一种基于同态标签的动态云存储数据完整性验证方法*

胡德敏[a,b], 余 星[a]
(上海理工大学 a. 光电信息与计算机工程学院; b. 计算机软件教研室, 上海 200093)

摘 要: 在云存储服务中,为使用户可以随时验证存储在云存储服务器上数据的完整性,提出一种基于同态标签的动态数据完整性验证方法。通过引入同态标签和用户随机选择待检测数据块,可以无限次验证数据是否完好无损,并支持数据动态更新;可信第三方的引入解决了云用户与云存储服务供应商因数据完整性问题产生的纠纷,实现数据完整性的公开验证;然后给出该方法的正确性和安全性分析,以及该方法的性能分析;最后通过实验验证了该方法是高效可行的。

关键词: 云计算; 云存储; 同态标签; 数据完整性; 公开验证
中图分类号: TP393 **文献标志码**: A **文章编号**: 1001-3695(2014)05-1362-04
doi:10.3969/j.issn.1001-3695.2014.05.018

Dynamic cloud storage data integrity verifying method based on homomorphic tags

HU De-min[a,b], YU Xing[a]
(*a. School of Optical-Electrical & Computer Engineering, b. Dept. of Computer Software, University of Shanghai for Science & Technology, Shanghai* 200093, *China*)

Abstract: In order to allow users can always verify the integrity of the data stored in cloud storage server, this paper proposed a dynamic data integrity verifying method based on homomorphic tags. Users could infinitely verify whether data integrity intact by introducing a homomorphic tags and user randomly selected data blocks to be verified, and support data dynamic updating; trusted third party auditor (TPA) could solve disputes of data integrity between users and cloud storage provider to realize the public validation of the data; and analysed correctness, security and performance of this method; finally the proposed method that could detect cloud storage data integrity was verified by experiment.

Key words: cloud computing; cloud storage; homomorphic tag; data integrity; public validation

0 引言

云存储服务的诞生与发展,以及存储即服务的理念,为广大云用户提供了廉价的存储空间,同时也对传统的数据存储方式发起了挑战。尽管云存储具有使用方便、按需收费、价格低廉等特点,但其推广过程却是很缓慢。从 Twinstrata 公司 2012 年的最新调查可以看出,只有 20% 的人愿意使用云存储服务来存储自己的私有数据,大约有 50% 的人愿意使用云存储服务来进行数据备份及灾难恢复等作业,因此云存储数据安全问题是进一步推广云存储服务的主要障碍之一。云存储数据安全问题的研究主要包含数据的完整性、机密性等问题。

云存储数据的完整性检测是验证不可信的云存储服务器是否能保证数据的完好无损,避免云用户存储在其中数据被窜改或删除。云存储数完整性研究主要集中在可证明数据持有(provable data possession,POR)方案和可恢复证明(proof of retrievability,PDP)方案。POR 与 PDP 的主要区别是:PDP 方案支持检测数据完整性,但无法保证数据可恢复性;POR 可以确保存储数据的可恢复性[1]。

Ateniese 等人[2]提出基于对称密码技术构造 PDP 方案。该方案由用户设定挑战次数和待检测数据块,把响应作为云数据存储在客户端,因此更新和检测次数是有限的。Erway 等人[3]提出了两种动态的数据持有性证明方案,一种是基于等级的鉴别跳表(rank-based authenticated skip lists),另一种是 RSA 加密算法的树结构。它们的主要目标是实现数据的动态性,即实现数据插入操作。肖达等人[4]在国内最早提出了使用数据持有性检测方法,其思想是在一个挑战—应答协议,用户要求服务器计算其随机指定文件中的若干个数据块的 hash 值,并与对应的校验块一起返回。通过这种随机抽样校验的方法,在保证足够置信度的同时降低了持有性检查的计算和通信开销,通过循环队列的挑战更新机制使得用检测者发起有效挑战的次数。该方法虽然解决了用户的挑战次数不受限制,但其数据完整性检测是在客户端进行的,这样无形之中就增加了用户的计算成本和通信成本。此外,该方案并不支持公开性验证,很难解决云用户与云供应商因数据完整性问题产生的纠纷。陈龙等人[5]提出了一种基于纠错编码思想的细粒度数据完整性检测方法——完整性指示码,该方法给出了完整性指示码的几个性质,然后设计了指示单个错误的组合单错码,分析

收稿日期: 2013-07-18; **修回日期**: 2013-08-14 **基金项目**: 国家自然科学基金资助项目(61202376);上海市教委科研创新资助项目(13YZ075);上海市教委"晨光计划"资助项目(10CG49)

作者简介:胡德敏(1963-),男,上海人,副教授,博士,主要研究方向为计算机网络、云计算、软件理论等(shanghaiusst@126.com);余星(1987-),男,湖北应城人,硕士研究生,主要研究方向为云计算、计算机网络.

