软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2016,27(6):1417−1431 [doi: 10.13328/j.cnki.jos.004995]
©中国科学院软件研究所版权所有.
E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

云存储中支持数据去重的群组数据持有性证明*

王宏远[1,2], 祝烈煌[1,2], 李龙一佳[1,2]

[1](北京理工大学 计算机学院,北京 100081)

[2](北京市海量语言信息处理与云计算应用工程技术研究中心, 北京, 100081)

通讯作者: 祝烈煌, E-mail: liehuangz@bit.edu.cn

摘 要: 数据持有性证明(provable data possession,简称 PDP)和数据可恢复性证明(proofs of retrievability,简称 POR)是客户端用来验证存储在云端服务器上数据完整性的主要技术.近几年,它在学术界和工业界的应用广泛,很多 PDP 和 POR 方案相继出现.但是由于不同群组的特殊性和独特要求,使得群组 PDP/POR 方案多样化,并且群组应用中的许多重要功能(例如数据去重)没有被实现.如何构造高效及满足群组特定功能和安全需求的 PDP/POR 方案,已经引起了人们的广泛关注.给出了一种支持数据去重的群组 PDP 方案(GPDP),基于矩阵计算和伪随机函数,GPDP 可以在支持数据去重的基础上,高效地完成数据持有性证明,并且可以在群组中抵抗恶意方选择成员攻击.在标准模型下证明了 GPDP 的安全性,并且在百度云平台上实现了 GPDP 的原型系统.为了评估方案的性能,使用了 10GB 的数据量进行实验和分析,结果表明:GPDP 方案在达到群组中数据去重的目标的基础上,可以高效地保证抵抗选择攻击和数据持有性,即:预处理效率高于私有验证方案,而验证效率高于公开验证方案(与私有验证效率几乎相同).另外,与其他群组 PDP/POR 方案相比,GPDP 方案将额外存储代价和通信代价都降到了最低.

关键词: 群组数据持有性证明;选择攻击;数据去重;云存储;云计算

中图法分类号: TP311

中文引用格式: 王宏远,祝烈煌,李龙一佳.云存储中支持数据去重的群组数据持有性证明.软件学报,2016,27(6):1417−1431. http://www.jos.org.cn/1000-9825/4995.htm

英文引用格式: Wang HY, Zhu LH, Lilong YJ. Group provable data possession with deduplication in cloud storage. Ruan Jian Xue Bao/Journal of Software, 2016,27(6):1417−1431 (in Chinese). http://www.jos.org.cn/1000-9825/4995.htm

Group Provable Data Possession with Deduplication in Cloud Storage

WANG Hong-Yuan[1,2], ZHU Lie-Huang[1,2], LI Long-Yi-Jia[1,2]

[1](School of Computer, Being Institute of Technology, Beijing 100081, China)

[2](Beijing Engineering Research Center of Massive Language Information Processing and Cloud Computing Application)

Abstract: Provable data possession (PDP) and proofs of retrievability (POR) are techniques for a client to verify the integrity of outsourced data in cloud storage. Recently, numerous PDP and POR schemes have been proposed while the techniques are widely used in academic and industrial community. However, due to specific and unique requirements of different groups, PDP/POR schemes vary and many functionalities such as data deduplication have not been implemented. How to construct an efficient group PDP/POR scheme to meet these unique requirements of functionality and security has received much attention. In this paper, a group PDP with deduplication

* 基金项目: 国家自然科学基金(61272512, 61100172); 国家高技术研究发展计划(863)(2013AA01A214); 教育部新世纪优秀人才计划(NCET-12-0046); 北京市自然科学基金(4121001)

Foundation item: National Natural Science Foundation of China (61272512, 61100172); National High-Tech R&D Program of China (863) (2013AA01A214); Program for New Century Excellent Talents in University (NCET-12-0046); Natural Science Foundation of Beijing (4121001)

收稿时间: 2015-08-13; 修改时间: 2015-10-09; 采用时间: 2015-12-05; jos 在线出版时间: 2016-01-21

CNKI 网络优先出版: 2016-01-22 11:01:39, http://www.cnki.net/kcms/detail/11.2560.TP.20160122.1101.011.html

(GPDP) is presented. Based on matrix calculation and pseudo-random function, GPDP can efficiently guarantee data possession with deduplication, as well as defend against selective opening attacks of a malicious party.The security of GPDP in the standard model is proved and a prototype based on GPDP scheme in a realistic cloud platform of Baidu is implemented. To evaluate the performance of GPDP, this work utilizes data size of 10GB for experiments and analysis. The result of experiments show that GPDP can guarantee data possession efficiently with deduplication and protect against selective opening attacks. In particular, the performance is superior to private schemes in the phase of pre-process and public schemes in the phase of verification (as efficient as private scheme in the phase of verification). Furthermore, GPDP reduces the extra storage and communication cost to a minimum than the other PDP/POR schemes applied in a group.

Key words: group provable data possession; selective opening attack; data deduplication; cloud storage; cloud computing

云计算作为新一代计算模式已经被广泛应用于不同领域的实践场景中,它为用户带来了低廉的运维成本、按需可扩展的性能配置以及更高效的计算能力.云存储基于云计算将网络中不同类型的存储设备进行整合,对外提供数据存储和业务访问功能.目前,随着全球数据量爆炸式增长,数据分析的价值对社会和商业智能产生了巨大的影响,越来越多的用户应用云存储服务对海量数据进行存储管理.

然而,这种数据外包的存储模式存在着一系列安全问题[1,2].近几年,云服务提供商暴露出的数据丢失问题引起了人们的广泛关注.例如:2011 年,阿里云(http://www.ctocio.com.cn/cloud/401/12178901.shtml)服务器磁盘错误,在维护过程中执行了重启操作,导致期间的数据丢失;2012 年,谷歌 Gmail 邮箱(http://cio.zol.com.cn/228/2281017.html)也爆发大规模数据丢失的问题,大约 15 万谷歌邮箱用户数据被删除.可见,云服务事故导致的数据缺失有可能对用户造成不可估量的损失,甚至给一个企业带来灭顶之灾.而委托给云存储服务的海量数据由于规模庞大,验证其完整性也存在着很多难点.因此,近年来,云存储中数据的完整性验证成为了广受关注的研究热点.

为了解决这个问题,很多不同的 PDP(provable data possession,数据持有性证明)[3−5]和 POR(proofs of retrievability,数据可恢复性证明)[6−8]方案被提出.然而,在实际情况中可能出现这样一个问题:对于某些公司(比如互联网公司)来说,每天数据呈爆炸式增长,比如购买记录,一旦数据损坏或丢失将对公司造成巨大损失.因此需要将数据存储在外部的云服务器上,并且有专业人员定期进行完整性和正确性的检查,我们将检测人员抽象为某个群组中的成员.如果维护数据的成员数量很少,当他们被恶意收买或者密钥被攻破或丢失时,恶意方就可以伪造验证标签,在云端数据被删除或者被修改的情况下,也能通过完整性验证,因此,数据的安全性很容易受到威胁,进而对公司造成重大损失.例如,携程网(http://news.sohu.com/20150528/n413987338.shtml)疑似遭受内部员工删除数据,致使公司损失惨重.

为了增强外包存储数据的安全性,可以增加检测成员以及相应密钥的数量,使用多个密钥对同一份数据分别进行验证,这样可以有效避免恶意方对群组中某些成员进行选择攻击(或者群组中某些成员与恶意方合谋).但由于现阶段的数据持有性技术的限制,N 个密钥就会对应 N 份验证标签,当增加密钥数量的时候,验证标签也会呈爆炸式增长.例如最著名的 S-PDP 方案[9,10]中,验证标签的数据量约为原始文件总数据量的 4%,如果将其应用于群组中,在增加 100 个验证密钥的情况下,标签数据量就是原始文件数据量的 4 倍,这种存储代价是非常大的.据我们的调研判断,目前没有一个 PDP/POR 方案可以满足群组中使用多个密钥和一份验证标签就能分别完成持有性验证,即,同时支持数据去重和抵抗选择攻击的 PDP/POR 方案目前是不存在的.

综上所述,我们需要一个群组数据持有性证明方案满足如下要求:

- 群组中的数据持有性证明:在群组中,成员可以使用各自的密钥对共享的数据进行持有性检验,如果数据被修改或者丢失,则可以被至少一个群组成员及时检测出来;
- 支持共享数据去重:群组中,成员共享同一份文件,在每个成员的密钥不相同的情况下,可以使用同一份标签值分别进行验证.区别于之前的方案中,每个密钥对应于一份标签值的情况.支持标签值数据去重的方案,可以减少存储空间,尤其在群组成员的数量比较多的情况下,存储空间减少的优势更加明显;
- 抵抗选择攻击:由于在群组中多个拥有不同密钥的成员对应于同一份验证标签,为了安全起见,要求当群组中部分成员被攻击密钥泄露的情况下(相当于部分成员与敌手合谋),敌手(可能为云服务器、恶意

第三方或二者共谋的情况)不能根据这些信息伪造出合法的标签值,并通过群组中其他合法成员的持有性验证;

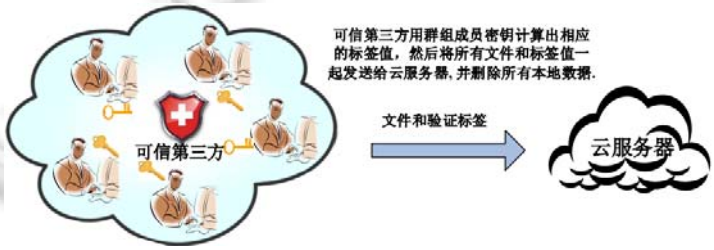
- 高效性和实用性:在满足上述条件的情况下,要求方案可以在实际的云平台上实现且高效运行,即,尽可能减少存储代价、计算代价和通信代价.

