一种层次式远程数据持有检测方法

马海峰[1,2] 杨家海[2] 姚念民[3] 关明山[1]

(黑龙江科技大学计算机与信息工程学院 哈尔滨 150022)[1]

(清华大学网络科学与网络空间研究院 北京 100084)[2]

(大连理工大学计算机科学与技术学院 大连 116024)[3]

摘 要 在云存储环境下,云服务器并不完全可信。用户如何以较低开销验证云上数据的完整性成为用户日益关心的问题。目前已提出多种保护方法,这些方法在认证多个文件时需要对文件逐一进行认证,因此当文件数很大时其计算和通信开销仍较大。针对此问题,提出一种层次式远程数据持有检测方法。该方法与远程数据持有检测方法相结合,能提供高效且安全的远程数据完整性保护,并支持动态数据操作。对提出的方法进行了安全性分析和实验评估,结果表明,提出的方法安全可靠,在较低的漏检率下,相比远程数据持有检测方法有45%~80%的性能提升。

关键词 云存储,数据完整性,数据持有证明,同态标签,哈希树

中图法分类号 TP309 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.03.014

Hierarchical Remote Data Possession Checking Method

MA Hai-feng[1,2] YANG Jia-hai[2] YAO Nian-min[3] GUAN Ming-shan[1]

(Institute for Computer and Information Engineering, Heilongjiang University of Science and Technology, Harbin 150022, China)[1]

(Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing 100084, China)[2]

(Institute for Computer Sciences and Technology, Dalian University of Technology, Dalian 116024, China)[3]

Abstract In cloud storage environment, the storage servers may not be fully trustworthy. How to verify the integrity of the cloud data with a lower overhead for users has become increasingly concerned problem. While many methods have been proposed, these methods in the verification of multiple files need to authenticate them one by one. So the computation and communication overhead is still expensive when the number of file is large. Aiming at this problem, a hierarchical remote data possession checking method was proposed. Combining with remote data possession checking method, the proposed method can provide more efficient and secure remote data integrity protection, and support dynamic data operation. The security analysis and experimental evaluation results show that the proposed method is safe and reliable, and it can gain 45%~80% performance improvement with lower false negative rate.

Keywords Cloud storage, Data integrity, Provable data possession, Homomorphism tag, Hash tree

1 引言

目前越来越多的企业和个人用户希望将应用和数据迁移到云上,但云存储环境下的云服务器并不完全可信,云存储安全问题是进一步推广云存储服务的主要障碍。云存储安全主要包括数据完整性、机密性、可靠性和隐私保护等几方面。云存储数据完整性检测是检验云存储服务器上的用户数据是否完好,避免云用户存储在其中的数据被篡改或删除。

目前云存储数据完整性研究主要集中在两个方面:数据持有性证明(Provable Data Possession, PDP)[1]和可恢复性证明(Proof Of Retrievability, POR)[2]。其基本思想都是利用某种形式的挑战-应答协议,并通过基于伪随机抽样的概率性检查方法降低通信和计算的开销。PDP方法是通过挑战-应答协议向用户证明其文件是完好无损的,它可以检测到大于某个比例的数据损坏,但不保证整个文件是可取回的。POR方法是将伪随机抽样和冗余编码(如纠错码)结合,同样是通过挑战-应答协议向用户证明其文件是完好无损的,而且用户能够以足够大的概率从服务器取回文件。

PDP仅支持静态类型数据的远程存储安全保护,不能对数据进行动态操作。Ateniese等人提出基于公钥加密的支持文件动态更新的PDP方案[3],该机制能够提供数据信息文件分块的更新、删除等基本操作,但是不支持对数据信息文件分

到稿日期:2016-01-05 返修日期:2016-05-11 本文受黑龙江省教育厅科学技术研究项目(12533052)资助。

马海峰(1977-),男,博士,副教授,主要研究方向为存储安全、性能分析,E-mail:mhf_2000@163.com;杨家海(1966-),男,博士,教授,主要研究方向为计算机网络;姚念民(1974-),男,博士,教授,主要研究方向为网络存储、无线传感器网络;关明山(1974-),男,硕士,讲师,主要研究方向为存储安全。

