一种支持完美隐私保护的批处理数据拥有性证明方案

庞晓琼 任孟琦 王田琪 陈文俊 聂梦飞
(中北大学大数据学院 太原 030051)

摘 要 数据拥有性证明技术是当前云存储安全领域中的一项重要研究内容,可使用户无须下载所有文件就能高效地远程校验用户数据是否完整存储于云服务器。现实中,用户趋向于委托第三方验证机构 TPA 代替自己来验证数据的完整性;然而,多数支持第三方公开审计的数据拥有性证明方案通常只考虑恶意服务器是否能够伪造标签或证明的问题,鲜有考虑恶意 TPA 可能会窃取用户隐私的情况。近几年,一些既针对服务器保证数据的安全性又针对 TPA 实现数据隐私保护的数据拥有性证明方案逐渐被提出,但多应用于单云服务器环境下;个别应用在多云服务器环境下可支持批量审计的方案,或者不能有效抵抗恶意云服务器的攻击,或者无法实现针对 TPA 的零知识隐私保护。因此,文中在 Yu 等工作的基础上,提出了一个多云服务器环境下支持批量审计的数据拥有性证明方案。所提方案既可保证针对恶意云服务器的安全性,还可实现针对 TPA 的完美零知识隐私保护。性能分析及仿真实验表明所提方案是高效且可行的。

关键词 零知识隐私,批处理校验,数据拥有性证明,云存储安全

中图法分类号 TP309 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2018.11.019

Perfect Privacy-preserving Batch Provable Data Possession

PANG Xiao-qiong REN Meng-qi WANG Tian-qi CHEN Wen-jun NIE Meng-fei
(School of Data Science and Technology, North University of China, Taiyuan 030051, China)

Abstract Provable data possession is an important research direction in the field of current cloud storage security. It allows user to verify whether his outsourced data stored in the cloud sever are complete without downloading all files efficiently and remotely. Currently, users tend to entrust TPA, a Third Party Auditor, to verify the integrity of their data instead of themselves. However, most of public auditing PDP schemes only consider whether malicious servers can forge data labels or proofs, rarely consider the case of whether malicious TPA may steal user's data privacy. In recent years, some of PDP schemes that both ensure the data security in server and protect the data privacy for TPA have been proposed and applied in single-cloud server. Few of batch PDP protocols applied in multi-cloud server can effectively resist the malicious cloud server's attack and achieve zero knowledge privacy for TPA. Therefore, based on the work proposed by Yu et al, this paper proposed a batch PDP scheme which can both guarantee the data security for malicious cloud servers and realize the perfect data privacy for TPA. Performance analysis and simulation experiments demonstrate the efficiency and feasibility of the proposed protocol.

Keywords Zero knowledge privacy, Batch verification, Provable data possession(PDP), Cloud storage security

随着云计算的飞速发展,云存储安全问题引起了学术界和产业界的广泛关注。用户将数据存储于远程服务器并享受其带来的数据分析、处理、存储等服务的便利。但是,用户也需要一种机制来验证存储在服务器上的数据是否遭到恶意损毁或删除。数据拥有性证明(Provable Data Possession, PDP)技术能使用户在无须下载原数据的情况下,通过简单的交互,以很高的效率校验存储在远程服务器上的数据的完整性。

早期的 PDP 方案多考虑由用户完成远程校验工作,用户以概率抽查的方式对远程服务器发起挑战,服务器利用存储的数据和元数据产生回应并返回给用户,用户根据服务器的回应判断其数据是否完整。数据完整性验证工作给用户带来了计算和存储方面的负担,因此用户更倾向于委托第三方代理机构 TPA(Third Party Auditor)代替自己来完成审计工作。TPA 的引入虽然有效地减轻了用户的审计负担,但却带来了新的安全问题,即恶意 TPA 可能从多次审计结果中分析用户数据的相关信息,造成用户数据的隐私泄露,因此在支持

到稿日期:2018-07-15 返修日期:2018-09-24 本文受山西省青年自然科学基金(201601D021075),山西省回国留学人员科研项目(2015-083),山西省研究生教育改革研究项目(2018JG62)资助。

庞晓琼(1982—),女,博士,讲师,硕士生导师,CCF 会员,主要研究方向为信息安全与密码学、复杂系统的故障预测与健康管理,E-mail:xqpang@nuc.edu.cn(通信作者);**任孟琦**(1994—),女,硕士生,主要研究方向为信息安全与密码学;**王田琪**(1993—),女,硕士生,主要研究方向为信息安全与密码学;**陈文俊**(1980—),男,博士生,主要研究方向为金融信息安全;**聂梦飞**(1994—),女,硕士生,主要研究方向为信息安全与密码学。

公开审计的 PDP 方案中需要考虑恶意 TPA 造成用户数据隐私泄露的问题。同时，在现实应用中，用户会将多个文件数据存储至多个云服务器，TPA 会收到用户请求处理多个文件的任务，此时若 TPA 能对服务器返回的有关用户多个文件的证明进行批处理校验，则能有效提高审计效率并降低通信复杂度。

因此，研究在多服务器环境下支持批量审计，且实现针对云服务器的安全性和 TPA 零隐私性的数据拥有性证明方案，具有理论意义和实践价值。

1　相关工作

Ateniese 等[1]于 2007 年首次提出 PDP 方案，方案中用户随机选取部分数据块索引向服务器发起挑战，服务器根据收到的挑战计算证明，用户通过校验该证明判断其远程数据的完整性。Ateniese 等[2]于 2008 年基于对称加密技术提出了一种高效且安全可证的 PDP 方案，但该方案不能实现公开审计。

为了支持公开校验，Wang 等[3]于 2009 年首次引入第三方审计员(TPA)实现了 PDP 方案的公开审计。Zhang 等[4]于 2014 年提出一个单服务器环境下支持 TPA 公开校验且对多文件批处理的 PDP 方案。2015 年至 2016 年期间，Yu 等[5-7]对公开审计 PDP 方案中存在的恶意服务器伪造证明的攻击行为进行了研究，并提出了改进方案。然而，上述方案均没有考虑到公开审计中针对 TPA 保护用户数据隐私性的问题。

为了实现针对 TPA 的隐私性，Wang 等[8]于 2010 年利用随机掩码技术提出了一种隐私保护的 PDP 方案。Hao 等[9]于 2011 年给出“针对 TPA 隐私性”的形式化定义，并在此定义下提出可证明安全的 PDP 方案。Yu 等[10]于 2015 年利用零知识隐私加强了文献[9]中针对 TPA 的隐私性，并提出了一种改进的 PDP 方案。Yu 等[11]于 2017 年又提出了一种基于身份的支持完美隐私保护的 PDP 方案。但是，上述方案均是在单服务器环境下提出的。

为了提高多服务器环境下的应用效率，也有学者对多服务器环境下支持批处理的 PDP 方案进行了研究。Zhu 等[12]于 2012 年首先构造了多云服务器环境下的合作 PDP 模型，方案支持针对 TPA 的隐私保护，但被指出容易遭受恶意服务器的攻击。He 等[13]于 2013 年设计了一种多云服务器环境下支持批量审计的 PDP 方案，该方案实现了针对 TPA 的隐私保护，但没有给出针对恶意服务器的安全性证明。Wang 等[14]于 2015 年提出基于身份的分布式 PDP 方案，该方案在多云服务器环境下支持 TPA 对用户多个文件的批处理校验，但没有针对 TPA 的数据隐私性证明，且在恶意服务器删除用户数据的情况下，仅保存哈希值也能通过完整性验证。

基于以上分析，本文在 Yu 等[11]工作的基础上，提出了分挑战和聚合证明算法，实现了多云服务器环境下支持公开批量审计的 PDP 方案。该方案既保证了针对恶意云服务器的安全性，又实现了针对 TPA 的零知识隐私保护。

2　预备知识

2.1　双线性对

设 q 是一个大素数，群 G_1 和 G_2 是两个阶为 q 的乘法循环群，g 为群 G_1 的生成元，群 G_1 到 G_2 的一个双线性映射 e：$G_1\times G_1\rightarrow G_2$ 满足如下性质：

(1)双线性：对于任意的 $u,v\in G_1$ 和 $x,y\in Z_q$，有 $e(u^x,v^y)=e(u,v)^{xy}$。

(2)非退化性：$e(g,g)$ 的值≠群 G_2 中的单位元。

(3)可计算性：对于任意的 $u,v\in G_1$，存在一个有效的算法可以计算 $e(u,v)$。

2.2　离散对数相等协议

设 q 是一个大素数，群 G 是阶为 q 的乘法循环群，g_1 和 g_2 均为 G 的生成元，证明者 P 向验证者 V 证明：群元素 Y_1，Y_2 在 g_1,g_2 上有相同的离散对数 x。协议交互如下：

