# 云环境下基于代数签名持有性审计的大数据安全存储方案

徐 洋[1,2] 朱 丹[1] 张焕国[2] 谢晓尧[1]
(贵州师范大学贵州省信息与计算科学重点实验室 贵阳 550001)[1]
(武汉大学计算机学院 武汉 430072)[2]

**摘 要** 针对存储在云端的大数据的安全性和动态更新问题,提出一种基于代数签名持有性审计的大数据安全存储方案。构建可信第三方审计者,利用代数签名(AS)技术对大数据进行数据持有性审计(DPA)以确保数据的完整性。另外,基于分而治之(DC)思想构建一种新型数据结构,使数据拥有者可以动态地进行修改、插入和删除操作,同时通过减少平移数据块的数量来降低操作的计算复杂度。实验结果表明,该方案能够有效地检测恶意操作,提供了较高的数据安全性,同时大大降低了服务器和审计端的计算量。

**关键词** 大数据,安全存储,代数签名,持有性审计,数据动态更新
**中图法分类号** TP393 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2016.10.032

## Big Data Storage Security Scheme Based on Algebraic Signature Possession Audit in Cloud Environment

XU Yang[1,2] ZHU Dan[1] ZHANG Huan-guo[2] XIE Xiao-yao[1]
(Key Laboratory of Information and Computing Science of Guizhou Province, Guizhou Normal University, Guiyang 550001, China)[1]
(School of Computer, Wuhan University, Wuhan 430072, China)[2]

**Abstract** For the issues of the security and dynamic updating of the big data stored in the cloud, a big data storage security scheme based on algebraic signature possession audit was proposed. It builds trusted third party auditor to make data possession audit (DPA) for the big data by using algebraic signature (AS) technology to ensure the integrity of data. In addition, a new type of data structure is built based on the idea of divide and conquer (DC), allowing the data owner to dynamically modify, insert and delete data. At the same time, the computation complexity is reduced by reducing the number of translated data blocks. Experimental results show that the proposed scheme can detect the malicious operation effectively, provide higher data security, and greatly reduce the computation of the server and the audit side.

**Keywords** Big data, Storage security, Algebraic signature, Possession audit, Data dynamic updating

目前,许多机构会产生大量的敏感数据,例如私人信息、经济数据和电子健康记录等,这些大数据给存储系统造成了很大压力。随着云计算概念的提出,越来越多的组织和个人开始将存储需求外包给云服务商以节省部署和维护存储空间的开销[1]。然而,存储在云服务器上的外包大数据可能会在未授权的情况下丢失数据。为了追求额外的收益,云存储服务商也会刻意删除很少访问的数据,并隐藏损坏数据来保护服务商的声誉[2];另外,当黑客获取控制云服务的权限时,他们就有能力发动伪装攻击或重放攻击,通过使用旧编码数据代替存储在损坏云服务器中的数据,以此来破坏编码数据的线性无关性。因此,云存储的安全性对数据的保密性、完整性和可用性构成了挑战。

然而,传统的认证技术不适合应用于大数据云存储中,例如传统哈希函数和签名。在这种情况下,需要一种有效方法来认证云端的大数据[3]。为了解决云计算中大数据完整性问题,研究者开发了数据持有性审计(Data Possession Audit, DPA)[3,4]技术,通过生成一种随机挑战信息来认证数据的完整性,从而防止云存储供应商删除或篡改数据。

另外,由于现存的DPA技术需要频繁审计以及频繁地进行数据传输,因此导致额外的审计计算负载和通信负载,这对许多数据拥有者是一种很大的负担,特别是当用户使用具有有限计算资源的移动设备时尤为明显[5];另一方面,由于数据具有动态属性,因此DPA方法需支持不同类型数据的动态更新,需要构建一种数据结构类型,如二叉树。然而,现有DPA方法中使用的数据结构不能有效支持大数据更新,特别是当数据更新比较频繁时。这是因为审计过程必须在有限调整数据结构次数内对大量数据块进行再平衡,但该过程需要较高的计算量[6]。因此,很有必要设计一种支持大规模数据更新的新数据结构。

