计算机科学
COMPUTER SCIENCE

基于身份的密钥隔离的多云多副本可证数据持有方案

周杰, 王化群

引用本文

周杰, 王化群. 基于身份的密钥隔离的多云多副本可证数据持有方案[J]. 计算机科学, 2025, 52(1): 401-411.

ZHOU Jie, WANG Huaqun. Identity-based Key-insulated Provable Multi-copy Data Possession in Multi-cloud Storage [J]. Computer Science, 2025, 52(1): 401-411.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.231200081

基于身份的密钥隔离的多云多副本可证数据持有方案

周 杰 王化群
南京邮电大学计算机学院 南京 210023
(zhoujiewish@163.com)

摘 要 可证数据持有方案(Provable Data Possession,PDP)可以让用户在不下载全部数据的情况下验证其外包数据是否完好无损。为了提高外包数据的可用性和安全性,许多用户将数据的多个副本存储在单云服务器上,但是单云服务器在发生故障或者其他意外情况时,用户存储的数据副本也会遭到破坏因而无法恢复原始数据。同时,许多可证数据持有方案依赖于公钥基础设施(Public Key Infrastructure,PKI)技术,存在密钥管理问题。此外,现有的可证数据持有方案大多是在用户端使用密钥对数据进行处理。由于用户端的安全意识较弱或者安全设置较低,密钥可能会有泄露的风险。恶意云一旦获得了用户端的密钥,就可以通过伪造虚假的数据持有证明来隐藏数据丢失的事件。基于上述问题,提出了一种基于身份的密钥隔离的多云多副本可证数据持有方案(Identity-Based Key-Insulated Provable Multi-Copy Data Possession in Multi-Cloud Storage,IDKIMC-PDP)。基于身份的可证数据持有方案消除了公钥基础设施技术中复杂的证书管理。多云多副本确保了即使在某个云服务器上的副本被篡改或者被破坏的情况下,用户仍然可以从其他云服务器上获取副本并恢复数据。同时,方案中使用了密钥隔离技术实现了前向和后向安全。即使某一时间段内的密钥泄露,其他时间段内云存储审计的安全性也不会受到影响。给出了该方案的正式定义、系统模型和安全模型;在标准困难问题下,给出了该方案的安全性证明。安全性分析表明,IDKIMC-PDP 方案具有强抗密钥泄露性、可检测性以及数据块标签和证明的不可伪造性。实验结果表明,与现有的多云多副本相关方案相比,IDKIMC-PDP 方案具有相对较高的效率。

关键词:可证数据持有;密钥隔离;基于身份的签名;多云多副本
中图分类号 TP309

Identity-based Key-insulated Provable Multi-copy Data Possession in Multi-cloud Storage

ZHOU Jie and WANG Huaqun
School of Computer Science,Nanjing University of Posts and Telecommunications,Nanjing 210023,China

Abstract Provable data possession(PDP) allows users to verify that their outsourced data is intact without downloading all the data. To improve the availability and security of outsourced data,many users store multiple copies of their data on a single server. In case of a single cloud server failure or other unexpected circumstances,the data copy stored by users will be damaged and the original data cannot be restored. At the same time,many PDP schemes rely on the technique of public key infrastructure(PKI), which has key management problems. In addition,most of the existing PDP schemes use the key to process the data on the client side. Because the security awareness of the client is weak or the security settings are low,the key may be exposed. Once the malicious cloud obtains the client's key,it can hide the event of data loss by forging false proof of data possession. Based on the above problems,we propose a scheme called identity-based key-insulated provable multi-copy data possession in multi-cloud storage. Identity-based PDP scheme eliminates complex certificate management in the technique of public key infrastructure. Multi-copy in multi-cloud ensures that if all copies in one cloud server are tampered with or corrupted,users can still obtain copies from other cloud servers and recover data. At the same time,the key-insulated technology is used to realize forward and backward security. Even if the key is exposed in a certain period of time,the security of cloud storage auditing in other periods of time is not affected. The formal definition,system model and security model of the scheme are given. The security proof of the scheme is given under the standard difficult problem. The security analysis shows that the proposed scheme has strong anti-key leakage,detectability and unforgeability of data block authenticator and proofs. Experimental results show that compared with the existing multi-cloud

到稿日期:2023-12-12 返修日期:2024-05-29
基金项目:国家自然科学基金(U23B2002)
This work was supported by the National Natural Science Foundation of China(U23B2002).
通信作者:王化群(whq@njupt.edu.cn)

and multi-copy related schemes, the proposed scheme has relatively high efficiency.

Keywords Provable data possession, Key-insulated, Identity-based signature, Multi-copy in multi-cloud

1 引言

随着云计算技术的高速发展，用户每日产生的数据激增，有限的本地存储资源已无法满足用户的需求。云服务器可以通过提供数据存储和计算等服务，减轻用户存储管理和维护的负担[1]。用户在将数据存储到云服务器的同时，也失去了对数据的控制权。但是，云服务提供商(Cloud Service Provider，CSP)并不是完全可信的[2]。CSP 可能为了减少成本，节省存储资源，会故意删除部分不常被用户访问的数据。此外，当云服务器出现软硬件故障或遭受恶意攻击等不可恢复事件后，其中存储的数据可能被损坏，但对于一些不常使用的数据，用户难以及时察觉。因此，用户需要对云存储中的数据进行完整性检测，以确保数据的完整性。可证数据持有是解决相关问题的基本方法。另一方面，可证数据持有方案必须是高效的，以使其适应容量有限的终端设备。因此，如何设计出安全且高效的可证数据持有方案是远程数据完整性验证的重点问题。

基于身份的签名和加密方案通常依赖于密钥保存安全的假设。然而，随着越来越多的签名和加密过程部署在不安全的设备上(例如移动设备)，密钥泄露似乎是不可避免的。密钥泄露可能是对密码系统最具破坏力的攻击，因为密钥泄露意味着系统完全失去安全性。对于可证数据持有方案来说，也是如此。密钥隔离是对抗密钥泄露最常用的方法之一。Dodis 等于 2002 年提出了密钥隔离的方案，它的主要思想是：将用于签名或者加密的密钥分成两个部分，一部分由用户保存，另一部分存放在一个物理安全的设备上[3]。用户在当前时间段处理数据的密钥由两部分的密钥组合而成。物理安全设备的密钥在整个生命周期内不会改变，只有用户的临时密钥会随着时间段更新。密钥隔离方案保证了当前时间段的密钥泄露不会推导出之前和之后时刻的密钥。

使用密钥隔离的方式可以解决 PDP 模型中密钥泄露带来的风险。然而，现有的少量方案只在 PKI 系统中实现了单云环境上的抗密钥泄露的 PDP 方案，给证书管理带来了较大的负担。同时，考虑到分布式的多云场景在实际应用中的需要，提出一个基于身份和密钥隔离的多云多副本方案是有必要的。

2 相关工作

近年来，多种可证数据持有方案被相继提出。2007 年，Ateniese 等首次提出了可证数据持有模型，并实现了两个基于 RSA 的 PDP 方案[4]。在该模型中，云服务器只需访问文件的一小部分数据即可生成数据完整性证据，用户则无需下载完整文件就可通过挑战-响应的方式得到云中数据的概率完整性证明。这种概率抽样检测的方式减小了验证和通信开销。但是，文献[4]中的 PDP 方案仅支持静态数据的验证。考虑到实际应用中数据的动态操作，Erway 等提出了支持完全动态数据更新的 PDP 方案[5]。由于用户或者云服务器的验证结果都无法使对方信服，因此引入了第三方验证者(Third Party Auditor，TPA)来执行验证[6]。后来，各种各样的 PDP 方案被相继提出，这些方案主要致力于改进安全性、计算效率、额外存储空间和通信效率。然而，这些方案只适用于单云存储服务，应用于分布式的多云环境时效率非常低下。为了解决多云环境中数据持有证明问题，2012 年，Zhu 等提出一个协同 PDP 方案[7]，首次将其适用的这种分布式云计算环境称为多云。为了消除传统 PKI 系统中复杂的证书管理流程，提高计算和通信效率，Wang 等提出了基于身份的 PDP 方案，使用用户的身份信息生成对应的密钥[8]。考虑到区块链具有去中心化、公开透明防篡改以及匿名性等特点，学者们[9-11]相继提出了基于区块链的 PDP 方案。将区块链技术与 PDP 方案相结合，增强了 PDP 协议的可靠性和运行透明度[12]。为了满足某些特定场景的应用，Wang 等首次研究了在基于身份的公钥加密中满足匿名性、激励性和远程数据完整性检查的协议[13]。Zhang 等提出了支持隐私保护的可验证云端分享方案[14]。Li 等首次提出了基于区块链和数字孪生的同步可证数据持有方案[15]。Yang 等提出了基于压缩云存储的高效身份可证明数据占有协议，在该模型中，云存储审计可以只使用加密的数据块，并且能够通过自验证的方式实现，原始数据块可以由外包数据重构[16]。为了提高数据的安全性和可用性，Curtmola 等提出了一种用于云服务器中多个副本完整性检查的 PDP 方案，数据所有者可以生成多个副本并将所有副本存储在云服务器上[17]。后来，支持动态更新的多副本方案被相继提出[18-21]。许多多副本的 PDP 方案是基于单云服务器场景的，但是单云服务器在发生故障或者其他意外情况时，用户存储的数据副本也会遭到破坏而无法恢复原始数据。于是，Li 等提出了基于身份的多云多副本的 PDP 方案[22]。在多云多副本的场景下，即使某个云服务器上的所有副本都被篡改或者破坏，用户仍然可以从其他云服务器上获取副本并恢复数据。考虑到区块链的优势，Miao 等提出了基于区块链的多云多副本 PDP 方案[23]。从目前的研究可见，PDP 方案已经比较成熟。但随着当下人工智能、机器学习、量子计算等新兴技术的发展，未来基于 PDP 的云存储数据完整性验证机制在关联应用场景、强化安全能力和提高验证效率方面仍然需要深入的研究。

