

基于联盟链的数据溯源机制

赵守才, 曹利峰, 杜学绘

(信息工程大学, 河南 郑州 450001)

摘要: 随着大数据时代的到来, 数据的产生和流转速度获得空前增长。区块链技术的出现为数据的真实性验证提供了新的解决思路。随着区块链技术的发展, 不同区块链间数据流转需求逐渐增加, 而不同区块链间数据流转又带来了新的安全性问题, 如跨链数据传递易泄露、非法访问造成数据泄露不易发现等。针对上述问题, 提出一种基于联盟链的数据溯源机制。设计了跨区块链的数据溯源模型, 引入私有数据管道来实现跨链数据传递的安全性, 通过授权与访问日志实现对用户行为的记录, 保证非法越权访问的可追溯性。为提高数据溯源查询效率, 采用链上链下同步存储机制, 将每次交易前的数据流转状态经加密存储至数据库, 并将其索引存储在区块链交易中, 从而实现链上链下数据的一一对应, 同时在区块中引入 Merkle 山脉存储区块摘要, 提高区块合法性检验效率。根据数据存储形式及跨链数据交互机制, 设计了数据溯源算法, 并将溯源结果以有序树的形式展示。针对电商行业的跨域数据溯源场景, 基于 Fabric 搭建了联盟链溯源实验环境, 并采用 Go 语言模拟测试了区块数量较大、交易数量较多情况下的数据溯源性能。结果显示, 随着区块高度及数据流转次数增加, 数据验证及溯源效率具有显著优势。

关键词: 区块链; 数据溯源; 跨链; 联盟链

中图分类号: TP391

文献标志码: A

DOI: 10.11959/j.issn.2096-109x.2023067

Data traceability mechanism based on consortium chain

ZHAO Shoucai, CAO Lifeng, DU Xuehui

Information Engineering University, Zhengzhou 450001, China

Abstract: With the unprecedented growth in the speed of data generation and circulation in the era of big data, the emergence of blockchain technology provides a new solution for data authenticity verification. However, with the increasing demand for data flow between different blockchains, new security issues arise. Cross-chain data transmission can lead to data leakage, and detecting data leakage caused by illegal access becomes challenging. To address these problems, a data traceability mechanism based on a consortium chain was proposed. A

收稿日期: 2022-09-01; 修回日期: 2023-03-02

通信作者: 曹利峰, caolf302@sina.com

基金项目: 中原英才计划豫组通[2021]44号

Foundation Item: Zhongyuan Talent Plan Yu Group Pass [2021] No. 44

引用格式: 赵守才, 曹利峰, 杜学绘. 基于联盟链的数据溯源机制[J]. 网络与信息安全学报, 2023, 9(5): 92-105.

Citation Format: ZHAO S C, CAO L F, DU X H. Data traceability mechanism based on consortium chain[J]. Chinese Journal of Network and Information Security, 2023, 9(5): 92-105.

cross-blockchain data traceability model was designed, incorporating private data pipelines to ensure the security of cross-chain data transmission. User behaviors were recorded through authorization and access logs, ensuring the traceability of illegal unauthorized access. To improve the efficiency of data traceability and query, an on-chain and off-chain synchronous storage mechanism was adopted. The state of data flow before each transaction was encrypted and stored in the database, and its index was stored in the blockchain transaction. This enables a one-to-one correspondence between on-chain and off-chain data. Additionally, Merkle trees were introduced into the block body to store block summaries, enhancing the efficiency of block legitimacy verification. Based on the data storage form and cross-chain data interaction mechanism, a data traceability algorithm was designed. The traceability results were displayed in the form of an ordered tree. An experimental environment for consortium chain traceability was built using fabric, targeting the cross-domain data traceability scenario in the e-commerce industry. The GO language was used to simulate and test the data traceability performance with a large number of blocks and transactions. The results demonstrate that with the increasing number of blocks and transactions, the proposed data traceability mechanism maintains satisfactory performance.

