

证明.　给出CDH的一个实例(g,G_1,g^a,g^b)。如果敌手A_{II}能以不可忽略的概率ε赢得Game II，模拟器B就能借助敌手A_{II}的能力以不可忽略的概率计算出g^{ab}。B和A_{II}的交互过程如下：

(1)**初始化**：模拟器B随机挑选一个值$s\in Z_q^*$作为主密钥并生成公共参数，B把主密钥和公共参数发送给A_{II}。

(2)**查询阶段**：敌手A_{II}会适应性地向模拟器B进行查询，查询过程如下：

$SecretValueQuery$：模拟器B保存着一个列表$L_1=\{(ID,ppk_{ID},s_{ID},Y)\}$，B会检查$ID^*$是否位于$L_1$中，其中，$Y\in\{0,1\}$为抛一枚硬币所得的随机数。如果$ID^*$在列表$L_1$中，B检查是否有$Y^*=1$。如果$Y^*=1$，B会输出失败并停机；否则$Y^*=0$，B会检索出$s_{ID}^*$并返回给敌手$A_{\mathrm{II}}$。

如果ID^*不在列表L_1中，B随机选取一个值$x\in Z_q^*$并设置$s_{ID}^*=x$，B抛掷一枚硬币得到$Y^*\in\{0,1\}$，假定$Y^*=0$的概率是η，则$Y^*=1$的概率是$1-\eta$。如果$Y^*=1$，B将会停机；如果$Y^*=0$，B计算$ppk_{ID^*}=g^x$并把新元组$(ID^*,ppk_{ID^*},s_{ID^*},Y^*)$放入$L_1$中。最终，返回$s_{ID^*}$给敌手$A_{\mathrm{II}}$。

$PartialPublicKeyQuery$：敌手A_{II}对于任意身份ID^*进行适应性查询。如果元组$(ID^*,ppk_{ID^*},s_{ID^*},Y^*)$位于列表$L_1$中，模拟器B会给敌手$A_{\mathrm{II}}$返回$ppk_{ID^*}$。如果列表$L_1$不包含元组$(ID^*,ppk_{ID^*},s_{ID^*},Y^*)$，模拟器B会随机选取一个值$x\in Z_q^*$并抛掷一枚硬币$Y^*\in\{0,1\}$。如果$Y^*=0$，模拟器B设置$s_{ID^*}=x$，$ppk_{ID^*}=g^x$；如果$Y^*=1$，模拟器B设置$ppk_{ID^*}=$

$(g^a)^x$。B把新元组$(ID^*, ppk_{ID^*}, s_{ID^*}, Y^*)$放入列表$L_1$中并给敌手$A_{\mathrm{II}}$返回$ppk_{ID^*}$。

H_1Query:敌手A_{II}对任意(ID^*, ppk_{ID^*})进行适应性查询。模拟器B维护一个列表$L_{h1}=\{(ID, ppk_{ID}, h_1)\}$。如果$(ID^*, ppk_{ID^*})$位于$L_{h1}$中,模拟器B会检索出$(ID^*, ppk_{ID^*}, h_1^*)$并给敌手$A_{\mathrm{II}}$返回$Q^*=g^{h_1^*}$。否则,模拟器B会选择一个随机值$h_1^*\in Z_q^*$并将元组$(ID^*, ppk_{ID^*}, h_1^*)$放入列表$L_{h1}$中,最终计算出$Q^*=g^{h_1^*}$并返回给敌手$A_{\mathrm{II}}$。

$PublicKeyQuery$:敌手A_{II}对于任意身份ID^*适应性地进行公钥查询。模拟器B维护一个列表$L_2=\{(ID, pk_{ID})\}$。如果ID^*在列表L_2中,直接返回pk_{ID^*}给敌手A_{II}。如果ID^*不在列表L_2中,模拟器B会执行$PartialPublicKeyQuery$,从列表L_{h1}中检索出h_1^*,若h_1^*不存在,会选择一个随机值$h_1^*\in Z_q^*$并将元组$(ID^*, ppk_{ID^*}, h_1^*)$放入列表$L_{h1}$中。

接下来,B会检查是否$Y^*=0$。若$Y^*=0$,B会从列表L_1中检索出s_{ID^*},最终计算出$pk_{ID^*}=(g^{h_1})^{s_{ID^*}}=(Q^*)^{s_{ID^*}}$,将元组$(ID^*, pk_{ID^*})$放入列表$L_2$中并返回$pk_{ID^*}$。否则$Y^*=1$,B会从列表$L_1$中检索出$s_{ID^*}$,最终计算出$pk_{ID^*}=(g^a)^{h_1\cdot s_{ID^*}}=(Q^*)^{a\cdot s_{ID^*}}$,将元组$(ID^*, pk_{ID^*})$放入列表$L_2$中并返回$pk_{ID^*}$给敌手$A_{\mathrm{II}}$。