2002 年，Dodis 等提出了密钥隔离方案[3]，用以解决密钥泄露带来的风险问题。其主要思想是：加密的密钥分成两个部分，一部分保存在物理安全的协助器上，另一部分由用户保存。当需要使用密钥进行加密时，将两部分的密钥进行组合以产生新的临时密钥。其中，协助器的密钥在整个时间周期内都不会改变，而用户进行加密的临时密钥会随着时间段更新。对于一个(t,N)密钥隔离方案来说，即使 t 个时间段的密钥泄露，也不会影响其余 $N-t$ 个时间段内密钥的安全。密钥隔离方案保证了当前时间段的密钥泄露不会推导出之前和之后时刻的密钥，即密钥隔离方案实现了前向和后向安全。此后，多个密钥隔离的方案被提出。Weng 等提出了基于

身份和密钥隔离的签名方案，该方案具有强密钥隔离和无限时间段等特点[24]。每个客户端可以定期更新自己的密钥，而对应的公钥保持不变。为了减少聚合签名中密钥暴露带来的风险，同时保持基于身份签名体制的优势，Vasudeva 等提出了基于身份的密钥隔离聚合签名方案[25]。然而，密钥隔离方案需要进行频繁的密钥更新，这会增加协助器密钥泄露的概率。于是，Hanaoka 等提出了平行密钥隔离加密方案[26]，使用两个协助器交替更新密钥，其中一个协助器的密钥独立于另一个协助器的密钥。该方案减小了协助器密钥泄露的可能性，增加了系统的安全性。Hou 等将平行密钥隔离应用到了工业物联网中，提出了工业物联网中基于证书的密钥隔离聚合签名方案[27]，该方案可以抵抗全选密钥攻击。Cui 等将密钥隔离与可搜索加密相结合，提出了面向工业物联网的并行密钥隔离多用户可搜索加密方案[28]。

考虑到 PDP 方案中密钥可能有泄露的风险，Yu 等首次提出了抗密钥泄露的云存储审计方案[29]。该方案中使用了二叉树结构和先序遍历方法来更新密钥，但是其只实现了前向安全性。后来，Yu 等又进一步提出了云储存中基于 PKI 的强抗密钥泄露审计方案[30]。在该方案中，一个时间段的密钥泄露不影响其他时间段的云存储审计的安全性。

本文在文献[22]和文献[30]的基础上进一步创新，将密钥隔离的思想与可证数据持有方案进行结合，提出了一个基于身份的密钥隔离的多云多副本可证数据持有方案。用户将需要上传到云服务器的数据分成多个数据块，然后产生多个副本。用户对数据块进行处理的密钥由两部分组成，一部分由用户自己保存，另一部分放在物理安全的密钥更新协助器上。新的时间段开始时，由协助器产生更新消息，并通过安全信道传递给用户。用户将上一时间段的临时密钥与更新消息组合，从而产生当前时间段的临时密钥，然后使用当前时间段的临时密钥完成数据的处理和完整性验证。通过使用密钥隔离技术，实现了前向和后向安全的可证数据持有方案。同时，本文中的方案也支持无限时间段的密钥更新。

3　预备知识

3.1　双线性映射

设 G_1 和 G_2 是阶为 q 的乘法循环群，且 q 为大素数。g 是 G_1 的生成元，$e:G_1\times G_1\rightarrow G_2$ 是双线性映射，它满足以下性质。

1）双线性：对于任意的 $P,Q\in G_1$，$a,b\in Z_q^*$，有 $e(P^a,Q^b)=e(P,Q)^{ab}$。

2）非退化性：存在元素 $P,Q\in G_1$，使得 $e(P,Q)\neq 1$。

3）可计算性：对于任意的 $P,Q\in G_1$，$e(P,Q)$可有效计算。

3.2　CDH 困难问题假设

定义 1（CDH 问题，Computation Diffie-Hellman Problem）　假设 G_1 是一个乘法循环群，g 是 G_1 的生成元。给定数据元组 (g,g^a,g^b)和未知数 $a,b\in Z_q^*$，计算 g^{ab}。

定义 2（CDH 假设）　对于任何概率多项式算法 A，A 解决群 G_1 上的 CDH 问题的优势是可忽略不计的，其定义如下：

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

其中，ε 表示一个可以忽略的值。

3.3　符号说明

本文中使用到的符号及其对应的含义如表 1 所列。

表 1　符号说明

Table 1　Notations and their descriptions

符号	说明
G_1	阶为 q 的乘法循环群
G_2	阶为 q 的乘法循环群
Z_q^*	$\{1,2,\cdots,q-1\}$
g	G_1 的生成元
H_1,H_2,H_3	3 个不同的哈希函数
(msk,mpk)	系统主私钥/主公钥对
(hsk,hpk)	协助器私钥/公钥对
(ID,SK_{ID})	基于用户身份的公钥/私钥对
N	文件副本数量
n	每个文件副本的数据块数量
$F=\{m_1,m_2,\cdots,m_n\}$	文件 F 分成 n 个数据块
m_{ij}	第 i 个副本中的第 j 个数据块
$F_{ij}=\{m_{ij}\}_{(1\leqslant i\leqslant N,1\leqslant j\leqslant n)}$	第 i 个副本被分成 n 个数据块
CF	文件副本集合
Fid	文件名标识符
F_i	第 i 个文件副本
T_i	副本 F_i 的标签集合 T_i
Cid_i	存储副本 F_i 的云服务器标识
UK_t	t 时刻协助器的更新信息
TSK_t	t 时刻用户的临时签名密钥
$SSig$	处理文件名的签名算法
(ssk,spk)	$SSig$ 算法对应的私钥/公钥对
$chal=(c,k_1,k_2)$	TPA 生成的挑战参数
$C=\{(v_j,a_j)\}$	被挑战数据块索引和对应的随机系数
ξ	存储文件 Fid 副本的云服务器数量
CT_i	云服务器 Cid_i 中文件 Fid 多个副本的索引集合
$P_i=\{\sigma_i,M_i\}$	云服务器 Cid_i 中的数据完整性证明
$P=\{t,\sigma,M,R,U,T_{Fid}\}$	CO 返回的最终完整性证明

4　方案设计

4.1　系统模型

本文中的 IDKIMC-PDP 协议由 6 个实体组成：密钥生成中心（Key Generation Center，KGC），用户 Client，云组织者（Cloud Organizer，CO），云存储服务器（Cloud Storage Server，CSS），第三方审计（Third Party Auditor，TPA），以及密钥更新协助器 Helper。

1）KGC 为用户和 Helper 生成私钥，并通过安全通道分别返回给用户和协助器。

2）Client 使用云存储服务，将数据存储在云服务器上。生成外包文件的标签，并和文件一起存储到云服务器上。

3）CSS 为用户提供存储服务。它拥有巨大的存储容量和强大的计算能力来维护用户的数据。

4）CO 是 CSS 的组织者。在存储文件副本时，用户首先将副本发送给 CO。CO 根据用户的要求向目标 CSS 分发不同的副本。当挑战文件完整性时，TPA 首先向 CO 发送挑战请求，CO 将挑战分发给相应的 CSS。当从 CSS 收到所有不同的证明后，CO 将它们汇总为完整的证明并发送给 TPA。

5）TPA 代表数据所有者验证所有外包文件的完整性。用户和 CSS 都相信 TPA 诚实地执行验证工作。

6）Helper 是一个物理安全的设备，作用是为所有时间段

生成用户的更新密钥。用户使用对应时间段的更新密钥来生成新特定时间段的临时签名密钥。

基于上述实体以及实体之间的联系，构建出的系统模型如图 1 所示。

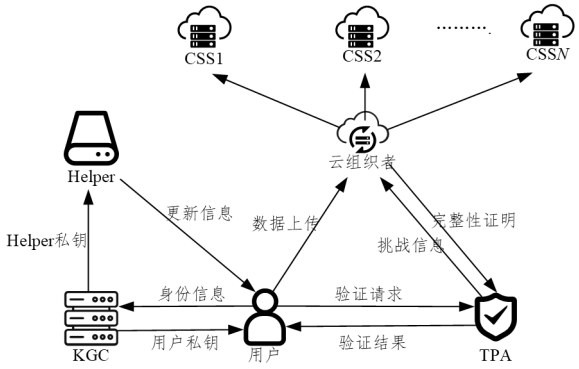

图 1 系统模型

Fig. 1 System model

4.2 详细算法流程

IDKIMC-PDP 方案的具体算法流程如下。

假设用户计划将文件 F 外包给云服务器。根据文件大小，数据所有者将 F 分成 n 块，即 $F=\{m_1,m_2,\cdots,m_n\}$，其中 m_i 表示文件 F 的第 i 个数据块。系统的初始时刻为 $t=0$。构造的方案包括 10 个多项式算法，详细描述如下。

