基于区块链的公平可验证数据持有方案

唐 飞[1,2],冯 卓[1],黄永洪[2]
(1. 重庆邮电大学计算机科学与技术学院,重庆 400065; 2. 重庆邮电大学网络空间安全与信息法学院,重庆 400065)

摘 要: 针对传统可证明数据持有(Provable Data Possession,PDP)方案中要求客户端是诚实的这一问题,基于区块链技术提出了公平的可证明数据持有方案. 在传统PDP方案中,总是假定服务器是半诚实而客户端是可信的,这对服务器而言是不公平的. 在基于区块链的公平PDP方案中,用于检验的元数据不再由客户端生成,而是由区块链节点生成并对其达成共识. 因此,借助区块链的分布式信任性质可以实现PDP方案的互信机制,保证客户端和云服务器之间的公平性. 同时,利用哈希函数、Pedersen承诺实现高效的公平PDP方案. 分析所提方案的安全性、计算开销、通信开销以及冗余率. 分析结果表明,在保障安全性的基础上,所提方案比同类方案具有更优的计算开销、通信开销及冗余率.

关键词: 可证明数据持有方案;互信机制;公平性;区块链
基金项目: 国家重点研发计划基金(No.2018YFB0803905);国家自然科学基金(No.61702067);重庆市自然科学基金(cstc2017jcyjAX0201,cstc2020jcyj-msxmX0343)
中图分类号: TP309 **文献标识码:** A **文章编号:** 0372-2112(2023)02-0406-10
电子学报URL:http://www.ejournal.org.cn **DOI:**10.12263/DZXB.20210161

Fair Provable Data Possession Scheme Based on Blockchain

TANG Fei[1,2], FENG Zhuo[1], HUANG Yong-Hong[2]
(1. *Department of Computer Science and Technology*, *Chongqing University of Posts and Telecommunications*, *Chongqing*, 400065, *China*;
2. *Department of Cyber Security and Information Law*, *Chongqing University of Posts and Telecommunications*, *Chongqing*, 400065, *China*)

Abstract: In order to solve the problem that the client is required to be honest in traditional provable data possession (PDP) schemes, we propose a fair provable data possession scheme based on blockchain. The traditional PDP schemes always assume that the cloud server is semi-honest but the client is trusted, which is unfair to the cloud server. In our scheme, the metadata used for verification is no longer generated by the client but by the blockchain node. A consensus on metadata is reached by the nodes in the blockchain. Therefore, based on the property of distributed trust of the blockchain, we can realize the mutual trust mechanism of the PDP scheme, which can ensure the fairness between the client and the cloud server. We firstly use hash function and Pedersen Commitment to construct an efficient fair PDP scheme. Then, we analyze the security, computing overhead, communication overhead and redundancy of our proposed scheme. The results show that this scheme is not only secure, but also superior to similar schemes in both the cost and redundancy.

Key words: provable data possession scheme; mutual trust mechanism; fairness; blockchain
Foundation Item(s): National Key Research and Development Program Fund (No. 2018YFB0803905); National Natural Science Foundation of China (No.61702067); Chongqing Natural Science Foundation of China (No.cstc2017jcyjAX0201, No.cstc2020jcyj-msxmX0343)

1 引言

近年来,云计算得到了极大的发展,被认为是继大型计算机、个人计算机和互联网之后的第四次信息技术革命[1]. 云计算的核心概念是管理和调度连接到网络的大量计算资源,以形成计算资源池,为用户提供服务. 云存储是云计算的基础,使用户可以将其数据存储在云服务器上,以节省存储成本和简化复杂的管理任务. 此外,云存储还有助于用户从不同终端灵活访问其存储的数据.

云服务器一般被认为是半诚实的,用户需要进行完整性校验以保证所存数据未被恶意篡改或删除. 早

收稿日期:2021-01-24;修回日期:2021-06-24;责任编辑:王天慧

在2004年,Deswarte等人[2]就提出了如何检验不可信服务器上文件的完整性问题,他们基于哈希函数和Diffie-Hellman密钥交换协议构造了数据完整性校验方案.随后,Ateniese等人[3]首次形式化定义了可证明数据持有(Provable Data Possession,PDP)方案的概念,通过"挑战-应答"交互协议的方式结合概率抽样方法实现云存储数据的快速完整性验证.Sebé等人[4]提出了PDP方案应满足的要求,具体包括:(1)验证者只需要一个摘要即可完成验证;(2)存在内部和外部攻击的情况下,协议能够保持安全;(3)验证者和证明者之间权利平等;(4)通信开销低;(5)可无限次验证.此后,Ateniese等人[5]提出了支持数据更新和删除操作的动态PDP方案.为了实现数据插入操作,Erway等人[6]提出了一种基于认证跳表结构的全动态PDP方案.此后,研究者提出了许多具有不同性质的PDP方案,如文献[7]提出的隐私保护PDP方案、文献[8]提出的代理PDP方案、文献[9]提出的两方协作PDP方案等.研究者们还根据Ateniese等人[3]的方案做了许多后续工作,例如文献[10~13]等.

传统PDP方案中,客户端和云服务器始终是对立关系,这导致无论是客户端还是云服务器充当验证者都无法使对方信服.针对这一问题,研究者们引入了第三方验证者(Third Party Auditor,TPA)来辅助验证操作.例如,Wang等人[14]基于Merkle树提出一种全动态公开验证的PDP方案;杜瑞忠等人[15]基于DDCT表提出一种多副本公开验证的PDP方案.TPA的引入一定程度上增加了验证结果的公正性和可信度,但用于完整性校验的元数据依然由客户端生成并上传.因此,如果客户端上传错误的元数据,则会造成云服务器始终无法通过验证.针对这一问题,Shen等人[16]于2017年提出一种可公开审计的PDP方案.他们设计出一种服务于云服务器的机制.该机制要求云服务器在客户端上传元数据至TPA时都必须对TPA上的所有元数据进行验证,而这一步骤会导致验证效率低下.为了解决上述问题,本文致力于寻找一种更为合适的第三方验证机制,在保证高效验证的基础上实现客户端和云服务器的公平性与互信性.