该码的基本性能,提供了在低出错率的条件下 hash 数据大幅度压缩的方法,从而验证数据的完整性。该方法只设计了单个错误的组合单错码,应用场景单一,在指示多个错误等其他一些实际情况中就不适用了。Shacham 等人[6]提出一个基于伪随机函数的数据完整性检测方案,不支持公开性验证。Boneh 等人[7]提出了一个基于 BLS 签名的数据完整性验证方案,支持数据的公开性验证。RSA 实验室的 Bowers 等人[8]提出了一个 POR 的理论架构,用于改进已有的方案。他们指出数据的更新和公开性验证依然是未解决的公开性问题。

综上所述,现有的关于数据完整性验证方案存在以下不足:a)只能进行有限次数据完整性验证;b)大部分数据完整性验证方案是基于公钥加密技术,其计算开销太大;c)有些方案不支持公开性验证;d)有些方案不适用于云存储海量数据服务模式。基于以上研究,本文提出一种基于同态标签的动态数据完整性检测方法,用户可以无限次地验证数据的完整性,并支持公开性验证以及数据的动态性;利用同态标签可以极大地减少数据完整性检测方案带宽需求。本文方法的所有服务都在云存储服务器上,减少了云用户的通信开销和计算开销。通过方法的正确性、安全性、性能分析以及实验结果证明该方法是可行的。

1　数据的完整性检测

1.1　同态标签

同态是从一个代数结构到另一个代数结构的映射,它保持所有相关的结构不变,即存在映射 $\Phi: X \to Y$,满足[1]:

$$\Phi(x \cdot y) = \Phi(x) \circ \Phi(y)$$

其中:$\cdot$ 是 X 上的运算,$\circ$ 是 Y 上的运算。

同态标签是基于同态概念而生成的标签,在数据完整性检测过程中,可以利用同态标签的同态性来验证数据是否完整。同态标签具有以下两个性质:a)通过使用同态标签,可使用少量特定的数据块进行数据的完整性检测,而无须对所有的数据块进行检测,降低了通信开销和计算开销;b)对于任意两个数据块 m_i 和 m_j,$m_i + m_j$ 的标签信息 $T(m_i + m_j)$ 可以由它们各自的标签信息 $T(m_i)$、$T(m_j)$ 生成,即 $T(m_i + m_j) = T(m_i) * T(m_j)$。

1.2　云存储数据完整性验证

数据完整性证明方案由生成密钥(KeyGen)、标签生成(TagBlock)、挑战(Challenge)、证据生成(ProofGen)和证据验证(ProofVerify)五个阶段组成。

用户在云平台注册后,可调用密钥生成算法生成密钥参数,其中私钥用户自己保存,公钥公开,用于生成文件块标签信息。当用户上传文件到云服务器时,云服务器对文件进行预处理。首先将文件 F 划分成 n 个子块,得到文件子块 $m_i (1 \leqslant i \leqslant n)$;然后将每个文件子块分成 k 个基本块,得到基本块文件 $m_{i,j} (1 \leqslant i \leqslant n, 1 \leqslant j \leqslant k)$;再调用 TagBlock 算法和选定的密钥参数为每个基本块 $m_{i,j}$ 生成相应的标签信息 $T_{i,j}$;文件子块的标签信息 T_i 由基本块标签信息 $T_{i,j}$ 聚集得到。通过构建一个状态表(state table,S-Table)组织所有的文件子块 m_i,其中 BN 表示文件子块的物理序号,SN 表示文件子块的插入顺序,FB 表示文件子块。云存储服务器存储由密钥、标签信息集合和 S-Table 表构成的预处理后的文件。

1)密钥生成阶段(KenGen)　由密钥生成算法 KeyGen 完成对公私对(pk, sk)的生成,随机生成两个大素数 p 和 q,且 $p \neq q$,计算 $N = p \times q$ 和 $\varphi(N) = (p-1)(q-1)$;生成随机数 $e (e \in Z_N)$,使得 $\gcd(e, \Phi(N)) = 1$。则公钥 $pk = (N, e)$,私钥 $sk = (p, q)$,私钥用户自己保存,公钥公开,用于生成文件块标签信息。