1) $Setup(1^k)\rightarrow(params,msk,hsk)$：输入一个安全参数 k，KGC 产生系统主密钥 msk、协助器密钥 hsk 和相关的公开参数 $params$。选取 2 个阶为 q 的乘法循环群 G_1 和 G_2，1 个双线性映射 $e:G_1\times G_1\rightarrow G_2$。$g$ 是 G_1 的生成元。选取 3 个不同的哈希函数 $H_1:\{0,1\}^*\rightarrow G_1$，$H_2:\{0,1\}^*\rightarrow G_1$ 和 $H_3:\{0,1\}^*\times G_1\rightarrow G_1$，一个伪随机排列 $\pi:Z_q^*\times\{1,\cdots,n\}\rightarrow\{1,\cdots,n\}$ 和一个伪随机函数(PRF) $\phi:Z_q^*\times Z_q^*\rightarrow Z_q^*$。KGC 随机选取 $x\in Z_q^*$ 作为系统主私钥 msk，计算 $P_0=g^x$ 作为系统的主公钥 mpk。选取 $y\in Z_q^*$ 作为协助器 Helper 的私钥 hsk，计算 $P_H=g^y$ 作为对应的公钥 hpk。KGC 秘密保存系统私钥 msk，将密钥 hsk 通过安全信道发送给协助器。最后，KGC 公开系统参数 $params=(q,g,G_1,G_2,e,P_0,P_H,H_1,H_2,H_3,\pi,\phi)$。

2) $Extract(ID,msk,hsk,params)\rightarrow TSK_0$：对于给定的用户 Client 的身份信息 ID，KGC 运行该算法产生用户的初始签名密钥 TSK_0。计算 $SK_{ID}=H_1(ID)^x$，$Q_t=H_2(ID,t)^y$，其中 t 表示时间片段。计算 $TSK_0=SK_{ID}\cdot Q_0=H_1(ID)^x\cdot H_2(ID\|0)^y$ 作为用户的初始签名密钥。

3) $UMGen(t,hsk)\rightarrow UK_t$：在时间段 t 开始时，协助器 Helper 根据此算法来计算 t 时间段的更新消息。协助器 Helper 按照如下公式计算：

$$\begin{aligned}UK_t&=(Q_t\cdot Q_{t-1}^{-1})^y\\&=[H_2(ID\|t)\cdot H_2(ID\|t-1)^{-1}]^y\end{aligned}$$

然后，协助器将更新信息 UK_t 通过安全信道发送给用户。

4) $CKeyUpdate(TSK_{t-1},UK_t)\rightarrow TSK_t$：当时间段 t 开始时，用户接收到来自协助器 Helper 的更新信息 UK_t。然后，用户按照如下公式计算 t 时间段的密钥。

$$\begin{aligned}TSK_t&=TSK_{t-1}\cdot UK_t\\&=H_1(ID)^x\cdot H_2(ID\|t-1)^y\cdot[H_2(ID\|t)\cdot H_2(ID\|t-1)^{-1}]^y\\&=H_1(ID)^x\cdot H_2(ID\|t)^y\end{aligned}$$

将 TSK_t 作为在时间段 t 内的签名私钥。

5) $RepGen(F,N)\rightarrow CF$：对于任何文件 F，用户生成 N 个不同的副本。这些副本彼此不同，因为如果所有副本都相同，云服务器可以串通欺骗用户，因为它们可以只共享一个副本，但声称所有副本都已存储。因此，利用一种具有扩散特性的加密算法 $E_k(\cdot)$ 来获得不同的副本，比如 AES 或者 DES。假设原始文件 F 被分成 n 个块，即 $F=\{m_1,m_2,\cdots,m_n\}$。于是，第 i 个副本表示为 $F_{ij}=\{m_{ij}\}_{(1\leqslant i\leqslant N,1\leqslant j\leqslant n)}$，其中单个数据块 m_{ij} 是通过等式 $m_{ij}=E_k(i\|m_j)$ 计算得出。

6) $TagGen(TSK_t,F_i,Cid_i)\rightarrow T_i$：用户执行此算法生成数据块标签。其中，符号 Cid_i 是存储第 i 个文件副本 F_i 的 CSS 标识。用户选取群 G_1 上的元素 U。用户再选取一个随机数 $r\in Z_q^*$ 并计算 $R=g^r$。然后，用户按照以下等式计算副本 F_i 中的每个数据块 m_{ij} 的标签。

$$T_{ij}=TSK_t\cdot(H_3(Fid\|Cid_i\|i\|j\|t,R)\cdot U^{m_{ij}})^r \quad (1)$$

用户重复计算式(1) $N\times n$ 次得出所有副本的标签集 T，并且创建如表 2 所列的记录表来存储 CSS 和对应副本索引之间的关系。

表 2 副本在 CSS 中的存储位置记录

Table 2 Copies storage location records in CSS

CSS 标识	副本索引
Cid_1	1,4,9
Cid_2	3,6
Cid_3	2,7
…	…

此外，用户需选取一个签名算法 SSig(比如 BLS 算法)来计算文件标识符的完整性。设 (ssk,spk) 是 SSig 算法对应的一对私钥和公钥，用户秘密保存 ssk，公开 spk。用户使用 SSig 算法计算 $T_{\mathrm{Fid}}=SSig_{ssk}(Fid\|R\|\mathrm{U}\|t)$ 作为文件名的标签。最后，用户将标签、记录表和 T_{Fid} 发送给组织者 CO。CO 根据记录表，将文件标签和对应的副本发送给对应的 CSS。此外，每个标签都可以通过式(2)进行验证。

$$e(T_{ij},g)=e(H_1(ID),P_0)\cdot e(H_2(ID\|t),P_H)\cdot e(H_3(Fid\|Cid_i\|i\|j\|t,R)\cdot U^{m_{ij}},R) \quad (2)$$

7) $Challenge(Fid)\rightarrow chal$：TPA 接收到验证通知后，将进行如下操作。为了检查所有名为 Fid 副本的完整性，TPA 随机选取两个值 $k_1,k_2\in Z_q^*$，将其分别作为伪随机排列和伪随机函数的随机种子。TPA 选择一个数 $c\in[1,n]$ 作为挑战的数据块数目。TPA 将 $chal=(c,k_1,k_2)$ 和文件名 Fid 发送给组织者 CO。假设有 ξ 个 CSS 存储了被挑战文件 Fid 的副本，CO 搜索记录表，然后将 $chal$ 发送给对应的 ξ 个 CSS。

8) $ProofGen(CF_i,chal)\rightarrow P_i$：当从 CO 接收到 $chal=(c,k_1,k_2)$ 后，存储了文件 Fid 副本的各个 CSS 生成各自的证明。假设云服务器 Cid_i 存储了文件 Fid 的副本，Cid_i 上多个文件副本 CF_i 的索引集合是 CT_i，易知 $\sum_{t=1}^{\xi}|CT_t|=N$。Cid_i 收到

了挑战集合 $C=\{(v_j,a_j)\}$，其中对于所有 $1\leqslant j\leqslant c$，$v_j=\pi(k_1,j)$，$a_j=\phi(k_2,j)$。对于任意的 $\beta\in CT_i$，索引为 Cid_i 的云服务器 CSS 计算每个副本上相应的证明，即 $\sigma_\beta=\prod_{(v_k,a_k)\in C}T_{\beta v_k}^{a_k}$，$M_\beta'=\sum_{(v_k,a_k)\in C}a_k m_{\beta v_k}$。然后，索引为 Cid_i 的云服务器计算出它的证明 $\sigma_i=\prod_{\beta\in CT_i}\sigma_\beta$，$M_i=\sum_{\beta\in CT_i}M'_\beta$。最后，标识为 Cid_i 的云服务器将证明 $P_i=\{\sigma_i,M_i\}$ 发送给组织者 CO。

9) $ProofAggre(\{P_i\}_{(1\leqslant i\leqslant\xi)})\rightarrow P$：CO 从 ξ 个云服务器接收到所有的证明之后，对这些证明进行聚合。CO 计算 $\sigma=\prod_{i=1}^{\xi}\sigma_i$，$M=\sum_{i=1}^{\xi}M_i$。组织者 CO 将最终的证明 $P=\{t,\sigma,M,R,U,T_{Fid}\}$ 发送给 TPA。

10) $Verify(ID,chal,P,\{Cid\}_{1\leqslant i\leqslant\xi})\rightarrow\{1,0\}$：TPA 在接收到来自 CO 的证明 P 之后，使用 $SSig_{ssk}(Fid\parallel R\parallel U\parallel t)$ 来验证 T_{Fid} 是否是一个有效的签名。如果不是，TPA 拒绝验证证明并且输出 0。否则，对于所有 $1\leqslant j\leqslant c$，TPA 计算 $v_j=\pi(k_1,j)$，$a_j=\phi(k_2,j)$，$C=\{(v_j,a_j)\}$。然后，TPA 验证式(3)是否成立。

$$\begin{aligned}e(\sigma,g)=&\,e(H_1(ID)^{\sum_{(v_k,a_k)\in C}a_k},P_0^N)\cdot e(H_2(ID\parallel t)^{\sum_{(v_k,a_k)\in C}a_k},P_H^N)\cdot e(\prod_{i=1}^{N}\prod_{(v_k,a_k)\in C}H_3(Fid\parallel Cid_i\parallel i\parallel v_k\parallel t,R)^{a_k}\cdot U^M,R)\end{aligned}\tag{3}$$

