DOI:10.3876/j.issn.1000-1980.2015.03.015

改进的基于同态哈希的云存储数据完整性验证方案

黄 石,刘文卓,曹天杰

(中国矿业大学计算机科学与技术学院,江苏 徐州 221116)

摘要:为了解决云存储用户数据完整性验证问题,在分析现有远程数据完整性校验方法的基础上,在标签生成的过程中加入同态哈希与伪随机数,提出一种支持动态数据与无限次挑战的同态哈希的数据完整性验证方案。通过安全性与性能的分析证明该方案的有效性,在保证远程数据完整性的同时,减少了存储空间的冗余和带宽消耗。

关键词:云存储;数据完整性;同态哈希;标签

中图分类号:TP302.7 **文献标志码**:A **文章编号**:1000-1980(2015)03-0278-05

An improved method of data integrity verification based on homomorphic hashing in cloud storage

HUANG Shi, LIU Wenzhuo, CAO Tianjie

(*School of Computer Science and Technology, China University of Mining and Technology, Xuzhou* 221116, *China*)

Abstract: Based on analysis of existing remote data integrity verification methods and using homomorphic hashing and pseudo-random numbers in the tag blocking stage, a data integrity verification method, supporting dynamic data and considering challenges to be unlimited, is proposed to deal with the problem of users' data integrity verification in cloud storage. The effectiveness of this method is verified through security and performance analyses. With a guarantee of maintaining the remote data integrity, the method can reduce the redundancy of storage space and the bandwidth consumption in data communication.

Key words: cloud storage; data integrity; homomorphic hashing; tag

1 数据完整性验证方案研究现状

远程数据验证允许客户端在一个不可信的服务器上检验外包数据的完整性。可恢复性证明POR(proof of retrievability)是Juels等[1]提出的完整性验证算法,其关键是将一些随机的数据块加入到存储的数据中,其插入的位置由伪随机序列决定,并使用了纠错码。这些数据块和数据本身没有任何关系,称之为"哨兵",这些"哨兵"起到了tag的作用,用于对数据进行完整性验证。数据持有性证明PDP(provable data possession)是Ateniese等[2]提出的,其2个显著特点是能够支持公开验证,以及在方案中使用了同态签名算法[3],这样就能使第三方以较小的开销来验证文件是否被完整存储。肖达等[4]提出数据持有性验证(data possession checking)的基本思想是验证者随机指定文件中若干个位置的数据块和相应的密钥,服务器计算哈希值并返回给验证者,验证者再比较哈希值与校验快是否一致,从而判定数据是否被正确持有,这是国内第一篇研究数据持有性验证的文章。由Ateniese[5]等提出的SPDP(scalable provable data possession)方案与PDP算法的区别在于新增了对动态数据的支持,采用了提前约定tag数目的标签组织方式,将生成的tag用对称密钥加密后保存在服务器或本地,每次挑战时使用一个tag进行验证,但该方案也限定了验证次数。Erway等[6]提出的DPDP(dynamic provable data possession)算法是使用了一种与树结构类似的跳表(skip

收稿日期:2014-11-12

作者简介:黄石(1979—),男,安徽桐城人,讲师,博士研究生,主要从事信息安全、云存储安全研究。E-mail:huangshi@cumt.edu.cn

list)结构来生成 tag,相比 SPDP 算法而言首次提出了一个可以支持动态数据的完整性验证方案。Chen 等[7]在 DPDP 算法的基础之上采用了 RS 码和柯西矩阵加强了原有算法的鲁棒性和动态更新的性能。除此以外还包括在私有云上的完整性验证方案 IPDP 和混合云上的完整性验证方案 CPDP 等[8]。

文献[9]针对上述一些算法中存在的问题提出了一种基于同态哈希的数据持有性证明方法,但该方法并不能防止服务器伪造与挑战块相对应的标签,从一定程度上说,无法很好地验证数据持有性。本文在文献[9]的基础上,结合文献[10]提出的同态哈希算法,提出一种支持动态数据和无限次挑战的基于同态哈希的数据完整性验证方案,在用户不用取回数据的前提下,很好地保证了远程数据的完整性并减少了存储空间的冗余与带宽消耗。

2　同 态 哈 希

同态是抽象代数中 2 个代数结构之间保持结构不变的映射。例如设 G,G'是两个群,f 是 G 到 G'的一个映射,若对任意的 $a,b\in G$ 都有 $f(ab)=f(a)f(b)$,则 f 就称为 G 到 G'的一个同态。