到稿日期:2015-12-07 返修日期:2016-03-09 本文受国家重点基础研究发展计划(973)项目(2014CB340600),国家自然科学基金重点项目(61332019),贵州省基础研究重大项目(黔科合JZ字[2014]2001号),贵州省科技合作计划重点项目(黔科合LH字[2015]7763号),住房和城乡建设部科学技术计划项目(2016-K3-009),贵州省科技创新人才团队项目(黔科合人才团队(2012)4009)资助。

徐 洋(1983-),男,博士后,副教授,主要研究方向为信息安全、大数据;朱 丹(1993-),男,硕士生,主要研究方向为信息安全;张焕国(1945-),男,教授,博士生导师,主要研究方向为信息安全、可信计算;谢晓尧(1952-),男,博士,教授,博士生导师,主要研究方向为大数据、信息安全。

针对上述大数据存储和动态更新问题，学者提出了多种数据安全存储技术，例如，文献[7]提出使用可证明数据持有(Provable Data Possession，PDP)方案来认证存储在云端数据的完整性，使用同态标签和块标签生成一种单独标签。然而，该方法使得服务器端的计算复杂度和整个系统的通信负载变高。目前，许多应用于大数据存储的数据可恢复证明(Proof of Retrievability，POR)方法不能有效支持数据动态更新，因为服务器不能保证数据块和加密码相关联。文献[8]提出基于 POR 方法扩展支持动态存储。在此模型中，数据可以在任意位置读写，同时该方案还可以对检验出的错误进行纠错。但是该方案只是限制次数的审计，在大量数据审计时不可能表现高效性。文献[9]提出了一种联合 Merkle 哈希树(MHT)和双线性聚合签名的动态远程数据审计方法。利用 MHT 结构从左到右排列叶节点，帮助识别更新块的位置。然而，该方法容易泄漏数据给审计端且使审计阶段的计算量较高，特别是对大数据文件。文献[10]设计了一种有效数据审计方案，使用双线性映射生成一种加密证据，并支持动态数据更新操作。然而，为了删除或插入数据块($i$)，审计端必须寻找块位置且平移其余($n-i$)个块，大大增加了审计端的计算负载。

为此，本文提出了一种基于代数签名(Algebraic Signature，AS)的 DPA 技术来实现云端大数据的安全存储。同时，基于分而治之(Divide and Conquer，DC)思想设计了一种新型数据结构来增强本文数据审计方法。该数据结构能有效支持动态数据操作，例如插入、删除和修改，使所提方法能以较低的计算负载实现大规模数据的更新和审计。实验结果表明，所提方案在具有较低计算负载的情况下，能够有效保证大数据的安全审计。

## 1 系统模型

本文提出的 DPA 方法结构中存在 4 个主要实体：1)数据拥有者(Data Owner，DO)，私人或企业上传数据到云空间，且可通过修改、删除和插入来更新外包数据；2)云存储提供商(Cloud Storage Provider，CSP)，CSP 拥有大量计算资源和存储系统，且对云服务器和 DO 数据进行管理；3)第三方审计者(Third Party Auditor，TPA)，TPA 用来帮助 DO 对数据进行审计，避免提供商对数据的恶意修改和删除；4)用户(User)，用户必须通过 DO 认证为可信用户后，才被允许访问特定类型的外包数据。图 1 描述了本文 DPA 方法中的主要组件和它们之间的相互作用。

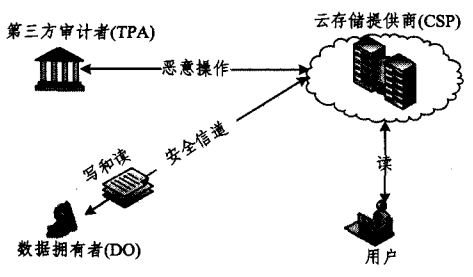


图 1 云存储中本文 DPA 的网络结构

## 2 基于代数签名的数据持有性审计

### 2.1 代数签名

代数签名[11]是一种具有同态性和代数性的哈希函数，以有限方法计算不可见信息的签名。代数签名的代数特性是指数据块代数签名的和与所对应的数据块和的代数签名结果一致。由 $n$ 个数据块($f[1],f[2],\cdots,f[n]$)组成的文件 $F$ 的代数签名计算如下：

$$S_\gamma(F)=\sum_{i=1}^{n} f[i]\cdot\gamma^{i-1} \tag{1}$$

其中，$\gamma$ 为有限域的一种元素，且向量 $\gamma=(\gamma_1,\gamma_2,\cdots,\gamma_n)$ 由非零元素组成。代数签名的一些性质解释如下。