如果式(3)成立则输出 1，否则输出 0。

如果所有的实体都能够诚实地执行协议，那么能通过式(3)验证文件副本的完整性。

式(3)的正确性证明如下所示：

$$\begin{aligned}e(\sigma,g)=&\,e(\prod_{l=1}^{\xi}\sigma_l,g)=e(\prod_{l=1}^{\xi}\prod_{\beta\in CT_l}\sigma_\beta,g)\\=&\,e(\prod_{i=1}^{N}\sigma_i,g)=e(\prod_{i=1}^{N}\prod_{(v_k,a_k)\in C}T_{\widetilde{w}_k}^{a_k},g)\\=&\,e(\prod_{i=1}^{N}\prod_{(v_k,a_k)\in C}(TSK_t\cdot(H_3(Fid\parallel Cid_i\parallel i\parallel v_k\parallel t,R)\cdot U^{m_{\widetilde{w}_k}})^r)^{a_k},g)\\=&\,e(\prod_{i=1}^{N}TSK_t^{\sum_{(v_k,a_k)\in C}a_k},g)\cdot e(\prod_{i=1}^{N}\prod_{(v_k,a_k)\in C}(H_3(Fid\parallel Cid_i\parallel i\parallel v_k\parallel t,R)\cdot U^{m_{\widetilde{w}_k}})^r)^{a_k},g)\\=&\,e((H_1(ID)^x\cdot H_2(ID\parallel t)^y)^{N\cdot\sum_{(v_k,a_k)\in C}a_k},g)\cdot e(\prod_{i=1}^{N}\prod_{(v_k,a_k)\in C}H_3(Fid\parallel Cid_i\parallel i\parallel v_k\parallel t,R)^{a_k}\cdot\prod_{i=1}^{N}\prod_{(v_k,a_k)\in C}U^{m_{\widetilde{w}_k}a_k},R)\\=&\,e((H_1(ID)^x)^{N\cdot\sum_{(v_k,a_k)\in C}a_k},g)\cdot e((H_2(ID\parallel t)^y)^{N\cdot\sum_{(v_k,a_k)\in C}a_k},g)\cdot e(\prod_{i=1}^{N}\prod_{(v_k,a_k)\in C}H_3(Fid\parallel Cid_i\parallel i\parallel v_k\parallel t,R)^{a_k}\cdot\prod_{i=1}^{N}\prod_{(v_k,a_k)\in C}U^{m_{\widetilde{w}_k}a_k},R)\\=&\,e(H_1(ID)^{\sum_{(v_k,a_k)\in C}a_k},P_0^N)\cdot e(H_2(ID\parallel t)^{\sum_{(v_k,a_k)\in C}a_k},P_H^N)\cdot e(\prod_{i=1}^{N}\prod_{(v_k,a_k)\in C}H_3(Fid\parallel Cid_i\parallel i\parallel v_k\parallel t,R)^{a_k}\cdot U^M,R)\end{aligned}$$

5　安全性分析

5.1　安全模型

用户端的秘密信息包含两个部分。一部分是 TSK_t，用于在时间段 t 内生成数据块标签；另一部分是 SK_{ID}，用于在时间段 t 内生成临时签名密钥 TSK_t。为了形式化安全模型，我们定义了敌手 A 和挑战者 C 之间的游戏，以显示敌手 A 如何攻击基于密钥隔离审计方案的安全性。在这个游戏中，考虑一个最强大的敌手，他可以在除一个密钥未暴露的时间段以外的所有时间段查询用户端的密钥。这个游戏包括以下几个阶段。

1) Setup 阶段：设置系统初始时间段为 $t=0$。挑战者 C 运行 Setup 算法产生系统主私钥 msk、密钥更新协助器的私钥 hsk 和其他系统参数 $params$。挑战者 C 秘密保存 msk 和 hsk，然后将 $params$ 发送给敌手 A。

2) RepGen 阶段：在时间段 t 内，挑战者 C 运行 RepGen 算法获得时间段 t 内初始文件的所有副本。

3) Queries 阶段：在时间段 t 内，敌手 A 可以对挑战者 C 进行多项式次询问。C 按照如下方式回答询问。

(1) Key-Query。敌手 A 发送任意身份 ID_i 查询对应的私钥。挑战者 C 执行 Extract 算法产生用户私钥 SK_{ID} 和时间段 t 的签名密钥 TSK_t，并将 SK_{ID} 和 TSK_t 发送给敌手 A。

(2) Hash-Query。在时间段 t 内，敌手 A 向挑战者 C 发送哈希询问，挑战者 C 将哈希结果返回给敌手 A。

(3) Tag-Query。在时间段 t 内，敌手 A 自适应地选择任意副本的任意数据块，然后询问对应的标签。挑战者 C 执行 TagGen 算法产生标签并返回给敌手 A。

4) Challenge 阶段：挑战者 C 选择一个时间段 t^*。在时间段 t^* 内，敌手 A 没有进行密钥询问。挑战者 C 执行 Challenge 算法产生一个挑战 $chal$，并将 $chal$ 发送给敌手 A。

5) Forgery 阶段：在时间段 t^* 内，敌手 A 执行 ProofGen 和 ProofAggre 算法产生用户信息为 ID_i 的数据块的完整性证明。所有的数据块标签是由敌手 A 进行 Tag-Query 询问产生的。敌手 A 将证明发送给挑战者 C。C 执行 Verify 算法验证证明，然后将结果返回给 A。如果 Verify 的结果为 1，那么敌手 A 赢得游戏。

上述安全模型指出，在密钥没有泄露的时间段内，如果敌手无法猜测出所有缺失的数据块，那么敌手就无法给出与挑战 $chal$ 对应的有效证明。在每个时间段内，敌手可以查询所有数据块的标签。除了进行挑战的时间段以外，敌手也可以查询其他所有时间段的密钥。敌手的目标是在时间段 t^* 内生成与挑战 $chal$ 对应的有效证明。

定义 3(强抗密钥泄露性)　如果敌手 A 赢得上述游戏的概率是可忽略不计的，那么本文中的 IDKIMC-PDP 协议具有强抗密钥泄露性。

此外，一个安全的 IDKIMC-PDP 协议还需要让客户相信，所有外包数据在很大程度上是完好无损的。于是，我们给出以下安全性定义。

定义 4$\left(\left(\frac{m}{n},1-\left(\frac{n-m}{n}\right)^c\right)\right.$可检测性$\left.\right)$　如果云中共存储了 n 个数据块，其中有 m 个数据块被损坏、修改或者删除，而其中有 c 个数据块被挑战，那么本文的方案就是 $\left(\frac{m}{n},1-\left(\frac{n-m}{n}\right)^c\right)$ 可检测的。

5.2 安全性证明

定理 1(强抗密钥泄露性) 如果 G_1 上的 CDH 问题是困难的,那么本文中的 IDKIMC-PDP 协议具有强抗密钥泄露性。

证明:我们定义了一系列游戏,并分析了连续游戏中敌手的行为差异。

游戏 0 游戏 0 与 5.1 节中定义的安全模型相同。

游戏 1 游戏 1 与游戏 0 只有一点不同:挑战者 C 保存了所有由他签名的标签。当敌手 A 提交一个由 SSig 算法生成但未被挑战者 C 签名的有效标签时,挑战者 C 将中止游戏。

分析 在游戏 1 中,如果敌手 A 会让挑战者中止游戏,那么其很容易利用敌手 A 构造攻击者来破解 SSig 签名方案。从现在开始,我们可以保证与敌手交互中的文件名 Fid 和时间 t 都是由挑战者 C 生成的。

游戏 2 游戏 2 与游戏 1 类似,只有一点不同:挑战者 C 保存了敌手 A 标签询问的响应列表。如果敌手成功了,但是证明 P 中的 R 与挑战者列表中保存的 R 值不同,那么挑战者将中止游戏。

分析 下面将证明,如果敌手 A 使挑战者 C 在游戏 2 中止游戏,那么我们能创建一个模拟器,该模拟器能够以不可忽略的概率解决 CDH 困难问题。模拟器类似于游戏 1 中的挑战者 C,但是有以下不同。

首先,给挑战者 C 一个 CDH 困难问题($g, X=g^a, Y=g^b$)。通过与敌手 A 的游戏交互,挑战者 C 将计算出 g^{ab}。挑战者 C 选择一个签名算法(spk, ssk)作为公私钥对,生成文件名 Fid 和时间段 t 的签名。假设敌手 A 进行了 q_k 次私钥查询和 q_s 次标签查询。

1)Setup 阶段:挑战者 C 随机选择 $x\in Z_q^*$ 作为 KGC 私钥 msk,设置 KGC 公钥 mpk 为 $P_0=g^x$ 和协助器公钥 $hpk=X$。挑战者 C 随机选择 $\alpha\in Z_q^*$,并计算 $U=g^\alpha$。最后,挑战者 C 将公共参数(g, U, P_0, X, spk)发送给敌手 A。

2)Query 阶段:H_1, H_2 和 H_3 被认为是由挑战者 C 管理并保存的随机预言。敌手 A 对这些随机预言进行询问,挑战者需要进行相应的回答。

(1)H_1 询问。敌手 A 对身份 ID^* 和时间段 t 进行 H_1 询问,挑战者 C 进行相应的响应,并用 H_1-$Table$ 表保存响应。挑战者 C 初始化表 H_1-$Table$ 为空。当敌手 A 在时间段 t 内进行 H_1 询问时,挑战者 C 进行如下操作。

①如果 H_1-$Table$ 中包含了与输入$\langle t, ID^*\rangle$对应的元组(t, ID^*, c, γ, h_1),挑战者 C 将 h_1 作为响应返回给敌手 A。

②否则,挑战者 C 随机抛掷一枚硬币 $c\in\{0,1\}$,产生 0 的概率是$\frac{q_k+q_s}{q_k+q_s+1}$,产生 1 的概率是$\frac{1}{q_k+q_s+1}$。当 $c=0$ 时,挑战者 C 随机选择一个数 $\gamma\in Z_q^*$,并计算 g^γ。然后,挑战者将元组($t, ID^*, 0, \gamma, h_1=g^\gamma$)添加到表 H_1-$Table$ 中。否则,挑战者 C 随机选择一个数 $\gamma\in Z_q^*$,并计算 Y^γ。然后,挑战者 C 将元组($t, ID^*, 1, \gamma, h_1=Y^\gamma$)添加到表 H_1-$Table$ 中。最后,C 将 h_1 返回给敌手 A。