H_2Query:A_{II}会用ω^*进行哈希查询,B维护一个列表$L_{h2}=\{(\omega, h_2)\}$。如果ω^*位于列表L_{h2}中,查询h_2^*并返回$(g^b)^{h_2^*}$给敌手A_{II}。如果ω^*不在列表L_{h2}中,B会选取$h_2^*\in Z_q^*$,同时把元组(ω^*, h_2^*)记录到列表L_{h2}中,最终返回$(g^b)^{h_2^*}$给敌手A_{II}。

H_3Query:A_{II}会用δ^*进行哈希查询,B维护一个列表$L_{h3}=\{(\delta, h_2)\}$。如果δ^*不在列表L_{h3}中,B会选取$h_3^*\in Z_q^*$,同时把元组(ω^*, h_3^*)记录到列表L_{h3}中,最终返回$g^{h_3^*}$给敌手A_{II}。

$TagQuery$:A_{II}适应性地用元组$(ID^*, m^*, \delta^*, \omega^*)$执行标签查询。模拟器B首先会检查列表$L_2$中对于$ID^*$所在元组的$Y^*$是否为0。如果$Y^*=1$,模拟器B会输出失败并停机。否则,模拟器B会从列表L_{h2}中获取h_2^*,从列表L_1和L_2中分别获取s_{ID^*}和pk_{ID^*},模拟器B通过$SetPrivateKey$算法计算出sk_{ID^*},通过$TagGen$算法计算出标签并返回给敌手A_{II}。

(3)**伪造阶段**:敌手A_{II}输出(Tag', ω', m', ID'),其中,Tag'是对身份为ID'的数据块m'进行伪造标签的结果。

(4)**分析**:如果敌手A_{II}赢得了Game II,模拟器B能获得等式:

$$e(Tag', g)=e(pk_{ID'}{}^{m'}, pk_{KGC})\cdot e(H_2(\omega')\cdot H_3(\delta'), ppk_{ID'}) \tag{6}$$

B会从L_1中查找出$(ID', ppk_{ID'}, s_{ID'}, Y')$,如果$Y^*=0$,模拟器B输出失败并停机;否则$Y^*=1$,B从$L_1$中获取$ppk_{ID'}=g^{a\cdot s_{ID'}}$,从$L_2$中获取$pk_{ID'}=(g^a)^{h_1'\cdot s_{ID'}}$,从$L_{h2}$中获取$h_2'$并计算出$H_2(\omega')=g^{b\cdot h_2'}$。根据上述提到的验证等式(6),有$e(Tag', g)=e\left((pk_{ID'})^{m'}, g^s\right)\cdot e\left(g^{b\cdot(h_2'+h_3')}, g^{a\cdot s_{ID'}}\right)$。

最终,模拟器B可以计算出

$$g^{ab}=\left(\frac{Tag'}{(pk_{ID'})^{s\cdot m'}}\right)^{\frac{1}{(h_2'+h_3')\cdot s_{ID'}}}。$$

如果模拟器B在与敌手A_{II}的交互过程中没有终止运行,那么模拟器与敌手进行了完美的交互模拟过程。模拟器在$SecretValueQuery$和$TagQuery$这两个查询过程中可能会发生终止,而对于$PartialPublicKeyQuery$、H_1Query、H_2Query、$PublicKeyQuery$和H_3Query这几个查询则会正常执行。因此,B会和敌手A_{II}进行完美交互模拟的概率要大于$\eta^{q_{sv}+q_{tag}}$。B正确输出g^{ab}的概率为$\varepsilon'\geqslant\varepsilon\cdot(1-\eta)\cdot\eta^{q_{sv}+q_{tag}}\geqslant\dfrac{\varepsilon}{(q_{sv}+q_{tag})\cdot 2e}$,相应的执行时间为$t'\leqslant t+O(q_{sv}+q_{ppk}+q_{h1}+q_{pk}+q_{h2}+q_{h3}+q_{tag})$。

6.3 证明信息的抗伪造性

定理4. 如果DL难题假设成立,敌手A_{III}仅能以可忽略的概率赢得Game III。

证明. 令挑战信息$chal=(c, k_1, k_2)$,如果敌手A_{III}输出完整性证明$proof'=(T', F')$并且以不可忽略的概率赢得了Game III,可以得到如下等式:

$$\prod_{j=1}^{d}e(T_j', g)\cdot\eta_j=e\left(\prod_{j=1}^{d}(pk_{h_j})^{F_j'}, pk_{KGC}\right)\cdot\prod_{j=1}^{d}e\left(\prod_{v_i\in C_j}\left(H_2(\omega_{v_i})\cdot H_3(\delta_j)\right)^{a_i}, ppk_{h_j}\right) \tag{7}$$