(1)承诺：P 随机选择 $\rho\in Z_q$，计算 $T_1=g_1^{\rho}$，$T_2=g_2^{\rho}$，将 T_1,T_2 发送给 V。

(2)挑战：V 随机选取 $c\in\{0,1\}^{\lambda}$，并将 c 返回给 P。

(3)回应：P 计算 $z=\rho-cx(\mathrm{mod}\ q)$，将 z 返回给 V。

(4)验证：V 检查 $T_1=g_1^zY_1^c\wedge T_2=g_2^zY_2^c$ 是否成立。若成立，则接受 P 的证明；否则拒绝。

以上协议可以转化为一种非交互式的零知识证明协议：$POK\{(x):Y_1=g_1^x\wedge Y_2=g_2^x\}$，即在挑战中令 $c=H(T_1\parallel T_2)$，H 为安全的哈希函数。

2.3　基于身份的签名

一种基于身份的签名方案由 4 个概率多项式时间算法组成。

(1)Setup(k)：初始化算法。输入为安全参数 k，输出为主私钥 msk 和主公钥 mpk。

(2)Extract(msk,ID)：提取算法。输入为用户的身份 ID 和主私钥 msk，产生一个用户私钥 sk 作为输出。

(3)Sign(ID,sk,m)：签名算法。输入为用户身份 ID、用户私钥 sk、消息 m，输出为消息 m 的签名 T。

(4)Verify(ID,m,T,mpk)：验证算法的输入为用户身份 ID、主公钥 mpk、消息 m 及其签名 T，若签名有效则输出 1，否则输出 0。

3　定义与模型

3.1　系统模型

多服务器环境下基于身份的批处理数据拥有性证明系统(Identity Based Batch Provable Data Possession，IDBPDP)包含 5 类实体：用户(DO)、云服务器(CS)、服务器组织者(Organiser)、第三方审计员(TPA)、秘钥生成中心(KGC)。

(1)用户。用户使用多个服务器提供的存储服务对全部文件进行预处理，对得到的所有数据块计算数据标签，将数据块及其对应的数据标签均匀地存储在不同的云服务器中。用户构建一张索引集合表用于记录与数据块相关的存储索引信息，并将其发送给服务器组织者。

(2)云服务器。云服务器是指由不同的云服务商提供的服务器资源，具有极大的存储空间和强大的计算能力。本系统设定有多个云服务器，每个云服务器上存储着用户提交的数据块和数据标签，在收到挑战时计算相关证明并返回。

(3)服务器组织者。服务器组织者在云服务器和第三方审计员之间起着连接作用。第三方审计员收到用户的审计请

求后，向服务器组织者发送总挑战，服务器组织者根据在其上存储的索引集合表向被挑战服务器分发若干个分挑战。服务器组织者根据多个服务器返回的分证明，计算聚合证明并返回给第三方审计员进行批校验。

(4)第三方审计员。经用户授权，第三方审计员验证远程服务器上存储数据的完整性。第三方审计员向服务器组织者发起总挑战，并对其返回的聚合证明进行批处理校验，将验证结果返回给用户。

(5)秘钥生成中心。秘钥生成中心生成系统的公开参数，并根据用户的身份 ID 和系统主私钥计算用户私钥，然后通过一个安全的通道返回给用户。

IDBPDP 系统模型如图 1 所示。

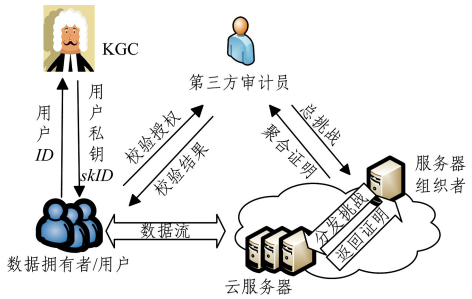

图 1 IDBPDP 系统模型

Fig. 1 System model of IDBPDP

3.2 符号定义

方案中涉及到的符号及其描述如表 1 所列。

表 1 符号及描述

Table 1 Notations and descriptions

符号	描述
mpk	KGC 使用的主公钥
msk	KGC 使用的主私钥
DO	用户
ID	DO 的身份
IDS	以 DO 身份进行的签名
sk	DO 的私钥
pk	DO 的公钥
CS_j	第 j 个服务器
M	DO 的文件集合
m_{jk}	存放在第 j 个服务器上的第 k 个文件块
T_{jk}	m_{jk} 的数据标签
c	TPA 选取的被挑战数据块的块数
U	TPA 选中的被挑战服务器的索引集合
$chal$	TPA 选取的总挑战
P_j	CS_j 返回的分证明
P	服务器组织者返回的聚合证明
$\mathcal{M}$	用户 DO 构建的索引集合表
$x \xleftarrow{R} Z_q$	从集合 Z_q 中随机且均匀地选取 x

3.3 方案框架

定义 1 多服务器环境下支持完美隐私保护的批处理数据拥有性证明方案 $IDBPDP=$(Setup, Extract, TagGen, Challenge, Prove, BatchVerify)包括 6 个算法。

(1) Setup(1^k)→($params$, msk, mpk)：初始化算法，由 KGC 运行，以安全参数 k 为输入，输出公共参数 $params$ 和主密钥对(mpk, msk)，msk 仅为 KGC 所知。

(2)Extract($params$, msk, ID)→(sk, pk)：提取算法，由 KGC 运行，以参数 $params$、主私钥 msk 和用户身份 $ID\in\{0,1\}^*$ 为输入，输出用户私钥 sk 和公钥 pk。

(3)TagGen($params$, ID, sk, $\{m_{jk}\}$)→($\{T_{jk}\}$, IDS, $\mathcal{M}$)：标签生成算法，由用户执行，以公共参数 $params$、用户身份 ID、用户私钥 sk 及其数据块集合$\{m_{jk}\}$为输入，计算数据标签的集合$\{T_{jk}\}$并输出，将$\{m_{jk}\}$和$\{T_{jk}\}$存储在服务器 CS_j 上。用户构建一张索引集合表 $\mathcal{M}$，用于记录与数据块相关的存储索引信息，并将表 $\mathcal{M}$ 及基于身份进行的签名 IDS 发送给 Organiser。

(4)Challenge($params$, n, ID)→($chal$, $\{chal_j\}$)，其包含两个子算法。

1)Challenge1($params$, n, ID)→($chal$)：总挑战算法，由 TPA 执行，以公共参数 $params$、用户身份 ID、随机选取的挑战块索引 n 为输入，生成总挑战 $chal$ 并输出，将 $chal$ 发送给 Organiser。

2)Challenge2($chal$, $\mathcal{M}$)→($\{chal_j\}$)：分挑战算法，由 Organiser 运行，Organiser 收到 TPA 发送的总挑战 $chal$ 后，依据索引集合表 $\mathcal{M}$ 计算分挑战$\{chal_j\}$，并将其发送给被挑战服务器$\{CS_j\}$。

(5)Prove($params$, $chal_j$, ID, $\{T_{jk}\}$, $\{m_{jk}\}$)→$\{P_j, P\}$，其包含两个子算法。

1)Prove1($params$, $chal_j$, ID, $\{T_{jk}\}$, $\{m_{jk}\}$)→$\{P_j\}$：分证明算法，由所有被挑战的服务器 CS_j 运行，收到挑战 $chal_j$ 的 CS_j 以公共参数 $params$、分挑战 $chal_j$、用户身份 ID、数据标签集合$\{T_{jk}\}$和 CS_j 上存储的文件块集合$\{m_{jk}\}$为输入，计算分证明 P_j，并将 P_j 发送给 Organiser。

2)Prove2($params$, $chal$, ID, $\{P_j\}$)→(P)：聚合证明算法，由 Organiser 运行，以公共参数 $params$、用户身份 ID、总挑战 $chal$ 和所有被挑战服务器返回的证明$\{P_j\}$为输入，计算聚合证明 P 并返回给 TPA。

(6)BatchVerify($params$, ID, $chal$, $\{P\}$)→$\{0,1\}$：批校验算法，由 TPA 运行，输入公共参数 $params$、总挑战 $chal$、用户身份 ID、Organiser 返回的证明 P。TPA 批校验证明 P 的有效性，若有效则输出 1，否则输出 0。