2)标签生成阶段(TagBlock)　云服务器对每个基本块 $m_{i,j} (i \leqslant n, j \leqslant k)$ 生成相应的标签信息 $T_{i,j} = e^{m_{i,j} h(m_{i,j}) f_i(j)}$,文件子块的标签信息由基本块标签信息聚集得到,即 $T_i = e^{\sum_{i=1,j=1}^{i=n,j=c} m_{i,j} h(m_{i,j}) f_i(j) f(i)}$,然后把文件块和文件块的标签信息关联到 S-Table 上。其中:$h(\cdot)$ 为同态哈希函数,为每个文件子块生成一个唯一的同态哈希值;$f(\cdot)$ 是伪随机函数,分别为每个基本块和文件子块生成一个伪随机数。

3)挑战阶段(Challenge)　用户随机输入一个系数 c,云服务器根据收到的 chal 的系数 c 去找到对应的 c 个文件子块,以及根据每个文件子块包含的基本块的标签信息计算出文件子块的标签信息 T_i。生成随机数 $r \in [1, 2^k - 1]$,将挑战信息 chal = $\{r, e\}$ 发送给云存储服务器。

4)证据生成阶段(ProofGen)　云存储服务器接收到挑战信息后,挑战信息包含公钥 pk、文件子块 m_i、基本块 $m_{i,j}$ 以及检测请求 chal。计算 $e_r = e^r \bmod N$,并计算输出验证信息 $R = (e_r)^{\sum_{t=1,j=1}^{t=c,j=k} m_{t,j} h(m_{t,j}) f_t(j) f(t)} \bmod N$。

5)证据验证阶段(ProofVerify)　取得数据的验证信息 R、待检测文件的公钥 pk、文件块 m_i 和其对应的标签信息 T_i 以及检测请求 chal,然后计算 $T = \prod_{t=1}^{c} T_t^r \bmod N$ 和检测信息 $R' = T^r \bmod N$,并验证 R 和 R' 是否相等。若存在 $R = R'$,则返回 success;否则返回 fail。

1.3　正确性分析

为验证用户存储数据的完整性,即验证 $R = R'$,计算步骤如下:

$$\begin{aligned} R' = T^r \bmod N &= (T_1^r \cdot T_2^r \cdot \cdots \cdot T_c^r) \bmod N = \\ & e^{rm_{1,1} h(m_{1,1}) f_1(1) f(1)} \cdot e^{rm_{1,2} f(m_{1,2}) f_1(2) f(1)} \cdot \ldots \cdot \\ & e^{rm_{1,k} f(m_{1,k}) f_1(k) f(1)} \cdot \ldots \cdot e^{rm_{4,k} f(m_{4,k}) f_4(k) f(4)}) \cdot \ldots \cdot \\ & e^{rm_{c,k} h(m_{c,k}) f_c(k) f(c)}) \bmod N = \\ & \prod_{t=1,j=1}^{t=c,j=k} (e^{rm_{t,j} h(m_{t,j}) f_t(j) f(t)}) \bmod N = \\ & (e_r)^{\sum_{t=1,j=1}^{t=c,j=k} m_{t,j} h(m_{t,j}) f_t(j) f(t)} \bmod N = R \end{aligned}$$

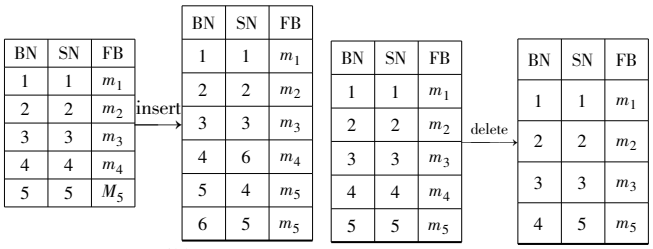
数据的完整性得证。若计算出 $R' \neq R$,则表明用户存在云存储服务器上的文件块被窜改或删掉。其中 T_i 为文件块标签信息,T 为多个文件子块的聚集标签信息,e 为小基数,$m_{t,j}$ 为基本块,$h(\cdot)$ 为同态哈希函数,$f(\cdot)$ 是一个伪随机函数,N 为两个不同的大素数 p 和 q 生成的参数。