随着区块链的快速发展,研究者们借助区块链的性质提出了基于区块链的PDP方案.例如,Wang等人[17]于2019年提出了一种基于区块链的私有验证PDP方案.他们利用区块链对元数据进行分布式存储,从而保证元数据的不可篡改性.Wang等人[18]提出了一个用于公开云存储审计的基于区块链的公平支付智能合约,避免出现客户端先支付云服务器后服务的情况.2020年,Li等人[19]提出了一种支持基于区块链的PDP分布式存储框架,解决了传统PDP方案的中心化存储所面临单点故障的潜在威胁.Chen等人[20]提出了基于区块链的智能城市动态PDP方案,实现了外包数据的动态操作.

现有PDP方案中,元数据均由客户端生成.一旦客户端恶意生成错误的元数据,云服务器就不可能通过验证,这会导致对云服务器不公平的问题.这一问题即便在TPA协助验证或区块链存储元数据等PDP方案中依然存在.针对该问题,本文提出一种基于区块链的公平PDP方案.在本方案中,用于完整性校验的元数据不再由客户端生成而是由区块链生成并分布式存储,从而防止客户端的恶意行为.基于区块链的分布式信任等性质实现PDP方案中客户端和云服务器之间的互信机制,进而实现数据完整性校验过程的公平性.本文的主要研究工作如下.

(1)首先,提出基于区块链的公平PDP方案的概念.与传统PDP方案的不同之处在于元数据的生成过程在区块链中完成而不是客户端自行产生.当验证不通过时,区块链可以通过搜集证据来判定是云服务器还是客户端作弊.方案提供了一种责任认定机制,可以同时保护客户端和云服务器的权益,实现PDP方案的公平性.

(2)其次,现有的大多数PDP方案需要大量的模幂、双线性配对等运算,效率偏低.为了提高PDP方案的可行性,本文主要采用哈希函数和Perdersen承诺,提高了验证效率.

(3)最后,分析了所提方案的安全性、计算开销、通信开销和冗余率等,并与相关方案做比较.

2　预备知识

2.1　区块链

区块链本质上来说是一个分布式的数据库[21],采用链表数据结构.数据区块由区块头和区块体组成,所有区块根据哈希值形成链式结构,从而实现不可篡改性.区块链的一般结构如图1所示,区块头包含前一个区块的哈希值Pre_Hash、指示该块生成时间的时间戳Timestamp、区块数据的Merkle根$Hash_{Root}$以及随机值Nonce等.区块体由当前区块中的所有交易构成.通过共识机制来实现共识节点数据的一致性,如工作量证明(Proof of Work,PoW)机制[21]、实用拜占庭容错(Practical Byzantine Fault Tolerance,PBFT)机制[22]等.正是由于区块链的这些特殊结构,区块链拥有去中心化、不可篡改性、分布式存储等特性.其中,去中心化特性可以解决各参与方权利不平等和互不信任的问题,保证区块链系统中的各参与方的公平性;不可篡改性和分布式存储特性可以解决数据在传输和流转过程中所有权的界定问题以及节点故障导致的数据丢失问题,保

证数据在区块链系统中的安全性.

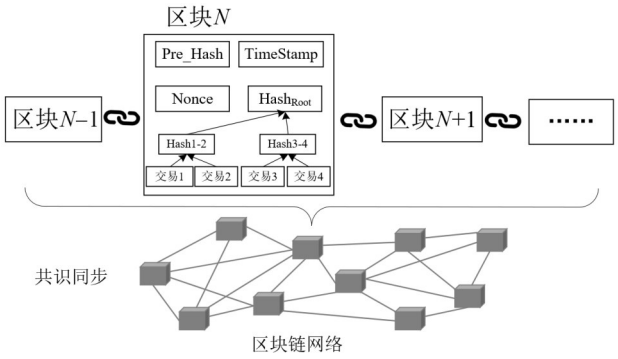

图1 区块链结构图

按照参与准入机制,区块链可分为公有链、私有链和联盟链. 其中,联盟链更适用于现实应用场景. 相比于私有链的运作空间和效率,联盟链的价值更大;而相比于公有链那种完全去中心化的不可控和隐私安全问题,联盟链更灵活,也具有更高的可控性.

尽管区块链存在着低效率、高能耗、高时延等通用问题,但相比于传统的可信中心又具有防单点故障、数据防篡改、分布式信任等优势. 目前,区块链技术已被广泛用于电子健康记录[23]、信用支付[24]、身份认证[25]等.

2.2 智能合约

智能合约是一种执行合约条款的计算机交易协议,无需第三方即可以数字方式实现可信交易,这些交易可追踪且不可逆转. 智能合约具有三个独特的性质,即自治、权力下放和自给自足[26]. 自治表示将合约部署在区块链上后,不再需要智能合约的创建者. 权力下放意味着在去中心化环境中,合约是在P2P网络的对等点之间分布并自行执行的. 自给自足表示智能合约能够通过提供服务来获取资金,当其需要花费资金时,就可以使用这些资金.

2.3 Pedersen承诺

Pedersen承诺方案[27]基于离散对数困难问题,具有无条件隐藏和计算绑定性质. 方案包括承诺者和验证者两个角色,具体步骤如下.

(1)系统建立:取一个大素数p且$p>2^{160}$. 取任意的$a,b\in Z_p$ 且 $4a^3+27b^2\neq 0$. 定义 $E_p(a,b)$: $y^2\equiv x^3+ax+b(\mathrm{mod}\,p)$,此方程的所有点集构成一个椭圆曲线. 受信任方选择椭圆曲线上的两个点R和S,其中$S=kR$(k是秘密值). 随后公布(p,a,b,R,S).