3.4 安全定义

从以下两个方面来考虑方案的安全性：1)针对恶意服务器的安全性；2)针对 TPA 的完美数据隐私保护。定义 2 刻画了针对恶意服务器的安全性游戏，游戏中敌手扮演不可信云服务器的角色，挑战者扮演用户的角色。完美隐私性指 TPA 在对用户的数据进行远程完整性验证时，得不到有关用户数据的任何信息，定义 3 通过模拟器 S 给出其形式化的定义。

定义 2(针对恶意服务器的安全性游戏)

(1) Setup：挑战者运行 Setup(1^k)，得到($params$, msk, mpk)，将($params$, mpk)发送给敌手，msk 私密保存。

(2)Queries：敌手 $\mathcal{A}$ 适应性地对挑战者 $\mathcal{V}$ 做 Extract Queries 和 TagGen Queries 两种询问。

1)Extract Queries：敌手 $\mathcal{A}$ 询问 ID_i 的私钥。$\mathcal{V}$ 运行算法 Extract($params$, msk, ID_i)，得到私钥 sk_i，并将其发送给 $\mathcal{A}$。

2)TagGen Queries：敌手 $\mathcal{A}$ 询问身份为 ID_i 的用户所拥

有的文件块 m_{jk} 的数据标签 T_{jk}，$\mathcal{V}$ 首先运行 Extract 得到私钥 sk_i，然后运行 TagGen($params$, ID, sk, $\{m_{jk}\}$)得到文件块集合$\{m_{jk}\}$的标签集合$\{T_{jk}\}$，并将$\{T_{jk}\}$发送给 $\mathcal{A}$。

(3)Prove：敌手 $\mathcal{A}$ 询问文件 M 的数据标签，并以用户身份 ID、文件块索引 n 为输入，在算法执行过程中，挑战者扮演 TPA 的角色，$\mathcal{A}$ 扮演证明者的角色。最后协议执行完毕时，敌手从挑战者处得到输出 P。

(4)Output：敌手选择身份为 $ID^{\dagger}$ 的用户，用户拥有的文件为 $M^{\dagger}$，要求敌手之前未对 $ID^{\dagger}$ 进行过 Extract Query，但对 $ID^{\dagger}$ 和 $M^{\dagger}$ 进行过 TagGen Query，那么如果敌手输出一个 ε-允许的证明者 $P^{\dagger}$，则称敌手赢得了游戏。现将 ε-允许和 ε-安全的定义描述如下：

1)ε-允许：如果具有欺骗性的证明者 $P^{\dagger}$ 能够有效地回应 ε-部分的完整性挑战，即 $\Pr[(\mathcal{V}(params, n, ID^{\dagger}) \rightleftharpoons P^{\dagger}) = 1] \geqslant \varepsilon$，则称证明者 $P^{\dagger}$ 是 ε-允许的。

2)ε-安全：敌手在任意时刻执行上述安全性游戏，并以 $ID^{\dagger}$ 和 n 为输入，输出一个 ε-允许的证明者 $P^{\dagger}$。若存在一个提取器算法 Extr，能够以不可忽略的概率从 $P^{\dagger}$ 中提取出 $M^{\dagger}$，即 Extr($params$, n, $ID^{\dagger}$, $P^{\dagger}$) = M，则称基于身份的 PDP 方案是 ε-安全的。

定义 3(针对 TPA 的完美隐私性游戏)　如果存在一个多项式时间内的非交互模拟器 S，对于公开输入 ID，$chal$，Tag 和非公开输入 M，以下两个量是计算上不可区分的，即 S 在与服务器的交互中，除了公开输入没有得到其他任何信息。

(1)$view_{TPA^*}$ ($Server_{R,chal,\mathcal{M},ID,Tag,TPA^*}$)。其中，$R$ 是协议中抛掷硬币随机产生的输出值。

(2)$S(chal, ID)$。

则称 PDP 方案针对欺骗性 TPA^* 具有完美隐私性。

4　具体方案

本节描述方案的具体细节。

(1)Setup(1^k)→($params$, msk, mpk)。输入安全参数 k，KGC 执行以下操作：

1)选择两个阶为 q 的乘法循环群 G_1 和 G_2，q 是一个大素数，取 G_1 的生成元为 g，在群 G_1 和 G_2 上选择一个双线性映射 $e: G_1 \times G_1 \rightarrow G_2$。

2)选择 Hash 函数(H_1, H_2, H_3)、伪随机函数 f 和伪随机置换 π，其中，$H_1: \{0,1\}^* \rightarrow G_1$，$H_2: \{0,1\}^* \rightarrow G_1$，$H_3: G_2 \rightarrow \{0,1\}^l$，$f: Z_q^* \times \{1,2,\cdots,n\} \rightarrow Z_q^*$，$\pi: Z_q^* \times \{1,2,\cdots,n\} \rightarrow \{1,2,\cdots,n\}$。

3)选择 $x \xleftarrow{R} Z_q^*$ 为主私钥 msk，主公钥 $mpk = g^x$，公开参数 $params = (G_1, G_2, q, g, e, H_1, H_2, H_3, l, f, \pi)$ 和主公钥 $mpk = g^x$，并秘密保存主私钥 $msk = x$。

(2)Extract($params$, msk, ID)→(sk, pk)。KGC 计算 DO 的公私钥，其中，公钥 $pk = H_1(ID)$，私钥 $sk = H_1(ID)^x = pk^x$，将 sk 通过安全信道发送给 DO。

(3)TagGen($params$, ID, sk, $\{m_{jk}\}$)→($\{T_{jk}\}$, IDS, $\mathcal{M}$)。用户 DO 执行以下操作：

1)DO 将所有文件分为 n 个块，并存放在 m 个服务器上，文件块 $m_{jk} \in Z_q$，$j \in [1,m]$表示存储文件块的服务器索引，$k \in [1,n]$表示文件块索引。用户随机选取 $\eta \longleftarrow Z_q$，并计算 $r = g^{\eta}$及数据块标签 $T_{jk} = sk^{m_{jk}} H_2(fname_{l_k} \| k)^{\eta}$，$fname_{l_k}$ 表示第 k 个文件块所属文件的文件名。用户将文件块集合$\{m_{jk}\}$及其对应的标签集合$\{T_{jk}\}$按索引发送给对应的服务器 CS_j。

2)DO 构建一张索引集合表 $\mathcal{M} = \{(fname_{l_k}, j, k)\}$用于记录数据块的存储索引信息，并将表 $\mathcal{M}$ 和 $IDS(r)$存放到Organiser。其中，表 $\mathcal{M}$ 包括 DO 的文件块索引 k、存储第 k 个块的服务器索引 j 和第 k 个块所属的文件名 $fname_{l_k}$。

(4)Challenge($params$, n, ID)→($chal$, $\{chal_j\}$)。TPA 执行以下操作：

1)Challenge1(Organiser←TPA)

①TPA 随机选取 c 个块进行挑战，$c \in [1,n]$；TPA 分别为 f 和 π 选取秘钥 $t_1, t_2 \in Z_q^*$，令挑战集合 $Q = (c, t_1, t_2)$。

②TPA 选择一个随机数 $\rho \in Z_q^*$，计算 $c_1 = g^{\rho}$，$Z = e(H_1(ID), mpk)$，$c_2 = Z^{\rho}$。

③产生一个知识证明 $pf: POK\{(\rho): c_1 = g^{\rho} \wedge c_2 = Z^{\rho}\}$，$g \in G_1$，$Z \in G_2$。

④令总挑战 $chal = (Q, c_1, c_2, pf)$，TPA 将 $chal$ 发送给 Organiser。

2)Challenge2(CS_j←Organiser)

①Organiser 验证 pf 的有效性，若有效则计算 $i_k = \pi_{t_1}(k)$，$1 \leqslant k \leqslant c$，查询表 $\mathcal{M}$ 中 i_k 值对应的记录，产生分挑战索引集合$\{\mathcal{M}_j\} = \{\{(fname_{l_{i_k}}, j, i_k) |_k \in [1,c]\}, j \in U\}$，其中 U 为被挑战块服务器所在的索引集合。

②Organiser 根据索引集合$\{\mathcal{M}_j\}$生成 $|U|$ 个分挑战$\{chal_j\}$，令 $chal_j = (\mathcal{M}_j, t_2)$，$chal = \bigcup_{j \in U} chal_j$。Organiser 向被挑战服务器 CS_j 分发分挑战 $chal_j$。