其中,d表明审计数据块相关的用户数量。假定对于挑战$chal$的真实证明为$proof=(T, F)$,有验证等如下:

$$\prod_{j=1}^{d} e(T_j, g)\cdot \eta_j = e\left(\prod_{j=1}^{d}\left(pk_{h_j}\right)^{F_j}, pk_{KGC}\right)\cdot \prod_{j=1}^{d} e\left(\prod_{v_i\in C_j}\left(H_2(\omega_{v_i})\cdot H_3(\delta_j)\right)^{a_i}, ppk_{h_j}\right) \quad (8)$$

因为敌手A_{III}赢得了Game III，所以存在$F\neq F'$但$T=T'$。根据以上两个等式(7)和(8)，有$\prod_{j=1}^{d}\left(pk_{h_j}\right)^{F_j'}=\prod_{j=1}^{d}\left(pk_{h_j}\right)^{F_j}$。对于$1\leqslant j\leqslant d$，设$\Delta F_j=F_j'-F_j$，我们可以得出$\dfrac{\prod_{j=1}^{d}\left(pk_{h_j}\right)^{F_j'}}{\prod_{j=1}^{d}\left(pk_{h_j}\right)^{F_j}}=1$，即$\prod_{j=1}^{d}\left(pk_{h_j}\right)^{\Delta F_j}=1$。根据该结论，*DL*难题可以通过如下步骤进行解决。

首先，给定两个群元素$u,f\in G_1$，其中$u=f^x$，通过解决*DL*难题计算出$x\in Z_q^*$。令$pk_{h_j}=\kappa_j=f^{\alpha_j}u^{\beta_j}$，其中$\alpha_j$和$\beta_j$分别为随机从$Z_q^*$选取的值。可得下列等式：

$$\prod_{j=1}^{d}(\kappa_j)^{\Delta F_j}=\prod_{j=1}^{d}\left(f^{\alpha_j}u^{\beta_j}\right)^{\Delta F_j}=f^{\sum_{j=1}^{d}\alpha_j\Delta F_j}u^{\sum_{j=1}^{d}\beta_j\Delta F_j}=1,$$

$$u=f^{-\frac{\sum_{j=1}^{d}\alpha_j\Delta F_j}{\sum_{j=1}^{d}\beta_j\Delta F_j}}。$$

由$u=f^x$可以计算出$x=-\dfrac{\sum_{j=1}^{d}\alpha_j\Delta F_j}{\sum_{j=1}^{d}\beta_j\Delta F_j}$。因为$F\neq F'$，至少有一个$\Delta F_j\left(1\leqslant j\leqslant d\right)$不为0。$\beta_j\left(1\leqslant j\leqslant d\right)$是从$Z_q^*$随机选取的一个随机值，所以$\sum_{j=1}^{d}\beta_j\Delta F_j=0$的概率仅有$\dfrac{1}{q}$。因此，能以不可忽略的概率$1-\dfrac{1}{q}$计算出正确的$x$值，这与已知的*DL*难题假设相矛盾。

定理5.　如果CDH难题假设成立，敌手A_{III}仅能以可忽略的概率赢得Game IV。

证明.

令挑战信息$chal=(c,k_1,k_2)$，如果敌手A_{III}输出完整性证明$proof'=(T',F')$并且以不可忽略的概率赢得了Game IV，可以得到如下等式：

$$\prod_{j=1}^{d} e(T_j', g)\cdot \eta_j = e\left(\prod_{j=1}^{d}\left(pk_{h_j}\right)^{F_j'}, pk_{KGC}\right)\cdot \prod_{j=1}^{d} e\left(\prod_{v_i\in C_j}\left(H_2(\omega_{v_i})\cdot H_3(\delta_j)\right)^{a_i}, ppk_{h_j}\right) \quad (9)$$

假定对于挑战*chal*的真实证明为$proof=(T,F)$，有验证等式如下：

$$\prod_{j=1}^{d} e(T_j, g)\cdot \eta_j = e\left(\prod_{j=1}^{d}\left(pk_{h_j}\right)^{F_j}, pk_{KGC}\right)\cdot \prod_{j=1}^{d} e\left(\prod_{v_i\in C_j}\left(H_2(\omega_{v_i})\cdot H_3(\delta_j)\right)^{a_i}, ppk_{h_j}\right) \quad (10)$$

因为敌手A_{III}赢得了*Game IV*，所以存在$F\neq F',T\neq T'$。将等式(9)和等式(10)相除，得到下列等式：

$$e\left(\prod_{j=1}^{d}\frac{T_j'}{T_j}, g\right)=e\left(\prod_{j=1}^{d} p\,k_{h_j}^{\Delta F_j}, pk_{KGC}\right) \quad (11)$$