(2)承诺:承诺者随机选择$r\in Z_p$来对$x\in Z_p$做出承诺,计算

$$C=C(x,r)=xR+rS\in E_p(a,b) \tag{1}$$

(3)打开:为了打开承诺,承诺者公布x和r,验证者验证等式$C=xR+rS$是否成立即可.

3 定义

3.1 方案模型

基于区块链的公平PDP方案的模型如图2所示. 方案包括三个角色:客户端、云服务器和区块链系统,具体如下.

(1)客户端(client):数据拥有者,他们希望将数据上传到云服务器进行外包存储,但这会使他们失去对数据的物理控制. 为了保证外包数据的安全,本文采用了数据完整性校验机制. 同时,为了实现整个数据完整性校验的真正公平,客户端将元数据生成交给区块链来完成. 客户端在外包存储之前可以在区块链上通过智能合约的形式发布一笔交易来匹配适当的节点帮其生成元数据,切断客户端和元数据的必然生成关系. 当校验失败时,客户端可以申请仲裁,发起赔偿.

(2)云服务器(cloud server):提供存储服务. 同时,云服务器还负责维护外包文件的完整性. 客户端会不定时发起对外包数据的完整性校验,而云服务器则需要提供证明作为回应. 为了体现整个校验的公平性,当完整性校验失败时,云服务器也可向区块链上传证据来证明其自身并无作弊行为.

(3)区块链(blockchain):它是处在客户端和云服务器之间的可信第三方. 为了方案的公平性,区块链在元数据生成阶段发挥作用,将使客户端无法伪造元数据. 同时,在仲裁阶段,由于区块链的不可篡改和分布式特性,区块链可以公平、公正、公开地处理来自客户端的仲裁请求.

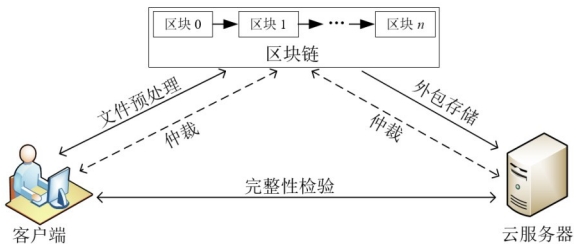

图2 公平PDP方案模型图

从上图中可以看出,文件在整个方案模型中是单向流通的,由客户端上传给区块链,再由区块链经过一系列处理后发送给云服务器. 而元数据的生成则是由区块链来完成,并且生成的元数据会经过一些操作后在区块链系统中达成共识. 这样既保证了元数据的正确性又保证了元数据的机密性. 此外,客户端和区块链之间的交互过程只有文件上传、接收元数据和仲裁. 相比于现有的工作,本方案借助区块链来计算用于完整性验证的元数据和充当第三方仲裁,从而实现整个PDP方案的公平性.

3.2 方案定义

定义1 传统的PDP方案包括系统建立、存储、挑战、证明生成和证明检查五个算法. 在基于区块链的公平PDP方案中,新增一个仲裁协议,用于判定作弊方. 方案具体定义如下.

(1)系统建立(setup):输入安全参数λ,建立区块链系统,并公开系统参数param.

(2)存储(store):输入文件F,输出元数据和承诺值. 该算法包括三部分:Store. node_sele决定出区块链中负责处理文件的节点;Store. file_pre生成元数据并将其返回给客户端;Store. upload将文件F上传到云服务器.

(3)挑战(challenge):客户端从元数据表任意选择一个值当作输入,输出挑战值chal,并将其发送给云服务器.

(4)证明生成(genproof):云服务器输入文件F和挑战值chal,输出一个证明P,并将其返回给客户端.

(5)证明验证(checkproof):客户端输入元数据、挑战值chal和证明P,如果验证通过,则输出1,否则,输出0.

(6)仲裁(arbitration):当验证不通过时,客户端可以向区块链发起仲裁请求. 然后,区块链要求云服务器上传证据. 最后,区块链判定到底是客户端还是云服务器的责任.

与传统PDP方案类似,基于区块链的公平PDP方案也需要满足正确性. 如果客户端和云服务器均诚实执行方案过程,则证明验证一定通过. 此外还要求,如果验证不通过,则区块链系统一定可以正确判定责任方.

3.3 安全性需求

公平的PDP方案需要满足以下安全需求.

抗伪造攻击. 在数据完整性校验的过程中,云服务器可能会为了更多的利益而破坏外包数据的完整性,而在进行数据完整性校验的时候,云服务器可能会伪造数据来欺骗验证者. 这要求本方案必须能够准确地发现云服务器是否存在作弊现象,这也是PDP方案的核心需求. 简言之,如果云服务器中的外包数据丢失或损坏,云服务器不可能通过任何方式来完成数据完整性校验.

抗重放攻击. 在数据完整性校验过程中,云服务器可能会使用前一个挑战对应生成的证明来冒充当前这个挑战对应的证明,并想以此来通过数据完整性校验. 这需要本方案能够去甄别该证明是否重复使用,以防止云服务器作弊成功.

抗合谋攻击. 在数据完整性校验的过程中,客户端可能会联合区块链进行合谋攻击,通过上传不正确的数据使云服务器返回的证明永远无法通过验证,从而让客户端获得欺诈性赔偿. 这需要本方案能够保证客户端上传到区块链的数据是正确的,使云服务器返回证明始终能够通过验证,除非云服务器返回错误的证明.

公平性. 在数据完整性校验的过程中,客户端和云服务器都有可能会为了利益作出恶意的行为,但是也有可能出现单方面的作弊行为,这就需要本方案既能发现作弊的一方也能保证另外一方的利益. 简言之,当外包数据是完整时,客户端无法骗取云服务器的赔偿. 反之,当外包数据不完整时,云服务器必然要赔偿客户端.

3.4 可行性需求

公平的PDP方案需要满足以下可行性需求.