在本文中,我们给出了一个可以同时保证数据去重和抵抗选择攻击的群组 PDP 方案(GPDP),并且在标准模型下证明了其安全性.根据 GPDP 方案,我们实现了一个 GPDP 原型系统.为了测试和评估其性能,我们将 GPDP 原型系统移植到现实的云服务器(百度云)中,并且使用了 10GB 的数据量进行实验.实验结果表明:GPDP 系统在保持高效快速的数据持有性证明的情况下,有效地做到了数据去重和抵抗选择成员攻击.

本文第 1 节简要介绍 GPDP 的模型.第 2 节中依时间顺序简单罗列相关工作.第 3 节给出构造方案使用的基础知识.第 4 节详细描述 GPDP 的定义和构造方法.然后,分别在第 5 节和第 6 节中给出安全性证明和实验分析.最后,第 7 节为总结和未来工作.

1 模型概述

根据交叉认证码和同态消息认证码可以构造一个高效的 GPDP 方案,并且可以在群组中同时达到数据去重和抵抗选择成员攻击的目的.一个 GPDP 方案分为两个阶段:预处理阶段和挑战验证阶段,如图 1 所示.



(a) GPDP 方案的预处理过程

(b) GPDP 方案的挑战和验证过程

Fig.1 The processes of GPDP scheme

图 1 GPDP 方案过程概述

在预处理过程中,可信第三方(the trusted party,简称 TTP)生成一系列密钥(密钥数量大于或等于群组成员个数),TTP 使用全部密钥和文件数据计算出相应的标签值,然后把文件集合和标签集合都发送给云服务器,把密钥分发给群组成员,并且将本地数据全部删除.注意,方案中假设 TTP 是安全的,即:它可以完全按照规定执行协议;正确计算数据并输出;在预处理结束后,严格按要求删除本地数据.

在挑战、证明和验证过程中,群组成员(一个或多个)向云服务器发送挑战,服务器根据某个成员 U 的挑战选择相应的数据块,产生数据持有性证明并发送给成员 U;然后,U 使用自己的密钥对证明进行验证.值得注意的是:每个成员都可以根据自己的时间和需求发送挑战;进行数据持有性的验证;并且每个成员的密钥和验证过程

都是相互独立的,从而多重保证了数据的安全性.

GPDP 方案的定义和构造将在第 4 节中详细给出.

2 相关工作

数据持有性证明(provable data possession,简称 PDP)的概念是由 Ateniese 等人[9]在 2007 年的 CCS 上首次提出来的.他们利用 RSA 方案来构造同态验证标签,并且使用了抽样的方法来避免客户端检索和下载整个文档,从而提高产生证明和验证证明的效率.同时,他们还给出一个支持公开验证方案,但是效率较低并且通信代价较大.同年,Juels 和 Kaliski[11]提出了数据可恢复性证明(proofs of retrievability,简称 POR)的概念.与 PDP 类似,POR 也是用来检查服务器端数据完整性和正确性的.它是将原始数据进行一定的编码(例如纠删纠错码),并且在数据中插入岗哨(用于检测的随机值).在证明和验证阶段,客户端向服务器发送一部分岗哨的位置信息;然后,服务器返回挑战位置的岗哨值;最后,客户端进行数据的比对验证.它的缺点非常明显,因为岗哨的数量是固定的,所以客户端只能进行有限次数的挑战和验证.2008 年,Shacham 和 Waters[12]提出了支持公开验证和私有验证的 POR 机制,用来解决 Juels 等人方案[11]的不足.借鉴 Ateniese 方案[9]的思想,CPOR 利用同态验证标签将证明压缩为一个较小的值,从而减小通信代价.同时,Ateniese 等人[13]构造了一个完全基于对称密码学的高效且安全可证的 PDP 方案,并且可以高效地支持数据的动态扩展(修改、删除、添加等操作),但缺点是只能进行有限次验证.2009 年,Dodis 等人[14]形式化证明了 Juels 和 Kaliski 的一个变形方案的安全性,进而构造了第一个不需要随机查询机并且可以无限次使用的 POR 方案.但这是一个理论的方案,并没有被实现.Ateniese 等人[15]给出了一个通用框架:从任何一个满足同态特性的身份认证协议构造出一个基于公钥密码学的同态线性认证协议(homomorphic linear authenticator,简称 HLA),因此,他们基于大数因数分解困难问题得到了第一个随机查询机模型下的可以无限次数使用的 POS(Proofs of Storage)协议.同年,Erway 等人[16]根据 Ateniese 的 S-PDP 模型提出一种可以支持存储数据更新的动态 PDP 模型(dynamic PDP,简称 DPDP).为了同时满足动态和静态存储的需求,Chen 等人[17]根据点检测法提出了高效的 RDC 模型.Zheng 等人[18]给出了一个新的概念:公平的动态 POR(fair and dynamic POR,简称 FDPOR),它可以保证服务器在动态存储中不能对用户的数据进行非法操作.FDPOR 是根据增量签名方案和基于范围的 2-3 树构造的,并且在其方案中第一次提到了不诚实的客户端,但是方案并没有有效地解决这个问题.随后,Zhu 等人[19,20]基于双线性映射和分层的哈希索引结构提出了合作的 PDP(cooperative PDP,简称 CPDP)方案.CPDP 适合用于混合云中的服务扩展和数据迁徙,缺点是双线性映射的计算效率太低.Hanser 等人[21]基于椭圆曲线提出了高效的 PDP 方案,它可以在使用同一个预处理过程和中间数据元的情况下,同时支持私有验证和公开验证.2013 年,Shi 等人[22]提出了高效实用的动态 POR 方案,比 Stefanov 等人[23,24]以及 Cash 等人[25]的方案减少了通信代价,提高了效率.后来,Wang 等人[26−29]将可信第三方(trusted third party,简称 TTP)引入到了 PDP 和 POR 方案中用来,实现保护数据隐私的公开验证.TTP 可以代替客户端来进行数据持有性的验证,并且在验证过程中无法获得数据的内容.缺点是 TTP 的假设很强,现实应用中很难被实现,并且可能会造成额外的代价.

近几年,也有一些方案考虑到了群组的概念,例如:Wang 等人[30,31]提出的支持身份隐私保护的群组方案分别给出了支持公开验证和私有验证的方案,并且在完整性验证的过程中可以保证群组成员身份信息的隐私性.后来,Wang 等人[32]又给出了允许群组成员撤销的公开验证方案,撤销的成员不能利用已知的信息伪造标签值或证明通过合法成员的验证,并且协议中不需要群组成员之间协商任何隐私信息.Wang 等人[33]提出了群组存储证明(group-oriented proofs of storage,简称 GPOS)的概念,可以将通信带宽限制在固定大小,在多用户公开验证的场景下保证用户的隐私.但是这些方案都没有涉及数据去重的问题,而且方案构造中都使用了双线性映射,相对于私有验证来说计算效率比较低.

密文的数据去重(data deduplication)技术早在云计算之前就被研究过[34,35],但直到 2010 年,数据去重才第一次被 Harnik 等人[36]应用在云计算和云存储中,因为在云存储中直接进行数据去重的协议很容易遭受攻击.例如:云服务器为了节省空间,将多个用户相同的文件删除只保留一份数据,就可能有不诚实的客户在没有文件的情

况下声明他拥有这份数据.Halevi 等人[37]提出了权限证明(proofs of ownership,简称 POW)的概念,一定程度上解决了这个问题.2012 年,Zheng 等人[38]提出了支持数据去重的 POS 方案,将 POW 和 POS 方案进行了结合,在公开验证的情况下,有效减少了重复数据的标签值的数量,从而降低了服务器端的存储代价.但是其方案没有涉及群组的概念,如果将其直接应用于群组中,则无法抵抗选择成员攻击,并且协议构造使用了双线性映射,因此计算效率比较低.

3 预备知识

3.1 交叉认证码

3.1.1 定 义

交叉认证码的概念是由 Fehr 等人[39]在 2010 年提出的.

定义 1(L-交叉认证码(简称 L-XAC)). 对于 $L\in N$, L-XAC 是由密钥空间 $X\kappa$、标签空间 $X\tau$和 3 个概率多项式算法 ***XGen***,***XAuth*** 和 ***XVer*** 构成.