(2)H_2 询问。挑战者 C 使用一个表 H_2-$Table$ 来保存敌手 A 对 H_2 的询问。首先,挑战者 C 初始化表 H_2-$Table$ 为空。当敌手 A 使用元组$\langle t, ID^*\rangle$进行 H_2 询问时,挑战者 C 进行如下操作。

①输入数据$\langle t, ID^*\rangle$,如果 H_2-$Table$ 存储了对应的元组(t, ID^*, λ, h_2),那么挑战者 C 将 h_2 返回给敌手 A。

②否则,挑战者 C 随机选择 $\lambda\in Z_q^*$,并且计算 Y^λ。然后,将($t, ID^*, \lambda, h_2=Y^\lambda$)添加到 H_2-$Table$ 中。最后,挑战者 C 将 h_2 返回给敌手 A。

(3)H_3 询问。挑战者 C 使用一个表 H_3-$Table$ 来保存敌手 A 对 H_3 的询问。首先,挑战者 C 初始化表 H_3-$Table$ 为空。当敌手 A 使用元组$\langle ID^*, Fid\parallel Cid_i\parallel i\parallel j\parallel t, R\rangle$进行 H_3 询问时,挑战者 C 进行如下操作。

① 输入数据$\langle ID^*, Fid\parallel Cid_i\parallel i\parallel j\parallel t, R\rangle$,如果 H_3-$Table$ 中存储了对应的元组($ID^*, Fid\parallel Cid_i\parallel i\parallel j\parallel t, R, \varphi, h_3$),那么挑战者 C 将 h_3 返回给敌手 A。

②否则,挑战者 C 随机选择 $\varphi\in Z_q^*$,并且计算 g^φ。然后,将($ID^*, Fid\parallel Cid_i\parallel i\parallel j\parallel t, R, \varphi, h_3$)添加到 H_3-$Table$ 中。最后,挑战者 C 将 h_3 返回给敌手 A。

(4)密钥询问。假设敌手 A 已经在时间段 t 内进行了 H_1 询问和 H_2 询问。当敌手 A 在时间段 t 内使用元组$\langle t, ID^*\rangle$进行密钥询问时,挑战者 C 检索 H_1-$Table$,获得了对应的元组(t, ID^*, c, γ, h_1)。挑战者 C 检索 H_2-$Table$,获得了对应的元组(t, ID^*, λ, h_2)。

①如果 $c=1$,C 中止游戏(将此事件表示为 E_1)。

②否则,C 计算 $TSK_t=h_1^{msk}\cdot X^\lambda$。

注意到:$TSK_t=h_1^{msk}\cdot h_2^{hsk}=h_1^{msk}\cdot g^{\lambda\cdot hsk}=h_1^{msk}\cdot X^\lambda$。最后,挑战者 C 将 TSK_t 返回给敌手 A。

(5)标签询问。当敌手 A 使用元组$\langle ID^*, m_{ij}, Fid, Cid_i, i, j, t\rangle$询问对应的标签时,挑战者 C 进行如下操作。

①首先,挑战者 C 从 H_1-$Table$ 表中获得元组(t, ID^*, c, γ, h_1)。

②如果 $c=1$,挑战者 C 将中止游戏(标记此事件为 E_2)。

③否则,挑战者 C 选择 $r\in Z_q^*$,并计算 $R=g^r$。C 再选择 $T_{ij\in R}G_1$。

④挑战者 C 将 $H_3(Fid\parallel Cid_i\parallel i\parallel j\parallel t)$的值定义为$(T_{ij}\cdot X^{-\lambda}\cdot h_1^{-msk})^{r^{-1}}/u^{m_{ij}}$。

注意到:

$$\begin{aligned}T_{ij}&=TSK_t\cdot(H_3(Fid\parallel Cid_i\parallel i\parallel j\parallel t)^r\cdot U^{rm_{ij}}\\&=((T_{ij}\cdot X^{-\lambda}\cdot h_1^{-msk})^{r^{-1}}/U^{m_{ij}})^r\cdot U^{rm_{ij}}\cdot h_1^{-msk}\cdot h_2^{hsk}\\&=T_{ij}\cdot X^{-\lambda}\cdot h_1^{-msk}\cdot h_1^{msk}\cdot X^\lambda\\&=T_{ij}\end{aligned}$$

最后,挑战者 C 将(t, R, T_{ij})返回给敌手 A。

3)Challenge 阶段。挑战者 C 选取一个时间段 t^*。在时间段 t^* 内,敌手 A 没有进行私钥询问。挑战者 C 给敌手 A 发送一个挑战 $chal=(c, k_1, k_2)$和时间段 t^*。同时,挑战者 C 要求敌手 A 在时间段 t^* 内提供相应数据块的证明 P。

4)Forgery 阶段。最后,敌手 A 输出一个证明 $P=(t^*, R, \sigma, M)$。挑战者 C 从 H_1-$Table$ 中获取($t^*, ID^*, c^*, \gamma^*, h_1^*$)。

(1)如果 $c^*=0$,挑战者 C 将中止游戏(标记此事件为 E_3)。

(2)否则，如果 R 和标签集合中真实的 R 不同，那么挑战者 C 从 H_2-*Table* 中获取元组 $(t^*, ID^*, \lambda^*, h_2 = g^{\lambda^*})$，从 H_3-*Table* 中获取相应的元组 $(ID^*, Fid \| Cid_i \| i \| j \| t^*, R, \varphi^*, h_3 = g^{\varphi^*})$。在这种情况下，$h_1^* = Y^{\gamma^*}$。如果伪造是有效的，那么：

$$\begin{aligned} e(\sigma, g) &= e\left(H_1(ID)^{\sum_{(v_k,a_k)\in C} a_k}, P_0^N\right) \cdot e\left(H_2(ID \| t)^{\sum_{(v_k,a_k)\in C} a_k}, P_H^N\right) \cdot e\left(\prod_{i=1}^{N} \prod_{(v_k,a_k)\in C} H_3(Fid \| Cid_i \| i \| v_k \| t, R)^{a_k} \cdot U^M, R\right) \\ &= e\left(Y^{\gamma^* \cdot \sum_{(v_k,a_k)\in C} a_k}, P_0^N\right) \cdot e\left(Y^{\lambda^* \cdot \sum_{(v_k,a_k)\in C} a_k}, X^N\right) \cdot e\left(\prod_{i=1}^{N} \prod_{(v_k,a_k)\in C} g^{\varphi^* \cdot a_k} \cdot g^{\alpha \cdot M^*}, R\right) \\ &= e\left(g, P_0^{N+a\gamma^* + \sum_{(v_k,a_k)} a_k}\right) \cdot e\left(Y, X^{N+\lambda^* \sum_{(v_k,a_k)\in C} a_k}\right) \cdot e\left(g, R^{\prod_{i=1}^{N} \prod_{(v_k,a_k)\in C} g^{\varphi^* a_k + \alpha M^*}}\right) \end{aligned}$$

从而，可以得出以下等式：

$$e(Y, X^{N+\lambda^* \sum_{(v_k,a_k)\in C} a_k}) = e(\sigma, g) \cdot e\left(g, P_0^{-(N+a\gamma^* + \sum_{(v_k,a_k)} a_k)}\right) \cdot e\left(g, R^{-(\prod_{i=1}^{N} \prod_{(v_k,a_k)\in C} g^{\varphi^* a_k + \alpha M^*})}\right)$$

①如果 $\sum_{(v_k,a_k)} a_k = 0$，那么挑战者 C 将中止游戏(标记此事件为 E_4)。

②否则，挑战者 C 可以计算出 g^{ab} 的值为：

$$\left(\sigma \cdot P_0^{-(N+a\gamma^* + \sum_{(v_k,a_k)} a_k)} \cdot R^{-(\prod_{i=1}^{N} \prod_{(v_k,a_k)\in C} g^{\varphi^* \cdot a_k + \alpha M^*})}\right)^{(N+\lambda^* \sum_{(v_k,a_k)} a_k)^{-1}}$$

概率分析：从上述游戏中分析出挑战者 C 不中止的概率为：

$$\begin{aligned} &Pr[\neg E_1 \wedge \neg E_2 \wedge \neg E_3 \wedge \neg E_4] \\ &\geqslant \left(\frac{q_k + q_s}{q_k + q_s + 1}\right)^{q_k} \cdot \left(\frac{q_k + q_s}{q_k + q_s + 1}\right)^{q_s} \cdot \frac{1}{q_k + q_s + 1} \cdot \frac{q-1}{q} \\ &= \left(\frac{q_k + q_s}{q_k + q_s + 1}\right)^{q_k + q_s} \cdot \frac{1}{q_k + q_s + 1} \cdot \frac{q-1}{q} \\ &\geqslant \frac{q-1}{\bar{e} \cdot q \cdot (q_k + q_s + 1)} \end{aligned}$$

其中，$\bar{e} = 2.71828\cdots\cdots$；$q_k$ 是敌手 A 进行密钥询问的次数；q_s 是敌手 A 进行标签询问的次数。

如果敌手 A 成功了，但是敌手 A 的证明 P 中的 R 与标签列表中的 R 不同，挑战者 C 就能以不可忽略的概率解决 G_1 上的 CDH 问题。

游戏 3　游戏 3 与游戏 2 类似，只有一点不同：挑战者 C 保存了敌手 A 进行标签询问的响应列表。挑战者 C 观察这个列表中的每一个实例。如果挑战者 C 发现存在一个实例，在实例中敌手 A 获得了成功，但是敌手 A 给出的聚合标签 σ 与实际的聚合标签 $\sigma = \prod_{\beta \in CT_i} \sigma_\beta$ 不相等，那么 C 将中止游戏。