(5)Prove($params$, $chal_j$, ID, $\{T_{jk}\}$, $\{m_{jk}\}$)→$\{P_j, P\}$。

1)Prove1(CS_j→Organiser)

①$\forall (fname_{l_{i_k}}, j, i_k) \in \mathcal{M}_j$，$CS_j$ 计算 $v_{i_k} = f_{t_2}(i_k)$，并计算其上被挑战数据块聚合值 μ_j'及其标签聚合值 T_j'，$\mu_j' = \sum_{i_k \in \mathcal{M}_j} v_{i_k} m_{ji_k}$，$T_j' = \prod_{i_k \in \mathcal{M}_j} T_{ji_k}^{v_{i_k}}$。

②CS_j 将分证明 $P_j = (\mu_j', T_j')$发送给 Organiser。

2)Prove2(Organiser→TPA)

①Organiser 聚合不同服务器返回的证明 P_j，$\mu' = \sum_{j \in U} \mu_j'$，$T' = \prod_{j \in U} T_j'$，并计算：

$$m' = H_3(e(T', c_1) \cdot c_2^{-\mu'}) \tag{1}$$

②Organiser 将聚合证明 $P = (r, m', IDS(r))$返回给 TPA。

(6)BatchVerify($params$, ID, $chal$, $\{P\}$)→$\{0,1\}$。

TPA 验证签名 IDS 的有效性，若有效则计算 $i_k = \pi_{t_1}(k)$，$v_{i_k} = f_{t_2}(i_k)$，并验证式(2)是否成立：

$$m' \stackrel{?}{=} H_3(e(\prod_{k \in [1,c]} H_2(fname_{l_{i_k}} \| i_k)^{v_{i_k}}, r^{\rho})) \tag{2}$$

上述具体方案的协议流程如图 2 所示。

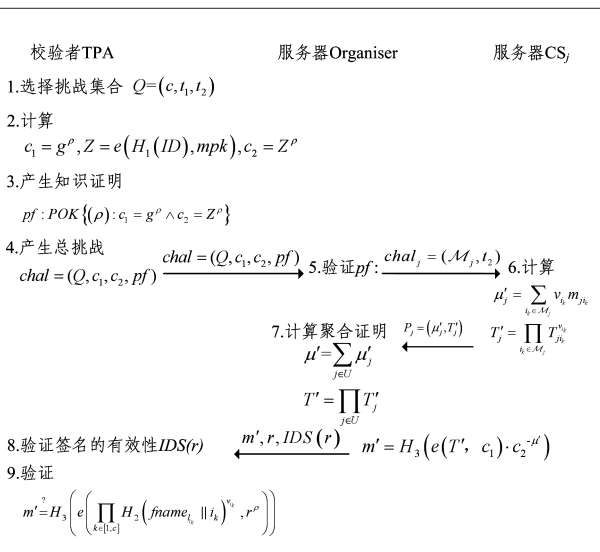

图 2 IDBPDP 协议的流程

Fig. 2 Process of IDBPDP protocol

5 正确性、安全性和隐私性证明

5.1 正确性证明

定理 1 若 TPA 和服务器都诚实地按照协议执行，则服务器返回的关于文件块和数据标签的证明就可以通过 TPA 的批处理校验。

证明：将式(1)变换如下：

$$\begin{aligned}m' &= H_3(e(T',c_1)\cdot c_2^{-\mu'})\\ &= H_3\left(\frac{e(T',c_1)}{e(H_1(ID),mpk)^{\rho\mu'}}\right)\\ &= H_3\left(\frac{e(\prod_{j\in U}\prod_{i_k\in\mathcal{M}_j}T_{ji_k}^{v_{i_k}},c_1)}{e(sk,g)^{\rho\sum_{j\in U}\sum_{i_k\in\mathcal{M}_j}v_{i_k}m_{ji_k}}}\right)\\ &= H_3\left(\frac{e(\prod_{k\in[1,c]}T_{ji_k}^{v_{i_k}},c_1)}{e(\prod_{k\in[1,c]}sk^{v_{i_k}m_{ji_k}},c_1)}\right)\\ &= H_3(e(\prod_{k\in[1,c]}H_2(fname_{i_k}\|i_k)^{\eta v_{i_k}},c_1))\\ &= H_3(e(\prod_{k\in[1,c]}H_2(fname_{i_k}\|i_k)^{v_{i_k}},r^{\rho}))\end{aligned}$$

证毕。

5.2 安全性分析[11]

下面证明一般性算法打破协议安全性的概率是可以忽略的。一般性算法通过一般性群来刻画，指在有限循环群中没有利用代数结构的算法。一般性算法的操作描述如下：设 p 是一个素数，$\xi_i:Z_p\to\{0,1\}^{\lceil\log_2 p\rceil}$ ($i\in\{1,2\}$)是两个独立的随机编码函数，一般性群 G_i 表示为 $G_i=\{\xi_i(x)\mid x\in Z_p\}$。由于一般性算法没有利用群的结构，因此对于 $\forall\xi_i(x)\in G_i$，除了值 $\xi_i(x)$，得不到其他任何信息。随机谕言机 O_i ($i\in\{1,2\}$) 分别用来模拟群 G_i ($i\in\{1,2\}$) 上的操作，即在 O_i 上输入 $\xi_i(a)$ 和 $\xi_i(b)$ 时，输出元素 $\xi_i(a+b)$ 表示群中两个元素 $\xi_i(a)$ 和 $\xi_i(b)$ 相乘的结果，输出 $\xi_i(a-b)$ 表示群中元素 $\xi_i(a)$ 和 $\xi_i(b)$ 相除的结果。随机谕言机 O_E 模拟群上的双线性操作 $\hat{e}$：$G_1\times G_1\to G_2$，输入元素 $\xi_1(a)$ 和 $\xi_1(b)$，O_E 返回 $\xi_2(a*b)$ 作为双线性运算的结果。

证明：令敌手 $\mathcal{A}$ 在随机谕言机 H_1，H_2，H_3 上分别做 q_1，q_2，q_3 次询问，通过构造模拟器 S 模拟 $\mathcal{A}$ 在一般性群模型中的视图，并从与 $\mathcal{A}$ 的交互中提取出全部被挑战数据块值，从而证明在随机谕言机模型下，本文方案针对一般性算法是可证明安全的。

(1)Settings：在一般性群模型和随机谕言机模型下，公共参数表示为 $(l,q,O_E,O_1,O_2,H_1,H_2,H_3,\xi_1,\xi_\alpha)$，其中 ξ_1 和 ξ_α 分别为元素 g 和 P_{pub} 的编码值。将 ξ_1 和 1 次常量多项式对应，将 ξ_α 和 α 次多项式对应。特别地，$\mathcal{A}$ 中任意元素将与多元多项式 $(\alpha,\{I_i\}_{i=1}^{q_I},\{f_{l_k,k}\}_{k=1}^{n\times q_f},\{\eta_t\}_{t=1}^{q_t},\rho)$ 对应，下面简要阐述多项式与元素的对应关系。

(2)Hash queries：$\mathcal{A}$ 对 ID_i 查询随机谕言机 H_1，S 返回一个随机比特串 ξ_{I_i}，并把它与多元多项式 I_i 相对应；$\mathcal{A}$ 对文件块 $fname_{l_k}\|k$ 查询随机谕言机 H_2，S 返回一个随机比特串 $\xi_{f_{l_k,k}}$，其对应的多项式为 $f_{l_k,k}$；$\mathcal{A}$ 对群 G_2 中的元素查询随机谕言机 H_3，S 返回一个随机比特串值，因为 H_3 返回的并不是群元素，所以没有多项式与之对应。

(3)Group G_1 operation(oracle O_1)：$\mathcal{A}$ 将元素 ξ_{F_1}，ξ_{F_2} 和两个整数 e_1，e_2 发送给 S，S 查找历史中是否有与多项式 F_1 和 F_2 对应的元素 ξ_{F_1}，ξ_{F_2} 存在(假设 F_1 与 F_2 不同)，回应如下：

1)如果元素 ξ_{F_1}，ξ_{F_2} 不存在，意味着 ξ_{F_1} 或 ξ_{F_2} 不是群中的有效元素，S 拒绝执行操作。

2)如果元素 ξ_{F_1} 和 ξ_{F_2} 存在，S 计算多项式 $F_3=e_1F_1+e_2F_2$，并查找历史中是否存在元素 $\xi\in G_1$ 和多项式 F_3 对应。