同态哈希(homomorphic hash)一直以来都被使用于对等网络中,通常与纠删码、网络编码一起抵抗攻击事件。在对等网络中,每个对等体会从其他的对等体那里直接获取原始数据块,因此可以用到 SHA1 这类哈希函数,通过比较所接收到的数据块的哈希值与原始哈希值来直接验证接收到的数据块的正确性。但 1 个对等体接收到的随机的编码包不能被源节点预先定义,因此标准的哈希函数在这里无法使用。但同态哈希函数却能使对等体发现伪造块的存在。

使用文献[10]中提出的同态哈希函数 $h_G(\cdot),h_G(\cdot)$有一组哈希参数 $G=(p,q,g)$,其中参数描述见表 1。g 的每个元素可以被表示成 $x^{(p-1)/q}\bmod p$,其中 $x\in Z_p$ 且 $x\neq 1$。

$$h_G(\cdot):\{0,1\}^k\times\{0,1\}^\beta\rightarrow\{0,1\}^{\lambda_p}$$

$$\mathrm{rand}(\cdot):\{0,1\}^k\times\{0,1\}^t\rightarrow\{0,1\}^t$$

其中 rand(·)是一个伪随机函数,作为一个伪随机数生成器来使用,用作初始化过程中同态哈希函数参数的生成、标签生成阶段随机数的生成以及挑战阶段确定随机抽取的用于挑战的数据块的选择,从而使挑战能够均匀覆盖所有的数据。

表 1　参数描述

Table 1　Parameter description

参数	描述
λ_p	离散对数安全参数(1 024 bit)
λ_q	离散对数安全参数(257 bit)
p	随机大素数 $\lvert p\rvert=\lambda_p$
q	随机大素数 $q\mid(p-1),\lvert q\rvert=\lambda_q$
β	数据块大小(16 kB)
m	每块中包含的子块数,$m=\lceil\frac{\beta}{\lambda_q-1}\rceil$(512 bit)
g	生成元 $1\times m$ 行向量,从 Z_p 中随机选取
G	哈希参数,包括(p,q,g)
n	文件分块数
seed	密钥流发生器的种子
MAXRAND	rand(·)可能输出的最大值
MINRAND	rand(·)可能输出的最小值

对于一个块的 b_i,哈希值可以按照式(1)计算:

$$h(b_i)=\prod_{k=i}^{m}g_k^{b_{k,i}}\bmod p \qquad (1)$$

原始块$(b_1,b_2,\cdots,b_n)$的哈希值分别为 $h(b_1),h(b_2),\cdots,h(b_n)$。

给出一个编码块 e_j 和一个系数向量$(c_{j,1},c_{j,2},\cdots,c_{j,n})$,则同态哈希函数 $h_G(\cdot)$满足下列等式:

$$h(e_j)=\prod_{i=1}^{n}h^{c_{j,i}}(b_i) \qquad (2)$$

这一特性就能够用来验证一个编码块的完整性。发布者首先需要预先计算每个数据块的同态哈希值,下载者下载这些同态哈希值,一旦收到了验证块就使用式(1)计算其哈希值,然后使用式(2)来验证该验证块的正确性。

3　基于同态哈希的远程数据完整性验证方案

3.1　方案描述

基于同态哈希的数据完整性验证方案的流程分为 5 个阶段,分别是初始化(Setup)、标签生成(TagBlock)、挑战阶段(Challenge)、证据生成(ProofGen)和证据验证(ProofVerify)阶段。

首先需要将文件分块存储,在后期的标签生成、证据验证等阶段都是以这些数据块为最小单位进行的。

在初始化阶段,主要生成一系列初始化参数用作哈希函数的生成上。在标签生成阶段,客户端使用伪随机数生成器产生一系列伪随机数,然后将文件块与伪随机数相乘得到标签。客户端将文件块 b_i、tag 以及 p、q 发送至服务器,客户端保存生成元 g、G 以及伪随机数生成器使用的种子 seed。在挑战阶段,客户端使用伪随机数生成器生成 k 个随机的挑战块发给服务器。在证据生成阶段,服务器计算出数据块与标签相应的证据 b_c 和 t_c,并将 b_c 和 t_c 返回客户端。在证据验证阶段,客户端用种子 seed 重新生成相应的伪随机数,验证服务器返回的 t_c 是否是客户端指定的 t_c,同时验证了此 t_c 是否对应正确的 b_c。