块的插入操作。Erway 等人[4]分别利用跳跃表(Skip List)和 RSA 树构建了基于秩次的认证字典,以此提出了支持全动态更新的 PDP 方案。

POR 机制不支持公开验证,且只能进行有限次验证。Shacham 等人[5]利用 Ateniese 的同态验证标签的思想,并运用 BLS 短签名构造了"同态认证元"(Homomorphic Authenticator)[6],有效地降低了检验过程的通信开销。该方案支持无限次数的挑战,但是存储认证元的开销较大。Bowers 等结合 Juels 和 Shacham 的研究工作,给出了 POR 协议的改进版本[7]。Wang 等人[8]在基本 BLS 签名方案中加入了随机数,从而保证挑战-应答过程中文件数据的隐私性,但其无法支持插入操作。陈兰香等人[9]引入远程数据持有检测(Remote Data Possession Checking,RDPC)概念。

2 远程数据持有检测方法

RDPC 是包括 PDP 和 POR 的全部远程数据完整性保护方案。本文提出的方法是与 RDPC 相结合的,下面对 RDPC 的相关内容进行简要说明。

2.1 相关概念

同态标签是基于同态生成的标签。基于同态的特性,同态标签可以使用较少的数据对较多的数据进行检测,这正符合远程数据完整性检查中计算和通信开销需较低的要求,因此可用来验证数据是否完整。同态标签具有如下性质:对任意两个数据块 m_i 和 m_j,它们和的标签信息等于各自标签信息的乘积,即 $T(m_i+m_j)=T(m_i)\times T(m_j)$。

Hash 树也称为 Merkle Hash Tree(MHT),是在数据认证中广泛使用的一种认证结构,能高效地认证一个元素是否被篡改。它一般被构建为一棵二叉树,它的叶结点是要认证数据的 hash 值。文中 hash 树被用来代替传统文件索引信息,同时用来认证数据和数据块的位置,单独的某一数据块的操作不影响其他数据块,解决了文件索引动态操作开销大的问题。

2.2 基本 RDPC 协议

基于基本 RDPC 协议,先定义若干函数和参数。$H(\cdot)$ 是同态 hash 函数;$h(\cdot)$是用来计算 hash 树的加密 hash 函数;主要参数有:λ_p,λ_q 为安全参数,p,q 为随机大素数,且满足 $p=\lambda_p$,$|q|=\lambda_q$,$q|(p-1)$,β 为块大小,m 为每个块的子块数,且 $m=\beta/(\lambda_q-1)$,g 为 $1\times m$ 行向量,K 是 hash 密钥,由 (p,q,g)组成,k 是私钥长度,s 是随机种子,F 为原始文件(也被看成是多个块的集合),R 是 hash 树的根,$sig_{sk}(h(R))$是树的根签名。

RDPC 协议由 5 个多项式时间算法组成:

1)$PublicKeyGen(1^k)\rightarrow(pk,sk)$,由客户端运行,用于产生公钥,它以安全参数 k 作为输入,输出公钥、私钥对(pk,sk);

2)$HomomorphicKeyGen(\lambda_p,\lambda_q,m,s)\rightarrow(p,q,g)$,由客户端运行,它以安全参数 λ_p,λ_q,m,s 作为输入,输出同态密钥(p,q,g);

3)$TagGen(sk,K,F)\rightarrow(T,sig_{sk}(h(R)))$,由客户端运行,以 sk、同态 hash 密钥 K 和文件 F 作为输入,输出文件标签集合 T 和 $sig_{sk}(h(R))$;

4)$ProofGen(F,T,chal)\rightarrow V$,由云服务器端运行,输入文件 F、标签 T 和挑战 $chal$,输出一个数据完整性证明 V;

5)$ProofCheck(pk,chal,V)\rightarrow\{TRUE,FALSE\}$,由客户端在接到证明 V 后运行,输入公钥 pk、挑战 $chal$ 和 V,如果文件认证成功则输出 TRUE,否则输出 FALSE。