分析　设与挑战 $chal = (c, k_1, k_2)$ 对应的诚实证明为 $P = \{t, \sigma, M, R, U, T_{Fid}\}$，从而该证明的有效性可以通过以下等式证明：

$$\begin{aligned} e(\sigma, g) &= e\left(H_1(ID)^{\sum_{(v_k,a_k)\in C} a_k}, P_0^N\right) \cdot e\left(H_2(ID \| t)^{\sum_{(v_k,a_k)\in C} a_k}, P_H^N\right) \cdot e\left(\prod_{i=1}^{N} \prod_{(v_k,a_k)\in C} H_3(Fid \| Cid_i \| i \| v_k \| t, R)^{a_k} \cdot U^M, R\right) \end{aligned}$$

设敌手 A 的证明为 $P = \{t, \sigma', M', R, U, T_{Fid}\}$，由于挑战者 C 中止了游戏，因此 $\sigma' \neq \sigma$，但是 σ' 可以使以下等式成立。

$$\begin{aligned} e(\sigma', g) &= e\left(H_1(ID)^{\sum_{(v_k,a_k)\in C} a_k}, P_0^N\right) \cdot e\left(H_2(ID \| t)^{\sum_{(v_k,a_k)\in C} a_k}, P_H^N\right) \cdot e\left(\prod_{i=1}^{N} \prod_{(v_k,a_k)\in C} H_3(Fid \| Cid_i \| i \| v_k \| t, R)^{a_k} \cdot U^{M'}, R\right) \end{aligned}$$

定义 $\Delta M = M' - M$。显然，$\Delta M \neq 0$，否则 $\sigma' = \sigma$，这与假设相矛盾。现在创建一个模拟器来解决 CDH 困难问题。

给模拟器输入一个元组 $(g, g^{a'}, \upsilon)$，模拟器最终输出 $\upsilon^{a'}$。模拟器充当游戏 2 中挑战者的角色，但是有以下不同：

在 Setup 阶段，模拟器设置 $U = g^{\lambda} \upsilon^{\eta}$，其中 $\lambda, \eta \in_R Z_q^*$。模拟器与真实场景一样，产生所有时刻的密钥。

H_3 被视作是由模拟器管理和存储的随机预言。当敌手 A 使用元组 $\langle ID^*, Fid \| Cid_i \| i \| j \| t, R\rangle$ 进行 H_3 询问时，模拟器检查 $\langle ID^*, Fid \| Cid_i \| i \| j \| t, R\rangle$ 是否出现在某个元组 $\langle ID^*, Fid \| Cid_i \| i \| j \| t, R, h, \tau, \gamma\rangle$ 中。如果是，模拟器将 h 返回给敌手；否则，模拟器选择 $\tau \in_R Z_q^*$，计算 $h = g^{\tau}$，并将元组 $\langle ID^*, Fid \| Cid_i \| i \| j \| t, R, h, \tau, *\rangle$ 加入到 H_3-*Table* 中。最后，模拟器将 h 返回给敌手 A。

当敌手 A 在时间段 t 内使用文件名 Fid 询问一个数据块 m_{ij} 的标签时，模拟器选择 $\zeta, r_{ij} \in_R Z_q^*$，计算 $R = (g^{a'})^{\zeta}$，设置 $h = g^{r_{ij}} / (g^{\lambda} \upsilon^{\eta})^{m_{ij}}$。模拟器可以通过以下过程计算数据块的标签：

$$\begin{aligned} T_{ij} &= TSK_t \cdot H_3(Fid \| Cid_i \| i \| j \| t, R)^{a'\xi} \cdot U^{a'\xi m_{ij}} \\ &= TSK_t \cdot (g^{r_{ij}} / (g^{\lambda} \upsilon^{\eta})^{m_{ij}})^{a'\zeta} \cdot (g^{\lambda} \upsilon^{\eta})^{a'\zeta m_{ij}} \\ &= TSK_t \cdot g^{r_{ij} a' \zeta} \end{aligned}$$

注意到模拟器知道临时签名密钥 TSK_t，因为模拟器在时间段开始时生成了密钥。模拟器将元组 $\langle ID^*, Fid \| Cid_i \| i \| j \| t, R, h, r_{ij}, \zeta\rangle$ 加入到表格 H_3-*Table* 中。

如果敌手的证明 $P = \{t, \sigma', M', R, U, T_{Fid}\}$ 中的 σ' 与真实的 σ 不同，但是敌手成功赢得了游戏，那么可以从从 H_3-*Table* 中提取出一个数据元组 $\langle ID^*, Fid \| Cid_i \| i \| j \| t, R, h, r_{ij}, \zeta\rangle$。通过将伪造标签 σ' 满足的等式与真实标签 σ 满足的等式相除，可以得到以下等式：

$$e(\sigma'/\sigma, g) = e(U^{\Delta M}, R) = e((g^{\lambda} \upsilon^{\eta})^{\Delta M}, R)$$

从而可以得到：

$$e(\sigma' \cdot \sigma^{-1} \cdot R^{-\lambda \Delta M}, g) = e(\upsilon^{\eta \Delta M}, g^{a'\zeta})$$

于是，从上式可以得出 CDH 困难问题的解为 $\upsilon^{a'} = (\sigma^{-1} \cdot \sigma' \cdot R^{-\lambda \Delta M})^{(\zeta \eta \Delta M)^{-1}}$。

因此，如果敌手在游戏 2 和游戏 3 中成功的概率之间存在不可忽略的差异，那么构建的模拟器可以解决 CDH 问题。从这里开始，σ 在任何成功的证明中都必须为真实的。

游戏 4：游戏 4 与游戏 3 类似，只有一点不同：挑战者 C 保存了敌手 A 进行标签询问的响应列表。挑战者 C 观察

这个列表中的每一个实例。如果挑战者 C 发现存在一个实例，在实例中敌手 A 获得了成功，但是敌手 A 给出的聚合信息 M 与实际的聚合信息 $M=\sum_{i=1}^{\xi}M_i$ 不相等，那么 C 将中止游戏。

分析：假设 $chal=(c,k_1,k_2)$ 是使挑战者中止游戏的挑战信息，$P'=\{t,\sigma',M',R',U,T_{Fid}\}$ 是敌手提供的证明，而 $P=\{t,\sigma,M,R,U,T_{Fid}\}$ 是真实的数据完整性证明。从游戏 2 和游戏 3 中，我们已经知道 $R=R',\sigma=\sigma'$。因此，只有 P' 中的 M' 会与 P 中的 M 不相同。定义 $\Delta M=M'-M$，显然，$\Delta M\neq 0$。如果敌手能在游戏 4 中使挑战者中止，那么可以构建一个模拟器来解决离散对数问题。

给定模拟器两个值 $g,\upsilon\in G_1$，目标是计算出一个值 x，使其满足 $\upsilon=g^x$。模拟器充当游戏 3 中挑战者的角色，但是有以下不同点。

在 Setup 阶段，模拟器设置 $U=g^{\lambda}\upsilon^{\eta}$，其中 $\lambda,\eta\in_R Z_q^*$。模拟器与敌手进行通信，直到敌手提供的聚合信息 M' 与真实的 M 不同。根据伪造标签 M' 满足的等式与真实标签 M 满足的等式，可以得到以下等式：

$$\begin{aligned}e(\sigma,g)&=e\left(H_1(ID)^{\sum_{(v_k,a_k)\in C}a_k},P_0^N\right)\cdot e\left(H_2(ID\|t)^{\sum_{(v_k,a_k)\in C}a_k},P_H^N\right)\cdot e\left(\prod_{i=1}^{N}\prod_{(v_k,a_k)\in C}H_3(Fid\|Cid_i\|i\|v_k\|t,R)^{a_k}\cdot U^M,R\right)\\&=e(\sigma',g)=e\left(H_1(ID)^{\sum_{(v_k,a_k)\in C}a_k},P_0^N\right)\cdot e\left(H_2(ID\|t)^{\sum_{(v_k,a_k)\in C}a_k},P_H^N\right)\cdot e\left(\prod_{i=1}^{N}\prod_{(v_k,a_k)\in C}H_3(Fid\|Cid_i\|i\|v_k\|t,R)^{a_k}\cdot U^{M'},R\right)\end{aligned}$$

所以可以推导出 $U^M=U^{M'}\Rightarrow U^{\Delta M}=1\Rightarrow g^{\Delta M\lambda}\upsilon^{\Delta M\eta}=1\Rightarrow\upsilon=g^{-\lambda/\eta}$。

因此，如果敌手在游戏 3 和游戏 4 中成功的概率之间存在不可忽略的差异，那么构建的模拟器可以解决离散对数问题。

定理 2 $\left(\frac{m}{n},1-\left(\frac{n-m}{n}\right)^c\right)$可检测性：如果云中共存储了 n 个数据块，其中有 m 个数据块被损坏、修改或者删除，而其中有 c 个数据块被挑战，那么本文的方案就是 $\left(\frac{m}{n},1-\left(\frac{n-m}{n}\right)^c\right)$可检测的。

证明：假设云中一共存储了 n 个数据块，其中有 m 个数据块被损坏、修改或者删除，而其中有 c 个数据块被挑战。于是，当且仅当 TPA 选择的挑战块中至少有一个损坏的数据块时，才能发现有损坏的数据块。我们使用离散随机变量 X 来表示挑战者的数据块与敌手更改的数据块相匹配的数量。使用 P_X 表示挑战者选择的至少一个区块与敌手改变的区块相匹配的概率。于是可以得到以下证明：

$$\begin{aligned}P_X&=P(X\geqslant 1)=1-P(X=0)\\&=1-\frac{n-m}{n}\times\frac{n-1-m}{n-1}\times\cdots\times\frac{n-c+1-m}{n-c+1'}\end{aligned}$$