(1)计算开销

计算开销主要存在于存储阶段中元数据与承诺值的生成、证明生成阶段中证明的生成以及证明验证过程中证明的验证. 因此,节省方案的计算开销可提高其执行效率.

(2)通信开销

通信开销主要存在于客户端、区块链、服务器三者之间的信息交互过程中. 某一环节的信息传递过慢就会造成整个方案运行的效率偏低,因此,需要控制通信开销,以增加方案的运行流畅.

(3)冗余率

假设上传的文件为F,其长度表示为$|F|$. 为了实现数据的远程完整性检验,需要生成用于验证的元数据MetaData,其长度为|MetaData|. 冗余率定义为$r=(|F|+|\text{MetaData}|)/|F|$. 从定义可以看出元数据越大,则冗余率越大,通信开销也越大. 相应地,冗余率越大导致整个系统的吞吐量越大,资源消耗也越多,例如区块链系统.

4 基于区块链的公平PDP方案

4.1 方案构造

基于区块链的公平PDP方案模型如图3所示. 验证部分包括五个阶段:系统建立、存储、挑战、证明生成和证明检查. 详细步骤如下.

步骤1 客户端首先发起一个文件预处理请求(交易),并发送给区块链.

步骤2 区块链中的节点收到这笔交易时,由区块链中的竞争机制决胜出一个获胜节点,并由获胜节点与客户端通信.

步骤3 客户端将文件发送给获胜节点,并让获胜节点对文件进行预处理. 同时,客户端在本地删除文件.

步骤4 获胜节点生成元数据并将其返回给客户端. 同时,将文件发送到云服务器.

步骤5 云服务器在收到来自获胜节点的文件之后,先确定文件真实性,再返回给获胜节点一个存储成功的值.

步骤6 客户端向云服务器发送挑战.

步骤7 云服务器收到客户端的挑战后,使用存储的文件生成证明并返回给客户端.

步骤8 客户端核实证明是否真实有效.

仲裁部分详细步骤如下:

步骤9 如果验证失败,则客户端向区块链发起仲裁请求以启动赔偿机制.

步骤10 云服务器将证据上传到区块链.

步骤11 区块链将责任认定的结果发送到客户端和云服务器. 确定责任后,客户端和云服务器之间必须进行补偿.

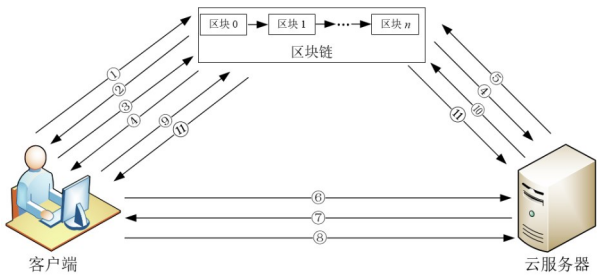

图3 公平PDP方案流程图

存储的文件表示为F,可以是源数据,也可以是源数据的密文,具体取决于客户端. 一般地,F应是加密后的文件. 本方案中的元数据生成过程和仲裁过程都是在区块链中进行的,而联盟链中的责任主体不仅可以胜任这些工作而且可以保护客户端的数据不被泄漏. 同时,由于联盟链中节点少,交易速度较快,交易成本低,因此,本文采用联盟链作为底层区块链系统. 此外,客户端还应创建一个集合L,初始化为ϕ. 系统初始化过程具体如下.

方案由五个阶段和一个协议组成,它们的详细描述如下.

(1)系统建立(setup). 取一个大素数p且$p>2^{160}$. 定义一条椭圆曲线E_p. 取椭圆曲线E_p上的点R和S. 定义一个Pedersen承诺函数:承诺者可随机选择$r\in Z_p$来对$x\in Z_p$做出承诺,形如

$$C(x,r)=xR+rS \tag{2}$$

定义一个安全密码哈希函数即$H:\{0,1\}^*\to Z_p^*$. 定义一个伪随机函数f和一个伪随机置换π即

$$f:Z_p^*\times\{1,2,\cdots,n\}\to Z_p^* \tag{3}$$

$$\pi:Z_p^*\times\{1,2,\cdots,n\}\to\{1,2,\cdots,n\} \tag{4}$$

最后,公开系统参数param$=\{H,C,f,\pi\}$,并将公开参数放到智能合约中,当客户端想使用参数来完成一些操作时,可以直接调用智能合约.

(2)存储(store). 包含步骤1~步骤5. 主要包含Store.node_sele,Store.file_pre和Store.upload三个子阶段.

Store.node_sele:决定出参与文件预处理的获胜节点. 客户端首先发起一个交易请求Re,交易结构如图4所示. 其次,客户端调用算法1节点竞争(选择)合约算法来决出区块链中有富余计算能力的节点来完成文件预处理的工作.

Size 文件大小	Type 文件类型	H_F 文件哈希	Reward 奖励	T_1 发起时间

图4 交易结构图

算法1 节点竞争(选择)合约算法

输入: 交易请求Re

输出: 交易反馈$\sigma_{\mathrm{Tx(rent)}}$

//客户端发起租赁请求

1 Client submits a rent Re(Size, Type, H_F, T_1)

//通过PBFT决出获胜节点

2 Host$\leftarrow$PBFT(Size, Type, H_F, T_1)

//生成交易信息

3 Transaction Tx(rent)$\leftarrow$(Client, Host, Size, Type, H_F, T_1)

4 Payment(Client, Reward) //客户端支付奖励

5 $\sigma_{\mathrm{Tx(rent)}}\leftarrow\mathrm{Sign}_{\mathrm{Host}}(\mathrm{Tx(rent)})$ //节点共识交易信息

Store.file_pre:在决出获胜节点之后,客户端与获胜节点之间通信,并通过调用算法2元数据生成合约算法来完成元数据生成的工作.