其中，$\Delta F_j=F_j'-F_j$。根据该结论，CDH难题可以通过如下步骤进行解决。

此处给出三个群元素$g,g^x,f\in G_1$，通过解决*CDH*难题计算出$f^x\in G_1$。令$pk_{KGC}=g^x$，$pk_{h_j}=g^{\alpha_j}f^{\beta_j}$，其中$\alpha_j$和$\beta_j$分别为随机从$Z_q^*$选取的值。根据等式(11)可得等式：

$$\prod_{j=1}^{d}\frac{T_j'}{T_j}=\prod_{j=1}^{d} p\,k_{h_j}^{\Delta F_j\cdot x}=\prod_{j=1}^{d}\left(g^{\alpha_j}f^{\beta_j}\right)^{\Delta F_j\cdot x} \quad (12)$$

最终，可以计算出*CDH*难题的结果$f^x=\left(\prod_{j=1}^{d}\dfrac{T_j'}{T_j\cdot pk_{KGC}^{\alpha_j\cdot\Delta F_j}}\right)^{\frac{1}{\sum_{j=1}^{d}\beta_j\cdot\Delta F_j}}$。因为$F\neq F'$，至少有一个$\Delta F_j\left(1\leqslant j\leqslant d\right)$不为0，$\beta_j\left(1\leqslant j\leqslant d\right)$是从$Z_q^*$随机选取的一个随机值，所以$\sum_{j=1}^{d}\beta_j\Delta F_j=0$的概率仅有$\dfrac{1}{q}$。因此，能以不可忽略的概率$1-\dfrac{1}{q}$计算出正确的$f^x$，这与*CDH*难题假设相矛盾。

6.4　陷门的抗伪造性

本节给出陷门抗伪造性的定理及其证明，基于离散对数难题的困难性，该定理确保了陷门机制在对抗恶意伪造攻击时的可靠性。

定理6.　在本文审计方案中，未经授权的第三方审计者无法通过伪造陷门来冒充DV执行数据完整性审计操作。

证明.

为了生成陷门，审计员需要使用自身私钥sk_V和用户公钥pk_O按照等式(2)计算出陷门。除DV以外的审计者仅知晓DV的公钥pk_V，而无法获取其私钥。由于离散对数问题的计算困难性，即使已知KGC的公钥$pk_{KGC}=g^x$，也无法逆向推导主密钥$sk_{KGC}=x$。因此，其他审计者无法通过计算$sk_V=\left(pk_V\right)^x$来获得DV的私钥值。

由于未经授权的TPA难以获取DV的私钥值，因此其无法通过伪造陷门来冒充DV进行数据完整性审计。

6.5 隐私保护性

定理7. 在本文审计方案中，当TPA中的DV向云存储服务器发出多次数据块审计挑战时，DV不能获取任何关于数据块m_i的信息。

证明.

为了检测CSP保存用户的数据情况，TPA会周期性地发出审计挑战，从而检查存储的数据是否保存完整。在证明生成阶段*ProofGen*，对每次的审计挑战都会生成一个相应的证明来证明自己存储的对应数据块m_i是完整的。TPA是半可信的，在正确执行协议的过程中想通过收到的证明信息获取某些数据块的信息，可能会提交特定的参数对某些数据块重复发出审计挑战。

然而，这种方法在该方案中不适用，因为在数据块聚合过程中添加了一个随机数$t_j \in Z_q^*$，$F_j' = \sum_{v_i \in C_j} a_i \cdot m_{v_i} + t_j$，它可以实现数据块的盲化。对于每次返回的都有不同的随机数，这对于DV来说难以通过解线性方程组的方式计算出数据块m_i，此外，η是双线性映射的结果，无法从η获取关于随机数t_j的有效信息。

因此，本方案具有对用户数据的隐私保护特性。

7 性能分析与实验结果

本文选择四个方案作为分析和实验的比较对象，方案[7]针对团队用户审计场景设计，采用无证书签名技术；方案[8]基于授权验证者机制，具有标签的不可伪造性和数据块的隐私保护功能；方案[24]是首个采用无证书签名技术的经典审计方案；方案[27]设计了一个支持数据拥有权限转移的审计方案。

7.1 性能分析

(1) 特性比较

表1对比了本文方案与方案[7-8,24,27]的特性差异，所有方案均基于无证书签名技术。在功能特性方面，本方案与方案[7,27]均支持成员撤销机制，但是仅本方案支持审计员更换功能。本方案与方案[8]共同支持授权审计和隐私保护功能。就工作场景而言，本方案和方案[7-8]均可应用于团队场景。相比于其他方案，本方案包含表1中的所有特性。

(2) 通信开销

本文对方案[7-8,24,27]的通信耗费进行了分析，总的通信耗费比较如表2所示。

本文方案中的通信开销用如下符号来定义。

表1 审计方案的特性比较

方案	无证书	成员撤销	更换审计员	授权审计	隐私保护	工作场景
方案[7]	√	√	×	×	×	团队
方案[8]	√	×	×	√	√	团队
方案[24]	√	×	×	×	×	个人
方案[27]	√	√	×	×	×	个人
本文方案	√	√	√	√	√	团队