Keywords: block chain, data traceability, across chain, consortium chain

0 引言

随着互联网技术的发展,数据创建的速度正以指数级增长。国际数据公司(IDC)预测全球的数据总量到2025年将会增长至175 ZB,届时中国将以48.6 ZB的数据总量成为最大的数据圈^[1]。海量的数据在网络中不同应用系统、组织之间共享和融合,这使得数据的生产方式和流转方式呈现出多样性、复杂性的特点^[2-3]。

在数据流转使用过程中,如何保证数据来源可信,以及对数据进行溯源追踪成为人们关注的重点^[4]。溯源技术用来追踪产品或者数据生成、流转过程,目前被广泛地应用在电子商务、科学实验、数据追踪等领域^[1]。目前溯源最常用的做法是在商品上附带二维码或者条形码。每经过一个环节,通过识别商品上的二维码或条形码,将生产或交易信息上传至数据中心,实现集中保存商品的溯源信息。这种传统的中心化溯源方式具有查询速度快、使用简单的优点,但同时存在易篡改和伪造的缺点,无法保证数据存储的安全性,很容易对数据的完整性造成破坏。同时,传统的数据溯源过程中,信息采集不透明将导致溯源信息可信度大打折扣。

区块链技术作为一种去中心化的分布式数据账本,为数据溯源管理提供了新的解决思路^[5]。区块链以其具有的不易篡改、去中心化、全网数据统一、方便追溯等特性得到了学术界和工

业界的广泛认可^[6]。区块链利用密码学技术来保证数据的传输和访问安全,其中每一个区块上记录的交易是上一个区块形成之后、本区块创建之前的所有活动,这种块链式结构保证了数据存储的完整性^[7],同时采用分布式共识算法实现数据在对等网络中的一致性^[8]。一旦新区块完成被加入区块链,该区块中的数据将获得永久记录并且无法更改,这种机制保证了区块链上存储数据的不易篡改的特性。区块中的时间戳机制,保证了区块数据操作时间的有迹可循,从而形成一个不易篡改、不易伪造及便于溯源的可信数据库^[9]。

随着区块链项目的井喷式发展,供应链、物流及金融等领域建立各自区块链平台。数据的跨系统、跨链的流转共享使得数据溯源变得更加复杂,跨链数据溯源成为溯源数据完整性必不可少的一个重要环节^[10]。以一次简单的网购平台购物为例,用户需要提交申请信息到网购平台,待网购平台审核后为用户创建账号。用户获取账号后,登录网购平台浏览商品,为保证商品的可靠真实性,用户能够访问商品的供应链溯源查看商品信息。待用户选定商品后,将付款信息发送给银行系统或第三方支付平台。待支付成功后,商家将货物及用户地址等相关信息提交给物流系统,同时用户或商家可对物流信息进行查询溯源。由此可以看出,一次简单的网上购物,至少需要银行系统、购物平台、商品供应链系统及物流系统之

间进行数据交互。

针对数据在不同系统中流转带来的跨链数据溯源需求,本文提出基于联盟链的数据溯源机制,主要贡献如下。

1) 设计了一个基于联盟链的跨链数据交互溯源模型,通过私有数据管道来保证敏感跨链数据安全性,并通过访问与授权日志,实现数据的访问控制及数据受访的可追溯性。

2) 针对区块链溯源过程中逐个交易查询造成的时延,采用链上存储数据单次流转交易,链下加密存储此交易前的全部交易溯源有序树,同时在区块中引入 Merkel 山脉,提高区块合法性验证效率。

3) 根据数据存储形式及跨链数据交互机制,设计了数据溯源算法,并以有序树的形式展示溯源结果。

1 相关工作

随着区块链技术的发展,越来越多的学者开始关注区块链技术在数据溯源领域的应用。目前人们对基于区块链技术的数据溯源研究主要集中在溯源数据存储、溯源信息安全及溯源效率等方面。

1) 溯源数据存储方面。樊玉琦等^[11]为缓解区块链节点数据存储压力,提出采用纠错码对区块链数据进行编码存储,以实现数据存储代价和数据读取性能的平衡。乔蕊等^[12]为解决物联网设备产生的大量动态数据存储与共享问题,提出了用于实现操作实体多维授权与动态数据存储的双联盟链结构,并给出了一种基于联盟链的动态数据溯源机制优化方案。Sun 等^[13]基于双链存储机制构造了一种基于区块链的数据加密存储方案,并结合 IPFS (inter planetary file system) 技术实现了电子病历的稳定存储与共享。刘炜等^[14]通过 IPFS 构建了一个大容量存储空间的传染病数据共享区块链模型,保证了区块链数据溯源信息存储的稳定性。张国潮等^[15]为解决区块链数据存储问题,提出一种基于门限秘密共享的区块链分片存储模型,缓解了单节点数据存储压力。