于是，可以得到 $P_X\geqslant 1-\left(\frac{n-m}{n}\right)^c$。因此，如果云中共存储了 n 个数据块，其中有 m 个数据块被损坏、修改或者删除，而其中有 c 个数据块被挑战，那么本文的方案就是 $\left(\frac{m}{n},1-\left(\frac{n-m}{n}\right)^c\right)$可检测的。

6 性能分析

6.1 功能比较

从实现的功能方面，将本文方案与文献[22]和文献[30]的方案进行了比较，结果如表 3 所列。文献[22]是基于身份的多云多副本可证数据持有方案。文献[30]是基于 PKI 的抗密钥泄露可证数据持有方案，该方案基于单云场景实现。与文献[22]相比，本文方案结合了密钥隔离的思想，实现了前向和后向安全的 PDP 模型，安全性比文献[22]更高。本文方案与文献[30]的方案都使用密钥隔离的思想实现了 PDP 模型。与文献[30]的方案相比，本文方案使用了基于身份的体制，消除了 PKI 中复杂的证书管理问题；同时，本文方案将密钥隔离的思想应用到了多云多副本场景。

表 3 功能比较

Table 3 Comparison of functions

方案	文献[22]	文献[30]	本文方案
基于身份	√	×	√
多云	√	×	√
多副本	√	×	√
密钥隔离	×	√	√

6.2 性能评估

从计算开销和通信开销两方面，将方案的性能总结如下。

1)计算代价

设 T_p，$T_{\exp}$ 和 T_{mul} 分别表示 G_1 上的配对、幂指和乘法运算。其他的一些运算，比如 Z_q 上的哈希、加法和乘法，是可以省略的，这是因为它们的计算代价几乎可以忽略不计。假设用户在 ξ 个云服务器上存储了 N 个不同的副本。每个云服务器上的副本数量是 $|CT_i|$。每个副本有 n 个数据块，TPA 每次挑战每个副本上的 c 个数据块。Extract 算法需要 $(2T_{\exp}+T_{\mathrm{mul}})$的代价来产生用户的初始签名密钥。UMGen 算法每次需要消耗 $(T_{\exp}+T_{\mathrm{mul}})$的开销计算更新消息。CKey-Update 算法每次需要 T_{mul} 的计算开销更新用户的临时密钥。为了产生所有副本的所有数据块标签，TagGen 算法执行一次的计算开销之和是 $N\cdot n\cdot(2T_{\exp}+2T_{\mathrm{mul}})$。Challenge 算法的计算开销几乎可以忽略不计。每个 CSS 运行 ProofGen 算法来生成完整性证明。假设 CSS 中存储了 L 个副本，所以 ProofGen 算法产生的计算开销是 $L\cdot c\cdot T_{\exp}+(L\cdot c-1)T_{\mathrm{mul}}$。CO 执行 ProofAgree 算法来聚合来自不同云服务器的证明，产生的计算开销是 $(\xi-1)\cdot T_{\mathrm{mul}}$。为了检测文件的完整性，TPA 执行 Verify 算法进行验证的计算开销是 $4T_p+(c\cdot N+5)T_{\exp}+c\cdot N\cdot T_{\mathrm{mul}}$。将本文 IDKIMC-PDP 方案的计算开销与文献[22]和文献[30]中方案的计算开销进行了比较，结果如表 4 所列。从计算代价上比较，本文方案在客户端进行密钥更新时的计算代价比文献[30]方案的低，仅需要进行一次群 G_1 上的乘法运算。但是，本文方案在产生数据块标签、生成证明和验证方面的开销比文献[30]方案的高，这是由于文献[30]的方案是基于 PKI 的密钥隔离的单云方案，而本文是基于身份的密钥隔离的多云多副本方案，本文中的应用场景比文献[30]中的场景复杂，因此本文方案的计算代价比

文献[30]方案的高。文献[22]是没有结合密钥隔离的多云多副本的可证数据持有方案。与文献[22]中的方案相比，当文件副本的数量相同时，本文方案在产生数据块标签、生成证明方面的时间开销与文献[22]方案接近，但是本文方案在验证阶段的时间开销比文献[22]方案的低。这是因为文献[22]进一步将每个数据块分割为 s 个扇区，所以在验证阶段需多进行 s 次群 G_1 上的乘法计算和($s-2$)次群 G_1 上的幂指计算，而本文方案中没有将数据块分割为扇区，因而验证的效率更高。同时，由于使用了密钥隔离，本文方案的安全性更高。

表 4　计算代价比较

Table 4　Comparison of computation costs

方案	文献[22]的方案	文献[30]的方案	本文方案
密钥更新	—	$T_{\exp}+T_{mul}$	T_{mul}
生成标签	$N\cdot n\cdot(2T_{\exp}+2T_{mul})$	$n\cdot(2T_{\exp}+2T_{mul})$	$N\cdot n\cdot(2T_{\exp}+2T_{mul})$
生成证明	$L\cdot c\cdot(T_{\exp}+T_{mul})$	$c\cdot T_{\exp}+(c-1)T_{mul}$	$L\cdot c\cdot T_{\exp}+(L\cdot c-1)T_{mul}$
聚合	$\xi\cdot T_{mul}$	—	$(\xi-1)\cdot T_{mul}$
验证证明	$3T_p+(c\cdot N+s+3)T_{\exp}+(c\cdot N+s)\cdot T_{mul}$	$3T_p+(c+2)T_{\exp}+(c+1)T_{mul}$	$4T_p+(c\cdot N+5)T_{\exp}+c\cdot N\cdot T_{mul}$

2)通信代价

为了验证云服务器上文件的数据完整性，TPA 将挑战信息 $chal=(c,k_1,k_2)$ 和文件标识 Fid 发送给 CO。于是，TPA 挑战请求的通信代价是 $\log_2 c+2\log_2 q+|Fid|$ 比特。然后，CO 将 $P=\{t,\sigma,M,R,U,T_{Fid}\}$ 作为挑战响应返回给 TPA。于是，挑战响应的代价是 $\log_2 q+3|G_1|+|T_{Fid}|$ 比特。将本文方案的通信代价与文献[22]和文献[30]中的方案进行了比较，结果如表 5 所列。与文献[30]的方案相比，本文方案的通信代价较低，这是由于文献[30]中的方案是由 TPA 生成了 c 对随机的数据块索引和系数对，然后传递给云服务器。而在本文中，TPA 只需要传递两个随机数种子给云服务器，由云服务器通过公开的伪随机排列和伪随机函数计算产生挑战的数据块索引和系数对，从而减少了数据传递。因此，本文方案的通信代价比文献[30]方案的低。与同是多云多副本场景的文献[22]相比，本文方案通信代价较低。这是由于文献[22]将每个数据块分割为 s 个扇区后，在返回证明阶段需要多传递($s-1$)个群 G_1 上的元素和($s-1$)个 Z_q^* 上的整数；而本文方案没有将数据块分割为扇区，因此在通信代价上，本文方案比文献[22]的方案更低。

表 5　通信代价比较

Table 5　Comparison of communication costs

方案	文献[22]的方案	文献[30]的方案	本文方案
发送挑战	$\log_2 c+2\log_2 q+\|Fid\|$	$\log_2 c+c\log_2 n+c\log_2 q$	$\log_2 c+2\log_2 q+\|Fid\|$
返回证明	$s\log_2 q+(s+2)\|G_1\|+\|T_{Fid}\|$	$\log_2 q+2\|G_1\|$	$\log_2 q+3\|G_1\|+\|T_{Fid}\|$

6.3　实验结果

基于 JPBC 库实现了本文方案。实验使用的设备为 DELL 笔记本电脑，配置为 AMD Ryzen 5 5625U with Radeon Graphics，24 GB 运行内存和 Win11 操作系统。对密钥更新、生成数据块标签、生成某一时间段内 N 个副本的完整性证明和相应的验证时间开销进行测试，并与文献[22]和文献[30]方案进行了比较。

在第一个实验中测试了在密钥更新阶段的时间开销。对从时间段 1 到时间段 6 进行密钥更新的时间开销进行了测试，并将本文进行密钥更新的时间开销与文献[30]方案的进行了对比，实验结果如图 2 所示。

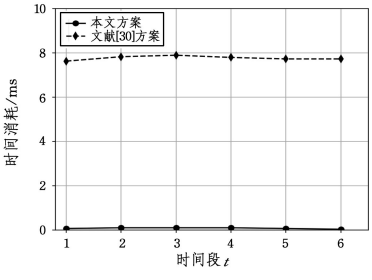

图 2　密钥更新的计算开销比较

Fig. 2　Computation cost of key update

从图 2 可以看出，本文方案执行 1 次密钥更新的平均时间约为 0.1 ms，时间开销很小，而文献[30]执行 1 次密钥更新的平均时间约为 7.8 ms。这是由于文献[30]进行密钥更新时多进行了一次群 G_1 上的幂指运算。因此，在密钥更新方面，本文方案比文献[30]的方案效率高。

在第二个实验中测试了生成标签的计算开销。在实验中，我们创建了一个大小约为 6 MB 的文件和对应的 10 个副本，每个副本分成了 2 000 个数据块，在模拟文献[22]的实验中进一步将每个数据块分割为 200 个扇区，然后将 10 个副本和标签存放在 3 个 CSS 上。我们计算了生成 100 到 600 个数据块标签需要的时间开销，并与文献[22]进行了比较。从 100 个数据块开始，每次测试增加 100 个数据块，实验结果如图 3 所示。可以看出，当数据块数量相同时，两种方案在生成数据块标签上的时间比较接近。因此，文中的方案生成数据块标签的效率与文献[22]中的方案相近。