3 层次式远程数据持有检测方法

与 RDPC 相结合,本文提出一种层次式远程数据持有检测(Hierarchical Remote Data Possession Shecking,H-RDPC)方法。

3.1 基本思想

H-RDPC 的基本思想是在对云上大量连续文件进行认证时,基于挑战应答模式进行分层次的认证,基本过程是:先由访问粒度和检测文件数确定第一层认证的文件,开始进行粗粒度的挑战应答式认证,即对较大间隔下的若干文件进行认证,若认证成功,则认为其邻近区域的文件均完整,不再认证;若认证失败时,则对其邻近的文件以较细粒度进行认证;当再次认证失败时,对所有认证失败文件附近的文件以更细粒度进行认证,以此类推,直到检测粒度最小为止。

3.2 算法描述

H-RDPC 方法包括建立阶段(Setup)和挑战阶段(Challenge),前者为文件生成认证信息,后者检测云上文件。下面对 H-RDPC 进行算法描述。

(1)建立阶段

客户端(Client)调用 $PublicKeyGen()$产生公钥和私钥,调用 $HomomorphicKeyGen()$产生同态 hash 密钥;接着客户端调用 $TagGen()$为文件计算标签 Tag,以标签为叶结点构建 hash 树,并对根结点 R 进行签名。对每个文件做上述操作,将所有文件及 hash 树等认证信息发送到云上。

(2)挑战阶段

客户端先确定要认证的文件数 N 和初始检测粒度 x,接着产生随机密钥 e 和要挑战的文件块数 c,并将挑战 $chal=\langle e,c,N,x\rangle$发送到云上。云服务器对第 $k\cdot x$ 个文件依次执行 $ProofGen(F',T,chal)\rightarrow V$,其中 $1\leqslant k\leqslant\lfloor N/x\rfloor$,即计算出数据块的和 B 以及标签的乘积 T;然后云服务器计算较少量的校验信息 ω,它是从 hash 树叶结点到根的兄弟结点。云服务器将 B,ω,T 和签名的 R 等信息作为证明发到客户端。客户端在收到证明后调用 $ProofCheck()$验证文件的完整性,即用 ω 生成新的根结点 R',认证 R 与 R'是否匹配,若失败则返回 FALSE;否则对 B 计算同态标签 $H_K(B)$,验证与 T 是否匹配,如不匹配则返回 FALSE。对于认证结果为 FALSE 文件,对其邻近的$\lfloor x/2\rfloor$处的文件发起类似的第二层挑战-应答式认证,以此类推,直到检测粒度最小。最后返回 N 个文件的认证结果。

3.3 动态数据更新操作

H-RDPC 完全支持动态更新操作,包括对云上数据的修改、插入和删除。设文件 F 和相关信息已被保存在服务器

上，下面以数据修改为例进行说明。假定客户端想修改第 i 个数据块 b_i 为 b_i^*，修改过程如下：

1）客户端基于 b_i^* 产生标签 T_i^*，接着构建更新请求并将其发送到服务器上；

2）云服务器接受请求后，执行更新操作：用 b_i^* 取代 b_i，同时更新 F^*，用 T_i^* 取代 T_i，得到辅助认证信息 ω_i^*，在 hash 树中用 b_i^* 的 hash 值 $H_K(b_i^*)$ 取代 b_i 的 hash 值 $H_K(b_i)$，重新计算出 hash 树新的根结点 R^*，并向客户端发送证明 $V_{update}=\{H_K(b_i^*),\omega_i^*,R^*\}$；

3）客户端用 $\{H_K(b_i^*),\omega_i^*\}$ 计算出 R，检测 R 与 R' 是否相等，如不相等，则返回 FALSE，否则产生新的根签名 $sig_{sk}(h(R)*)$ 并发送到云上更新。