1.4　数据动态更新

数据的更新主要包含三种操作:a)插入,在指定数据块后面插入新的数据块;b)修改,对指定的数据块进行修改;c)删除,删除指定的数据块。数据的更新主要在于更新用户存储的 S-Table 表,以及云服务器上存储的相应的标签信息。在三种更新操作中,S-Table 表的变化如图 1 所示。

当向云存储服务器中第 i 个数据块后插入一个新的数据块 t 时(图 1(a)),新数据块的物理序号 BN 为 $i+1$,插入顺序 SN 为数据块中最大的 SN 值加 1。首先把新的数据块 m_t 分成

基本块 $m_{t,j}$,然后根据其对应的标签信息 $T_{t,j}$ 计算数据块标签信息 T_t,并关联数据块 m_t 和标签信息 T_t,最后更新 S-Table。



(a)插入数据块 (b)删除数据块

图 1 数据块更新方法图

当用户要修改云存储服务器中的第 i 个数据块时,首先取得第 i 个数据块 m_i、m'_i;对新的数据块 m'_i 进行预处理操作后计算生成数据块标签信息 T'_i;把第 i 个节点下的数据块 m_i 和标签信息 T_i 更新为新 m'_i,并关联 T'_i,最后更新 S-Table。

当用户要删除存储在服务器中的第 i 个数据块时(图 1(b)),查询 S-Table 表,取得第 i 个数据块 m_i,删除第 i 个节点和对应的数据块 m_i,以及关联的标签信息 T_i,其后续节点的 BN 值全都减 1,SN 值保持不变,然后更新 S-Table。

1.5 可信第三方判定

文献[9]将每一次数据完整性验证交给可信第三方,这样会产生额外的通信负载和给可信第三方带来巨大的计算开销;并且每多进行一次数据完整性验证的同时,增大了数据泄密的可能性。在本文中只有当云用户与云存储服务供应商之间因数据的完整性验证而产生纠纷时,可申请可信第三方(TPA)介入。在减小了与可信第三方之间的通信负载的同时,提升了数据的机密性。

由于判断数据是完整性断定式中的参数 r、e,标签信息 T_i 等会在 TPA 中存有备份,且双方都对其签名认证过,而用户的公钥 pk 是公开的。此时为了公开判定云用户存在云存储服务器上的完整性,TPA 和云存储服务器只需要不端重复挑战阶段、证据生成阶段和证据验证阶段的工作。此时 TPA 就等同于云用户,其判定结果是公正的。

2 安全性证明

为了验证本方案的安全性,构建一个数据持有游戏,如果攻击者赢得此游戏,则攻击者能正确地获得全部密文数据块和签名标签信息。

定理 在大整数因式分解困难性问题的假设下,本文数据完整性检测方法是安全的。

证明 首先进行如下游戏描述:

a)生成密钥。用户调用 KeyGen 算法生成密钥,将公钥送给攻击者。

b)询问。攻击者选择数据块 $m_i(1\leqslant i\leqslant n)$ 发送给云用户,云用户调用 TagBlock 算法,根据数据块 m_i 包含的基本块的标签信息,计算收到的数据块 m_i 的标签信息 T_i,然后所有的标签信息发送给攻击者。

c)挑战。用户调用 Challenge$(pk,T_i)\rightarrow$chal 生成一个挑战 chal 发送给攻击者,要求攻击这根据此挑战信息计算出一个数据完整性证据 R。

d)伪造。攻击者根据挑战信息 chal 和持有的数据块 $m_i(1\leqslant i\leqslant n)$ 以及标签信息 $T_i(1\leqslant i\leqslant n)$ 生成一个检测信息 R,并将其发送给用户。

e)验证。云用户调用 ProofVerify 算法检验攻击者返回的证据 R 是否正确,若正确则攻击者赢得本次游戏,表明攻击者能正确得到所有的密文数据块和标签信息。