①若有，则返回 ξ_{F_3} 给 $\mathcal{A}$。

②若没有，S 随机选择一个比特串 ξ_{F_3} 与多项式 F_3 对应，并将 ξ_{F_3} 返回给 $\mathcal{A}$。

其中，ξ_{F_3} 表示 ξ_{F_1} 的 e_1 次和 ξ_{F_2} 的 e_2 次群操作运算结果。

(4)Group G_2 operation(oracle O_2)：与随机谕言机 O_1 的操作相同。

(5)Pairing operation(oracle O_E)：$\mathcal{A}$ 将元素 ξ_{F_1}，ξ_{F_2} 发送给 S，询问 O_E 结果，S 查找历史中是否有与多项式 F_1 和 F_2 对应的元素 ξ_{F_1}，ξ_{F_2} 存在，回应如下：

1)如果元素 ξ_{F_1}，ξ_{F_2} 不存在，则 ξ_{F_1} 或 ξ_{F_2} 不是群中的有效元素，S 拒绝执行操作。

2)如果元素 ξ_{F_1} 和 ξ_{F_2} 存在，则 S 计算多项式 $F_3=F_1\cdot F_2$，并在群 G_2 中寻找历史是否存在元素 $\xi\in G_2$，其对应的多项式为 F_3。

①若有，则返回 ξ_{F_3} 给 $\mathcal{A}$。

②若没有，S 随机选择一个比特串 ξ_{F_3} 与多项式 F_3 对应，并将 ξ_{F_3} 返回给 $\mathcal{A}$。

其中，ξ_{F_3} 代表元素 ξ_{F_1} 和 ξ_{F_2} 的双线性运算结果。

(6)Extraction Query：$\mathcal{A}$ 将用户身份 ID 发送给 S，S 查找历史中是否存在元素 $\xi_{I_i}=H_1(ID)$，若没有则对 ID 进行 H_1 询问，得到 $\xi_{I_i}=H_1(ID)$ 和与元素 ξ_{I_i} 对应的多项式 I_i；然后 S 查找历史中是否存在 $\xi\in G_1$，其对应的多项式为 αI_i，并做如下回应：

1)若存在，则 S 返回 ξ 给 $\mathcal{A}$。

2）否则，S 随机选择一个比特串 ξ_{aI_i}，与多项式 αI_i 对应，并将 ξ_{aI_i} 返回给 $\mathcal{A}$。

（7）TagGenQuery：$\mathcal{A}$ 将身份 ID 和文件块集合 $M=(m_{j1},\cdots,m_{jn})(j\in[1,m])$ 发送给 S，S 对 ID 查询 Extraction，查找历史中是否存在与多项式 I_i 对应的元素 $\xi_{I_i}=H_1(ID)$ 及与多项式 αI_i 对应的元素 $\xi_{aI_i}=H_1(ID)^{\alpha}$，若没有，则对 ID 进行 Extraction Query。S 查找历史中是否存在与多项式 $f_{l_k,k}$ 对应的元素 $\{H_2(fname_{l_k}\parallel k)\}$，若不存在，则 S 对 $fname_{l_k,k}$ 做 H_2 查询。S 随机选择一个比特串 ξ_{η_i} 与多项式 η_i 对应，并对 $\forall k\in[1,n]$ 计算多项式 $F_{jk,i}=\alpha I_i m_{jk}+fname_{l_k,k}\eta_i$，然后查找是否存在与多项式 $F_{jk,i}$ 对应的元素 $\xi_{jk,i}$，做如下回应：

1）若存在，则 S 返回 $\xi_{jk,i}$ 作为对用户 ID 的文件块 m_{jk} 的标签。

2）否则，S 随机选择比特串 $\xi_{jk,i}$ 与多项式 $F_{jk,i}$ 对应，并将 $\xi_{jk,i}$ 返回给 $\mathcal{A}$。同时，S 将元素 ξ_{η_i} 和对 r_i 的身份认证签名 $IDS(r_i)$ 返回给 $\mathcal{A}$。

令 $(\xi_{\eta_i},\xi_{j1,i},\xi_{j2,i},\cdots,\xi_{jn,i},IDS(r_i))$ 作为 $\mathcal{A}$ 对用户文件块集合 M 的标签询问回应。

S 指定身份为 ID 的用户，并从 ID 拥有的文件块 $M=(m_{j1},m_{j2},\cdots,m_{jn})$ 中选取 k 个进行挑战，$k\in[1,n]$。要求敌手 $\mathcal{A}$ 之前未对 ID 进行过 Extraction Query。假设 $\mathcal{A}$ 对 ID 进行过哈希查询和文件块标签查询，并设 $H_1(ID)$ 为哈希查询得到的值，$H_1(ID)$ 对应的多项式为 I；$(\xi_{\eta_i},\xi_{j1,i},\xi_{j2,i},\cdots,\xi_{jn,i},IDS(r))$ 为 $\mathcal{A}$ 以身份 ID 和文件块 $(m_{j1},m_{j2},\cdots,m_{jn})$ 为输入得到的标签询问回应。接下来，S 扮演挑战者角色，敌手 $\mathcal{A}$ 扮演服务器的角色，交互执行协议。

（8）challenge：S 随机选择挑战块数 $c\in[1,n]$ 和秘钥 t_1，$t_2\in Z_q^*$。然后，S 随机选择比特串 ξ_{c_1} 与多项式 ρ 对应，随机选取 ξ_Z 和多项式 αI 对应，ξ_{c_1} 和 ξ_Z 分别表示群 G_1 中元素 c_1 和群 G_2 中元素 Z，且 S 随机选取元素 ξ_{c_2} 与多项式 $\alpha I\rho$ 对应。最后，S 生成与 ξ_{c_1} 和 ξ_{c_2} 相关的知识证明 pf，并将 pf 返回给 A。

（9）Polynomial Evaluation：由上文可知，I 为元素 $H_1(ID)$ 对应的多项式，令 η 为 ξ_η 对应的多项式，$f_{l_k,k}$ 为 $H_2(fname_{l_k}\parallel k)$ 对应的多项式，则文件块 m_{jk} 的标签 T_{jk} 对应的多项式为 $F_{jk}\equiv m_{jk}\alpha I+f_{l_k,k}\eta$。此时，$S$ 随机选取的变量值包括以下几种：$\alpha,I,\{f_{l_k,k}\}_{k\in[1,c]},\eta,\rho$。换句话说，对于 $\mathcal{A}$ 得到的所有元素，其多元多项式由 $c+4$ 个变量组成。

（10）Prove：最后 $\mathcal{A}$ 返回 m'，r 和 $IDS(r)$。

在随机谕言机模型下，$\mathcal{A}$ 成功地返回值 $m'=H_3(e(\prod_{k\in[1,c]}H_2(fname_{l_{i_k}}\parallel i_k)^{v_{i_k}},r^{\rho}))$，需要对群 G_2 中的元素 $\xi=e(\prod_{k\in[1,c]}H_2(fname_{l_{i_k}}\parallel i_k)^{v_{i_k}},r^{\rho})$ 做 H_3 询问，且由于身份认证签名的不可伪造性，元素 r 与之前标签询问时返回给 $\mathcal{A}$ 的 r 值相同。因此，如果 $\mathcal{A}$ 以不可忽略的概率赢得游戏，则 $\mathcal{A}$ 需要返回与多项式 F 对应的元素值 ξ，而 F 需满足关系式：

$$F\equiv\sum_{k\in[1,c]}v_k f_{l_k,k}\rho\eta(\bmod q)\tag{3}$$

下面证明如果 $\mathcal{A}$ 能在进行一些群操作的询问后成功返回元素值 ξ，则 S 可以从与 $\mathcal{A}$ 的交互中提取出全部文件块值 $\{m_{jk}\}$。

挑战阶段之前，$\mathcal{A}$ 可以询问群 G_1 中的元素，每次询问 $\mathcal{A}$ 最多得到一个新的群元素，因此 $\mathcal{A}$ 得到的总体元素数量有限。元素对应的多元多项式表示为：

$$A_0+A_1\alpha+A_2\eta+A_3 I+\sum_{k\in[1,c]}A_{4,k}f_{l_k,k}+\sum_{k\in[1,c]}A_{5,k}(m_{jk}\alpha I+f_{l_k,k}\eta)$$