第三个实验评估生成证明和进行验证的计算时间开销。在实验二的基础上，测试了在不同数量的挑战块下生成证明和进行验证的时间开销，并与文献[22]中的方案进行对比。在实验中，将挑战数据块的数量从 10 增加到 60，每次测试增加 10 个数据块，实验结果如图 4 所示。从图 4 可以看出，无论是本文方案还是文献[22]的方案，生成证明的时间开销都与挑战块的数量呈线性关系。当挑战的数据块数量相同时，两种方案生成证明的时间开销比较接近。因此，本文方案在

生成证明的效率上与文献[22]方案相近。从图5可以看出，在验证阶段，本文方案的效率比文献[22]方案高，这是因为文献[22]中进一步将每个数据块分割为 s 个扇区，所以在验证阶段需多进行 s 次群 G_1 上的乘法计算和 $(s-2)$ 次群 G_1 上的幂指计算。而在本文方案中没有将数据块分割为扇区，因此，本文的 IDKIMC-PDP 方案在验证阶段效率比文献[22]方案的效率高。

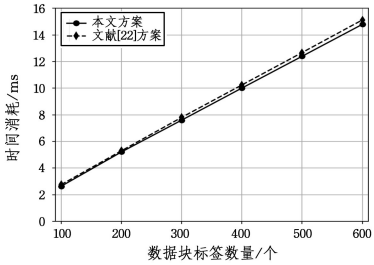

图3 生成数据块标签的计算开销比较

Fig. 3 Computation cost of tag generation

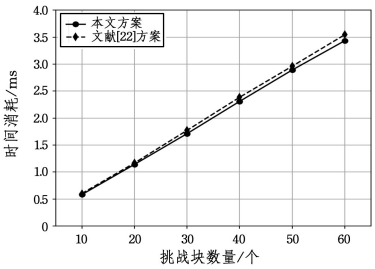

图4 生成证明的计算开销比较

Fig. 4 Computation cost of proof generation

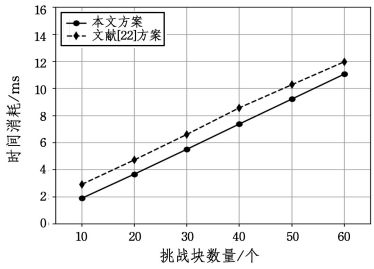

图5 验证的计算开销比较

Fig. 5 Computation cost of verification

结束语 本文提出了一个基于身份的密钥隔离的多云多副本可证数据持有方案，用于检查分布式云服务器上副本的完整性。在 CDH 假设上，证明了方案的安全性。基于身份的加密签名体制消除了 PKI 中复杂的证书管理。密钥隔离技术降低了密钥泄露带来的风险，实现了前向和后向安全的 PDP 模型。实验结果表明，与现有的多云多副本有关方案相比，本文方案具有相对较高的效率。同时，本文方案由于结合了密钥隔离的思想，安全性更高。未来，我们将探索更多抗密钥泄露的技术，并将其应用到 PDP 模型中。

参 考 文 献

[1] WANG W, REN L, CHEN L, et al. Intrusion detection and security calculation in industrial cloud storage based on an improved dynamic immune algorithm[J]. Information Sciences, 2018, 501: 543-557.

[2] ZAFAR F, KHAN A, MALIK S U R, et al. A survey of cloud computing data integrity schemes: design challenges, taxonomy and future trends[J]. Computers & Security, 2017, 65: 29-49.

[3] DODIS Y, KATZ J, XU S H, et al. Key-insulated public key cryptosystems[C]// Proceedings of the Eurocrypt 2002. Berlin, Heidelberg: Springer, 2002: 65-82.

[4] 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. New York: Association for Computing Machinery, 2007: 598-609.

[5] ERWAY C, KÜPÇÜ A, PAPAMANTHOU C, et al. Dynamic provable data possession[J]. ACM Transactions on Information and System Security, 2015, 17(4): 15.

[6] 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.

[7] ZHU Y, HU H X, YU M Y, et al. Cooperative Provable Data Possession for Integrity Verification in Multicloud Storage[J]. IEEE Transactions on Parallel and Distributed Systems, 2012, 23(12): 2231-2244.

[8] WANG H Q, WU Q H, QIN B, et al. Identity-based remote data possession checking in public clouds[J]. IET Information Security, 2014, 8(2): 114-121.

[9] CHEN R N, LI Y N, YU Y, et al. Blockchain-BasedDynamic Provable Data Possession for Smart Cities[J]. IEEE Internet of Things Journal, 2020, 7(5): 4143-4154.

[10] WANG H Q, WANG Q H, HE D B. Blockchain-Based Private Provable Data Possession[J]. IEEE Transactions on Dependable and Secure Computing, 2021, 18(5): 2379-2389.

[11] DU J M, DONG G F, NING J G, et al. A Blockchain-Assisted Certificateless Public Cloud Data Integrity Auditing Scheme[J]. IEEE Access, 2023, 11: 123018-123029.

[12] YANG X, WU L B, ZHANG Z Z, et al. Survey on Blockchain-based Integrity Validating for Cloud Data[J]. Journal of Chinese Computer Systems, 2023, 44(11): 2369-2376.

[13] WANG H Q, HE D B, YU J, et al. Incentive and Unconditionally Anonymous Identity-Based Public Provable Data Possession [J]. IEEE Transactions on Services Computing, 2019, 12(5): 824-835.

[14] ZHANG X J, LIU Q, ZHENG S, et al. Verifiable Cloud Data Sharing Scheme that Supports Privacy Protection[J]. Computer Engineering, 2023, 49(3): 49-57.

[15] LI T, WANG H Q, HE D B, et al. Synchronized Provable Data Possession Based on Blockchain for Digital Twin[J]. IEEE Transactions on Information Forensics and Security, 2022, 17: 472-485.

[16] YANG Y, CHEN Y J, CHEN F, et al. An Efficient Identity-Based Provable Data Possession Protocol With Compressed Cloud Storage[J]. IEEE Transactions on Information Forensics and Security, 2022, 17: 1359-1371.

[17] CURTMOLA R, KHAN O, BURNS R, et al. MR-PDP: Multi-

ple-Replica Provable Data Possession[C]// Proceedings of the 28th International Conference on Distributed Computing Systems. IEEE，2008：411-420.

[18] YUAN Y L，ZHANG J B，XU W S. Dynamic Multiple-Replica Provable Data Possession in Cloud Storage System[J]. IEEE Access，2020，8：120778-120784.

[19] LIU Z P，LIU Y，YANG X W，et al. Integrity Auditing for Multi-Copy in Cloud Storage Based on Red-Black Tree[J]. IEEE Access，2021，9：75117-75131.

[20] ZHOU L，FU A M，YANG G M，et al. Efficient Certificateless Multi-Copy Integrity Auditing Scheme Supporting Data Dynamics[J]. IEEE Transactions on Dependable and Secure Computing，2022，19(2)：1118-1132.

[21] ZHOU L，FU A M，MU Y，et al. Multicopy provable data possession scheme supporting data dynamics for cloud-based Electronic Medical Record system[J]. Information Sciences，2021，545：254-276.

[22] LI J G，YAN H，ZHANG Y C. Efficient Identity-Based Provable Multi-Copy Data Possession in Multi-Cloud Storage[J]. IEEE Transactions on Cloud Computing，2022，10(1)：356-365.

[23] MIAO Y，HUANG Q，XIAO M Y，et al. Blockchain Assisted Multi-Copy Provable Data Possession With Faults Localization in Multi-Cloud Storage[J]. IEEE Transactions on Information Forensics and Security，2022，17：3663-3676.

[24] WENG J，LIU S L，CHEN K F，et al. Identity-Based Key-Insulated Signature with Secure Key-Updates[C]// Proceedings of International Conference on Information Security and Cryptology. Berlin，Heidelberg：Springer，2006：13-26.

[25] VASUDEVA REDDY P，GOPAL P V S S N. Identity-based key-insulated aggregate signature scheme[J]. Journal of King Saud University-Computer and Information Sciences，2017，29(3)：303-310.

[26] HANAOKA G，HANAOKA Y，IMAI H. Parallel Key-Insulated Public Key Encryption[C]// Proceedings of the PKC 2006. Berlin，Heidelberg：Springer，2006：105-122.

[27] HOU Y，XIONG H，HUANG X，et al. Certificate-Based Parallel Key-Insulated Aggregate Signature Against Fully Chosen Key Attacks for Industrial Internet of Things[J]. IEEE Internet of Things Journal，2021，8(11)：8935-8948.

[28] CUI J，LU J，ZHONG H，et al. Parallel Key-Insulated Multiuser Searchable Encryption for Industrial Internet of Things[J]. IEEE Transactions on Industrial Informatics，2022，18(7)：4875-4883.

[29] YU J，REN K，WANG C，et al. Enabling Cloud Storage Auditing With Key-Exposure Resistance[J]. IEEE Transactions on Information Forensics and Security，2015，10(6)：1167-1179.

[30] YU J，WANG H Q. Strong Key-Exposure Resilient Auditing for Secure Cloud Storage[J]. IEEE Transactions on Information Forensics and Security，2017，12(8)：1931-1940.

ZHOU Jie，born in 1997，postgraduate. His main research interests include cryptography and information security.

WANG Huaqun，born in 1974，Ph. D，professor. His main research interests include cryptography，blockchain and cloud computing security.

（责任编辑：柯颖）