**性质 1** 长度为 $r$ 的块 $f[i]$ 串联 $f[j]$ 后的代数签名计算式为：

$$S_\gamma(f[i]\,\|\,f[j])=S_\gamma(f[i])\oplus r^{\gamma}S_\gamma(f[j]) \tag{2}$$

**性质 2** 文件 $F$ 中一些块的签名总和等于每个块的签名总和。

$$\begin{aligned}&S_\gamma(f[1])+S_\gamma(f[2])+\cdots+S_\gamma(f[m])\\&\quad=S_\gamma(f[1]+f[2]+\cdots+f[m])\end{aligned} \tag{3}$$

证明：假设文件 $F$ 分为 $m$ 个块，且每个块由 $n$ 个分区组成，则：

$$\begin{aligned}&S_\gamma(f[1])+S_\gamma(f[2])+\cdots+S_\gamma(f[m])\\&=\sum_{j=1}^{n} f[1][j]\cdot\gamma^{j-1}+\sum_{j=1}^{n} f[2][j]\cdot\gamma^{j-1}+\cdots+\sum_{j=1}^{n} f[m][j]\cdot\gamma^{j-1}\\&=\sum_{j=1}^{n}\gamma^{j-1}\cdot(f[1][j]+f[2][j]+\cdots+f[m][j])\\&=S_\gamma(f[1]+f[2]+\cdots+f[m])\end{aligned}$$

其中，$f[i][j]$ 表示文件 $F$ 中块 $i$ 的第 $j$ 个比特位。

**性质 3** 两个文件 $F$ 和 $G$ 的代数签名和等于每个文件签名和。

$$S_\gamma(F+G)=S_\gamma(F)+S_\gamma(G) \tag{4}$$

证明：由 $n$ 个文件组成的两个文件 $F$ 和 $G$ 的代数签名和为：

$$\begin{aligned}S_\gamma(F)+S_\gamma(G)&=\sum_{i=1}^{n} f[i]\cdot\gamma^{i-1}+\sum_{i=1}^{n} g[i]\cdot\gamma^{i-1}\\&=\sum_{i=1}^{n}\gamma^{i-1}(f[i]+g[i])=S_\gamma(F+G)\end{aligned}$$

### 2.2 数据审计方案

本文中，第三方审计者根据不同数据库的持有性证明来验证云存储服务器所持有数据的完整性。本文使用数据分段技术将输入文件 $F$ 分为 $m$ 个数据块，其中每个数据块包含 $n$ 个分区。如果最后块的分区数量少于 $n$，则必须设置 $f[m][j]=0$，如果 $j\leqslant n$，则增大块的大小。提出的数据审计方案包含以下步骤：

初始化阶段：DO 必须首先使用密钥生成算法 $KeyGen(K)->(PK,SK)$ 生成公钥和私钥，其中 $K$ 为一种安全参数。因此，使用式(5)计算基于块代数签名的输入文件中每个块的特定标签(元数据)：

$$T_i=S_\gamma(f[i]\,\|\,(ID_F\,\|\,i\,\|\,L_i\,\|\,V_i)) \tag{5}$$

其中，$f[i]$ 为文件 $F$ 的第 $i$ 个块，$ID_F$ 为文件的特定 $ID$，$L_i$ 为 DCT 表中文件的逻辑数量，$V_i$ 表示数据块的版本号。为了预防重放攻击，DO 必须计算每个数据块的校验值 $C_i=S_\gamma(ID_f\,\|$