$\mathcal{A}$ 已知上式中所有的系数。同样地，群 G_2 中的元素对应的多项式表示为 $\sum c_i F_i * F_i'$，F_i 和 F_i' 为群 G_1 中元素对应的多项式，c_i 是常数项系数，$\mathcal{I}$ 是身份为 ID_i 的不同用户集合。显然，在挑战前，$\mathcal{A}$ 没有得到变量 ρ，故无法返回值 ξ，使其对应多项式满足等式(3)。

挑战阶段之后，$\mathcal{A}$ 得到另外两个元素 ε_{c_1} 和 ε_{c_2}，其对应多项式分别为 ρ 和 $I\alpha\rho$，$\rho\in G_1$，$I\alpha\rho\in G_2$。此时，$\mathcal{A}$ 得到的所有群 G_2 中的元素的多元多项式可表示为：

$$\begin{aligned}&A_0+A_1\rho+A_2\alpha+A_3\eta+A_4 I+\sum_{k\in[1,c]}A_{5,k}f_{l_k,k}+\sum_{k\in[1,c]}A_{6,k}(m_{jk}I\alpha+f_{l_k,k}\eta)+A_7 I\alpha\rho+B_1\rho^2+B_2\rho\alpha+B_3\rho\eta+B_4\rho I+\sum_{k\in[1,c]}B_{5,k}f_{l_k,k}\rho+\sum_{k\in[1,c]}B_{6,k}(m_{jk}I\alpha\rho+f_{l_k,k}\eta\rho)+B_7 I\alpha\rho^2+C_1\alpha^2+C_2\alpha\eta+C_3\alpha I+\sum_{k\in[1,c]}C_{4,k}f_{l_k,k}\alpha+\sum_{k\in[1,c]}C_{5,k}(m_{jk}I\alpha^2+f_{l_k,k}\eta\alpha)+C_6 I\alpha^2\rho+D_1\eta^2+D_2 I\eta+\sum_{k\in[1,c]}D_{3,k}f_{l_k,k}\eta+\sum_{k\in[1,c]}D_{4,k}(m_{jk}I\alpha\eta+f_{l_k,k}\eta^2)+D_5 I\alpha\rho\eta+E_1 I^2+\sum_{k\in[1,c]}E_{2,k}f_{l_k,k}I+\sum_{k\in[1,c]}E_{3,k}(m_{jk}I^2\alpha+f_{l_k,k}I\eta)+E_4 I^2\alpha\rho+\sum_{k\in[1,c]}\sum_{h\in[1,c]}F_{1,k,h}f_{l_k,k}f_{l_h,h}+\sum_{k\in[1,c]}\sum_{h\in[1,c]}F_{2,k,h}(m_{jk}I\alpha f_{l_h,h}+f_{l_k,k}f_{l_h,h}\eta)+\sum_{k\in[1,c]}F_{3,k}I\alpha\rho f_{l_k,k}+\sum_{k\in[1,c]}\sum_{h\in[1,c]}G_{1,jk,jh}(m_{jk}I\alpha+f_{l_k,k}\eta)(m_{jh}I\alpha+f_{l_h,h}\eta)+\sum_{k\in[1,c]}G_{2,k}(m_{jk}I^2\alpha^2\rho+f_{l_k,k}\eta I\alpha\rho)+H_1 I^2\alpha^2\rho^2\end{aligned}\tag{4}$$

$\mathcal{A}$ 和 S 已知式(4)中所有系数，将式(4)与式(3)联立，则 $\forall k\in[1,c]$，$B_{6,k}=v_k$，$A_7=\sum_{k\in[1,c]}m_{jk}v_k$，其他系数均为 0。通过 $|c|$ 次交互，S 从系数 $A_7=\sum_{k\in[1,c]}m_{jk}v_k$ 中可以计算出全部的 m_{jk} 值，即通过 $|c|$ 次交互 S 可以提取出全部的文件块。证毕。

5.3　隐私性证明[11]

协议针对 TPA 完美地保护用户数据隐私，我们通过构造模拟器 S（在不知道文件块集合 $\{m_{jk}\}$ 和标签集合 $\{T_{jk}\}$ 的情况下），以黑盒的方式访问验证者 V^*（即恶意的 TPA），来模拟远程数据拥有性证明协议。

首先，将 $r,IDS(r)$ 发送给 S。因为 $r=e(g,g)^{\eta}$，η 是由用户选取的随机值，所以将 $r,IDS(r)$ 发送给 S 不会影响协议的完美隐私性。S 对 V^* 的挑战回应如下：

（1）S 收到 V^* 的挑战 $chal=(c_1,c_2,Q,pf)$ 后，解析知识证明 $pf:POK\{(\rho):c_1=g^{\rho}\wedge c_2=Z^{\rho}\}$，并根据 pf 的合理性，S 可从 $c_1=g^{\rho}$ 和 $c_2=e(H_1(ID),mpk)^{\rho}$ 中得到值 ρ。

（2）S 解析集合 $Q=(c,t_1,t_2)$ 计算 $i_k=\pi_{t_1}(k)$，$v_{i_k}=f_{t_2}(i_k)$，并根据文件名 $fname_{l_k}$ 计算值 $m'=H_3(e(\prod_{k\in[1,c]}H_2(fname_{l_{i_k}}\parallel i_k)^{v_{i_k}},r^{\rho}))$，最后 S 输出 $(r,m',IDS(r))$ 作为挑战回应。

综上，S 的输出和 $VIEW_{P,V^*}$ 的视图同分布，以上模拟是完备的。

6 性能分析及实验对比

6.1 性能分析

以下分析中，n 表示用户的文件块总数，c 表示 TPA 选中的被挑战块数量，t 表示被挑战的服务器数量，c_j 表示被挑战服务器 CS_j 上的挑战块数量，显然有 $\sum_{j=1}^{t} c_j = c$。令 E_{G_1}，E_{G_2} 分别表示群 G_1，G_2 上单个指数运算的开销，M_{G_1}，M_{G_2} 分别表示群 G_1，G_2 上单个乘法运算的开销，M_{Z_p} 表示群 Z_p 上单个乘法运算的开销，A_{Z_p} 表示群 Z_p 上单个加法运算的开销，P 表示一个双线性对运算的开销，H 表示单个哈希函数运算的开销。

(1)计算复杂度分析

1)秘钥生成中心(KGC)：生成系统参数、主公钥 mpk 及用户私钥 sk。KGC 的主要计算开销为 $2E_{G_1}+1H$。

2)用户(DO)：将全部文件分为 n 个块并为所有数据块计算数据标签 $T_{jk}=sk^{m_{jk}}H_2(fname_{l_k}\|k)^{\eta}$。DO 的计算开销为 $(2n+1)E_{G_1}+nH+nM_{G_1}$，该工作是一次性完成的。

3)被挑战服务器(CS_j)：计算关于被挑战数据块的证明。CS_j 的计算开销为 $c_jE_{G_1}+(c_j-1)M_{G_1}+c_jM_{Z_p}+(c_j-1)A_{Z_p}$。

4)服务器组织者(Organiser)：根据不同服务器返回的证明计算聚合证明，并将其发送给 TPA。Organiser 的计算开销为 $P+H+E_{G_2}+M_{G_2}+(t-1)M_{G_1}+(t-1)A_{Z_p}$。

5)验证者(TPA)：生成挑战并对 Organiser 返回的证明进行批量校验。根据离散对数相等协议，TPA 生成挑战需要执行 1 次双线性操作和 6 次幂运算操作。TPA 批量校验证明的计算开销为 $P+(c+1)E_{G_1}+(c+1)H+(c-1)M_{G_1}$。

(2)通信复杂度分析

1)初始阶段：云用户除了将文件块 $\{m_{jk}\}$ 和数据标签 $\{T_{jk}\}$ 发送给对应的云服务器 CS_j 之外，还需要把索引集合表 $\mathcal{M}$ 发送给 Organiser，表 $\mathcal{M}$ 中包含 $3n$ 个 Z_q 中的元素。

2)挑战阶段：在 Challenge1 中 TPA 将总挑战 $chal=(Q,c_1,c_2,pf)$ 发送给 Organiser，其中 $Q=(c,t_1,t_2)$ 包含 c 个整数、2 个 Z_q 中的元素，pf 中包含 6 个 Z_q 中的元素，因此 $chal$ 的二进制长表示为 $\log_2 c_1+\log_2 c_2+\log_2 t_1+\log_2 t_2+\log_2 pf+c\log_2 n$。在 Challenge2 中，Organiser 发送分挑战 $chal_j=(\mathcal{M}_j,t_2)$ 给被挑战服务器 CS_j，$\mathcal{M}_j$ 包含 c 个 Z_q 中的元素和 $2c$ 个整数，因此 $chal_j$ 的长度为 $(c+1)\log_2 q+2c\log_2 n$。