将文件 $\boldsymbol{F}$ 表示成一个 $m\times n$ 的矩阵,矩阵中的每个单元都是 Z_p 中的元素。对 m 的选择保证每一个元素都小于 2^{λ_q-1},因此小于 q。

$$\boldsymbol{F}=(b_1b_2\cdots b_n)=\begin{pmatrix}b_{11} & \cdots & b_{1n}\\ \vdots & & \vdots\\ b_{m1} & \cdots & b_{mn}\end{pmatrix}$$

此时,$\boldsymbol{F}$ 的第 j 列仅仅与文件 $\boldsymbol{F}$ 第 j 个消息块相关,写成 $b_j=(b_{1,j},\cdots,b_{m,j})$,因此对于 2 个文件块的加法运算只需要将相应的列向量直接相加来实现。也就是说,将文件的第 i 块和第 j 块相加,只需计算:

$$b_i+b_j=(b_{1,i}+b_{1,j},\cdots,b_{m,i}+b_{m,j})\bmod q$$

初始化阶段参数的生成使用文献[10]中的初始化算法,具体的算法实现见图 1。

$\mathrm{Setup}(\lambda_q,\lambda_p,m,s)\rightarrow G=(p,q,g)$	$x_j=h_G(b_j)=\prod_{i=1}^{m}g_i^{b_{i,j}}\bmod p$
do	$r_j=\mathrm{rand}(\mathrm{seed})$
$q\leftarrow q\mathrm{Gen}(\lambda q)$	then $\mathrm{Tag}=[x_1\cdot r_1,x_2\cdot r_2,\cdots,x_n\cdot r_n]$
$p\leftarrow p\mathrm{Gen}(\lambda_p)$	return $\mathrm{Tag}=[t_1,t_2,t_3,\cdots,t_n]$, $t_j=x_j\cdot r_j$
while $p=0$ done	the client sends $\boldsymbol{F}$, Tag, p, q to the server, saves G and seed
for($i=1$ to m) do	challenge:
do	the client select k blocks to challenge randomly
$x\leftarrow f(p-1)+1$	for($j=1$ to k) do
$g^i\leftarrow x^{(p-1)/q}(\mathrm{mod}p)$	$r_j'=\lceil \mathrm{rand}(\mathrm{time}())/(\mathrm{MAXRAND}-\mathrm{MINRAND})\times n\rceil$
while $g_i=1$ done	then the client sends $<b_{r_j'},k>$ to the server
done	ProofGen:
return (p,q,g)	for($i=1$ to $b_{r_j'}$) do
$q\mathrm{Gen}(q,\lambda_q)$	$b_c=(\sum_{i=r_1'}^{r_k'}b_i)\bmod q$
do	$t_c=(\prod_{i=r_1'}^{r_k'}t_i)\bmod p$
$q\leftarrow\mathrm{rand}(2^{\lambda_q})$	return (b_c,t_c)
while q is not prime done	ProofVerify:
return q	the client verifies b_c, recalls rand(seed) to produce $r_1'\rightarrow r_k'$
$p\mathrm{Gen}(q,\lambda_p)$	verify: $t_c\overset{?}{=}g^{b_c}\cdot r_1\cdot r_2\cdot\cdots\cdot r_k\bmod p$
for($i=1$ to $4\lambda_p$) do	if the equation holds, it indicates that the file is intact
$X\leftarrow\mathrm{rnd}(2^{\lambda_p})$	
$c\leftarrow X(\mathrm{mod}2p)$	
$p\leftarrow X-c+1//p\equiv 1(\mathrm{mod}2p)$	
If p is prime then return p	
done	
return 0	
$\mathrm{TagBlock}(G,\boldsymbol{F})$	
$G=(p,q,g)\boldsymbol{F}=(b_1,b_2,\cdots,b_n)$	
for($j=1$ to n) do	

图 1 基于同态哈希的数据完整性验证方案描述

Fig. 1 Description of data integrity verification method based on homomorphic hashing

3.2 对动态数据的支持

对动态数据的支持包括插入、删除 2 个基本操作。数据插入以数据块为最小单位进行操作,每次可以在不同的位置插入一个或多个数据块。假设需要插入的数据块为 b_s,客户端计算该数据块的标签,以及重新计算第 m 块以后的标签,再将更新以后的 $\boldsymbol{F}'$ 和 tag′上传到服务器,然后立即发出一次对该数据块的验证以确保数据的上传过程是正确的。在进行数据块的删除操作时,首先确定需要删除的数据块,用 b_t 表示,再用该文档对应的 seed 重新生成随机数序列,对删除位置之后的数据块重新取对应位置的随机数,计算标签,将删除的数据块的位置和更新后的标签传至服务器,删除服务器上相应的数据块,并计算更新该数据块之后的标签,具体过程如图 2 所示。