$i \| L_i \| V_i$)，其中 $i$ 为块序列号。产生所有块标签后，DO 根据云中对应的标签$\{f[i], T_i, C_i\}_{i=1}^{n}$上传数据块，并将块的代数标签发送给第三方审计者，同时删除本地拷贝文件。

挑战阶段：为了认证外包数据块的完整性，DO 需要生成一种挑战消息。由 $c$ 个随机数据块组成的消息作为挑战消息($chal=\{CS_i\}_{i=1}^{c}$)，通过具有随机选择键的伪随机置换键来预防出现自期望块指数的服务器。

当云服务器接收到挑战消息时，通过式(6)，根据挑战信息和对应标签，生成块($\sigma$)和审计标签($u$)的证据消息，并将这些信息发送给第三方审计者。

$$\mu=\sum_{i=CS_1}^{CS_C} T_i \oplus C_i$$
$$\sigma=\sum_{i=CS_1}^{CS_C} f[i] \tag{6}$$

审计阶段：第三方审计者接收到($\mu,\sigma$)并将其作为证据，然后根据 $\sigma$ 计算该证据的代数签名，并与 $u$ 进行比较。由于代数签名的特殊性质使得签名操作与求和操作可调换顺序，因此在云存储服务器可靠的情况下，此次比较应该相等。即使用式(7)认证数据块的完整性：

$$S_Y(\sigma) \stackrel{?}{=} \mu \tag{7}$$

为了提升所提算法的安全性，DO 使用设置步骤中的 DO 私钥来签名文件，随后在认证步骤中使用 DO 公钥来认证签名。

## 3 数据动态更新

动态数据更新为数据认证方法的一个必要特征，不需要下载数据就能完成外包数据更新过程。然而现存许多方法不能支持该特征。本文基于分而治之(DC)[12]的思想构建一种新的数据结构：分而治之列表((Divide and Conquer Table，DCT)来实现数据的动态更新操作。

DCT 包括两个重要组成部分：1)逻辑序列($L_i$)，描述了数据块的原始指针；2)版本号($V_i$)，基于更新数量描述当前块的版本号。如果 DO 更新一个数据块，则 DCT 中对应的 $V_i$ 加 1，表示修改的块。外包数据的物理位置与 DCT 中每个块的指针相匹配。DO 在上传外包数据到云端之前，会为每个文件生成 DCT 数据结构。

尽管该类型数据结构能有效实现小文件的动态更新操作，然而在更新大量文件时会增加认证的计算负载。例如，为了将一个新数据块插入到第 $i$ 个数据块后面，审计端必须移动 $n-i$ 个数据块，这将增加审计阶段的计算负载。因此，为了解决该问题，并减小 DCT 大小且缓解节点再平衡问题，本文将 DCT 分为 $k$ 个单独的数据结构，每个数据结构大小为 $\frac{n}{k}$。因此，在使用新 DCT 结构下，当新数据块插入到第 $i$ 个块后，DO 仅需要移动$\frac{n}{k}-i$个块。之后的实验结果也证明了本文数据结构能有效完成大规模数据文件的更新操作。下面将讨论本文算法如何完成动态数据操作。

### 3.1 数据修改

数据修改是数据审计技术的一种重要需求，将文件($f[i]$)的第 $i$ 个块修改为 $f'[i]$时，DO 运行修改算法步骤如下。

(1)在 DCT 中，通过比较 $i$ 和每个 DCT 的范围来寻找查询块($i$)的位置，然后块版本号增加 1，即 $V_i=V_i+1$。

(2)使用式(8)生成修改后数据块的新标签：

$$T_i'=S_Y(f'[i] \| (ID_F \| i \| L_i \| V_i'))$$
$$C_i'=S_Y(ID_F \| i \| L_i \| V_i') \tag{8}$$

(3)生成一个包含($ID_F, i, f'[i], T_i', C_i'$)的修改消息，并将其传输给 CSP。

当 CSP 接收到修改消息后，用块 $f'[i]$代替 $f[i]$且将数据块($T_i, C_i$)的标签变为($T_i', C_i'$)。图 2 显示了 DO 修改块 $f[7]$时(在每个 DCT 中存在 5 个实体)对 DCT 的影响。