表2 通信耗费的比较

方案	*DO*→*CSP*	*DV*→*CSP*	*CSP*→*DV*
方案[7]	$\lvert F_{id}\rvert + n\lvert G_1\rvert$	$2\lvert Z_q^*\rvert + \lvert n\rvert$	$d\left(\lvert G_1\rvert + \lvert Z_q^*\rvert\right)$
方案[8]	$\lvert F_{id}\rvert + n\lvert G_1\rvert$	$\lvert Z_q^*\rvert + \lvert n\rvert$	$\lvert G_1\rvert + \lvert G_2\rvert$
方案[24]	$\lvert F_{id}\rvert + n\lvert G_1\rvert$	$c\lvert Z_q^*\rvert + c\lvert n\rvert$	$\lvert G_1\rvert + n\lvert Z_q^*\rvert$
方案[27]	$\lvert F_{id}\rvert + n\lvert G_1\rvert$	$c\lvert Z_q^*\rvert + c\lvert n\rvert$	$2\lvert G_1\rvert$
本文方案	$\lvert F_{id}\rvert + n\lvert G_1\rvert$	$2\lvert Z_q^*\rvert + \lvert n\rvert$	$d\left(\lvert G_1\rvert + \lvert G_2\rvert + \lvert Z_q^*\rvert\right)$

$|n|$：表示整数集合$[1,n]$中的元素长度。$|Z_q^*|$：表示群Z_q^*中的元素长度。$|G_1|$：表示循环群G_1中的元素长度。$|G_2|$：表示循环群G_2中的元素长度。$|F_{id}|$：表示文件的大小。

在数据上传阶段，数据拥有者将文件及对应的标签相关信息上传至云服务器，该阶段通信耗费为$|F_{id}| + n|G_1|$。在挑战阶段，数据验证者向云服务器发送挑战信息$chal = (c, k_1, k_2)$，该阶段的通信耗费为$2|Z_q^*| + |n|$。在证明阶段，云服务器接收挑战信息后，计算并返回相应的证明信息$proof = (\Phi, \eta)$，该阶段的通信耗费为$d\left(|G_1| + |G_2| + |Z_q^*|\right)$。以上通信耗费中，$n$为数据块数量，$d$为审计数据中所包含的用户数量，$c$为审计数据块的数量。

(3) 计算开销

假定T_{exp}、$T_{G_1 mul}$、$T_{G_2 mul}$、T_{mul}分别代表群G_1中元素的指数运算耗费、群G_1中元素的乘法运算耗费、群G_2中元素的乘法运算以及群Z_q^*中元素的乘法运算耗费；计算过程中的加法运算和哈希运算由于耗时很短不再统计。假定在挑战与证明过程中审计的数据块数量为c，审计数据中所包含的用户的数量为d；在执行用户撤销或更换审计员时重新生成数据块标签的数量为N。

在标签生成阶段，用户执行*TagGen*算法生成一个数据块标签的计算耗费为$2T_{exp} + 2T_{G_1 mul}$；在进行数据块验证时，云服务器的验证计算耗费为

$T_{exp}+T_{G_1mul}+T_{G_2mul}$；在进行数据完整性证明生成时，云服务器执行*ProofGen*算法的计算耗费为$(c+d)T_{exp}+(c-d)T_{G_1mul}$；当用户撤销时，撤销过程需要将旧用户的数据块标签转换为新用户的数据块标签，用户撤销的后续数据块标签生成只需云服务器计算，计算耗费为$N(3T_{exp}+3T_{G_1mul})$。在整个用户撤销和更换审计员的过程中，本方案将用户端参与的大量计算过程迁移至云服务器端。

表3展示了本文方案与方案[7-8，24，27]的计算耗费对比结果。

表3　计算耗费的比较

	*Tag*生成	*Tag*验证	证明生成	用户撤销
方案[7]	$2T_{exp}+T_{G_1mul}$	$T_{exp}+T_{G_2mul}$	$cT_{exp}+(c-d)T_{G_1mul}$	不支持
方案[8]	$2T_{exp}+2T_{G_1mul}$	不支持	$(c+1)T_{exp}+(2c+1)T_{G_1mul}$	不支持
方案[24]	$3T_{exp}+2T_{G_1mul}$	$T_{exp}+T_{G_1mul}+T_{G_2mul}$	$cT_{mul}+cT_{exp}+(c-1)T_{G_1mul}$	不支持
方案[27]	$2T_{exp}+2T_{G_1mul}$	不支持	$cT_{exp}+(2c-1)T_{G_1mul}$	$(2N+9)T_{G_1mul}+(5N+2)T_{exp}$
本文方案	$2T_{exp}+2T_{G_1mul}$	$T_{exp}+T_{G_1mul}+T_{G_2mul}$	$(c+d)T_{exp}+(c-d)T_{G_1mul}$	$N(3T_{exp}+3T_{G_1mul})$