- ***XGen***$(1^k)\to K$:输入安全参数 k,随机产生密钥 $L\in X\kappa$;
- ***XAuth***$(K_1,K_2,\ldots,K_L)\to T$:输入密钥$(K_1,K_2,\ldots,K_L)$,输出标签 $T\in X\tau$;
- ***XVer***$(K_i,T)\to\{1,0\}$:输入第 i 个密钥 K_i 和标签 T,输出判定值 1 或者 0.

L-XAC 必须满足 3 个性质:正确性、抵抗模拟攻击和抵抗替换攻击:

- 正确性:对于所有的 $i\in[L]$,当 $K_1,\ldots,K_L\leftarrow$***XGen***(1^k)时,$fail_{\mathrm{XAC}}(k):=\Pr[$***XVer***$(K_i,$***XAuth***$(K_1,K_2,\ldots,K_L))\neq 1]$的概率是关于 k 的可忽略函数,表示为 $negl(k)$.它等于 k 的多项式运算和 k 的指数运算的比值,因此当 k 趋向于无穷大时,$negl(k)$无限接近于 0,所以概率是可忽略的;
- 抵抗模拟攻击:对于所有的 $i\in[L]$以及 $T'\in X\tau$, $Adv^{imp}_{\mathrm{XAC}}(k):=\max\limits_{i,T'}\Pr[XVer(K,T')=1\,|\,K\leftarrow XGen(1^k)]$ 的概率是可忽略的.其中,max 是指遍历所有的 i 和 T'.直观上说,当密钥 K 不是标签值 T'的生成密钥时,即使访问整个标签空间,也很难找到一个 T'可以通过 K 的验证;
- 抵抗替换攻击:对于所有的 $i\in[L]$,$K_{\neq i}=(K_j)_{j\neq i}\in X\kappa^{L-1}$ 以及所有(可能为随机的)函数 $F:X\tau\to X\tau$,下式的概率是可忽略的:

$$Adv^{sub}_{\mathrm{XAC}}(k):=\max_{i,K_{\neq i},F}\Pr\left[\begin{array}{c}T'\neq T\wedge\\ XVer(K_i,T')=1\end{array}\middle|\begin{array}{c}K_i\leftarrow XGen(1^k),\\ T:=XAuth(K_1,K_2,\ldots,K_L),\\ T'\leftarrow F(T)\end{array}\right].$$

直观上说,在拥有密钥 K_i 的情况下,可以很容易验证标签值 T 的正确性.但在没有 K_i 的情况下,即使拥有一个正确的标签值 T 和其他所有密钥 $K_{\neq i}$,也很难伪造一个合法的标签 T'并且可以通过 K_i 的验证.

3.1.2 L-XAC 方案

Fehr 等人给出了一个 L-XAC 的方案[39]如下:

G 是模数为 q 的有限域.设置初始值为 $X\kappa=G^2$ 和 $X\tau=G^L\cup\{\perp\}$,并且 ***XGen***(1^k)产生的密钥在 $X\kappa=G^2$ 中.对于 $K_1=(a_1,b_1),\ldots,K_L=(a_L,b_L)\in X\kappa$,认证标签 $T=$***XAuth***$(K_1,\ldots,K_L)$的计算方法如下:

对于全部 $i,1\leqslant i\leqslant L$,计算函数 $p_T(a_i)=b_i$,其中 $p_T(x)=T_0+T_1x+\ldots+T_{L-1}x^{L-1}\in G[x]$,从而计算得出唯一的向量值 $T=(T_0,\ldots,T_{L-1})\in G^L$.通过解线性方程组 $AT=B$ 可以高效地计算出 T 的值,其中:$A\in G^{L\times L}$ 是一个范德蒙矩阵,它的第 i 行为 $[1,a_i,a_i^2,\ldots,a_i^{L-1}]$;$B\in G^L$ 是一个值为$[b_1,\ldots,b_L]$的列向量.如果 $AT=B$ 无解或者有不止一个解,那么 T 设置为$\perp$.

对于任意的 $T\in X\tau$和 $K=(a,b)\in X\kappa$,当且仅当 $T\neq\perp$及 $p_T(a)=b$ 时,验证算法 ***XVer***(K_i,T)输出 1.

根据 Fehr 等人所证[39],上述 L-XAC 方案满足概率不等式如下:

$$fail_{\mathrm{XAC}}(k)\leqslant\frac{L(L-1)}{2q},Adv^{imp}_{\mathrm{XAC}}(k)\leqslant\frac{1}{q},Adv^{sub}_{\mathrm{XAC}}(k)\leqslant 2\cdot\frac{L-1}{q}.$$

由于 q 的大小取决于安全参数 k,假设 $q=2^k$,则上述概率不等式的右侧均为 k 的可忽略函数,因此满足 L-XAC

的 3 个性质.

3.2 同态消息认证码

同态消息认证码的概念是由 Dan Boneh[40]在 2009 年提出.

定义 2(同态消息认证码(HomMac)). HomMac 方案是由 4 个概率多项式时间算法 ***Gen***,***Sign***,***Combine*** 和 ***Verify*** 构成,如下所示(其中,向量用粗体形式表示):

- ***Gen***$(1^\lambda)\to Key$ 是产生密钥 Key 的概率多项式时间的算法,产生的密钥可能是对称密钥,也可能是公私钥对.输入安全参数λ,输出密钥 Key;
- ***Sign***$(Key,id,\boldsymbol{v}_i,i)\to t_i$ 是产生标签值的多项式时间的算法,输入密钥 Key,向量空间标识符 id,向量 $\boldsymbol{v}_i$ 和向量索引值 i,输出向量 $\boldsymbol{v}_i\in F_q^s$ 的标签值 t_i;
- ***Combine***$((\boldsymbol{v}_1,t_1,\alpha_1),\ldots,(\boldsymbol{v}_c,t_c,\alpha_c))\to(T,y)$是产生 HomMac 的多项式时间的算法.输入 c 个向量 $\boldsymbol{v}_1,\ldots,\boldsymbol{v}_c\in F_q^s$、相应的标签值 $t_1,\ldots,t_c$ 以及 c 个随机的常量参数$\alpha_1,\ldots,\alpha_c\in F_q$,然后计算相应标签值的总和 $T=\sum_{i=1}^{c}\alpha_i t_i\in F_q$,最后输出 T 和 y;
- ***Verify***$(Key,id,\boldsymbol{y},T)\to\{1,0\}$是用来验证 HomMac 的确定性算法.输入密钥 Key、向量空间标识符 id、向量 $\boldsymbol{y}\in F_q^s$ 和相应的 T,如果验证通过输出 1,否则输出 0.

同态消息认证码必须满足两个性质:正确性和安全性.定义如下所示:

- 正确性:对 $i=1,\ldots,m$, $\boldsymbol{v}_1,\ldots,\boldsymbol{v}_m\in F_q^s$ 以及向量空间标识符 $id,Key\in\kappa,t_i=$***Sign***$(Key,id,\boldsymbol{v}_i,i)$,假设$\alpha_1,\ldots,\alpha_m\in F_q$,那么概率 $fail_{\mathrm{HomMac}}(\lambda):=\Pr\left[\textbf{\textit{Verify}}\left(Key,id,\sum_{i=1}^{m}\alpha_i\boldsymbol{v}_i,\textbf{\textit{Combine}}((\boldsymbol{v}_1,t_1,\alpha_1),\ldots,(\boldsymbol{v}_m,t_m,\alpha_m))\right)\neq 1\right]$是可忽略的;
- 安全性: HomMac 安全性的定义类似于普通 Mac 的选择消息攻击的定义.假设敌手有能力根据自己的选择获取任意向量空间的标签值.对于每一个有唯一标识符 id_i 的向量空间 V_i,敌手产生一个合法的三元组$(id,\boldsymbol{y},t)$,其中,id 为新的标识符或者 $id=id_i\wedge\boldsymbol{y}\notin V_i$.这种伪造的概率是可忽略的.

随后,Dan Boneh 等人根据安全的伪随机函数和伪随机生成器构造了一个安全的(q,n,m) HomMac 方案[40],并证明了其正确性和安全性.

4 群组数据持有性证明方案

4.1 形式化定义

定义 3(支持数据去重的群组数据持有性证明(GPDP with deduplication,简称 GPDP)). GPDP 方案是由 4 个多项式时间的算法 ***KeyGen***,***TagGen***,***ProofGen*** 和 ***VerifProof*** 组成的.

- ***KeyGen***$(1^\lambda)\to\{K_l\}_{l=1}^{L}$ 是用来产生密钥的概率多项式时间的算法,输入安全参数λ,输出一系列相应密钥 $K_1,\ldots,K_L$.需要注意的是,密钥的个数应大于等于群组中成员的个数;
- ***TagGen***$(\{K_l\}_{l=1}^{L},m_i)\to T_i$ 是用来产生验证标签的多项式时间的算法,输入所有群组成员的密钥$\{K_l\}_{l=1}^{L}$和文件块 m_i,输出一个对应于文件块 m_i 的标签值 T_i.其中,i 是每个文件块 m_i 的索引;
- ***ProofGen***$(File,chal,\Sigma)\to\rho$是产生数据持有性证明的多项式时间的算法.输入文件块集合 $File$、标签值的集合Σ和挑战 $chal$,输出一个数据持有性证明ρ,其中,被证明的数据块是根据挑战 $chal$ 选择的;
- ***VerifProof***$(K_l,chal,\rho)\to\{1,0\}$是确定性的多项式时间的算法.输入第 l 个成员的密钥 K_l、挑战 $chal$ 和根据 $chal$ 产生的数据持有性证明ρ,如果验证通过输出 1,否则输出 0.