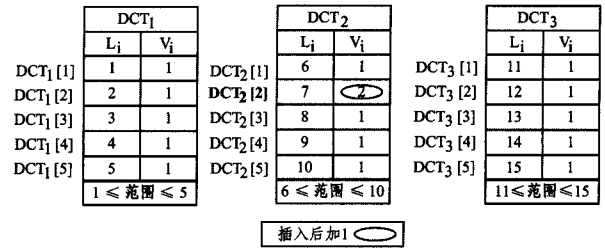


图 2 当每个表中块数量为 5 时，修改 $f[7]$后对 DCT 的影响

### 3.2 数据插入

数据插入操作是将新数据块($f^*[i+1]$)插入到文件($f[i]$)的第 $i$ 个块之后，DO 执行插入算法，其步骤如下。

(1)根据 DCT 范围，在 DCT 中寻找存储文件 $F$ 的第 $i$ 个块的位置，并确定该数据结构中新数据块($l$)的精确位置。

(2)将剩余块平移($\frac{n}{k}-l$)个位置，并在第 $i$ 个块后创建一个新行($L_{i+1}^*, V_{i+1}^*$)。

(3)设置新数据块的原始指针 $L_{i+1}^*=m+1$ 且新块的版本号 $V_{i+1}^*=1$，其中 $m$ 为 DCT 中块数量的最大值。

(4)当前 DCT 的最大范围和随后 DCT 的最小和最大范围均加 1。

(5)使用式(9)生成每个新数据块的标签：

$$T_{i+1}=S_Y(f^*[i+1] \| (ID_F \| i+1 \| L_{i+1}^* \| V_{i+1}^*))$$
$$C_{i+1}^*=S_Y(ID_F \| i+1 \| L_{i+1}^* \| V_{i+1}^*) \tag{9}$$

(6)生成一个包含($ID_F, i+1, f^*[i+1], T_{i+1}^*, C_{i+1}^*$)的插入消息，并发送该消息给 CSP。

根据接收到的插入消息，CSP 将该新块和对应的标签插入到文件的第 $i$ 个位置。图 3 描述了插入块对 DCT 结构的影响。为了在文件第 7 个块后插入新块，DO 仅需要平移 3 个实体就可在 $DCT_2$ 中插入新块($DCT_2[3]=\{16,1\}$)，并增加 $DCT_2$ 最大范围值和下一个 DCT 的最大和最小范围值。

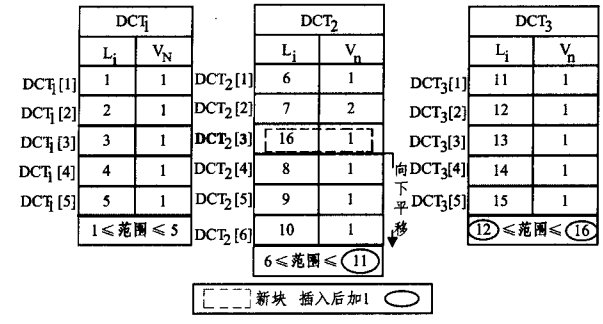


图 3 当每个表中块数量为 5 时，在 $f[7]$后插入新数据块对 DCT 的影响

### 3.3 数据删除

删除操作与插入操作相反，即从外包文件($f[i]$)中删除第 $i$ 个数据块。因此，为了删除数据块，DO 必须根据 DCT 最大和最小范围在 DCT 中寻找存储的第 $i$ 个数据块。然后在 DCT 中确定该块($p$)的位置，并将后面的块($\frac{n}{k}-p$)向上移动一个位置来删除该数据块。最后，DO 将包含($ID_F$，$i$)的删除消息传输给 CSP。图 4 显示了删除操作对 DCT 的影响，即通过向上平移一行($f[5]$)删除第 4 个数据块($f[4]$)，并减少 $DCT_2$ 和 $DCT_3$ 的最大和最小范围以及第一个 DCT 的最大范围。