在游戏中,攻击者返回的数据完整性证据$\{m'_i,r',e'\}$,云用户根据返回的数据完整性证据计算验证信息,即 $R'=(e'_{r'})^{\sum_{i=1,j=1}^{i=c,j=k}m'_i,jf(m'_{i,j})f_i(j)f(i)} \bmod N$ 而 $R=e_r^{\sum_{i=1,j=1}^{i=c,j=k}m_{i,j}h(m_{i,j})f_i(j)f(i)} \bmod N$。当攻击者正确地拥有全部数据块和标签时,必有 $m'_i=m_i$,$r'=r$,$e'=e$,则 $R\equiv R'$ 成立。当攻击者窜改或删除其中一部分密文块 $m_i(1\leqslant i\leqslant n)$ 或标签信息时 $T_i(1\leqslant i\leqslant n)$,并希望通过数据的完整性验证,则必须伪造出合适的 $m'_i=m_i$ 或者 T_i 使得 $R=R'$ 成立,即攻击者有能力伪造出两个随机大素数 p 与 q,使得 $p\neq q$ 且 $g^p=g^q \bmod N$,其中 g 是 Z_N^* 的一个生成元。因此,$p-q=k\varphi(N)$ 成立,从而 $p-q$ 可以用来分解大整数 N。因此,在大整数因式分解困难性问题的假设下,若攻击者想要赢得上述数据完整性验证游戏,则攻击者必须正确地持有全部的密文块和所有的标签信息。

证毕。

3 性能分析

3.1 通信开销

通信开销主要集中在用户生成检测请求和返回检测结果两个阶段。在生成检测请求阶段,用户发送检测请求$\{r,e\}$到云存储服务器,其大小为 $k+|N|$ 比特。在返回结果阶段,云服务器返回检测结果$\{\text{success},\text{fail}\}$到客户端,其大小为 k 比特,所有总的通信开销为 $2k+|N|$ 比特,通信开销非常小。

3.2 存数开销

云存储服务器的存储开销主要有密钥存储,其开销为 $2|N|+|p|+|q|$ 比特;文件 m 的存储,其比特为 $|m|$ 个比特;存储每个数据块为 N,则存储文件块和基本块的总开销为 $2n\times|N|$。

3.3 计算开销

计算开销主要集中在标签生成、检测请求生成、验证信息生成和和验证完整性四个阶段。

a)标签生成阶段 总共要为 n 个数据块生成标签信息,计算复杂度为 $O(n)$。根据欧拉定理[10],由于 $\gcd(e,N)=1$,那么 $e^{\varphi(N)} \bmod N=1$。由于取模运算比模指运算高效得多,在这里只考虑幂指操作的开销。所以,标签生成阶段的计算开销为 $(n+n\times k)\times T_{\exp}(|N|,N)$,其中 n 表示数据块数;$n\times k$ 表示基本块数;$T_{\exp}(\text{len},\text{num})$ 表示一个整数的指数为 len 比特再取模 num 的模指运算的计算时间开销。

b)检测请求生成阶段 需要计算两个随机数(r,e),计算复杂度为 $O(1)$,其计算开销为 $T_{\text{prng}}(|N|)+T_{\text{prng}}(k)$。其中 $T_{\text{prng}}(\text{len})$ 表示生成一个 len 比特的伪随机数的计算开销时间。

c)验证信息生成阶段 计算复杂度为 $O(n)$。云服务器首先需要计算 $e_r=e^r \bmod N$,此过程执行模指运算,计算时间开销为 $T_{\exp}(|N|,N)$。然后需要进行 $n\times k+n$ 次伪随机数生成和计算 R。在计算 $\sum_{i=1,j=1}^{i=n,j=k}m_{i,j}h(m_{i,j})f_i(j)f(i)$ 需要进行 $n\times k$ 次大型乘法计算,因为 $f_i(j)$、$f(i)$ 和 m_i 的长度分别为 d 比特、d 比特和 l 比特,$h(m_{i,j})$ 为 h 比特,那么计算完每个 $m_{i,j}h(m_{i,j})f_i(j)f(i)$ 后,然后计算它们和 $\sum_{i=1,j=1}^{i=n,j=k}m_{i,j}h(m_{i,j})f_i(j)f(i)$。因此验

证信息阶段的总计算开销为

$$T_{\exp}(|N|,N)+(n\times k+n)T_{\text{prng}}(d)+n\times k\times T_{\text{mul}}(2d+l+h)+n\times k\times T_{\text{add}}(2d+l+h)$$

其中:$T_{\text{mul}}(\text{len})$表示几个 len 比特的数相乘的计算开销,$T_{\text{add}}(\text{len})$表示几个 len 比特的数相加的计算时间开销。

d)验证完整性阶段　计算复杂度为 $O(n)$。云存储服务器需要$(n+1)$次模指运算和$(n-1)$次模乘运算。整个阶段的计算开销为

$$(n+1)T_{\exp}(d,N)+(n-1)T_{\text{mul}}(|N|,N)$$

其中:$\text{sum}\times T_{\text{mul}}(\text{len},\text{num})$表示 sum 个长度为 len 比特的整数的取模 num 计算时间开销。