Algorithm:Insert	Algorithm:Delete
Assume that insert block b_s	Assume that delete block b_t
Client sends to server: Insert request	Client sends to server: Delete request
Client receive from server: $\boldsymbol{F}$、Tag	Client receive from server: $\boldsymbol{F}$、Tag
insert b_s	delete b_t
for(j=1 to n+1) do	for(j=1 to n−1) do
r_j=rand(seed)	r_j=rand(seed)
if($j \geqslant s$)	if($j \geqslant t$)
$x_j = h_G(b_j) = \prod_{i=1}^{m} g_i^{b_{i,j}} \bmod p$	$x_j = h_G(b_j) = \prod_{i=1}^{m} g_i^{b_{i,j}} \bmod p$
$t_j = x_j \cdot r_j$	$t_j = x_j \cdot r_j$
return Tag′=[t_1,t_2,t_3,…,t_{n+1}]	return Tag″=[t_1,t_2,t_3,…,t_{n-1}]
Client sends to server: $\boldsymbol{F}'$、Tag′	Client sends to server: $\boldsymbol{F}''$、Tag″

图 2　支持动态数据的描述

Fig. 2　Description of supporting dynamic data

4　安全性与性能分析

4.1　安全性分析

探测概率 P_{det} 是指客户端进行一次挑战验证过程中,服务器能够成功地对其中的一些数据块进行修改或删除的概率[5],可由式(3)描述:

$$P_{\text{det}} = \left(1 - \frac{d}{n}\right)^k \tag{3}$$

式中:d——被损坏或删除的数据块;k——1 次验证中包含的数据块数。若文档中的一些重要字节被删除或篡改了,可以使用纠错码处理这些微小变化,从而重新恢复原始文档,但纠错码对于含有大量数据块的文档是不适用的。

此方案中服务器端保存所有的数据本身、p、q、数据块及对应的标签。客户端保存生成标签使用的伪随机数种子 seed,大素数 p、q 以及生成元 g。该方案的安全性是基于可验证标签的,标签的构造使用了同态哈希算法和一系列伪随机数。

在 challenge 阶段,挑战者 challenger 随机生成 k 个挑战块发给 A,A 生成挑战块的完整性验证 P,若 P 通过了验证,则认为 A 完成了一次成功的欺骗。

服务器接收到挑战 b_{rj} 后,在于分析一个概率多项式时间算法 PPT(probabilistic ploynomial time)中是否能够在概率多项式时间内找到一对碰撞。若对于时间复杂度为 $\tau(\lambda)$ 的概率多项式时间敌手满足:

$$Pr[h_G(b_i) = h_G(b_i') \mid b_i \neq b_i'] < \varepsilon(\lambda)$$

式中 $\varepsilon(\lambda)$ 是一个足够小的数,则证明此算法是安全的。文献[11]给出了函数族 h_G 是一个碰撞抵抗哈希函数的证明过程。

假设 A 删除了 challenger 挑战的数据块,从而将任意的数据块及其对应的标签返回给 challenger。此时,虽然能够验证其返回的 b_c 与标签 t_c 是正确对应的,但由于 A 不知道构造标签使用的随机数 r_i,因此 challenger 只需要将收到的数据块进行同态哈希后,用与生成标签相同的种子生成伪随机数后重新构造标签,并与 A 返回的标签进行比较,就能验证 A 返回的数据块与标签是否是 challenger 指定的。

4.2　性能分析

在生成标签时取数据块大小 β 为 16 kB,同态哈希输出 λ_p 为 1 024 bit,因此此哈希函数将文件的存储空间减为原来的 λ_p/β=1/128,粗略判断这种 Tag 的组织方式对减小存储冗余的效果十分明显。该方案在支持动态数据的同时,还能够支持无限多次的挑战,表 2 是几种方案的比较(其中 DIVH 是本文的方案)。其中 t 表示挑战次数,s 表示每块中数据分片的个数,c 表示 CPDP 方案中混合云服务器的个数。

从表 2 可以看出,本文的方案在时间复杂度上与 PDP 方案是相同的。在支持动态数据的同时,本文方案能够支持无限多次挑战,因为挑战块是用户随机生成的。每次选取不同的数据块,服务器也无法猜出生成元 g 的准确数值,与 SPDP 方案中只能事先约定挑战的次数生成 token 相比有很大的改进。但在添加和删除