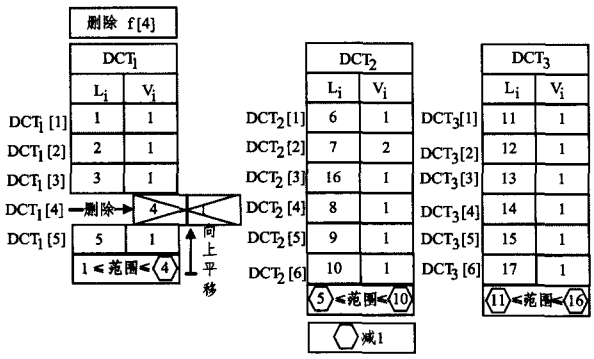


图 4 当每个表中块的数量为 5 时，删除 DCT 中 $f[4]$对 DCT 的影响

## 4 实验及分析

### 4.1 安全性能分析

代数签名在使用中可取足够长的签名以使代数签名以极低的概率碰撞。一个 256B 的签名仅有 $2^{-256}$ 的可能性发生碰撞，这对于一个未持有任何秘密信息的攻击者来说，构造签名碰撞是极难完成的[13]。挑战中选取的数据块数量可根据用户的要求进行改变，这将防止云存储服务器对固定数量的预先计算并存储所有可能的校验标签值。成功的攻击需要满足两个条件[14]：1)完成对原始数据的破坏；2)没有被审计机制所发现。下面计算了本文审计机制对恶意行为的理论检测率。

本文基于随机采样策略来执行本文审计策略。采样技术将输入文件($F$)分为许多块($m$)，并随机选择一些块($c$)作为挑战信息用于批处理过程。接着，分析本文数据审计算法对不端行为的检测性能。

假设 CSP 修改 $m$ 个外包块中的 $y$ 个块，则损坏块的概率为 $p_y=\frac{y}{m}$。设 $c$ 为挑战阶段 TPC 需要认证的数据块数量，$n$ 为每个块的分区数量。设 $x$ 为 DO 选择的离散块的数量，这些块与 CSP 修改的块相匹配。那么，TPC 对恶意修改的检测率 $p_x(x\geqslant 1)$计算如下：

$$\begin{aligned}p_x(x\geqslant 1)&=1-p_x(x=1)\\&=1-(\frac{m-y}{m})\cdot(\frac{m-y-1}{m-1})\cdots(\frac{m-y-c+1}{m-c+1})\\&=1-(1-\frac{y}{m})\cdot(1-\frac{y}{m-1})\cdots(1-\frac{y}{m-c+1})\\&=1-\prod_{i=0}^{c-1}(1-\frac{y}{m-i})\end{aligned}\tag{10}$$

另外，

$$(1-\frac{y}{m-i})\leqslant(1-\frac{y}{m})\Rightarrow\prod_{i=0}^{c-1}(1-\frac{y}{m-i})\leqslant(1-\frac{y}{m})^c$$

$$\Rightarrow 1-\prod_{i=0}^{c-1}(1-\frac{y}{m-i})\geqslant 1-(1-\frac{y}{m})^c\tag{11}$$

所以，本文审计策略对恶意行为的理论检测率为：

$$p_x(x\geqslant 1)\geqslant 1-(1-\frac{y}{m})^c\tag{12}$$

在配置为 Intel Core i5-2450M CPU，2.5GHz 和 6GB RAM 的 PC 机上，使用 C++实现本文算法，通过实验研究本文审计策略的实际检测性能。选取 1GB 外包文件，并将其分为 125000 个数据块，设定固定检测数据块数量为 1000。在损毁数据块数量从 1 到 5000 块变化的情况下进行审计性能实验，结果如图 5 所示。可以看出，当存在很少损毁数据块时，检测率几乎为 0，这是因为损毁数据块数量没有达到检测阈值；当存在较多损毁数据块时，本文审计策略的成功检测率几乎达到 100%。