4　实验及结果分析

本文利用 Eucalyputs 技术搭建小型云平台,使用两台计算机和一台 Dell 服务器作为客户端和云服务器,首先配置了 Eucalyputs 和云平台的核心组建,配置了统一的网络通信时间;然后配置了云服务器与客户端的 SSH 服务和监听服务,以及配置了 MIRACL 库;最后根据验证本文提出云存储数据完整性方法的需要做以下实验:

实验 1　首先对三个 10M 的文件(文件 A、B、C)进行处理分块签名后存储;然后对文件 A 删除 10% 的文件块,对文件 B 修改 10% 的文件块,对文件 C 不作任何处理;最后对这三个文件进行完整性检测。检测结果如表 1 所示。

表 1　文件完整性检测结果

文件名	检测结果
A	fail
B	fail
C	success

实验结果表明,当文件被删除和窜改后,不能通过本文云存储数据完整性检测,而没有修改的文件可以通过,即证明了本文所提出方法的可行性。

实验 2　首先对一个 10 MB 的文件进行处理分块签名后存储;然后对该文件分别窜改 0.1%、0.2%、0.4%、0.6%、0.8%、1%、1.5% 的文件块,最后对该文件进行完整性检测,实验次数分别为 10、20、30、40。实验结果如表 2 所示。

表 2　窜改文件比例与算法有效率的关系

实验次数	窜改率/%						
	0.1	0.2	0.4	0.6	0.8	1.0	1.5
10	8	9	9	10	10	10	10
20	15	17	19	20	20	20	20
30	23	28	30	30	30	30	30
40	34	36	39	40	40	40	40

实验结果表明,对一个 10 MB 文件窜改 0.6% 左右后,文件完整性检测的成功率为 100%,算法检测效率较高,可以准确检测出文件的完整性是否被破坏。

实验 3　首先对一个 100 MB 的文件进行处理分块签名后存储,然后分别取 50、100、150、200、250、300、400 个数据块进行多次完整性检测。实验结果如表 3 所示。

实验结果表明,在进行数据完整性检测时,当数据块为 200 个左右时,成功检测数据完整性次数,算法具有较高的有效率,验证的数据块量比较小,用户能方便快速地知道其所存数据的完整性是否被破坏。

表 3　数据块量与算法有效率的关系

实验次数	数据块量/个						
	50	100	150	200	250	300	400
10	7	8	9	10	10	10	10
20	16	17	17	20	20	20	20
30	23	25	28	29	30	30	30
40	34	37	38	40	40	40	40

实验 4　对于同一个文件 F,选择不同的文件块大小,对完整性检测计算实验开销。选择一个 10 MB 的文件,依次选取 2、4、8、16、32、64、128 KB 大小,每次选取 460 块,然后记录每次实验所花费的时间和通信开销。实验结果如表 4 所示。

表 4　完整性检测过程计算开销

BlockSize/KB	2	4	8	16	32	64	128
检测请求阶段/s	0.15	0.14	0.13	0.16	0.15	0.14	0.16
验证信息生成阶段/s	0.65	0.63	0.64	0.63	0.65	0.64	0.62
验证完整性阶段/s	0.81	0.84	0.82	0.89	0.85	0.80	0.83
通信开销/KB	52	53	57	51	54	56	53

实验结果表明,在进行数据完整性检测时,文件块的大小对检测计算开销几乎没有什么影响,各个过程的计算时间基本保持稳定,检测请求生成阶段计算时间约为 0.14 s,验证信息生成阶段计算时间约为 0.64 s,数据完整性验证阶段计算时间约为 0.83 s,对于一个 10 MB 的文件整个检测过程大概需要 1.61 s,计算时间总体较小,符合预期设计目标。实验环境比较稳定的情况下,每次实验通信开销始终稳定在 50 几 KB,通信开销较小,符合预期设计目标。

5　结束语

本文提出了一种基于同态标签的数据完整性检测方法,证明了该方法的正确性和安全性,并对其通信开销、存储开销和计算开销等性能进行了分析。所有验证过程中的计算工作是在云端服务器上完成,减少了网络带宽的消耗。此外,本文方法还实现了公开性验证,解决了用户与云存储供应商因数据完整性问题而产生的纠纷,以及实现了数据的动态更新。下一步的研究重点是在进行申请第三方公开性验证时,如何确保数据的机密性,或者不需要引入可信第三方就可以实现公开性验证。

参考文献:

[1] 陈兰香. 一种基于同态 hash 的数据持有性证明方法[J]. 电子与信息学报,2011,33(9):2200-2204.