3)响应阶段：Prove1 中服务器 CS_j 返回证明 $P_j=(\mu_j,T_j)$ 给 Organiser，其中包含 2 个 Z_q 中的元素，长度为 $\log_2\mu_j+\log_2 T_j$。所有被挑战服务器返回的证明包含 $2t$ 个 Z_q 中元的素。Prove2 中 Organiser 返回证明 $P=(r,m',IDS(r))$ 给 TPA，因为身份签名通常包含椭圆曲线上的两个点(320 位)，所以证明 P 的长度为 $\log_2 r+l+320$。

(3)存储复杂度分析

1)服务器(CS_j 和 Organiser)：服务器 CS_j 上存储着用户在其上存放的所有文件块 M_j 及其对应的数据标签，因此存储内容如图 3 所示。

M_j	$Tags$

图 3 服务器 CS_j 上的存储内容

Fig. 3 Stored content in CS_j server

选取阶为 160 bit 的椭圆曲线，则文件块及数据标签的存储开销为 $\log_2 M_j+\lceil \log_2(M_j)/\log_2 q \rceil 160$。

Organiser 在被挑战务器和 TPA 之间起连接作用，其上需存储一张索引集合表，如图 4 所示。

$fname_{l_k}$	j	k

图 4 Organiser 上的存储内容

Fig. 4 Stored content in Organiser

为了保证 r 值不受到外部或内部的敌手干扰，还需将值 r 及其身份签名 $r\|IDS(r)$ 存放在 Organiser 上，Organiser 的存储开销为 $c(\log_2 fname_{l_k}+\log_2 m+\log_2 n)+\log_2 r+len(IDS(r))$。

2)验证者：TPA 进行批处理校验，需存储挑战集合 Q、选取的随机值 ρ 和文件名 $fname_{l_k}$，其存储开销为 $\log_2 t_1+\log_2 t_2+\log_2 q+c\log_2 n+c\log_2 fname_{l_k}$。

6.2 性能及安全性对比

我们在多服务器环境下对 Yu 等[11]的方案进行拓展，并将本文方案与文献[11]拓展后的方案及 Wang 等[14]的方案进行对比，结果如表 2 所列。

表 2 不同方案的效率、功能及安全性比较

Table 2 Comparison of efficiency, function and security for different schemes

	方案	本文 IDBPDP 方案	Yu 等[11]方案的拓展	Wang 等[14]的方案
计算复杂度	用户端	$(2n+1)E_{G_1}+nH+nM_{G_1}$	$(2n+1)E_{G_1}+nH+nM_{G_1}$	$n(s+1)E_{G_1}+nsM_{G_1}+nh+nsH$
	单个服务器 CS_j 端	$c_jE_{G_1}+(c_j-1)M_{G_1}+c_jM_{Z_p}+(c_j-1)A_{Z_p}$	$c_jE_{G_1}+(c_j-1)M_{G_1}+c_jM_{Z_p}+(c_j-1)A_{Z_p}+(M_{G_2}+E_{G_2}+P+H)$	$c_jE_{G_1}+(c_j-1)M_{G_1}+s(c_jh+c_jM_{Z_p}+(c_j-1)A_{Z_p})$
	Organiser 服务器端	$M_{G_2}+E_{G_2}+P+H+(t-1)M_{G_1}+(t-1)A_{Z_p}$	—	$(t-1)M_{G_1}+s(t-1)A_{Z_p}$
	TPA 端	$P+(c+1)E_{G_1}+(c-1)M_{G_1}+(c+1)H$	$t(cP+(c+1)E_{G_1}+(c-1)M_{G_2}+(c+1)H)$	$(c+s+1)E_{G_1}+2P+(c+s)M_{G_1}+cH$
通信复杂度	Challenge1	$\log_2 c_1+\log_2 c_2+c\log_2 n$ $\log_2 t_1+\log_2 t_2+\log_2 pf$	$t\left(\log_2 c_1+\log_2 c_2+c\log_2 n \atop \log_2 t_1+\log_2 t_2+\log_2 pf\right)$	$\log_2 t_1+\log_2 t_2+\log_2 n$
	Challenge2	$(c+1)\log_2 q+2c\log_2 n$		$\log_2 q+c\log_2 n$
	Prove1	$\log_2\mu_j+\log_2 T_j$	$\log_2\mu+\log_2 T+l+\log_2 r+320$	$s\log_2\mu_j+\log_2 T_j$
	Prove2	$l+\log_2 r+320$		$s\log_2\mu+\log_2 T$
是否支持批处理校验		是	否	是
是否针对服务器安全		是	是	否
是否针对 TPA 隐私		是	是	否

(1)IDBPDP 与 Yu 等[11]方案的对比：由表 2 可看出，在服务器端的计算开销上，IDBPDP 通过 Organiser 进行聚合证明，因此相对于 Yu 等拓展后的方案，本文方案中每个云服务器减少了 1 次幂运算、1 次双线性对运算、1 次群乘法运算和 1 次哈希运算，因为此部分工作由 Organiser 完成。在 TPA 端的计算开销上，IDBPDP 支持对多个服务器返回的证明进行批处理校验，与在多服务器环境下简单拓展之后的 Yu 等的方案对比，本文方案减少了$(ct-1)$次双线性运算、$(t-1)(c+1)$次群 G_1 上的幂运算、$(t-1)(c-1)$次群上的乘法运算和$(t-1)(c+1)$次哈希运算操作，有效提高了 TPA 端的审计效率。

(2)IDBPDP 与 Wang 等[14]方案的对比：Wang 等的方案将每个文件块都分成了 s 个区，由此使得用户端、单个服务器端及 TPA 端的计算开销均增大；而在 Organiser 上的计算开销低于本文方案，这是因为为支持针对 TPA 的用户数据隐私保护，本文方案在 Organiser 上增加了 1 次双线性运算和 1 次乘幂运算，但是增加的计算开销是可以忽略的。

通信开销上，本文方案的通信复杂度为 $O(c+t)$，Wang 等方案和 Yu 等方案的通信复杂度分别为 $O(sc)$ 和 $O(tc)$，因此本文方案拥有更低的通信复杂度。

6.3　实验对比

下面分别从用户、单个服务器及 TPA 端的计算开销上，对本文方案、Yu 等在多服务器环境下简单拓展的方案和 Wang 等的方案进行了仿真实验对比。其中，Wang 等的方案设置分区数 $s=20$。实验环境如下：PC 硬件配置为 Intel Core2Duo 处理器、4 GB 内存，操作系统为 Ubuntu 16.04 LTS 32 位。利用 PBC 库、GMP 库和 Miracl 库，使用 gcc 编译执行，使用 PBC 库中的 a.param 参数设置双线性对。

(1)用户计算数据标签 TagGen

用户对全部文件进行分块，并为每个数据块计算数据标签。本文方案中用户有多个文件，设为 10 个，用户总文件大小为 2～20 MB，设置每个数据块的大小为 2 kB，则相应的数据块数为 1000～10000，设置步长为 2 MB，观察随着文件的增大本文方案在 TagGen 上的计算开销，实验结果如图 5 所示。从图 5 可以看出，TagGen 的时间随着文件大小的增长呈线性增长，与性能分析的结果一致。

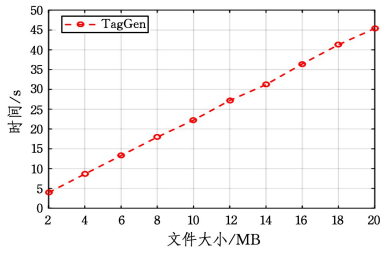

图 5　文件大小增加时 TagGen 的计算开销

Fig. 5　Computational overhead of TagGen when size of files increases

此外，固定文件大小为 1 MB，每个数据块的大小为 2 kB，共有 500 个数据块。观察本文方案与 Wang 等方案中 TagGen 的计算开销随数据块数的增加的变化情况，实验结果如图 6 所示。可以看出，两个方案中用户的 TagGen 计算开销均随数据块数的增加呈线性增长，且本文方案的计算耗费显著优于 Wang 等的方案，与性能分析结果一致。