GPDP 系统是由 GPDP 方案构造的,分为两个阶段,过程如下所示:

- 预处理阶段

可信第三方 TTP 运行 ***KeyGen*** 和 ***TagGen***,产生群组成员的密钥$\{K_l\}_{l=1}^{L}$,并使用所有密钥产生每个数据块

m_i 的标签值 T_i.然后将密钥 K_l 分发给群组中成员 U_l,将文件集合 $File=\{m_i\}_{i=1}^n$ 和标签集合 $\Sigma=\{T_i\}_{i=1}^n$ 发送给云服务器 S,然后删除所有本地文件数据.在密钥个数多于成员个数情况下,可以将剩余未分发的密钥暂时存储于 TTP 上,在群组添加新成员后,将密钥分发出去.

- 挑战和验证阶段

群组成员 U_l(可以为一个或者多个)根据各自需求进行数据持有性的验证,U_l 产生一个挑战 $chal$ 发送给 S,S 根据挑战 $chal$ 选取相应的数据块,运行 ***ProofGen*** 产生持有性证明ρ并发送给 U_l,U_l 使用自己的密钥 K_l 运行 ***VerifProof*** 对ρ进行验证.

在第 2 个阶段中,每个成员都可以使用自己的密钥进行数据持有性证明的验证,每个成员的挑战和验证过程是互相独立的,可以同时进行也可以分别进行.因此,群组中某些成员被攻击后不会泄露其他成员的信息.另外,第 2 阶段可以被多次运行,用来保证数据的完整性和正确性.

4.2 具体GPDP方案实现

将文件分成 n 个文件块 $m_{\langle i\rangle}$,$1\leqslant i\leqslant n$,其中每个文件块包括 L 个小文件块:$m_{\langle i\rangle}=(m_{i,1},\ldots,m_{i,L})$.并且 $f:\kappa\times I\to F_q$ 是一个伪随机函数,$\pi:\kappa\times I\to I$ 是一个伪随机置换.q 为有限域 F_q 的阶,它的大小取决于安全参数λ(例如 $q=2^\lambda$).则 GPDP 的密钥空间为 $\kappa=F_q^3$,标签空间为 $F_q^L\cup\{\perp\}$.

具体方案细节如图 2 所示.

KeyGen$(1^\lambda)\to(a_k,b_k,c_k)$:
输入安全参数λ,输出一系列密钥$(a_k,b_k,c_k)\in\kappa$,$1\leqslant k\leqslant L$,其中,L 是群组允许包含成员的最大个数.
TagGen$((a_k,b_k,c_k),m_{\langle i\rangle})\to T_{\langle i\rangle}$:
标签值的计算方式如下:

$$\begin{bmatrix}1,a_1,a_1^2,\ldots,a_1^{L-1}\\1,a_2,a_2^2,\ldots,a_2^{L-1}\\\vdots\\1,a_L,a_L^2,\ldots,a_L^{L-1}\end{bmatrix}\begin{bmatrix}T_{i,1}\\T_{i,2}\\\vdots\\T_{i,L}\end{bmatrix}=\begin{bmatrix}f_{b_1}(i)\\f_{b_2}(i)\\\vdots\\f_{b_L}(i)\end{bmatrix}+\begin{bmatrix}1,c_1,c_1^2,\ldots,c_1^{L-1}\\1,c_2,c_2^2,\ldots,c_2^{L-1}\\\vdots\\1,c_L,c_L^2,\ldots,c_L^{L-1}\end{bmatrix}\begin{bmatrix}m_{i,1}\\m_{i,2}\\\vdots\\m_{i,L}\end{bmatrix}$$

通过解 L 元一次方程式组,可以高效地计算出标签值 $T_{\langle i\rangle}=(T_{i,1},\ldots,T_{i,L})$.
输出文件集合和标签集合$\{m_{\langle i\rangle},T_{\langle i\rangle}\}_{1\leqslant i\leqslant n}$.
ProofGen$(\{m_{\langle i\rangle}\}_{1\leqslant i\leqslant n},\{T_{\langle i\rangle}\}_{1\leqslant i\leqslant n},chal)\to\rho$:
根据挑战得到临时密钥(k_1,k_2)和抽样的文件块数量 c:$chal\to(k_1,k_2,c)$.
对于 $1\leqslant z\leqslant c$:

1. 计算每个抽取的文件块的索引号: $i_z=\pi_{k_1}(z)$;
2. 计算每个抽取的文件块的相关参数: $v_z=f_{k_2}(z)$.

然后分别计算: $\begin{cases}\tau_1=v_1T_{i_1,1}+v_2T_{i_2,1}+\ldots+v_cT_{i_c,1}\\\tau_2=v_1T_{i_1,2}+v_2T_{i_2,2}+\ldots+v_cT_{i_c,2}\\\vdots\\\tau_L=v_1T_{i_1,L}+v_2T_{i_2,L}+\ldots+v_cT_{i_c,L}\end{cases}$ 和 $\begin{cases}\omega_1=v_1m_{i_1,1}+v_2m_{i_2,1}+\ldots+v_cm_{i_c,1}\\\omega_2=v_1m_{i_1,2}+v_2m_{i_2,2}+\ldots+v_cm_{i_c,2}\\\vdots\\\omega_L=v_1m_{i_1,L}+v_2m_{i_2,L}+\ldots+v_cm_{i_c,L}\end{cases}$.

输出向量$(\tau_1,\ldots,\tau_L)$和$(\omega_1,\ldots,\omega_L)$.
VerifProof$(\rho,chal,(a_k,b_k,c_k))\to\{1,0\}$:
根据挑战得到相应的信息:$chal\to(k_1,k_2,c)$.
同样,对于 $1\leqslant z\leqslant c$:

1. 计算每个抽取的文件块的索引号: $i_z=\pi_{k_1}(z)$;
2. 计算每个抽取的文件块的相关参数: $v_z=f_{k_2}(z)$.

验证过程如下:
分别计算: $\tau=\tau_1+a_k\tau_2+\ldots+a_k^{L-1}\tau_L,\omega=\omega_1+c_k\omega_2+\ldots+c_k^{L-1}\omega_L,\sigma=v_1f_{b_k}(i_1)+\ldots+v_cf_{b_k}(i_c)$.
验证$\tau=\sigma+\omega$是否成立:如果成立,则验证通过并输出 1;否则输出 0.

Fig.2 The detailed construction of GPDP scheme

图 2 具体 GPDP 方案实现

注意:

1. 为了避免 ***TagGen*** 中方程组无解或多解,要求密钥 a 必须两两不相等,即, $a_{r_1} \neq a_{r_2} \wedge r_1 \neq r_2, 1 \leqslant r_1, r_2 \leqslant L$; 另外,为了增加随机性,我们要求密钥 b 和 c 都两两不相等,即, $b_{r_1} \neq b_{r_2} \wedge c_{r_1} \neq c_{r_2} \wedge r_1 \neq r_2, 1 \leqslant r_1, r_2 \leqslant L$;
2. 群组中每个成员的挑战验证过程是相互独立的,因此,某些成员被攻击密钥泄露后不会影响其他成员的验证,并且敌手也没有办法伪造一个合法的标签值来通过合法成员的验证;
3. 文件块索引 i 是一个全局变量,即使在多文件情况下,也可以使用统一的索引值;
4. L 是群组内成员个数的上界,方案允许群组成员个数 $x \leqslant L$ 的情况下正常运行,如果在现实应用中有增加成员数量的情况,可以让 TTP 在密钥产生阶段适当增大 L 的值.

5 安全性分析

5.1 安全模型

首先,我们使用数据持有性游戏和抵抗选择攻击的游戏来定义 GPDP 的安全模型.

- 游戏 1:数据持有性游戏

设置:挑战者运行 ***KeyGen***(1^λ)产生一系列密钥 $\{K_l\}_{l=1}^L$,并且保持其私密性.

询问:敌手适应性地选择文件块向挑战者进行询问:敌手选择一个文件块 m_i 并发送给挑战者,然后,挑战者运行 ***TagGen***$(\{K_l\}_{l=1}^L, m_i) \to T_i$ 来计算标签值 T_i 并且发送给敌手,这个查询过程可以被执行多次.然后,敌手存储所有的文件块集合 $File=(m_1,\ldots,m_n)$和标签值集合$\Sigma=(T_1,\ldots,T_n)$.

挑战:挑战者产生一个挑战 $chal$ 并且向敌手请求文件块 $m_{i_1},\ldots,m_{i_c}$ $(1 \leqslant i_c \leqslant n)$ 的持有性证明.

伪造:敌手根据挑战 $chal$ 计算一个持有性证明ρ并发送给挑战者.