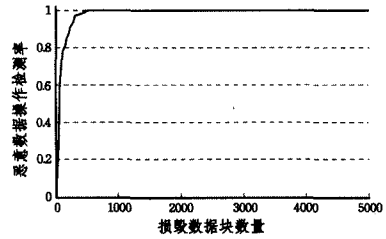


图 5 不同损毁数据块数量下，恶意操作的检测率

### 4.2 计算性能比较

从算法计算负载方面评估本文算法的性能，其中计算负载包括通信负载、审计计算负载和动态数据更新计算负载[1]。1)通信负载：不同审计阶段，审计端和服务器端平移数据的数量。2)数据审计计算负载：数据审计的每个阶段需执行的任务量。认证端的计算负载为用于认证外包数据完整性所需的计算资源；服务器端的计算负载为响应阶段中处理和生成证据消息所需的计算资源。3)动态数据更新的计算负载：DO 执行插入、删除和修改操作的计算负载。

将所提算法和文献[9,10]的算法进行比较，实验结果为 20 次实验的均值。实验中，设定 DO 将 1GB 文件分为大小为 8kB 的 125000 个块，并上传块到云端。更新块的数量以间隔 8 从 100 增加到 1000，实验结果如图 6 所示。可以看出本文算法计算负载最低，计算时间最短。这是因为在文献[9]提出的方法中，为了插入或删除块($i$)，审计端必须精准找到 MHT 树中块($i$)的位置，并重新计算新叶子节点的哈希值和现存节点到根节点的路径，这将导致审计端的高计算负载。在文献[10]提出的方法中，找到块($i$)的位置后，对每次插入或删除操作，审计端需要平移其余($n-i$)块。如果重复许多次，该过程在审计端的计算负载会增加。本文方法使用大小为 12500 块的 10 个 DCT 代替大小为 12500 块的单个 DCT，来降低审计端的计算复杂度。

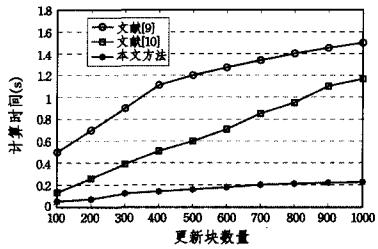


图 6　更新数据块数量对计算时间的影响

为了研究所提方法中分区($k$)对更新数据的计算复杂度的影响,设定了具有不同分区($k$)的 1GB 外包文件场景。本文方法在审计阶段的计算负载为 $O(n/k)$,所以 $k$ 越大,计算负载越低,但这也会增加块位置寻找的难度。在 1GB 外包文件中进行多次实验,获得一个近似最优数量为 $\sqrt{125000}\cong 352$。图 7 显示了在不同分区数量($k$)和近似最优数量 352 下,审计端的计算负载。可以看出,实验结果验证了上述分析结论。当分区数量 $k=100$ 时,插入 1000 个数据块的计算负载为 0.16s;当增加分区数量到 353 时,计算负载降低到 0.14s。

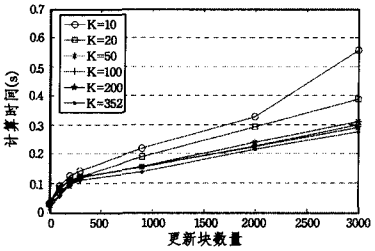


图 7　不同更新块数量下,分区数量对计算时间的影响

**结束语**　本文提出了一种基于代数签名的 DPA 技术来实现云端大数据的安全存储。同时,根据分而治之思想设计了一种新型数据结构来改进本文数据审计方法。该数据结构能有效支持动态数据操作,例如插入、删除和修改,通过减少平移数据块数量来降低数据更新和审计过程的计算负载。实验结果表明,所提方案具有较高的数据丢失检测率,且大大减少了数据计算量。

下一步将融入纠错码机制来扩展本文方法,使其能够对被删除的大数据文件进行恢复。

## 参 考 文 献


[1] Li H, Sun W H, Li F H, et al. Secure and Privacy-Preserving Data Storage Service in Public Cloud [J]. Journal of Computer Research and Development, 2014, 51(7): 1397-1409(in Chinese)
李晖,孙文海,李凤华,等. 公共云存储服务数据安全及隐私保护技术综述[J]. 计算机研究与发展,2014,51(7):1397-1409