表2 方案比较

Table 2 Scheme comparison

方案	O(服务端)	O(客户端)	O(通信)	修改	插入	删除
PDP	$O(t)$	$O(t)$	$O(1)$			
SPDP	$O(t)$	$O(t)$	$O(t)$	✓		✓
DPDP	$(t\lg n)$	$O(t\lg n)$	$O(t\lg n)$	✓	✓	✓
CPDP	$O(t+s)$	$O(t+s)$	$O(cs)$	✓	✓	✓
IPDP	$O(t+s)$	$O(t+s)$	$O(s)$	✓	✓	✓
DIVH	$O(t)$	$O(t)$	$O(1)$	✓	✓	✓

操作方面来讲,与SPDP方案类似,也需要对所有涉及相应数据块的token取回进行更新。

5 结 语

在分析和总结前人方法的基础上,提出了一种基于同态哈希的数据完整性验证方案,并通过安全性与性能分析证明了此方案的可行性与有效性。在本文的基础之上,仍需进一步研究如何实现更少的冗余存储空间,如何使单次挑战覆盖更多的数据、更快的验证速度、占用更少的带宽以及对数据的可恢复性。

参考文献:

[1] JUELS A, KALISKI B S, POR S. Proof of rerievability for large files[C]//Proc of the 14th ACM Conf on Computer and Communications Security. New York:ACM,2007:584-597.

[2] ATENIESE G,BURNS R,CURTMOLA R,et al. Provable data possession at untrusted stores[C]//Proc of the 14th ACM Conf on Computer and Communications Security. New York:ACM,2007:598-609.

[3] JOHNSON R,MOLNAR D,SONG D,et al. Homomorphic signature schemes[C]//Proc of CT-RSA. New York:Springer,2002:244-262.

[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):1660-1668. (XIAO Da,SHU Jiwu,CHEN Kang,et al. A practical data possession checking scheme for networked archival storage[J]. Journal of Computer Research and Development,2009,46(10):1660-1668. (in Chinese))

[5] ATENIESE G,DI P R,MANCINI L V,et al. Scalable and efficient provable data possession[C]//Proc of the 4th International Confon Security and Privacy in Communication Netowrks (SecureComm 2008). New York:ACM,2008:1-10.

[6] ERWAY C,KUPCU A, PAPAMANTHOU C, et al. Dynamic provable data possession[C]//Proc of the 16th ACM Conf on Computer and Communications Security (CCS 2009). New York:ACM,2009:213-222.

[7] CHEN B, CURTMOLA R. Robust dynamic provable data possession [C]//Distributed Computing Systems Workshops (ICDCSW),32nd International Conference on. Macau:IEEE,2012:515-525.

[8] ZHU Y,WANG H,HU Z,et al. Cooperative provable data possession[R]//Beijing:Peking University and Arizona University,2010.

[9] 陈兰香. 一种基于同态hash的数据持有性验证方法[J]. 电子与信息学报,2011,33(9):2199-2204. (CHEN Lanxiang. A Homomorphic Hashing based provable data possession[J]. Journal of Electronics & Information Technology:2011,33(9):2199-2204. (in Chinese))

[10] KROHN M,FREEDMAN M J,MAZIERES D. On-the-fly verification of rateless erasure codes for efficient content distribution [C]//Procof IEEE Symposium on Security and Privacy. Lee Badger:IEEE,2004:226-240.

[11] BELLARE M,GOLDREICH O,GOLDWASSER S. Incremental cryptography:the case of hashing and signing[C]//Advances in Cryptology-CRYPTO'94. Berlin:Springer Berlin Heidelberg,1994:216-233.

[12] HAO Zhuo,ZHONG Sheng,YU Nenghai. A privacy-preserving remote data integrity checking protocol with data dynamics and pubic verifiability[J]. IEEE Transactions Oil Knowledge and Data Engineering,2011,23(9):1432-1437.

[13] WANG Qian,WANG Cong,REN Kui,et al. Enabling pubic auditability and data dynamics for storage security in cloud computing [J]. Transactions on Paralel and Distributed Systems,2011,22(5):847-859.

[14] 冯登国,张敏,张妍,等. 云计算安全研究[J]. 软件学报,2011,22(1):71-83. (FENG Dengguo,ZHANG Min,ZHANG Yan,et al. Study on cloud computing security[J]. Journal of Software,2011,22(1):71-83. (in Chinese))

[15] 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. (YU Yangyang,YU Huiqun,FAN Guisheng. An approach to verifying the data integrity of cloud storage[J]. Journal of East China University of Science and Technology:Natural Science Edition,2013,39(2):211-216. (in Chinese))