算法2 元数据生成合约算法

输入:文件F

输出:元数据MetaData

1: **if**$H(F)==H_F$**then** //判断客户端的文件是否与交易中的哈希值对应

2: 确定为文件挑选n个随机值

3: **for**$(1\leqslant i\leqslant n)$ **do**

4: $r_i=f(i)$ //生成随机值

5: $h_i=H(F\|r_i)$ //计算哈希值

6: $C_i=C(h_i,r_i)=h_iR+r_iS$ //计算承诺值

7: **end for**

8: **else quit**

9: **end if**

10: MetaData$=(r_i,h_i)_{i\in[1,n]}$

11: **return** MetaData //将元数据返回给客户端

12: **delete** MetaData //节点删除元数据

13: $\sigma_{C_i}\leftarrow\mathrm{Sign}_{\mathrm{Host}}(C_{i,i\in[1,n]})$ //节点共识承诺值

Store.upload:获胜节点在完成上述操作后,将文件F上传到云服务器,然后删除本地文件F. 云服务器收到文件F之后,先对其做哈希运算得到H'_F,再对比链上的H_F. 若两者相等,则云服务器返回给获胜节点一个存储成功的值,表明文件已成功存储在云服务器中. 获胜节点在收到这个值之后,对其达成共识. 随后,客户端则可以通过查看链上数据来知晓云服务器是否成功存储文件.

(3)挑战(challenge). 包含步骤6. 在完成了文件外包存储之后,客户端开始进行挑战阶段. 算法3是挑战生成算法,将挑战发起请求作为输入,挑战值作为输出. 最后,将挑战值发送给云服务器.

算法3　挑战生成算法

输入:请求 request

输出:挑战值 chal

1: Client initiates a *request* to CloudServer
2: $j=\pi(c)$　　//c代表客户端发起的第c次挑战,j代表随机值的索引
3: **if** $r_j \notin L$ **then**
4:　$L \leftarrow r_j$　　//将随机值写进集合L中
5:　chal$=(c, r_j)$　　//生成挑战值
6:　$\sigma_{\text{chal}} \leftarrow \text{Sign}_{\text{Client}}(\text{chal})$　　//将挑战值上链
7:　**return** chal　　//将挑战值返回给云服务器
8: **else jump** 2
9: **end if**

(4)证明生成(genproof). 包含步骤7. 云服务器收到了来自客户端的挑战之后,根据自身本地存储的文件来生成对应的证明. 算法4是证明生成算法,将挑战值作为输入,证明作为输出. 最后,将证明返回给客户端.

算法4　证明生成算法

输入:挑战值 chal

输出:证明P

1: $h'_j=H(F\|r_j)$　　//用于最终完整性检验的哈希值
2: evidence$=(r_j, h'_j)$　　//将(r_j, h'_j)临时保存,以便随后仲裁阶段使用
3: $P=(c, r_j, h'_j)$　　//生成证明
4: **return** P　　//将证明返回给客户端

(5)证明验证(checkproof). 包含步骤8. 客户端收到了云服务器的证明之后,需要验证其正确性. 如果验证通过则说明文件完整,反之则说明文件不完整或客户端恶意验证. 算法5是证明验证算法,将证明作为输入,输出为0/1. 若验证不通过,客户端告诉云服务器结果,并向给区块链提起仲裁申请.

(6)仲裁(arbitration). 包含步骤9~步骤11区块链收到请求后,向云服务器索要证据. 最后,根据自身所拥有的参数来完成客户端和云服务器之间的仲裁阶段. 区块链通过调用算法6仲裁合约来判定是客户端还是云服务器的责任.

算法5　证明验证算法

输入:证明P

输出:成功/失败 1/0

1: **if** $h_j == h'_j$ **then**　　//判断哈希值是否相同
2:　**return** 1
3: **else return** 0
4:　request$'=(c, j, r_j, h_j)$　　//客户端向区块链系统申请仲裁
5: **end if**

算法6　仲裁合约算法

输入:请求 request′、证据 evidence

输出:客户端/云服务器 1/0　　//1代表客户端,0代表服务器

1: **if** evidence.r_j == request′.r_j **then**　　//防重放攻击
2:　$C'_j=C(h'_j, r_j)=h'_jR+r_jS$　　//通过云服务器的证据生成承诺值
3:　**if** $C_j == C'_j$ **then**　　//判断承诺值是否相同
4:　　**return** 1
5:　**else return** 0
6:　**end if**
7: **else return** 0
8: **end if**

正确性分析如下. 假设客户端需要委托云服务器保存的数据为F,且未将此文件进行分块处理. 在预处理阶段,让n个随机值依次与文件F进行哈希运算得到相应的n个哈希值$h_i=H(F\|r_i)_{i\in[1,n]}$. 当客户端想要验证文件$F$是否完整时,用户任意选择一个随机值$r_i$发送给云服务器,云服务器收到挑战后,计算$h'_i=H(F\|r_i)$并返回给客户端. 客户端则可以通过验证该等式$h'_i=H(F\|r_i)=h_i$来证明该算法的正确性.

4.2　安全性分析

在其他PDP方案的基础上,本方案还重点关注了公平性. 在Shen等人[16]的工作中,客户端与TPA存在频繁交互,容易导致客户端和TPA发起合谋攻击,无法保证方案的公平性. 此外,Wang等人[17]和Chen等人[20]的方案利用区块链的特性实现PDP方案的正确性和不可伪造性,但都不能确定元数据的真实性,同样也无法确保方案的公平性. 本方案则可以利用区块链的特性来实现PDP方案的公平性.

(1)抗伪造攻击

客户端每次挑战的随机值都是从本地元数据表中任意抽取,这个随机值的样本空间大小不可能让云服务器能一次性猜出客户端选择的挑战值. 同时,每个随机值之间没有必然联系,更不可能让云服务器在经过多轮验证之后,猜测出下一次的挑战值. 在云服务器无法猜测挑战值时,它就更不可能伪造出能够通过数据完整性校验的证明. 因此,本方案可以抵御来自云服务

器的伪造攻击.