如果 ***VerifProof***$(K',chal,\rho) \to 1$,其中, $K' \in \{K\}_{l=1}^L$,则敌手赢了这个游戏.

直观地说,敌手赢了游戏的概率可忽略地接近于抽取器ε抽取出挑战 $chal$ 所对应文件块 $m_{i_1},\ldots,m_{i_c}$ $(1 \leqslant i_c \leqslant n)$的概率.

- 游戏 2:抵抗选择攻击的游戏

设置:挑战者运行 ***KeyGen***(1^λ)产生一系列密钥 $\{K_l\}_{l=1}^L$,并且保持其私密性.

选择攻击:敌手可以适应性地选择密钥询问,注意,询问的密钥数量必须小于 L.挑战者将被询问的密钥发送给敌手,并保存好剩余密钥.询问结束后,敌手保存密钥 $\{K_{l_1},\ldots,K_{l_r}\}$,挑战者剩余隐私密钥 $\{K_l\}_{l=1}^L / \{K_{l_1},\ldots,K_{l_r}\}$.

伪造:敌手可以获得文件块集合 $File=(m_1,\ldots,m_n)$、标签值集合$\Sigma=(T_1,\ldots,T_n)$和密钥集合 $\{K_{l_1},\ldots,K_{l_r}\}$,运行函数 ***TagGen*** 并输出其中一个标签值 T_r' 并且 $T_r' \neq T_r$,其中,T_r 为文件块 m_r 的原始标签值.

如果 ***VerifProof***$(K',chal,\rho) \to 1$,其中 $K' \in \{K_l\}_{l=1}^L / \{K_{l_1},\ldots,K_{l_r}\}$,则敌手赢了这个游戏.

如果方案是抵抗选择攻击的,那么敌手赢了游戏的概率是可忽略的,即,满足如下不等式:

$$\Pr\left[\begin{array}{c} \textbf{\textit{TagGen}}(\{K_{l_1},\ldots,K_{l_r}\},m_r) \to T_r' \\ \textbf{\textit{TagGen}}(\{K_l\}_{l=1}^L, m_r) \to T_r \end{array} \wedge T_r' \neq T_r \wedge \begin{array}{c} \textbf{\textit{VerifProof}}(K',chal,\rho) \to 1 \\ K' \in \{K_l\}_{l=1}^L / \{K_{l_1},\ldots,K_{l_r}\} \\ \textbf{\textit{ProofGen}}(File, T_r') \to \rho \end{array}\right] \leqslant negl(\lambda).$$

要注意的是:在伪造阶段,敌手之所以可以获得 $File$ 和Σ,是因为敌手可以与云服务器 S 合谋,甚至 S 本身也可以是敌手.

5.2 安全性证明

定理 1. 假设 f 是安全的伪随机函数且 TTP 是安全可信的第三方,则 GPDP 方案是在标准模型下支持数据去重的、并且可以抵抗选择攻击的、安全的群组数据持有性方案.

证明:

(1) 正确性和可靠性

在一个 GPDP 方案中,正确性即对合法证明的验证,可靠性即对非法证明的验证.

- 对合法证明的验证

对于所有的 l,$1\leqslant l\leqslant L$,当$(a_l,b_l,c_l)\leftarrow \boldsymbol{KeyGen}(1^\lambda)$时,对于任意一个合法的证明,验证失败的概率为

$$fail_{\mathrm{GPDP}}(\lambda):=\Pr[\boldsymbol{VerifProof}(K_l,chal,\boldsymbol{ProofGen}(\{m_{\langle i\rangle}\}_{1\leqslant i\leqslant n},\{T_{\langle i\rangle}\}_{1\leqslant i\leqslant n},chal))\neq 1].$$

由于 F 是一个安全的伪随机函数,则标签计算的等式可以表示为 $AT=X.A$ 为 a 的矩阵,T 为标签值的列矩阵,X 为一个随机矩阵.则根据 L-XAC 的安全模型可以得出: $fail_{\mathrm{GPDP}}(\lambda)\leqslant \dfrac{L(L-1)}{2q}$.

- 对非法证明的验证

对于所有的 l,$1\leqslant l\leqslant L$,任意一个非法证明$\rho'\notin\{\boldsymbol{ProofGen}(File,\Sigma,chal)\}$通过验证的概率为

$$Adv_{\mathrm{GPDP}}^{illeg}(\lambda):=\max_{l}\Pr[\boldsymbol{VerifProof}(K_l,chal,\rho')=1\,|\,K_l\leftarrow \boldsymbol{KeyGen}(1^\lambda)],$$

其中,max 是指遍历所有的 l.可以得出: $Adv_{\mathrm{GPDP}}^{illeg}(\lambda)\leqslant \dfrac{1}{q^c}$.

(2) 抵抗选择攻击

运行游戏 2 如下所示:

设置:挑战者运行 $\boldsymbol{KeyGen}(1^\lambda)$产生一系列密钥$\{K_l\}_{l=1}^L$,并且保持其私密性.

选择攻击:敌手可以适应性地选择密钥向挑战者进行询问,询问结束后,敌手保存密钥$\{K_{l_1},...,K_{l_r}\}$,挑战者剩余隐私密钥为$\{K_l\}_{l=1}^L/\{K_{l_1},...,K_{l_r}\}$.

伪造:敌手利用已知的信息:文件块集合 $File=(m_1,\ldots,m_n)$、标签值集合$\Sigma=(T_1,\ldots,T_n)$和密钥集合$\{K_{l_1},...,K_{l_r}\}$,运行函数 $\boldsymbol{TagGen}(\{K_{l_1},...,K_{l_r}\},File)$ 并输出标签值 $T_r',T_r'\neq T_r$,其中,T_r 为文件块 m_r 的原始标签值,即:

$$T_r=\boldsymbol{TagGen}(\{K_l\}_{l=1}^L,m_r).$$

根据 L-XAC 的抵抗替换攻击,可得出 GPDP 方案抵抗选择攻击的概率为

$$Adv_{\mathrm{GPDP}}^{select}(\lambda):=\max_{File,\Sigma}\Pr\left[\begin{array}{c|c} T_r'\neq T_r\wedge & K'\leftarrow\{K_l\}_{l=1}^L/\{K_{l_1},...,K_{l_r}\} \\ \boldsymbol{VerifProof}(K',\rho)=1 & T_r:=\boldsymbol{TagGen}(\{K_l\}_{l=1}^L,m_r) \\ & T_r'\leftarrow Forge \\ & \rho\leftarrow\boldsymbol{ProofGen}(File,T_r') \end{array}\right]=Adv_{\mathrm{XAC}}^{sub}(\lambda)\leqslant 2\cdot\frac{L-1}{q}.$$

由前文可知,q 是λ的指数函数,因此上述概率函数都是可忽略的.

(3) 数据持有性

在证明数据持有性时,我们将参数进行简化,将 v_z 都设置为 1,矩阵用大写字母进行表示,使用伪随机函数 f 和随机数 r 分别在现实环境和理想环境中执行游戏 1.

- 现实环境中

设置:挑战者运行 $\boldsymbol{KeyGen}(1^\lambda)$产生一系列密钥$\{K_l\}_{l=1}^L$,并且保持其私密性.

询问:敌手选择文件块 m_i 并发送给挑战者,然后,挑战者运行 $\boldsymbol{TagGen}(\{K_l\}_{l=1}^L,m_i)$ 来计算标签值 T_i,过程如下:使得 $T=A^{-1}CM+A^{-1}B$,其中,M 为文件块列矩阵,A,C 是密钥矩阵, $B=[f_{b_1}(i),f_{b_2}(i),...,f_{b_L}(i)]^T$.然后,将标签矩阵转秩成向量格式 $T_i=T^T$ 并发送给敌手,这个查询过程可以被执行多次.然后,敌手存储所有的文件块集合 $File=(m_1,\ldots,m_n)$和标签值集合$\Sigma=(T_1,\ldots,T_n)$.

挑战:挑战者产生一个挑战 $chal$ 并且向敌手请求文件块 $m_{i_1},...,m_{i_c}(1\leqslant i_c\leqslant n)$ 的持有性证明.

伪造:敌手根据挑战 $chal$ 计算一个持有性证明ρ,并发送给挑战者.

- 理想环境中

使用随机数代替伪随机函数,则 $B=[r_1,r_2,\ldots,r_L]^T$.对所有的随机数进行记录,并且在验证过程中使用相应的随机数进行计算.

假设敌手可以在 M 发生改变的情况下完成验证,即在理想环境下,敌手可以成功找到 T' 使得 $T'=A^{-1}CM'+$

$A^{-1}B$,其中,M' 为改变后的 M.

由于 A 和 C 为随机生成的密钥,B 为纯随机数,则可得不等式如下所示:

$$A_{\mathrm{GPDP}}^{ideal}=\Pr[T'\mid T'=A^{-1}CM'+A^{-1}B]=\Pr[T'\mid T'=R_1M'+R_2]\leqslant\frac{1}{2^{\lambda\cdot L}},$$

其中,R_1,R_2 为随机数矩阵.