[2] Hong H S, Sun Z X. Big data Storage Security Based on Cloud Computing [J]. Journal of Nanjing University of Posts and Telecommunications(Natural Science), 2014, 34(4): 26-32(in Chinese)
洪汉舒,孙知信. 基于云计算的大数据存储安全的研究[J]. 南京邮电大学学报(自然科学版),2014,34(4):26-32

[3] Xiao D, Yang L, Liu C, et al. Efficient Data Possession Auditing for Real-World Cloud Storage Environments [J]. Ieice Trans. inf. & Syst, 2015, 98(4): 796-806

[4] An B Y, Gong Z, Xiao D, et al. Data Possession Audit with an Implicit Trusted Third-party for Cloud Storage [J]. Journal of Harbin Engineering University, 2012, 33(8): 1039-1045(in Chinese)
安宝宇,宫哲,肖达,等. 具有隐式可信第三方的云存储数据持有性审计[J]. 哈尔滨工程大学学报,2012,33(8):1039-1045

[5] Qin Z G, Wang S Y, Zhao Y, et al. An Auditing Protocol for Data Storage in Cloud Computing with Data Dynamics [J]. Journal of Computer Research and Development, 2015, 52(10): 2192-2199(in Chinese)
秦志光,王士雨,赵洋,等. 云存储服务的动态数据完整性审计方案[J]. 计算机研究与发展,2015,52(10):2192-2199

[6] Li M C, Zhang L, Guo C. Accurate Location in Batch Dynamic Provable Data Possession [J]. Applied Mechanics & Materials, 2014, 51(3): 9-12

[7] Li C L, Chen Y, Tan P X, et al. Towards Comprehensive Provable Data Possession in Cloud Computing [J]. Wuhan University Journal of Natural Sciences, 2013, 18(3): 265-271

[8] Cash D, Küpçü A, Wichs D. Dynamic Proofs of Retrievability via Oblivious RAM [J]. Journal of Cryptology, 2015, 7881: 1-36

[9] Wang Q, Wang C, Ren K, et al. Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing [J]. IEEE Transactions on Parallel & Distributed Systems, 2011, 22(5): 847-859

[10] Yang K, Jia X. An Efficient and Secure Dynamic Auditing Protocol for Data Storage in Cloud Computing [J]. IEEE Transactions on Parallel & Distributed Systems, 2013, 24(9): 1717-1726

[11] Jiao W Z, Wang G Q, Zhai Z J, et al. Algebraic Signature based Data Possession Checking for Cloud Storage [J]. Journal of Northeast Normal University (Natural Science Edition), 2013, 45(4): 55-61(in Chinese)
焦文喆,王国庆,翟正军,等. 云存储下基于代数签名的数据持有性检查方法[J]. 东北师大学报(自然科学版),2013,45(4):55-61

[12] Sun W, Zhu Z L. Graph Classification Algorithm based on Divide and Conquer Strategy and Hash Linked List [J]. Computer Engineering & Science, 2013, 35(3): 145-149(in Chinese)
孙伟,朱正礼. 基于分而治之及 Hash 链表的图分类算法[J]. 计算机工程与科学,2013,35(3):145-149

[13] Ameen J N, Mohamed J J, Begam N N. Dynamic Auditing Protocol for Efficient and Secure Data Storage in Cloud Computing [J]. Compusoft International Journal of Advanced Computer Technology, 2014, 3(6): 263-272

[14] Ni J B, Yu Y, Mu Y, et al. On the Security of an Efficient Dynamic Auditing Protocol in Cloud Storage [J]. IEEE Transactions on Parallel & Distributed Systems, 2014, 25(10): 2760-2761

[15] Huang Zhi-hong, Wu Li-li, Zhang Bo. Network Security Threats and Prevention on Cloud Computing [J]. Journal of Chongqing University of Technology(Natural Science), 2012, 26(8): 85-90(in Chinese)
黄志宏,巫莉莉,张波. 基于云计算的网络安全威胁及防范[J]. 重庆理工大学学报(自然科学版),2012,26(8):85-90