7.2　实验结果

为了评估本方案的性能，本文基于JPBC密码学库(Java Pairing-Based Cryptography)进行性能评估实验。实验环境采用Windows1164位操作系统，硬件配置为Intel Core i5-8300H处理器(2.30 GHz)和16 GB内存。在实验设计中，密码学参数选用typeA型素数阶椭圆曲线，其中基域长度为512位，群阶数为160位。

(1)　数据块标签生成

图6展示了用户上传数据时*TagGen*算法在不同数据块规模下的计算耗时。实验采用1000为增量单位，标签生成的数量从1000增加到9000。结果显示，标签计算耗时随着数据块数量线性增长。在本方案中，生成1000个标签耗时约8.8秒，生成9000个标签耗时约76秒。由于本方案支持标签更新功能，其计算耗时较另外三个方案[7-8,27]略高。

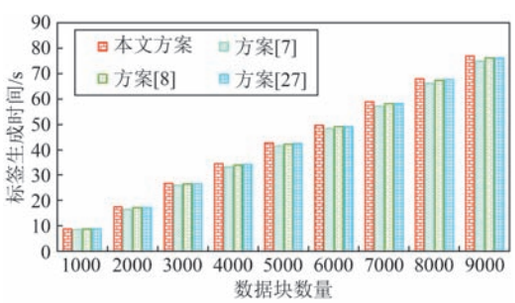

图6　标签生成的时间耗费

(2)　证明生成

当云服务器收到来自审计员的挑战信息后，云服务器就必须对该挑战信息生成相应的证明。图7展示了*ProofGen*算法在不同挑战数据块规模下的计算耗时。

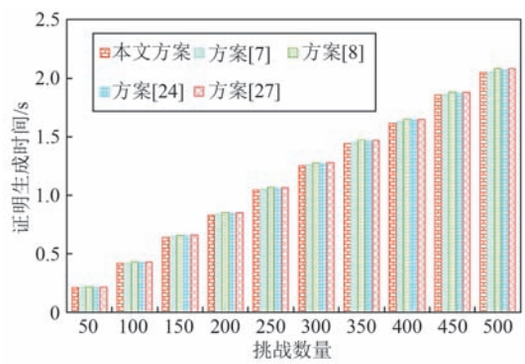

图7　证明生成的时间耗费

在仿真实验中，实验以50为增量单位，测试的挑战块数量从50逐渐增加到500。结果表明，证明生成的耗时随挑战块数量线性增长，证明生成时长也仅从0.2秒增至2.0秒左右。此外，本方案与方案[7，24]性能相当。在挑战块数量从50逐渐增加到500的情况下，本方案与方案[8，27]相比具有稍低的计算耗时，时间差距从3毫秒增至24毫秒，且差距随挑战数量增加而扩大。

(3)　用户撤销和更换审计员的数据块标签变更

用户在执行撤销和更换审计员操作时，数据块标签需要相应更新。方案[7]给出了传统方案的介绍，传统方案要求新用户下载全部数据块及其标签进行本地验证后重新生成并上传标签。在本方案中，云服务器可以通过与新旧两个用户进行轻量级交互之后，在保证密钥安全的前提下，由云服务器代为完成标签更新操作。

图8对比了用户撤销场景下传统方案与本方案的性能表现。对于900个数据块标签的更新，本方案需要7秒左右，而传统方案在忽略网络传输的情况下需要16秒左右，节省了近54%的时间。值得注

意的是，传统方案完全依赖用户本地计算资源，本文方案通过将主要计算任务转移至云服务器，节省了大量用户端的计算资源和网络带宽消耗。

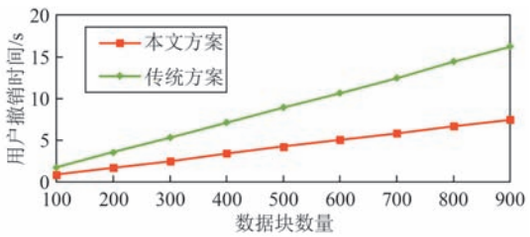

图8 用户撤销的时间耗费

图9对比了审计员更换场景下传统方案与本方案的时间效率。对于传统方案，用户需要执行的计算操作与用户撤销引发的计算操作相同。当数据块数量从100增至900时，本方案对应的计算耗时也仅从2毫秒到16毫秒，将传统方案时间压缩至0.1%左右，大幅减少了计算时间。与传统方案相比，本方案不仅优化了计算时间，还降低了用户端的资源消耗，实现了更高效的审计员更换流程。

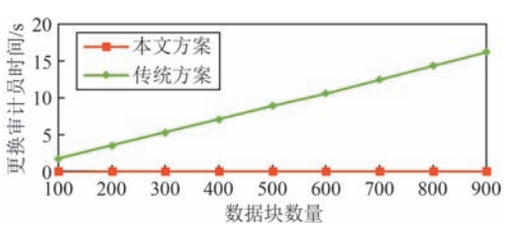

图9 更换审计员的时间耗费

(4)证明验证

当审计员收到来自云服务器的证明信息后，审计员需要执行证明验证操作。在本文方案中，证明验证支持多用户批量验证。此时，应考虑不同组内成员数量对验证时长引发的变化。图10展示了在相同数据块总数情况下，不同团队成员数量的证明验证耗时情况。