(2)抗重放攻击

从方案模型来分析,当云服务器发起重放攻击之后,其返回的证明肯定无法通过数据完整性验证. 客户端一定会向区块链系统发起仲裁请求,让区块链来判断云服务器是否存在作弊行为. 当云服务器将自己之前产生的证明当作证据上传至区块链之后,肯定可以打开对应的Pedersen承诺,从而证明自身并没有作弊. 但是在挑战生成阶段,本文将产生的挑战上链,同时在使用云服务器上传的证据之前要先进行证据筛查,如果不匹配则可以直接说明云服务器存在作弊行为. 因此,本方案可以抵御来自云服务器的重放攻击.

(3)抗合谋攻击

对整个数据完整性校验过程分析可以得到,首先,在元数据生成阶段,客户端只能与区块链中的一个节点进行交互通信,所以客户端不可能与整个区块链进行合谋攻击. 其次,当客户端发起交易的时候,交易中包含着文件大小、类型、Hash值和时间戳. 这些内容可以防止客户端随后上传的实际文件与它准备上传的文件不匹配的情况. 这也可以保证客户端不与获胜节点进行合谋攻击. 最后,每当获胜节点生成一对元数据的时候就会进行一次Pedersen承诺,并将承诺值上链,这也保证了客户端无法在后续的步骤中与获胜节点进行合谋攻击. 综上所述,本方案可以抵御来自客户端的合谋攻击.

(4)公平性

从防止作弊方面分析,本方案用区块链来代替客户端生成元数据,使元数据生成过程与客户端分离,再用Pedersen承诺将元数据绑定并共识上链. 既能防止客户端在源头发起作弊又能避免客户端发起恶意数据完整性检验. 本方案还可以通过选取随机值的范围空间来防止服务器猜到随机值而作弊. 这极大地保证了数据完整性检验中两方的约束公平. 此外,除防止作弊外,还应考虑从处罚方面分析. 如果云服务器确实产生了作弊行为,那么它必须支付给客户端一定的赔偿金;同样地,如果客户端也产生了恶意骗取赔偿金的作弊行为,那么它必须支付给云服务器一定的处罚金. 客户端可以通过向区块链发起仲裁请求来申请赔偿金,在区块链采集了两方上传的证据之后就可以判断出到底是客户端还是云服务器在产生作弊行为. 这样既保证了客户端的权利又保证了服务器的利益,从而实现了数据完整性检验中两方的收益公平.

4.3 效率分析

对整个数据完整性检验过程进行效率分析,主要分为计算开销、通信开销和冗余率,并将本方案与其他现有的基于区块链的PDP方案相比较. 本方案在元数据生成的时候并不存在分块操作,这有区别于现有的PDP方案. 因此,只能对方案的各个流程进行对比分析.

(1)计算开销

由于本方案需要产生n对元数据来做数据完整性检验,且文件预处理与客户端分离,本文定义哈希计算开销为C_{hash},Pedersen承诺计算开销为C_{commit}. 在存储阶段,计算开销则是$nC_{\text{hash}}+nC_{\text{commit}}$,取$n=1000$. 在挑战阶段和证明检查阶段,计算开销几乎可以忽略不计,因为不涉及分块操作. 在证明生成阶段,计算开销为C_{hash}.

为了展现出本方案的计算优势,本文对比了其他两种基于区块链的PDP方案:文献[17]和文献[20]. 文献[17]基于RSA问题,将文件分成$n_1=|F|/1024$块,挑战块为c个. 定义模N的指数计算开销为C_{exp},模余计算开销为C_{mod},大整数相乘计算开销为C_{Bmul}. 在块标签生成阶段,计算开销为$(n_1+1)C_{\text{exp}}+n_1C_{\text{mod}}$. 在挑战阶段,计算开销可忽略不计,因为没有指数运算. 在证明生成阶段,计算开销为cC_{Bmul}. 在证明检验阶段,计算开销为$(c+1)C_{\text{exp}}+C_{\text{mod}}$. 文献[20]基于双线性映射,将文件分成$n_2=|F|/160$块,挑战块为$c$个. 定义双线性配对计算开销为$C_{\text{pair}}$,定义群上的乘法计算开销为$C_{\text{mul}}$. 在块标签生成阶段,计算开销为$n_2(s+1)C_{\text{exp}}+n_2sC_{\text{mul}}$. 挑战阶段计算开销忽略不计. 在证明生成阶段,计算开销为$cC_{\text{exp}}+(c-1)C_{\text{mul}}$. 在证明检验阶段,计算开销为$2C_{\text{pair}}+(c+s+1)C_{\text{exp}}+(c+s)C_{\text{mul}}$. 本文对方案的计算开销进行测试. 实验采用的是物理机和本地的虚拟机相结合的方式来进行计算开销的测试. 物理机测试平台为HUAWEI MateBook 14,操作系统为Windows 10,处理器为Intel i7-8565 U,主频为1.8 GHz,内存为8 GB;本地虚拟机操作系统为Ubuntu Kylin18.04.4 LTS. 测试代码由基于Java对的密码库(JPBC)[28]和基于Python编程语言实现的代码所提供. 其中,客户端运行在物理机上,而区块链系统和云服务器运行在虚拟机上. 最终可以得到如下七个结果:$C_{\text{exp}}=27.388$ ms,$C_{\text{pair}}=21.982\,7$ ms,$C_{\text{mod}}=0.298$ ms,$C_{\text{mul}}=0.243\,2$ ms,$C_{\text{hash}}=573$ ms,$C_{\text{Bmul}}=5.347\,5$ ms,$C_{\text{commit}}=0.012\,8$ ms. 以存储的文件$|F|=1Gb$为例,则$n_1=1.073\,7\times10^6$,$n_2=6.871\,9\times10^6$,$c=3.9728\times10^4$,s记为1. 本文制作了表1以更直观地进行比较.