数据插入和删除与修改过程类似，限于篇幅，在此略去。

4　安全性分析

为了验证本方法的安全性，在数据所有者和攻击敌手之间构建一个数据持有性游戏，如果敌手赢得此游戏，则表示敌手能正确地持有所有密文数据块和标签信息。

定理 1　在大整数因式分解困难性问题的假设下，本文提出的完整性检测方法对检测到的文件是安全的。

证明：首先进行如下游戏描述。

1）生成密钥：云用户调用 *PublicKeyGen*() 生成密钥，并将公钥发送给敌手；

2）询问：敌手选择一个数据块 $m_i(1\leqslant i\leqslant n)$ 发送给云用户，云用户调用 *TagGen*()，计算收到数据块的标签信息 T_i，再将所有标签信息发送给敌手；

3）挑战：云用户发起挑战 *chal*，要求敌手根据挑战信息计算出数据持有证明；

4）伪造：敌手根据挑战信息 *chal*、数据块 m_i 以及标签信息 T_i 生成数据持有证明 V 并发送到云用户；

5）认证：云用户调用 *ProofCheck*() 检测敌手返回的证明 V 是否正确，若正确则敌手赢得本次游戏。

在游戏中，云用户根据敌手返回的数据持有证明 V 来计算验证信息，当敌手正确持有全部数据块和标签时，必有 $H_K(B)=T$，即 $T_i=H_K(m_i)$。敌手篡改或删除数据块 m_i 或标签信息 T_i 时，必须伪造出适合的 m_i' 和 T_i'，使得 $T_i'=H_K(m_i')$，$H_K(B)=T$ 成立，即敌手有能力构造出两个随机大素数 p 和 q，使得 $p\neq q$，且 $g^p=g^q \bmod N$，因此 $p-q=k\varphi(N)$ 成立，从而 $p-q$ 可以用来分解大整数 N。而在大整数因式分解困难的假设下，除敌手非正确持有全部的密文数据块和标签信息，否则不可能赢得此数据持有性游戏，证明结束。

5　算法性能评价

为验证和评价 H-RDPC 算法的有效性，进行以下的实验：实验运行在一台 Intel i5 CPU 2.5，4G 内存的 PC 机上，在 Ubuntu12.04 环境下基于 PBC library 实现了需要的加密和 hash 函数，PBC 版本为 0.5.11。选择 RDPC 作为比较对象，基于 PBC 实现了 H-RDPC 和 RDPC，在评估方案时考虑了通讯和计算代价。

实验 1　不同文件数下的性能评价

设检测的多个文件中有 10%的连续文件损坏，文件大小为 10MB，块大小为 200kB，基于 RDPC 和粒度为 2，5，10 下的 H-RDPC 方法，分别对 100，200，300，400 和 500 个文件进行完整性认证，结果如图 1 所示。由图 1 可知，H-RDPC 的性能明显优于 RDPC，在此实验环境下，当文件数改变且检测粒度为 2，5 和 10 时，H-RDPC 相比 RDPC 平均分别有 80%，71.8%和 45%的性能提升。

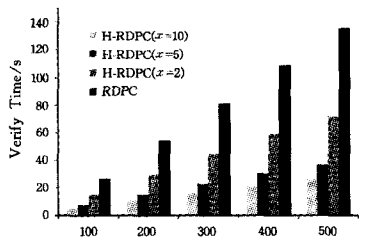

图 1　不同文件数下 H-RDPC 与 RDPC 的性能比较

实验 2　不同损坏文件比例下的性能评价

设要检测 300 个文件，文件大小为 10MB，块大小为 200kB，检测粒度为 5，损坏的文件比例分别为 1%，5%，10%，15%和 20%，RDPC 与 H-RDPC 的性能比较结果如图 2 所示。由图可知，当损坏的文件比例逐渐提高时，因 RDPC 是逐个文件比较，开销较大且固定，而 H-RDPC 认证开销随着比例提高而缓慢增加。在此实验环境下，当损坏的文件比例从 1%增加到 20%，H-RDPC 相比 RDPC 平均有 72%的性能提升。