2) 溯源信息安全方面。针对数据溯源过程中

存在的数据篡改、泄露等问题,贾大宇等^[16]通过对现有的数据溯源模型及标准的扩展,提出了一种分层次的数据溯源安全模型,保证了数据溯源信息的完整性和可信性。刘耀宗等^[17]为提高数据溯源安全,提出基于区块链的 RFID 大数据溯源安全模型,通过建立数据溯源全程链式路径,保证大数据溯源的安全管理。针对溯源数据流转过程中数据易泄露等问题,Li 等^[18]结合多签名密钥分发及零知识证明技术设计了安全跨链体系及数据隐私保护机制,实现了溯源数据的链上链下双验证。Neisse 等^[19]从智能合约角度出发,基于区块链构造了主体合约和逻辑合约两个数据审计和溯源模型,并评估了溯源模型的可扩展性等性能,实现了对区块链数据的安全溯源。

3) 溯源效率方面。余涛等^[20]为解决区块链数据查询面临的查询功能单一、查询效率不足等问题,将 Fabric 联盟链和 MySQL 结合提出一种区块链数据的关系查询方案,提高了区块链数据溯源查询效率。Lai 等^[21]为缩短区块链溯源查询响应时间,设计了一个独立于 Merkel 树的索引结构来支持高效的块内数据查询。刘炜等^[22]为解决区块链查询效率低等问题,结合布隆过滤器及 Merkel 树提出一种面向区块链溯源的高效查询方法,在保证数据不易篡改的同时提高了区块链数据溯源查询效率。Peng 等^[23]通过构建应用层和中间层,提出了可验证的查询结构,同时为实现区块链溯源数据的快速查询,对区块链存储数据进行了抽取重组。

2 模型架构

2.1 跨链模型

基于联盟链的数据溯源结构模型如图 1 所示,图中共包含 4 条应用链。为实现数据的跨链溯源,各应用链分别包含应用链插件、授权与访问日志及跨链网关等模块。跨链中心部分除包含各应用链组织代理外,还含有用于不同链之间隐私数据传递的私有数据通道。

应用链插件用于连接应用链和跨链网关,实现应用链和跨链网关之间的解耦。应用链插件按照跨链网关和应用链交互的需求确定适合跨链交互的插件接口。通过引入支持动态加载插件的方