通过上表的对比可以发现,本方案在计算开销方面远远要小于其他两种方案,主要原因在于本方案没有复杂的模幂运算. 本文方案的证明检验阶段只涉及一次简单的哈希值比对,因此时间消耗可忽略不计.

区块链在计算元数据的过程中,需要运行共识算

表1　计算开销对比

方案	元数据生成	证明生成	证明检验
文献[17]	$(n_1+1)C_{\exp}+n_1C_{\mathrm{mod}}$ $=2.972\,6\times10^6$ ms	cC_{Bmul} $=2.124\,5\times10^5$ ms	$(c+1)C_{\exp}+C_{\mathrm{mod}}$ $=1.088\,1\times10^6$ ms
文献[20]	$n_2(s+1)C_{\exp}+n_2sC_{\mathrm{mul}}$ $=3.780\,9\times10^8$ ms	$cC_{\exp}+(c-1)C_{\mathrm{mul}}$ $=1.097\,7\times10^6$ ms	$2C_{\mathrm{pair}}+(c+s)C_{\mathrm{mul}}$ $+(c+s+1)C_{\exp}$ $=1.097\,8\times10^6$ ms
本文方案	$nC_{\mathrm{hash}}+nC_{\mathrm{commit}}$ $=5.730\,1\times10^5$ ms	C_{hash} $=753$ ms	忽略不计

法,这个共识时间取决于具体所选的区块链系统. 本文参照文献[17]、文献[20]的方式,只对所提方案进行实验验证分析其效率,不考虑区块链本身的执行开销.

(2)通信开销

一般地,哈希值长度取决于所选大素数p的长度λ_p. 算法安全性与λ_p的值成正比,与时间开销成反比,所以λ_p不宜太小. 在本方案中,λ_p取1024位,随机值的长度选择为256 bit,椭圆曲线中点的长度选择为160 bit.

在存储阶段,客户端发送给区块链$|F|$位数据,区块链在文件预处理之后返回给客户端(1024+256)n位元数据以及发送给服务器$|F|$位数据. 在挑战阶段,客户端发送给服务器256 bit数据. 在证明生成阶段,服务器发送给客户端1 024 bit数据. 在证明检查阶段,通信开销可忽略不计.

为了展现出本方案的通信优势,本文对比了其他两种基于区块链的PDP方案:文献[17]和文献[20]. 对于文献[17],在块标签生成阶段,客户端发送给服务器$|F|+|F_H|$ bit数据. 同时,客户端发送给区块链1 024n_1+$|F_H|$ bit数据. 在挑战阶段,客户端发送给服务器2 048+$|n_1|$ bit数据. 在证明生成阶段,服务器发送给客户端1 023+$|c|+|F_i|$ bit数据. 在证明检查阶段,区块链发送给客户端1 024c bit数据. 对于文献[20],在块标签生成阶段,客户端发送给服务器160c+1 024(cn_2+1)位数据. 在挑战阶段,客户端发送给服务器$l\log_2 p$位数据. 在证明生成阶段,服务器发送给客户端1 024(lc+1)+160s bit数据. 在证明检查阶段,通信开销可忽略不计. 同(1),以$|F|$=1 GB文件为例,取l=10,$\log_2 p$=256. 本文制作了表2以更直观地进行比较.

表2　通信开销对比

方案	元数据生成	挑战	证明生成
文献[17]	$1\,024n_1+\|F\|+2\|F_H\|$ $=2.147\,5\times10^9$ bit	$2\,048+\|n_1\|$ $=2\,078$ bit	$1\,023+\|c\|+\|F\|/n_1$ $=2\,062$ bit
文献[20]	$160c+1\,024(cn_2+1)$ $=7.043\,2\times10^9$ bit	$l\log_2 p$ $=2\,560$ bit	$1\,024(lc+1)+160s$ $=4.068\,1\times10^8$ bit
本文方案	$(1\,024+256)n+2\|F\|$ $=2.148\,8\times10^9$ bit	256 bit	1 024 bit

通过上表的对比可以发现,本方案在综合通信方面的开销低于其他方案,特别是挑战和证明生成阶段.

区块链在运行过程中需要进行额外通信,以保证分布式节点的数据一致性. 本文参照文献[17]、文献[20]的方式,只对方案本身所产生的通信开销进行对比分析,不考虑区块链本身的额外通信开销.

(3)冗余率

PDP方案的冗余率定义为$r=(|F|+|\mathrm{MetaData}|)/|F|=1+|\mathrm{MetaData}|/|F|$,其中,$|F|$表示文件的大小,$|\mathrm{MetaData}|$表示元数据的大小. 同(1),以$|F|$=1 Gb文件为例. 通过对其他两个方案的分析,本文发现文献[17]的冗余率为$1+1\,024\ \mathrm{bit}\times1.073\,7\times10^6/1\ \mathrm{Gb}=2$,文献[20]的冗余率为$1+1\,024\ \mathrm{bit}\times6.871\,9\times10^6/1\ \mathrm{Gb}\approx4$. 而本文的冗余率为$1+(1\,024+256)\ \mathrm{bit}\times1\,000/1\ \mathrm{Gb}\approx1$. 因此,本文所提方案的冗余率最低.

5　结束语

本文提出了基于区块链的公平可证明数据持有方案. 与传统的可证明数据持有方案相比,本文方案不仅可以保护客户端的权益,还可以保护服务器的利益,从而实现可证明数据持有方案的公平性. 此外,基于可行性考虑,本文主要采用了哈希函数与Perdersen承诺,校验效率极高.

参考文献

[1] BHATT D. A revolution in information technology - cloud computing[J]. Walailak Journal of Science & Technology, 2012, 9(2): 107-113.

[2] DESWARTE Y, QUISQUATER J, SAIDANE A. Integrity and Internal Control in Information Systems. VI[M]. Boston: Springer, 2004: 1-11.