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

[3] ERWAY C, KÜPÇÜ A, PAPAMANTHOU C, *et al*. Dynamic provable data possession[C]//Proc of the 16th ACM Conference on Computer and Communications Security. New York: ACM Press, 2009: 213-222.

[4] 肖达,舒继武,陈康,等. 一个网络归档存储中实用的数据持有性检查方案[J]. 计算机研究与发展,2009, 46(10):1600-1668.

[5] 陈龙,王国胤. 一种细粒度数据完整性检测方法[J]. 软件学报,2013,20(4):902-909.

[6] SHACHAM H, WATERS B. Compact proofs of retrievability[C]// Proc of the 14th International Conference on the Theory and Application of Cryptology and Information Security. Berlin: Springer-Verlag, 2008: 90-107.

(下转第 1395 页)

统计该区域的所有患者都在哪些时段内发病,这就需要使用时态集合上的并运算来处理关于该病例的电子医疗时态数据。同理,在某些非时态属性一致的前提下,举例设患者的时态变量值分别为 $D_{R_1}=\{[5,8]\}$,$D_{R_2}=\{[3,7],[9,16]\}$,$D_{R_3}=\{[2,8],[20,25],[5,12]\}$,则

$$
\begin{aligned}
&D_{R_1}\cup D_{R_2}\cup D_{R_3}\rightarrow\\
&\rightarrow\left\{\begin{array}{l}\langle[5,8],[3,7],[2,8]\rangle\\ \langle[5,8],[3,7],[20,25]\rangle\\ \langle[5,8],[3,7],[5,12]\rangle\\ \langle[5,8],[9,16],[2,8]\rangle\\ \langle[5,8],[9,16],[20,25]\rangle\\ \langle[5,8],[9,16],[5,12]\rangle\end{array}\right.\rightarrow\left\{\begin{array}{l}\langle[2,8]\rangle\\ \langle[3,8],[20,25]\rangle\\ \langle[3,12]\rangle\\ \langle[2,16]\rangle\\ \langle[5,16],[20,25]\rangle\\ \langle[5,16]\rangle\end{array}\right.\rightarrow\\
&\rightarrow\{\langle[2,8]\rangle\cup\langle[3,8],[20,25]\rangle\cup\langle[3,12]\rangle\cup\\
&\langle[2,16]\rangle\cup\langle[5,16],[20,25]\rangle\cup\langle[5,16]\rangle\}\\
&\rightarrow\{[2,16],[20,25]\}
\end{aligned}
$$

4　结束语

时态属性作为刻画事物的一个重要维度,对时态信息检索、时态知识推理、时态数据挖掘等研究领域均有深远的影响。针对海量的非结构化电子医疗时态数据,以时态集合作为基本的时态数据存储结构,本文提出使用 Hadoop 平台下分布式存储系统 HBase 进行数据存取,并在此基础上借助 Map/Reduce 并行编程模式对存储的时态数据进行分析和处理,首先将时间点和时态区间为时态属性值的时态数据以时态集合的形式重新构建、扩展时态区间的关系代数运算为时态集合的关系代数演算提供基础性支撑,并提出在 Map/Reduce 下时态关系代数演算算法,实现以时态集合为对象的交、并等基本关系代数运算操作,以满足用户对超大规模和高并发的非结构化时态数据的存储需求与高效的查询访问请求。需要指出的是,本文的研究工作是在单一时间粒度的前提下开展的确定性时态信息的研究,是较为理想下的时态数据处理模型。然而在现实生活中涉及的多粒度模糊的不确定时态信息,不仅需要进行不同时态粒层的映射转换,而且还需额外地增加时态因子以描述时态信息的不确定性程度。鉴于问题的复杂性,同时限于篇幅,将另行讨论。

参考文献:

[1] ALLEN J F. Maintaining knowledge about temporal intervals[J]. Communications of the Association for Computing Machinery, 1983,26(11):832-843.

[2] COMBI C, POZZI G, ROSSATO R. Querying temprol clinical databases on granular trends [J]. Journal of Biomedcal Informatics, 2012,45(2):273-291.