根据假设:f 是一个安全的伪随机函数,则敌手无法区分协议是在现实环境中还是在理想环境中执行的,因此在现实环境中,敌手伪造的概率: $A_{\mathrm{GPDP}}^{real}\cong A_{\mathrm{GPDP}}^{ideal}\leqslant\frac{1}{2^{\lambda\cdot L}}$. □

6 实验结果和分析

实验环境:所有数据使用百度云服务器进行存储和计算,假设方案中服务器、TTP 和客户端有着相同的计算能力,则所有实验都是运行在16G内存的Red Hat Enterprise Linux AS release 4 (Nahant Update 3)操作系统上,算法使用的密码学库是版本为 0.9.7a 的 OpenSSL.实验中测试的时间数据包含磁盘文件 I/O 产生的时间消耗,实验采用转速 7 200,接口类型是 SATA 6gb/s 的硬盘.由于每次测试的结果会有一定差别,所以全部实验结果都是进行了多次测试和比较后得出的.

GPDP 中使用的抽样方法类似于 S-PDP[9]:n 为存储在云服务器上的文件块的总数,t 为被修改或者被删除的文件块的数量,c 为挑战指定的需要抽样的文件块数量.假设 X 为抽样到的并且已经被改动或被删除的文件块数量,那么 P_X 为至少有一块被改动或被删除的文件块被抽中的概率,即,可以检测出数据完整性的概率,那么可以得到概率为

$$P_X=P\{X\geqslant 1\}=1-P\{X=0\}=1-\frac{n-t}{n}\cdot\frac{n-1-t}{n-1}\cdot\frac{n-2-t}{n-2}\cdot\ldots\cdot\frac{n-c+1-t}{n-c+1}.$$

最后得到不等式: $1-\left(\frac{n-t}{n}\right)^c\leqslant P_X\leqslant 1-\left(\frac{n-c+1-t}{n-c+1}\right)^c$.

容易看出:可以通过检测固定数量的文件块,产生数据持有性证明来以相当高的概率保证数据的完整性和正确性.例如:t 为 n 的 1%时,假设 P_X 概率至少为 99%,那么需要检测的文件块个数为 460.

6.1 各种PDP和POR方案对比

不同的 PDP 和 POR 方案的性能对比见表 1,其中,n 为总的文件块数量,c 为一次挑战中抽样的文件块的数量,w 为存储在服务器端的验证标签等额外存储,l 是每个大文件块中小文件块的数量,L 为群组中成员的最大数量.注意:在 GPDP 方案中,l 与 L 相等.

Table 1 The comparation of the performance of various PDP and POR schemes

表 1 不同 PDP 和 POR 方案的性能比较

方案名称	S-PDP[9]	CPOR[12]	POSD[38]	Oruta[31]	GPoS[33]	GPDP
支持数据持有性证明	是	是	是	是	是	是
是否可用于群组	是	是	是	是	是	是
支持数据去重	否	否	是	否	否	是
抵抗选择攻击	是	是	否	否	否	是
成员之间预处理密钥	不同	不同	不同	不同	不同	不同
成员之间验证密钥	不同	不同	不同	相同	不同	不同
预处理复杂性	$O(L\cdot n)$	$O(L\cdot n)$	$O(L\cdot n)$	$O(n)$	$O(n)$	$O(n)$
产生一次证明	$O(c)$	$O(c)$	$O(c)$	$O(c)$	$O(c)$	$O(c)$
验证一次证明	$O(1)$	$O(1)$	$O(1)$	$O(1)$	$O(1)$	$O(1)$
通信复杂性	$O(1)$	$O(l)$	$O(l)$	$O(l)$	$O(l)$	$O(L)$
存储空间复杂性	$O(n+w\cdot L)$	$O(n+w\cdot L)$	$O(n+w)$	$O(n+w\cdot L)$	$O(n+w\cdot L)$	$O(n+w)$

当 S-PDP 方案和 CPOR 方案用于群组时,相当于每个成员分别用自己的密钥对文件数据进行预处理和挑战验证操作,每个成员的密钥和标签值都是相互独立的,因此可以抵抗选择攻击.

从表 1 可以看出,当目前的 PDP 和 POR 方案应用于群组时,可以分为两类:

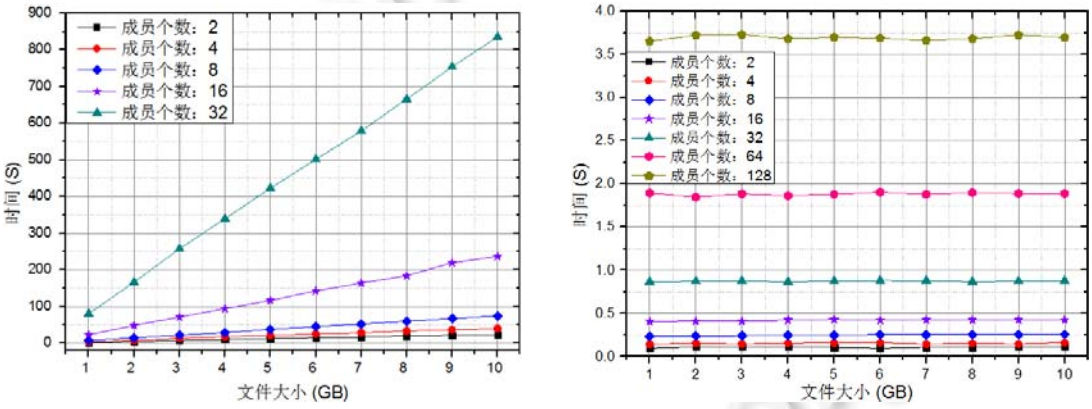
1) 群组中成员使用不同的密钥进行预处理和验证:在这种情况下,成员之间相互独立,其中某个或者某些成员被攻击,造成部分密钥泄露时,剩余的合法成员还能正确的完成数据持有性证明,达到了抵抗选择攻击的目标;但是每个成员使用各自密钥产生一份标签值,造成大量额外存储代价,不能达到数据去重的目标.在群组成员数量增加时,服务器的存储代价呈线性增长;

2) 群组中成员使用相同的密钥进行预处理和验证:在这种情况下,成员之间共享一个或者一组密钥,因此在预处理过程中只需要产生一份标签值,所有成员都可以进行数据持有性的验证,达到了数据去重的目标;但是由于所有成员共享密钥,所以当群组中部分成员被攻击导致密钥泄露时,恶意敌手(可能是云服务器,也可能是第三方与云服务器共谋)就可以根据泄露的密钥伪造标签值和数据持有性证明,从而通过剩余合法用户的验证,不能达到抵抗选择攻击的数据持有性证明的目标.

然而,GPDP 方案可以有效地解决上述问题:在同时达到数据去重和抵抗选择攻击的目标的前提下,高效地保证数据持有性证明的验证.

6.2 GPDP方案效率测试

图 3 给出了在不同成员个数情况下,GPDP 方案的性能测试结果.



(a) GPDP 预处理过程 (b) GPDP 挑战验证过程

Fig.3 The performance evaluation of GPDP with different member number

图 3 不同成员个数情况下 GPDP 方案性能测试

在预处理阶段,TTP 使用所有群组成员的密钥对文件进行预处理,产生验证标签,并将标签值集合和处理后的文件发送给云服务器,将密钥分发给每个成员后删除所有本地数据.我们实验中测试了预处理的所有计算时间,包括密钥产生时间、验证标签的计算时间以及必要的 I/O 时间,但不包括数据传输所使用的时间.

如图 3(a)所示:随着文件数据总量的增加,计算时间呈线性增长.这是因为对于固定的文件块大小,文件数据总量越大,文件块的数量也就越多,对应的标签值的总数量也会增加,因此,计算量也呈线性增长.另外,对于相同大小的文件块和文件总量,成员个数越多,计算时间越长.这是因为标签值的产生需要每个成员的密钥参与计算,因此成员数量增加时,需要计算的数据相应增加,因此计算时间相应增长.虽然当文件总量比较大以及成员个数比较多的时候计算时间相对较大,但是预处理过程中的计算是一次性的,并且可以在挑战验证过程中被无限次使用,因此实验结果是完全可以被接受的.

在挑战和验证阶段,任何群组的成员 U_k 都可以向云服务器发送挑战请求,云服务器根据挑战生成相应数据块的持有性证明并发送给成员 U_k,然后,U_k 根据自己的密钥对证明进行验证.在实验过程中,我们对产生证明、验证证明以及相应的 I/O 操作的时间进行了记录,忽略了数据传输使用的时间以及抽样过程中数据查找和比对的

I/O 时间.

从图 3(b)可以看出:对于不同的文件总量,在固定了文件块大小和群组成员个数的情况下,挑战和验证的时间是固定的.这是因为在实验中,我们将抽样的文件块数量固定取值为 460,因此一次产生证明和验证证明的计算时间与文件总量无关.但是随着成员个数的增加,需要计算的小文件块(每个文件块分成与群组成员的最大个数相同数量的小文件块)数量增加,因此计算量也相应增加.当群组成员个数比较少(20 以下)的时候,产生证明和验证证明的时间维持在 0.5s 以下;当群组成员数量比较大(100 以上)时,产生证明和验证证明的时间维持在 3.7S 左右.对于一个真实的系统来说,实验结果效率很高,可以应用于实际环境中.