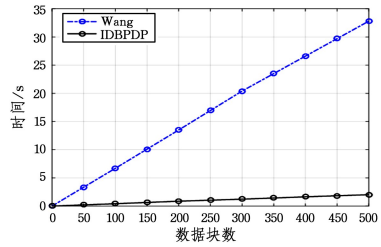

图 6　数据块数增加时 TagGen 的计算开销对比

Fig. 6　Comparison of computational cost for TagGen under increased number of blocks

(2)单个服务器端计算证明 Prove1 的计算开销

在 Prove1 阶段，收到分挑战的服务器需要计算分证明并返回给 Organiser，其计算开销与 TPA 选取的挑战块数量相关。因此，在相同条件下对比了 3 个方案 CS_j 的计算复杂度。实验设置用户拥有 10 个文件，总文件的大小为 20 MB，每个数据块的大小为 2 kB，共有 10000 个数据块，将这 10000 个数据块均匀存储到 10 个服务器上，则每个服务器上存储着用户的 1000 个数据块。若云服务器的数据块损毁率为 1%，则 TPA 挑战其上的 300(460)个数据块，就能够以 95%(99%)的概率检测出该服务器的损毁数据行为[15]。因此，令单个服务器上的被挑战块数为 300～460(相应的总挑战块数为3000～4600)，步长为 50，对比观察 3 个方案中服务器端的计算开销，实验结果如图 7 所示。由图 7 可以看出，本文方案在单个被挑战服务器端的计算开销最小。

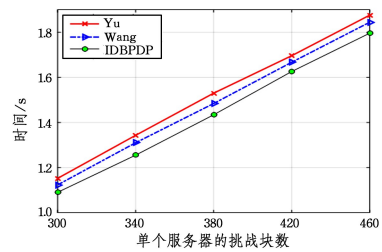

图 7　挑战块数增加时单个服务器 CS_j 端的计算开销对比

Fig. 7　Comparison of computational cost for CS_j's proof under increased number of challenged blocks

在 Prove2 阶段，Organiser 计算聚合证明并返回给 TPA 校验。当实验总挑战块数为 3000 时，本文方案在 Organiser 上的计算开销为 1.01 ms，Wang 等的方案在 Organiser 上的计算开销为 0.2 ms，故本文方案在聚合服务器上增加的计算开销是可忽略的，实验结果与性能分析的结果一致。

(3)TPA 批量校验证明 BatchVerify 的计算开销

在响应阶段，TPA 对 Organiser 返回的聚合证明进行批量验证。TPA 批校验 BatchVerify 的计算开销与其选取的挑战块数量相关，因此在相同条件下，选取 TPA 挑战的总块数为 3000～4600，步长为 200，观察 3 种方案在 TPA 端的计算

(下转第 154 页)

ABE 方案[J]. 软件学报,2012,23(10):2805-2816.

[16] SU J S,CAO D,WANG X F,et al. Attribute-based encryption schemes[J]. Journal of Software,2011,22(6):1299-1315. (in Chinese)
苏金树,曹丹,王小峰,等. 属性基加密机制[J]. 软件学报,2011,22(6):1299-1315.

[17] FENG D G,CHEN C. Research on attribute-based cryptography [J]. Journal of Cryptologic Research,2014,1(1):1-12. (in Chinese)
冯登国,陈成. 属性密码学研究[J]. 密码学报,2014,1(1):1-12.

[18] YAN X X,MENG H. Ciphertext policy attribute-based encryption schemesupporting direct revocation[J]. Journal on Communications,2016,37(5):44-50. (in Chinese)
闫玺玺,孟慧. 支持直接撤销的密文策略属性基加密方案[J]. 通信学报,2016,37(5):44-50.

[19] ZHANG K,MA J F,LI H,et al. Multi-authority attribute-based encryption with efficient revocation[J]. Journal on Communications,2017,38(3):83-91. (in Chinese)
张凯,马建峰,李辉,等. 支持高效撤销的多机构属性加密方案[J]. 通信学报,2017,38(3):83-91.

[20] SHAN Z Y,SUN Y F. A study of security attributes immediatere vocation in secure OS[J]. Journal of Computer Research and Development,2002,39(12):1680-1688. (in Chinese)
单智勇,孙玉芳. 安全操作系统安全属性即时撤消研究[J]. 计算机研究与发展,2002,39(12):1680-1688.

[21] FANG L,YIN L H,GUO Y C,et al. A survey of key technologies in attribute-based access control scheme[J]. Chinese Journal of Computers,2017,40(7):1680-1698. (in Chinese)
房梁,殷丽华,郭云川,等. 基于属性的访问控制关键技术研究综述[J]. 计算机学报,2017,40(7):1680-1698.

(上接第 137 页)

开销,实验结果如图 8 所示。从图 8 中可看出,批量校验有效降低了 TPA 端的审计开销,且本文方案的 TPA 批量审计效率优于 Wang 等的方案,与性能分析的析结果一致。

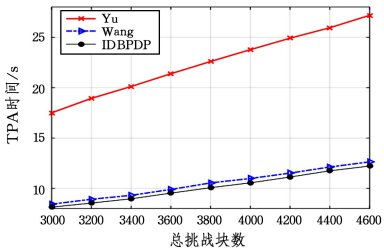

图 8 总挑战块数增加时 TPAverify 的计算开销对比

Fig. 8 Comparison of computational cost for TPAverify under increased total number of challenged blocks

以上实验表明,本文方案是高效且可行的。

结束语 本文提出了一个多服务器环境下支持完美隐私保护的批处理 PDP 方案,并对方案的正确性和安全性进行了证明。最后,将本文方案与在多服务器环境下简单拓展的 Yu 等的方案和 Wang 等的方案进行了理论分析对比与实验对比。实验结果表明,本文方案是高效且可行的。

参 考 文 献

[1] ATENIESE G,BURNS R,CURTMOLA R,et al. Provable data possession at untrusted stores[C]// ACM Conference on Computer and Communications Security. ACM,2007:598-609.

[2] ATENIESE G,PIETRO R D,MANCINI L V,et al. Scalable and efficient provable data possession[C]// Proceedings of the 4th International Conference on Security and Privacy in Communication Netowrks. ACM,2008:1-10.

[3] WANG Q,WANG C,LI J,et al. Enabling public verifiability and data dynamics for storage security in cloud computing[C]// European Conference on Research in Computer Security. Springer-Verlag,2009:355-370.

[4] ZHANG J,TANG W,MAO J. Efficient public verification proof of retrievability scheme in cloud[M]. Kluwer Academic Publishers,2014.

[5] YU Y,NI J,MAN H A,et al. Comments on a Public Auditing Mechanism for Shared Cloud Data Service[J]. IEEE Transactions on Services Computing,2015,8(6):998-999.

[6] YU Y,LI Y,NI J,et al. Comments on "Public Integrity Auditing for Dynamic Data Sharing With Multiuser Modification" [J]. IEEE Transactions on Information Forensics & Security,2016,11(3):658-659.

[7] YU Y,XUE L,MAN H A,et al. Cloud data integrity checking with an identity-based auditing mechanism from RSA[J]. Future Generation Computer Systems,2016,62(C):85-91.

[8] WANG C,WANG Q,REN K,et al. Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing[C]// Infocom,2010 Proceedings IEEE. IEEE,2010:1-9.

[9] HAO Z,ZHONG S,YU N. A Privacy-Preserving Remote Data Integrity Checking Protocol with Data Dynamics and Public Verifiability[J]. IEEE Transactions on Knowledge & Data Engineering,2011,23(9):1432-1437.

[10] YU Y,AU M H,MU Y,et al. Enhanced privacy of a remote data integrity-checking protocol for secure cloud storage[J]. International Journal of Information Security,2015,14(4):307-318.

[11] YU Y,MAN H A A,ATENIESE G,et al. Identity-based Remote Data Integrity Checking with Perfect Data Privacy Preserving for Cloud Storage[J]. IEEE Transactions on Information Forensics & Security,2017,PP(99):1-1.

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

[13] HE K,HUANG C,WANG J,et al. An efficient public batch auditing protocol for data security in multi-cloud storage[C]// 2013 8th ChinaGrid Annual Conference. IEEE,2013:51-56.

[14] WANG H. Identity-Based Distributed Provable Data Possession in Multicloud Storage[J]. IEEE Transactions on Services Computing,2015,8(2):328-340.

[15] ATENIESE G,BURNS R,CURTMOLA R,et al. Remote data checking using provable data possession[J]. Acm Transactions on Information & System Security,2011,14(1):1-34.