[3] COWLEY W, PLEXOUSAKIS D. A interval algebra for indeterminate time[C]//Proc of the 17th National Conference on Artificial Intelligence. Austin: AAAL, 2000:470-475.

[4] GAO Deng-feng,GENDRANO J A G, MOON B, *et al*. Main memory-based algorithms for efficient parallel aggregation for temporal databases [J]. Distributed and Parallel Databased, 2004, 16(2): 123-163.

[5] 叶小平. 基于时态变量对象关系模型及代数运算[J]. 计算机研究与发展,2007,44(11):1971-1979.

[6] JENSEN C S, SNODGRASS R T, SOO M D. Extending existing dependency theory to temporal databases [J]. IEEE Trans on Know ledge and Data Engineering,1996,8(4):563-582.

[7] BRUSONI V , CONSOLE L, TERENZIANI P, *et al*. Qualitative and quantitative temporal constraints and relational databases: theory, architecture, and applications [J]. IEEE Trans on Knowledge and Data Engineering,1999,11(6):948-968.

[8] 汤庸,刘海,郭欢,等. TempDB:时态数据管理系统[J] .计算机研究与发展,2010,47(z1):442-445.

[9] 王宁,杨扬,由海涌,等. 极大有序频繁项目集的时间属性分析方法[J]. 小型微型计算机系统, 2013,34(1):120-124.

[10] WBITE T. Hadoop:the definitive guide [M]. 3rd ed. [S. l.]: O' Reilly Media, Inc,2009:1-16.

[11] BORTHAKUR D, GRAY J,SARMA J S, *et al*. Apache Hadoop goes realtime at Facebook[C]//Proc of ACM SIGMOD International Conference on Management of Data. New York:ACM Press,2011:1071-1080.

[12] YANG Jin,TANG De-yu,ZHOU Yi. A distributed storage model for EHR based on HBase[C]//Proc of Information Management, Innovation Management and Industrial Engineering. Shenzhen: IEEE Press, 2011:369-372.

[13] FONG L L,GAO Yu-qing,GUERIN X R,*et al*. Toward a scale-out data-management middleware for low-latency enterprise computing [J]. IBM Journal of Research and Development,2013,57(3/4):1-14.

[14] KOEHLER M,KANIOVSKYI Y,BENKNER S. An adaptive framework for the execution of data-intensive mapReduce applications in the cloud[C]//Proc of Parallel and Distributed Processing Workshops and Phd Forum. Shanghai: IEEE Press, 2011:1122-1131.

[15] FRANKE C,MORIN S,CHEBOTKO A,*et al*. Efficient processing of semantic Web queries in HBase and MySQL cluster [J]. IT Professional,2013,15(3):36-43.

(上接第 1365 页)

[7] BONEH D,LYNN B,SHACHAM H. Short signatures from the Weil pairing[J]. Journal of Cryptology, 2004, 17(4): 297-319.

[8] BOWERS K D, JUELS A,OPREA A. Proofs of retrievability:theory and implementation[C]//Proc of ACM Workshop on Cloud Computing Security. New York:ACM Press,2009: 43-54.

[9] 于洋洋,虞慧群,范贵生.一种云存储数据完整性验证方法[J].华东理工大学学报:自然科学版, 2013,39(2):211-216.

[10] JONES G A,JONES J M. Elementary number theory[M]. [S. l.]: Springer-Verlag,1998.

[11] 彤丽,栗磊,李超零.一种可扩展的动态数据持有性证明方案[J].计算机应用研究,2013,30(7):2132-3135.

[12] YANG Yi, WANG Xin-yan, ZHU Sen-cun, *et al*. SDAP: a secure hop-by-hop data aggregation protocol for sensor networks[J]. ACM Trans on Information and System Security,2008,11(4):1-43.

[13] VARALAKSHMI P,DEVENTHIRAN H. Integrity checking for cloud environment using encryption algorithm [C]//Proc of International Conference on Receut Treuds in Information Technology. 2012:228-232.

[14] DESWARTE Y, QUISQUATER J J, SAIDANE A. Remote integrity checking[C]//Proc of IFIP Integrity and Internal Control in Information Systems. Berlin:Springer,2003:1-11.

[15] 曹夕,许力,陈兰香.云存储系统中数据完整性验证协议[J].计算机应用,2013,32(1):8-12.