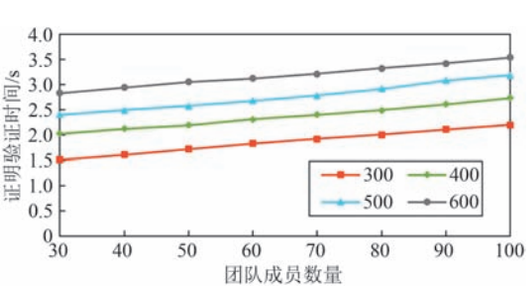

图10 证明验证的时间耗费

实验结果表明验证时间呈现双重线性增长特性。固定数据块数量时，验证时间随团队成员数量线性增长。当验证数据块数量为300，团队成员数量从30人增加至100人时，证明的验证时间从1.5秒增加到2.2秒。当验证数据块数量为400，团队成员数量从30人增加至100人时，证明的验证时间从2秒增加到2.7秒。

固定团队成员数量时，验证时间随数据块数量线性增加。特别地，当组成员固定为30人时，数据块从300增至600块，验证时间以约0.5秒的增量从1.5秒增至2.8秒。可以看出，本方案在不同团队成员数量或数据块数量的情况下对于证明验证过程均具有较好的适用性。

8 结束语

针对团队场景下云数据完整性审计方案中存在的授权审计问题，本文提出了一个面向团队场景的无证书云数据完整性授权审计方案。该方案在指定授权审计者的同时也支持高效地更换授权审计者。当用户退出时，方案还支持用户的变更操作，将用户数据的所有权进行转移，以便后续团队用户的使用。此外，该方案采用无证书签名方案，有效避免了证书管理和密钥托管问题。更换TPA和处理用户变更都需要对数据标签进行更新，本方案中更新操作的通信开销与标签数量无关，是一个恒定值。安全性分析表明本方案具有正确性、隐私保护性等特性，同时性能分析与实验结果也表明方案是高效可行的。

与现有研究类似，本文采用随机抽样方法实现完整性审计，但其存在无法针对特定属性(如价值、时间段等)进行重点审计的局限性。因此，未来研究将继续探索支持属性导向的审计方案，以满足更加多元的实际应用需求。

致 谢 在此，我们向对论文提出宝贵意见的审稿专家们表示衷心的感谢！

参 考 文 献

[1] Xiao L, Cao Y, Gai Y, et al. Review on the application of cloud computing in the sports industry. Journal of Cloud Computing, 2023, 12(1): 152

[2] Ren K, Wang C, Wang Q. Security challenges for the public cloud. IEEE Internet Computing, 2012, 16(1): 69-73

[3] Yan H, Liu Y, Qiu S, et al. Towards public integrity audition for cloud-iot data based on blockchain. Computer Systems