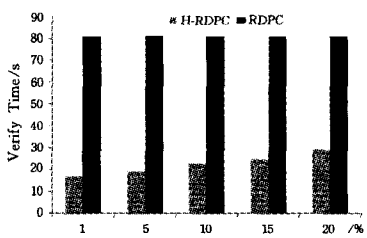

图 2　不同损坏文件比例下 H-RDPC 与 RDPC 的性能比较

结束语　本文提出一种层次式远程数据持有检测方法 H-RDPC，它能提供数据持有证明和数据完整性保护，能支持动态数据操作和无限次认证。本文对 H-RDPC 的安全性进行了分析，并进行了实验评估，评估结果表明，相比 RDPC，H-RDPC 有 45%～80%的性能提升，并保持较低的漏报率，即 H-RDPC是一种可行的远程数据持有检测方法。

参 考 文 献

[1] ATENIESE G, BUMS R, CURTMOLA R, et al. Provable data possession at untrusted stores[C]//14th ACM CCS. 2007: 598-609.

[2] JUELS A, BURTON S, KALISKI J. PORs: Proofs of Retrievability for Large Files [C]// Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS'07). Alexandria, Virginia, USA, USA: ACM Press, New York, NY, 584-597.

[3] ATENIESE G, DI PIETRO R, MANCINI L V, et al. Scalable

and efficient provable data possession[C]// 4th International Conference on Security and Privacy in Communication Networks. 2008.

[4] ERWAY C, KUPCU A, PAPAMANTHOU C, et al. Dynamic provable data possession[C]//16th ACM CCS. 2009:213-222.

[5] SHACHAM H, WATERS B. Compact Proofs of Retrievability [C]// Proceedings of the 14th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT'08). Melbourne, Australia, Berlin: Springer-Verlag, 90-107.

[6] TAN S, JIA Y, HAN W H. Research and Development of Provable Data Integrity in Cloud Storage [J]. Chinese Journal of Computers, 2015, 38(1): 164-177. (in Chinese)
谭霜,贾焰,韩伟红. 云存储中的数据完整性证明研究及进展[J]. 计算机学报,2015,38(1):164-177.

[7] BOWERS K D, JUELS A, OPREA A. Proofs of Retrievability: Theory and Implementation[C]// Proceeding(s) of ACM Workshop on Cloud Computing Security. Chicago, USA, 2009: 43-53.

[8] WANG C, WANG Q, REN K, et al. Privacy-preserving public auditing for data storage security in cloud computing[C]// 29th IEEE INFOCOM. 2010.

[9] CHEN L, ZHOU S, HUANG X, et al. Data dynamics for remote data possession checking in cloud storage[J]. Computers and Electrical Engineering, 2013, 39: 2413-2424.

(上接第 26 页)

聚类的准确率。实验结果表明,MRGAK-Medoids 算法在处理大数据时具有接近线性的加速比,同时聚类质量得到提升。

接下来的工作是如何改进遗传算子使得遗传操作可以快速收敛、如何降低 Reduce 的任务负载等问题。

参 考 文 献

[1] HAN J, KAMBER M. 数据挖掘:概念与技术(第 2 版)[M]. 范明,孟小峰,译. 北京:机械工业出版社,2007.

[2] LAI Y X, LIU J P, YANG G X. K-Means Clustering Analysis Based on Genetic Algorithm[J]. Computer Engineering, 2008, 34(20): 200-202. (in Chinese)
赖玉霞,刘建平,杨国兴. 基于遗传算法的 K 均值聚类分析[J]. 计算机工程,2008,34(20):200-202.

[3] TANG Z X. K-means Clustering Algorithm Based on Improved Genetic Algorithm[J]. Journal of Chengdu University (Nature Science Edition), 2011, 30(2): 162-164. (in Chinese)
唐朝霞. 一种改进的基于遗传算法的 K 均值聚类算法[J]. 成都大学学报(自然科学版),2011,30(2):162-164.