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

[4] SEBÉ F, DOMINGO F J, MARTINEZ B A, et al. Efficient remote data possession checking in critical information infrastructures[J]. IEEE Transactions on Knowledge and Data Engineering, 2008, 20(8): 1034-1038.

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

[6] ERWAY C C, KUPCU A, PAPAMANTHOU C, et al. Dynamic provable data possession[C]//Proceedings of The 15th ACM Conference on Computer and Communications Security. New York: ACM, 2009: 213-222.

[7] HAO Z, ZHONG S, YU N H. A privacy-preserving remote data integrity checking protocol with data dynamics and public verifiability[J]. IEEE Transactions on Knowledge and Data Engineering, 2011, 23(9): 1432-1437.

[8] WANG H Q. Proxy provable data possession in public clouds[J]. IEEE Transactions on Services Computing, 2013, 6(4): 551-559.

[9] ZHU Y, HU H X, AHN G J, et al. Cooperative provable data possession for integrity verification in multicloud storage[J]. IEEE Transactions on Parallel and Distributed Systems, 2012, 23(12): 2231-2244.

[10] ZHU Y, WANG H X, HU Z X, et al. Efficient provable data possession for hybrid clouds[C]//Proceedings of The 17th ACM Conference on Computer and Communications Security. New York: ACM, 2010: 756-758.

[11] CURTMOLA R, KHAN O, BURNS R, et al. MR-PDP: multiplereplica provable data possession[C]//Proceedings of The 28th International Conference on Distributed Computing Systems. Washington: ACM, 2008: 411-420.

[12] HAO Z, YU N H. A multiple-replica remote data possession checking protocol with public verifiability[C]//Proceedings of The 2010 Second International Symposium on Data, Privacy, and E-Commerce. Washington: ACM, 2010: 84-89.

[13] WANG Y J, WU Q H, QIN B, et al. Online/offline provable data possession[J]. IEEE Transactions on Information Forensics and Security, 2017, 12(5): 1182-1194.

[14] WANG Q, WANG C, LI J, et al. Enabling public verifiability and data dynamics for storage security in cloud computing[C]//Proceedings of The 14th European Symposium on Research in Computer Security. Berlin: ACM, 2009: 355-370.

[15] 杜瑞忠, 石朋亮, 田俊峰. 基于DDCT表的多副本完整性审计方案[J]. 电子学报, 2020, 48(1): 164-171.
DU R Z , SHI P L, TIAN J F. Multi-copy integrity audit scheme based on DDCT table[J]. Acta Electronica Sinica, 2020, 48(1): 164-171. (in Chinese)

[16] SHEN J, SHEN J, CHEN X F, et al. An efficient public auditing protocol with novel dynamic structure for cloud data[J]. IEEE Transactions on Information Forensics and Security, 2017, 12(10): 2402-2415.

[17] WANG H Q, WANG Q H, HE D B. Blockchain-based private provable data possession[J]. IEEE Transactions on Dependable and Secure Computing, 2019: 1-10.

[18] WANG H, QIN H, ZHAO M H, et al. Blockchain-based fair payment smart contract for public cloud storage auditing[J]. Information Sciences, 2020, 519: 348-362.

[19] LI Y N, YU Y, CHEN R N, et al. IntegrityChain: provable data possession for decentralized storage[J]. IEEE Journal on Selected Areas in Communications, 2020, 38 (6): 1205-1217.

[20] CHEN R N, LI Y N, YU Y, et al. Blockchain-based dynamic provable data possession for smart cities[J]. IEEE Internet of Things Journal, 2020, 7(5): 4143-4154.

[21] NAKAMOTO S. Bitcoin: A peer-to-peer electronic cash system[EB/OL]. (2018)[2021]. https://bitcoin.org/bitcoin.pdf.

[22] CASTRO M, LISKOV B. Practical byzantine fault tolerance[C]//Proceedings of The Third Symposium on Operating Systems Design and Implementation. Berkeley: ACM, 1999: 173-186.

[23] TANG F, MA S, XIANG Y, et al. An efficient authentication scheme for blockchain-based electronic health records[J]. IEEE ACCESS, 2019, 7: 41678-41689.

[24] 孙君, 熊关. SCMA mMTC系统中基于联盟区块链的无线电资源交易的信用支付[J]. 电子学报, 2019, 47(8): 1677-1684.
SUN J, XIONG G. Credit payment for radio resources transactions based on consortium blockchain in SCMA mMTC[J]. Acta Electronica Sinica, 2019, 47(8): 1677-1684. (in Chinese)

[25] 马晓婷, 马文平, 刘小雪. 基于区块链技术的跨域认证方案[J]. 电子学报, 2018, 46(11): 2571-2579.
MA X T, MA W P, LIU X X. A cross domain authentication scheme based on blockchain technology[J]. Acta Electronica Sinica, 2018, 46(11): 2571-2579. (in Chinese)

[26] THOMAS B. Smart Contracts - Blockchains in the Wings [M]. Boston: Springer, 2017: 169-184.

[27] PEDERSEN T P. Non-interactive and information- theoretic secure verifiable secret sharing[C]//Proceedings of The 11th Annual International Cryptology Conference on Advances in Cryptology. Berlin: Springer, 1991: 129-140.

[28] CARO D A, IOVINO V. JPBC: Java pairing based cryptography[C]//Proceedings of The 2011 IEEE Symposium on Computers and Communications. Washington: ACM, 2011: 850-855.

作者简介

唐　飞　男,1986年生,重庆垫江人. 博士,重庆邮电大学副教授、硕士生导师. 主要研究方向为公钥密码、隐私保护、区块链等.
E-mail: tangfei@cqupt.edu.cn

冯　卓　男,1997年生,四川广元人. 硕士,重庆邮电大学研究生. 主要研究方向为公钥密码、区块链.

黄永洪　男,1974年生,重庆永川人. 硕士,重庆邮电大学讲师. 主要研究方向为信息安全、密码学等.