Science & Engineering, 2022, 41(3): 22317
[4] Zhang Y, Xu C, Lin X, et al. Blockchain-based public integrity verification for cloud storage against procrastinating auditors. IEEE Transactions on Cloud Computing, 2021, 9(03): 923-937
[5] Fu A, Yu S, Zhang Y, et al. NPP: A new privacy-aware public auditing scheme for cloud data sharing with group users. IEEE Transactions on Big Data, 2017, 8(1): 14-24
[6] Lin Li, Tan Wen-Ting, Chu Zhen-Xing. A survey on outsourced data integrity auditing. Cyberspace Security, 2020, 11(11):9 (in Chinese)
(林莉、檀文婷、储振兴. 外包数据完整性审计综述. 网络空间安全, 2020, 11(11): 9)
[7] Li J, Yan H, Zhang Y. Certificateless public integrity checking of group shared data on cloud storage. IEEE Transactions on Services Computing, 2021, 14(01): 71-81
[8] Li R, Wang X A, Yang H, et al. Efficient certificateless public integrity auditing of cloud data with designated verifier for batch audit. Journal of King Saud University-Computer and Information Sciences, 2022, 34(10): 8079-8089
[9] Zhou R, He M, Chen Z. Certificateless public auditing scheme with data privacy preserving for cloud storage//Proceedings of the 2021 IEEE 6th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA), 2021: 675-682
[10] Al-Riyami S S, Paterson K G. Certificateless public key cryptography//Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security, 2003: 452-473
[11] Jarolimkova A. Data sharing: an integral part of research practice?. Qualitative and Quantitative Methods in Libraries, 2023, 12(4): 609-620
[12] Yan Y X, Wu L, Xu W Y, et al. Integrity audit of shared cloud data with identity tracking. Security and Communication Networks, 2019, 2019(1): 1354346
[13] Tian H, Nan F, Jiang H, et al. Public auditing for shared cloud data with efficient and secure group management. Information Sciences, 2019, 472: 107-125
[14] Li A, Tian G, Miao M, et al. Blockchain-based cross-user data shared auditing. Connection Science, 2022, 34(1): 83-103
[15] Liu Z, Wang S, Liu Y. Blockchain-based integrity auditing for shared data in cloud storage with file prediction. Computer Networks, 2023, 236: 110040
[16] Tian J, Yang Q. An arbitrable outsourcing data audit scheme supporting credit reward and punishment and multi-user sharing. Journal of Parallel and Distributed Computing, 2023, 178: 100-111
[17] Yi Zhang-Qian, Wen Lin-Ya, Zhang Wei-Xin. Efficient and revocable cloud storage auditing for shared data. Computer Systems & Applications, 2022, 31(3): 333-339 (in Chinese)
(仪张倩，温琳雅，张维鑫. 高效可撤销的共享数据云存储审计. 计算机系统应用, 2022, 31(3): 333-339)
[18] Yuan J, Yu S. Proofs of retrievability with public verifiability and constant communication cost in cloud//Proceedings of the 2013 International Workshop on Security in Cloud Computing. 2013: 19-26
[19] Worku S G, Xu C, Zhao J, et al. Secure and efficient privacy-preserving public auditing scheme for cloud storage. Computers & Electrical Engineering, 2014, 40(5): 1703-1713
[20] Venkatachalam A, Balaji N. Assurance on data integrity in cloud data centre using PKI built RDIC method. Journal of Information Security Research, 2020, 11(1):10-20
[21] Wang H, Wu Q, Qin B, et al. Identity-based remote data possession checking in public clouds. IET Information Security, 2014, 8(2): 114-121
[22] Yu Y, Au M H, Ateniese G, et al. Identity-based remote data integrity checking with perfect data privacy preserving for cloud storage. IEEE Transactions on Information Forensics and Security, 2016, 12(4): 767-778
[23] Ateniese G, Faonio A, Kamara S. Leakage-resilient identification schemes from zero-knowledge proofs of storage//Cryptography and Coding: 15th IMA International Conference, IMACC 2015. Oxford, UK, 2015: 311-328
[24] Wang B, Li B, Li H, et al. Certificateless public auditing for data integrity in the cloud//Proceedings of the 2013 IEEE conference on communications and network security (CNS), 2013: 136-144
[25] Miao Y, Miao Y, Miao X. Blockchain-based transparent and certificateless data integrity auditing for cloud storage. Concurrency and Computation: Practice and Experience, 2024, 36(27): e8285
[26] Worku S G, Xu C, Zhao J. Cloud data auditing with designated verifier. Frontiers of Computer science, 2014, 8: 503-512
[27] Huang Y, Shen W, Qin J. Certificateless cloud storage auditing supporting data ownership transfer. Computers & Security, 2024, 139: 103738
[28] Yan H, Li J, Zhang Y. Remote data checking with a designated verifier in cloud storage. IEEE Systems Journal, 2020, 14(2): 1788-1797
[29] Shao B, Zhang L, Bian G. Incentive public auditing scheme with identity-based designated verifier in cloud. Electronics, 2023, 12(6): 1308
[30] Liu J, Chen Q, Wu G, et al. Dynamic and privacy-preserving cloud auditing with updatable designated verifier//Proceedings of the 2023 4th Information Communication Technologies Conference (ICTC), 2023: 157-161

LIU Yi-Qi, M. S. His research interests include cryptography and information security.

XU Xian, Ph. D., associate professor. His research interests include cryptography, blockchain and information security.

LONG Yu, Ph. D., associate professor. Her research interests include cryptography and blockchain.

Background

This paper studies the authorized audit of data integrity in team scenarios in the field of cloud storage security. Data integrity audit schemes in team scenarios have been widely studied because they are more suitable for the reality of cloud data sharing. With the continuous development of cloud storage technology, researchers are currently committed to studying user revocation and privacy protection of audit schemes in team scenarios. In this paper, we propose a certificateless authorized audit scheme for cloud data integrity in team scenarios, which supports functional features such as user change, authorized audit, and multi-user batch audit. We implement the authorization and change operations of auditors through shared trapdoors. In the operation of changing auditors, users only need to interact with the cloud storage server in a simple way to allow the cloud storage server to update data tags without leaking private keys, saving a lot of bandwidth and user computing resources. In addition, we also blind the audit proof to protect the user's data privacy information while ensuring the correctness of the audit scheme. Performance analysis and experimental results show that the scheme meets multiple functional features. At the same time, the scheme is efficient and feasible for replacing authorized auditors and processing user changes. The research presented in this paper is supported by Shanghai's 2024 Science and Technology Innovation Action Plan (24BC3200500, 24BC3200300).