另外,上述计算效率与成员密钥是否被攻击没有关系.因为在预处理过程中,TTP 使用所有成员的密钥进行计算,不管密钥泄漏与否,对于固定的成员个数和文件块数量,计算时间是基本恒定的;而在挑战验证过程中,使用任意成员密钥都可以独立进行验证,即使有部分成员被攻击密钥泄漏,剩余任何一个合法成员使用密钥进行验证的时间是基本恒定的.方案支持抵抗选择攻击并不影响实际的执行效率.

6.3 与私有验证方案和公开验证方案进行效率对比

本节中,我们将 GPDP 方案分别与私有验证方案和公开验证方案进行了计算效率的对比.在预处理阶段,将不同群组成员个数的 GPDP 方案与 S-PDP 方案进行对比,S-PDP 作为数据持有性证明领域中的里程碑方案,高效且实用性强,因此作为我们实验的对比方案.

从图 4 可以看出:随着文件总量的增加,两个方案的预处理时间都呈线性增长.原因与前述类似,在固定了文件块大小和群组成员个数的情况下,文件总量越大,需要计算的标签值就越多,因此预处理的时间也越长.而对于相同的文件总量和文件块大小,当群组成员个数越大,计算标签时进行的运算操作次数越多,因此计算时间越长.另外,从图中很容易看出:对于相同的文件总量、固定的文件块大小以及相同的群组成员个数,S-PDP 预处理需要的时间比 GPDP 长.这是因为 S-PDP 预处理过程中的主要运算为指数和乘法运算,而 GPDP 预处理过程中只有乘法和加法运算,因此 GPDP 单次运算的效率比 S-PDP 高.当文件总量增加时,这种优势会更加明显.

图 5 为目前群组 PDP 和 POR 方案中通用的公开验证与私有验证方法的效率对比,为了方便直观,我们仅对比在客户端的验证时间.实验中,我们使用版本号为 0.5.14 的 Paring-Based Cryptography(PBC)库,模数 N 为 1 024 比特.从图中容易看到:在 I/O 影响可接受范围内,一次双线性映射操作的时间是一次指数运算时间的 9 倍左右;同时,是一次乘法运算时间的 350 倍左右.很明显的看出,双线性映射的操作是非常耗时的.

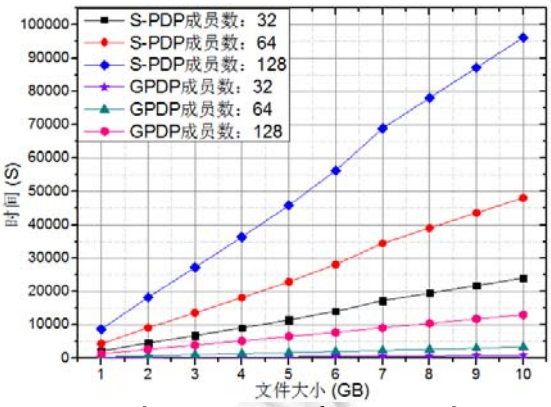


Fig.4 The comparation of pre-process between GPDP and S-PDP

图 4 GPDP 与 S-PDP 预处理过程效率对比

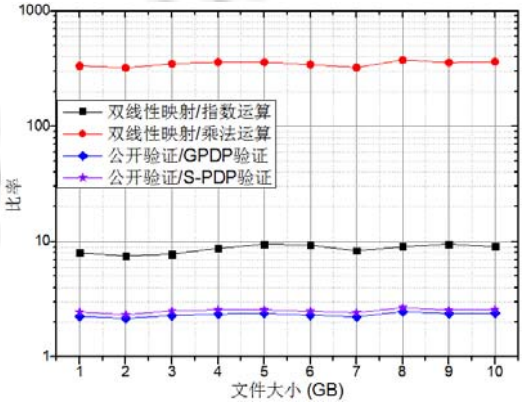


Fig.5 The comparation between public and private verification

图 5 公开验与私有验证效率对比

在验证阶段,我们在公开验证和私有验证方案中抽样的数据块数量都是 460.实验显示,公开验证的时间是私有验证的 2.5 倍左右.尽管对于一次验证来说时间优势不是特别明显,但是在一个真实的系统中,许多不同的客户端会进行成百上千甚至上万次验证操作,因此基于双线性映射的公开验证方案在实际系统中会变得低效

和不实用,而 GPDP 方案更适合应用于这样的现实场景中.

6.4 存储代价和通信带宽

6.4.1 存储代价

GPDP 需要的存储空间比其他群组 PDP 和 POR 方案都要少.在不支持数据去重的方案中,为了能够达到抵抗选择攻击的目标,需要使用多个密钥.对于同一份文件,每个密钥对应一份独立的标签值集合.以 S-PDP 为例,当群组中只有一个成员时,如果文件块大小为 4KB、模数 N 取值为 1 024 比特,则增加的额外存储空间为原始文件数据量的 3.125%.由于不同密钥对于同一份文件产生不同的标签值集合,因此当群组中成员增加时,标签值总量也成倍增加.例如:当群组成员个数为 100 时,额外存储空间为原始文件数据量的 3.125 倍.这种存储代价的线性增长对群组来说是非常棘手的问题.

而 GPDP 方案中,额外存储空间的大小与群组成员个数无关,仅与安全参数有关.例如:当文件块大小为 4KB、模数 N 取值为 1 024 比特时,无论群组成员个数为多少,增加的额外存储空间始终是原始文件数据量的 3.125%.因此,GPDP 方案可以有效地解决标签值数据去重的问题.

GPDP 方案在客户端的存储代价为 $O(1)$,与文件总量无关,仅与安全参数的大小有关.

6.4.2 通信带宽

在预处理阶段,GPDP 方案的传输代价比其他群组 PDP 和 POR 方案都小.因为对于相同的文件数据,GPDP 产生的标签值总量比其他方案少,因此发送给云服务器的总数据量比其他群组 PDP 和 POR 方案都少.

GPDP 方案在发送挑战阶段所需要的带宽为 $O(1)$,相当于挑战 *chal* 的长度(小于 0.4KB).在验证阶段所需要的带宽为 $O(L)$,其中,L 为群组中成员的最大数量,也就是每个大文件块划分成的小文件块的个数.假设模数 N 取值为 1 024 比特,则每次验证的传输带宽为 $0.25\cdot L$KB.实际应用中,比其他群组 PDP 和 POR 方案的通信带宽都要小一些.

7 结束语

本文给出了支持数据去重的群组 PDP 方案(GPDP),基于矩阵计算和伪随机函数,GPDP 可以高效地完成数据持有性证明,并且可以在群组应用中达到抵抗选择成员攻击以及数据去重的双重目的.

我们在标准模型下给出了 GPDP 的安全性证明,并且在百度云平台上实现了 GPDP 的原型系统.我们使用 10G 的数据量进行实验来分析和评估方案的性能,结果表明:GPDP 方案在达到群组中数据去重目标的前提下,高效地保证了抵抗选择攻击和数据持有性.直观地说,GPDP 方案的预处理效率高于私有验证方案,而验证效率高于公开验证方案(与私有验证效率几乎相同).另外,与其他群组 PDP 和 POR 方案相比,GPDP 方案将额外存储代价和通信代价都降到了最低.

在未来工作中,如何用新的工具来构造高效去重和抵抗选择攻击的群组 PDP 和 POR 方案,是一个很值得研究的方向.

致谢 在此,我们向对本文的工作给予支持和建议的同行,尤其是中国科学院信息工程研究所第一研究室薛锐老师讨论班上的同学和老师表示感谢.

References:

[1] Tan S, Jia Y, Han HW. Research and development of provable data integrity in cloud storage. Chinese Journal of Computers, 2015,38(1):164−177 (in Chinese with English abstract).

[2] Yu NH, Hao Z, Xu JJ, Zhang WM, Zhang C. Review of cloud computing security. Journal of Electritic, 2013,41(2):371−381 (in Chinese with English abstract).

[3] Yuan J, Yu S. Efficient public integrity checking for cloud data sharing with multi-user modification. In: Proc. of the 2014 IEEE INFOCOM. IEEE, 2014. 2121−2129. [doi: 10.1109/INFOCOM.2014.6848154]

[4] Zhu Y, Ahn GJ, Hu H, Yau SS, An HG, Hu CJ. Dynamic audit services for outsourced storages in clouds. IEEE Trans. on Services Computing, 2013, 6(2):227−238. [doi: 10.1109/TSC.2011.51]

[5] Rong L, Li L, Li CL. Extensible provable data possession scheme with data dynamics.Application Research of Computers, 2013,30(7):2132−2135 (in Chinese with English abstract).

[6] Juels A, KaliskiJr BS, Bowers KD. Proof of Retrievability for Archived Files. U.S. Patent 8381062, 2013-2-19.