[4] LI J B, YANG L, HUA B. Research on parallel K-Medoids algorithm on multi-core platform[J]. Application Research of Computers, 2011, 28(2): 498-500. (in Chinese)
李静滨,杨柳,华蓓. 基于多核平台并行 K-Medoids 算法研究[J]. 计算机应用研究,2011,28(2):498-500.

[5] LI J B, YANG L, CHEN N J. An improved parallel K-Medoids algorithm based on MapReduce[J]. Journal of Guangxi University (Nature Science Edition), 2014(2): 341-345. (in Chinese)
李静滨,杨柳,陈宁江. 基于 MapReduce 的改进 K-Medoids 并行算法[J]. 广西大学学报(自然科学版),2014(2):341-345.

[6] ZHANG X P, GONG K L, ZHAO G C. Parallel K-Medoids algorithm based on MapReduce[J]. Journal of Computer Application, 2013, 33(4): 1023-1025. (in Chinese)
张雪萍,龚康莉,赵广才. 基于 MapReduce 的 K-Medoids 并行算法[J]. 计算机应用,2013,33(4):1023-1025.

[7] JIANG Y B, ZHANG J M. Parallel K-Medoids Clustering Algorithm Based on Hadoop[C]// IEEE Beijing Section. Proceedings of 2014 IEEE 5th International Conference on Software Engineering and Service Science. IEEE Beijing Section, 2014: 4.

[8] ZHU Y, WANG F, SHAN X, et al. K-medoids clustering based on MapReduce and optimal search of medoids[C]// 2014 9th International Conference on Computer Science & Education (ICCSE). IEEE, 2014: 573-577.

[9] SRINIVASULU D L, REDDY A V, AKULA V S G, et al. Improving the Scalability and Efficiency of K-Medoids By Map Reduce [J]. International Journal of Engineering and Applied Sciences (IJEAS), 2015(4): 88-90.

[10] ALBA E, TROYA J M. A survey of parallel distributed genetic algorithms [J]. Complexity, 1999, 4(4): 31-52.

[11] GUO T C, MU C D. The Parallel Drifts of Genetic Algorithms [J]. System Engineering-Theory & Practice, 2002, 22(2): 15-23, 41. (in Chinese)
郭彤城,慕春棣,并行遗传算法的新进展[J]. 系统工程理论与实践,2002,22(2):15-23,41.

[12] WANG X L, LI Q. Application and Research on Parallel Genetic Algorithm[J]. Microcomputer Information, 2007, 23(9): 205-206. (in Chinese)
王小良,李强. 并行遗传算法研究及其应用[J]. 微计算机信息,2007,23(9):205-206.

[13] JOHAR F M, AZMIN F A, SUAIDI M K, et al. A review of genetic algorithms and parallel genetic algorithms on Graphics Processing Unit (GPU)[C]// 2013 IEEE International Conference on Control System, Computing and Engineering (ICCSCE). IEEE, 2013: 264-269.

[14] FERRUCCI F, SALZA P, KECHADI M T, et al. A Parallel Genetic Algorithms Framework based on Hadoop MapReduce[C]// The ACM Symposium. ACM, 2015: 1664-1667.

[15] Jin C, Vecchiola C, Buyya R. Mrpga: an extension of mapreduce for parallelizing genetic algorithms[C]// IEEE Fourth International Conference on EScience, 2008 EScience'08. IEEE, 2008: 214-221.

[16] FERRUCCI F, KECHADI M, SALZA P, et al. A Framework for Genetic Algorithms Based on Hadoop[J]. arXiv preprint arXiv: 1312.0086, 2013.

[17] ZHANG X, WANG J, WU F, et al. Genetic K-Medoids Spatial Clustering with Obstacles Constraints[C]// 2006 3rd International IEEE Conference on Intelligent Systems. IEEE, 2006: 826-831.

[18] SHENG W, LIU X. A genetic k-medoids clustering algorithm [J]. Journal of Heuristics, 2006, 12(6): 447-466.

[19] DEAN J, GHEMAWAT S. MapReduce: Simplified Data Processing on Large Clusters[C]// Conference on Symposium on Operting Systems Design & Implementation. 2004: 107-113.