[7] Armknecht F, Bohli JM, Karame GO, Liu ZR, Reuter CA. Outsourced proofs of retrievability. In: Proc. of the 2014 ACM SIGSAC Conf. on Computer and Communications Security. ACM Press, 2014. 831−843. [doi: 10.1145/2660267.2660310]

[8] 朱岩,王怀习,胡泽行,Gail-Joon AHN,胡宏新.数据可恢复性的零知识证明.中国科学:信息科学,2011,41(10):1227−1237.

[9] Ateniese G, Burns R, Curtmola R, Herring J, Kissaer L, Peterson Z, Song D. Provable data possession at untrusted stores. In: Proc. of the 14th ACM Conf. on Computer and Communications Security. ACM Press, 2007. 598−609. [doi: 10.1145/1315245.1315318]

[10] Ateniese G, Burns R, Curtmola R, Herring J, Khan O, Kissner L, Peterson Z, Song D. Remote data checking using provable data possession. ACM Trans. on Information & System Security, 2011,14(1):1165−1182. [doi: 10.1145/1952982.1952994]

[11] Juels A, Kaliski BS. Pors: Proofs of retrievability for large files. In: Proc. of the 14th ACM Conf. on Computer and Communications Security. ACM Press, 2007. 584−597. [doi: 10.1145/1315245.1315317]

[12] Shacham H, Waters B. Compact proofs of retrievability. In: Proc. of the Advances in Cryptology (ASIACRYPT 2008). Berlin, Heidelberg: Springer-Verlag, 2008. 90−107. [doi: 10.1007/978-3-540-89255-7_7]

[13] Ateniese G, Di Pietro R, Mancini LV, Tsudik G. Scalable and efficient provable data possession. In: Proc. of the 4th Int'l Conf. on Security and Privacy in Communication Networks. ACM Press, 2008. 9. [doi: 10.1145/1460877.1460889]

[14] Dodis Y, Vadhan S, Wichs D. Proofs of retrievability via hardness amplification. In: Proc. of the Theory of Cryptography. Berlin, Heidelberg: Springer-Verlag, 2009. 109−127. [doi: 10.1007/978-3-642-00457-5_8]

[15] Ateniese G, Kamara S, Katz J. Proofs of storage from homomorphic identification protocols. LNCS, 2009. 319−333. [doi: 10.1007/978-3-642-10366-7_19]

[16] Erway CC, Küpçü A, Papamanthou C, Tamassia R. Dynamic provable data possession. In: Proc. of the ACM Conf. on Computer and Communications Security. 2009. 213−222. [doi: 10.1145/1653662.1653688]

[17] Chen B, Curtmola R. Robust dynamic provable data possession. In: Proc. of the ICDCS Workshops. 2012. 515−525. [doi: 10.1109/ICDCSW.2012.57]

[18] Zheng Q, Xu S. Fair and dynamic proofs of retrievability. In: Proc. of the 1st ACM Conf. on Data and Application Security and Privacy. ACM Press, 2011. 237−248. [doi: 10.1145/1943513.1943546]

[19] Zhu Y, Wang H, Hu Z, Ahn G, Hu H, Yau SS. Efficient provable data possession for hybrid clouds. In: Proc. of the ACM Conf. on Computer and Communications Security. 2010. 756−758. [doi: 10.1145/1866307.1866421]

[20] Zhu Y, Hu H, Ahn G, Yu M. Cooperative provable data possession for integrity verification in multicloud storage. IEEE Trans. on Parallel Distributed Systems, 2012. 2231−2244.

[21] Hanser C, Slamanig D. Efficient simultaneous privately and publicly verifiable robust provable data possession from elliptic curves. In: Proc. of the 2013 Int'l Conf. on Security and Cryptography (SECRYPT). IEEE, 2013. 1−12.

[22] Shi E, Stefanov E, Papamanthou C. Practical dynamic proofs of retrievability. In: Proc. of the 2013 ACM SIGSAC Conf. on Computer & Communications Security. ACM Press, 2013. 325−336. [doi: 10.1145/2508859.2516669]

[23] Stefanov E, van Dijk M, Juels A, Oprea A. Iris: A scalable cloud file system with efficient integrity checks. In: Proc. of the 28th Annual Computer Security Applications Conf. ACM Press, 2012. 229−238. [doi: 10.1145/2420950.2420985]

[24] Stefanov E, Shi E, Song D. Towards Practical Oblivious RAM. NDSS the Internet Society, 2011.

[25] Cash D, Küpçü A, Wichs D. Dynamic proofs of retrievability via oblivious ram. In: Proc. of the Advances in Cryptology (EUROCRYPT 2013). Berlin, Heidelberg: Springer-Verlag, 2013. 279−295. [doi: 10.1007/978-3-642-38348-9_17]

[26] Wang Q, Wang C, Li J, Ren K, Lon WJ. Enabling public verifiability and data dynamics for storage security in cloud computing. In: Proc. of the Computer Security (ESORICS 2009). Berlin, Heidelberg: Springer-Verlag, 2009. 355−370. [doi: 10.1007/978-3-642-04444-1_22]

[27] Wang C, Wang Q, Ren K, Lou WJ. Privacy-Preserving public auditing for data storage security in cloud computing. In: Proc. of the 2010 IEEE INFOCOM. IEEE, 2010. 1−9. [doi: 10.1109/INFCOM.2010.5462173]

[28] Wang C, Wang Q, Ren K, Cao N, Lou WJ. Toward secure and dependable storage services in cloud computing. IEEE Trans. on Services Computing, 2012,5(2):220−232. [doi: 10.1109/TSC.2011.24]

[29] Wang C, Chow SM, Wang Q, Ren K, Lou WJ. Privacy-Preserving public auditing for secure cloud storage. IEEE Trans. on Computers, 2013,62(2):362−375. [doi: 10.1109/TC.2011.245]

[30] Wang B, Li B, Li H. Knox: Privacy-Preserving auditing for shared data with large groups in the cloud. LNCS, 2012. 507−525. [doi: 10.1007/978-3-642-31284-7_30]

[31] Wang B, Li B, Li H. Oruta: Privacy-Preserving public auditing for shared data in the cloud. IEEE Trans. on Cloud Computing, 2014,2(1):43−56. [doi: 10.1109/TCC.2014.2299807]

[32] Wang B, Li B, Li H. Panda: Public auditing for shared data with efficient user revocation in the cloud. IEEE Trans. on Services Computing, 2015,8(1):92−106. [doi: 10.1109/TSC.2013.2295611]

[33] Wang Y, Wu Q, Qin B, Chen XF, Huang XY, Zhou YY. Group-Oriented proofs of storage. In: Proc. of the 10th ACM Symp. on Information, Computer and Communications Security. ACM Press, 2015. 73−84. [doi: 10.1145/2714576.2714630]

[34] Douceur JR, Adya A, Bolosky WJ, Simon D, Theimer M, Research M. Reclaiming space from duplicate files in a serverless distributed file system. In: Proc. of the Int'l Conf. on Distributed Computing Systems. 2002. 617−624. [doi: 10.1109/ICDCS.2002.1022312]

[35] Storer MW, Greenan K, Long DE, Miller EL. Secure data deduplication. In: Proc. of the 4th ACM Int'l Workshop on Storage Security and Survivability. ACM Press, 2008. 1−10. [doi: 10.1145/1456469.1456471]

[36] Harnik D, Pinkas B, Shulman-Peleg A. Side channels in cloud services: Deduplication in cloud storage. Security & Privacy, IEEE, 2010,8(6):40−47. [doi: 10.1109/MSP.2010.187]

[37] Halevi S, Harnik D, Pinkas B, Shulman-Peleg A. Proofs of ownership in remote storage systems. In: Proc. of the ACM Conf. on Computer and Communications Security. 2011. 491−500. [doi: 10.1145/2046707.2046765]

[38] Zheng Q, Xu S. Secure and efficient proof of storage with deduplication. In: Proc. of the 2nd ACM Conf. on Data and Application Security and Privacy. CODASPY, 2012. 1−12. [doi: 10.1145/2133601.2133603]

[39] Fehr S, Hofheinz D, Kiltz E, Wee H. Encryption schemes secure against chosen-ciphertext selective opening attacks. LNCS, 2010, 6110:381−402. [doi: 10.1007/978-3-642-13190-5_20]

[40] Agrawal S, Dan B. Homomorphic MACs: MAC-Based integrity for network coding. In: Proc. of the ACNS. 2009. 292−305. [doi: 10.1007/978-3-642-01957-9_18]

附中文参考文献:

[37] 谭霜,贾焰,韩伟红.云存储中的数据完整性证明研究及进展.计算机学报,2015,38(1):164−177.

[38] 俞能海,郝卓,徐甲甲,张卫明,张弛.云安全研究进展综述.电子学报,2013,41(2):371−381.

[40] 彤丽,栗磊,李超零.一种可扩展的动态数据持有性证明方案.计算机应用研究,2013,30(7):2132−2135.

王宏远(1988－),女,山东潍坊人,博士生,CCF 学生会员,主要研究领域为云计算安全,数据持有性证明.

李龙一佳(1991－),男,硕士,主要研究领域为云计算安全,大数据安全.

祝烈煌(1976－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为安全协议分析,云计算安全,无线传感网安全.