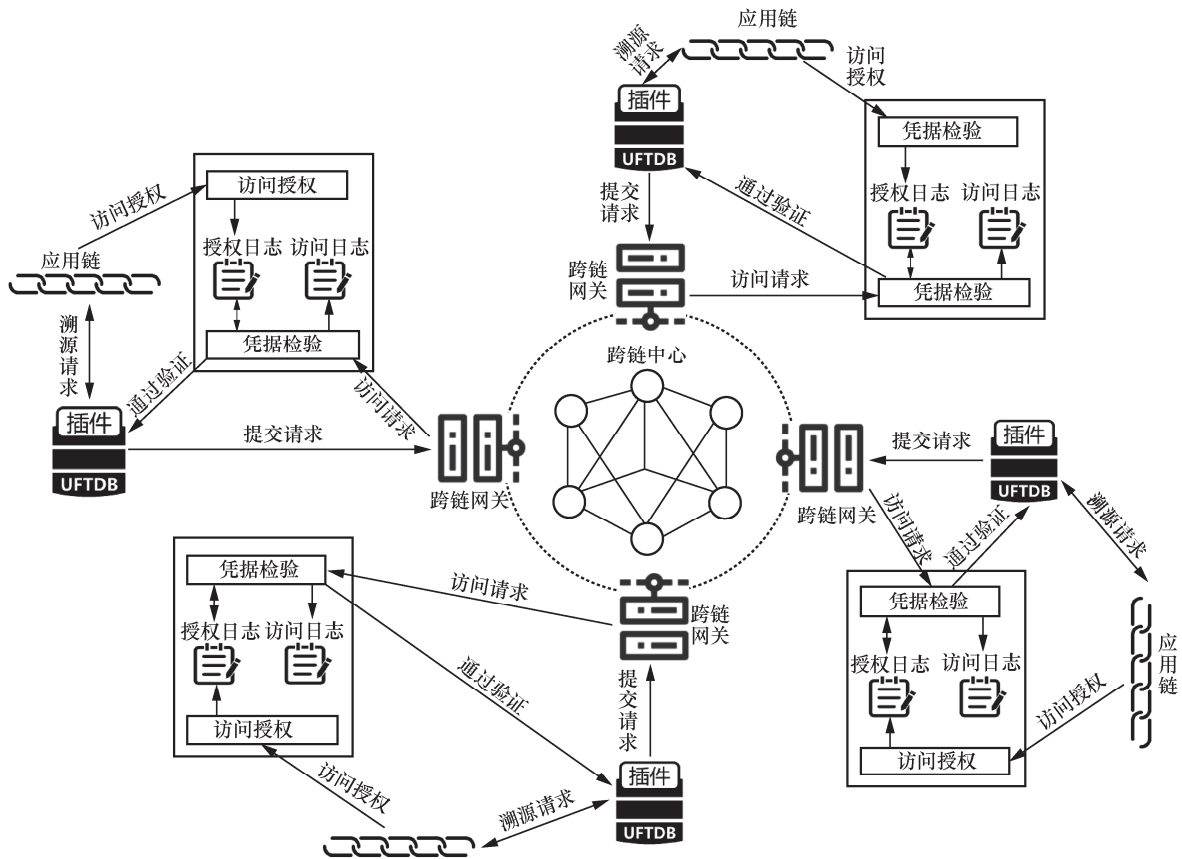


图1 基于联盟链的数据溯源结构模型

Figure 1 Data traceability structure model based on consortium chain

式完成应用链接入跨链系统，同时将应用链中用于跨链操作的部分请求封装到插件中，插件可根据应用链请求，向跨链网关转发跨链系统能够识别的统一请求方式。此外，应用链插件负责将通过验证后的跨链溯源请求解析为应用链能够识别的消息格式，并将其发送到应用链数据库执行查询。

授权与访问日志将不同系统区块链的授权行为和受访信息记录在相应的区块链日志上，根据区块链不易篡改等特性，任何人和组织都无法伪造授权与访问结果。通过授权行为日志追踪记录，快速溯源得到访问用户是否拥有访问权限，以此对用户的跨链访问行为做出限制，降低数据泄露的风险。访问行为日志用于记录用户跨系统、跨区块链的客体访问行为。通过授权行为及访问行为的联合溯源，判断用户访问的合法性，快速锁定用户的越权访问请求及已发生的越权访问行为。通过区块链授权与访问日志，跟踪查询用户越权等非法访问行为历史记录，以此作为用户好

坏评判指标之一。同时在数据发生泄露时，能够通过授权日志与访问日志的联合溯源，快速找到数据泄露的问题所在，提升系统数据安全性。

跨链网关可包含两种设计模式。一种为适合链对链的直连模式，这种模式只适用于区块链数量较少的小型跨链系统。另一种设计思想为引入中继链来进行跨链操作，这种模式适用于较多区块链间进行跨链互操作的场景。考虑到现实数据流转过程中可能涉及多系统、多条区块链间的数据信息互操作需求，本文采用第二种引入中继链的跨链网关模式，一条中继链即跨链中心的一个组织。根据功能将跨链网关分为监听模块（MM，monitor module）、路由分发模块（RDM，routing distribution module）、代理模块（AM，agent module）、同步模块（SM，synchronization module）、检查模块（CM，check module）及执行模块（PM，perform module）。跨链网关结构如图2所示。

监听模块：监听对应的系统应用链插件上提

交的跨链请求，对接收到的跨链请求信息检查。将通过检查的请求发送到 RDM，对未通过检查的跨链请求进行回滚操作。

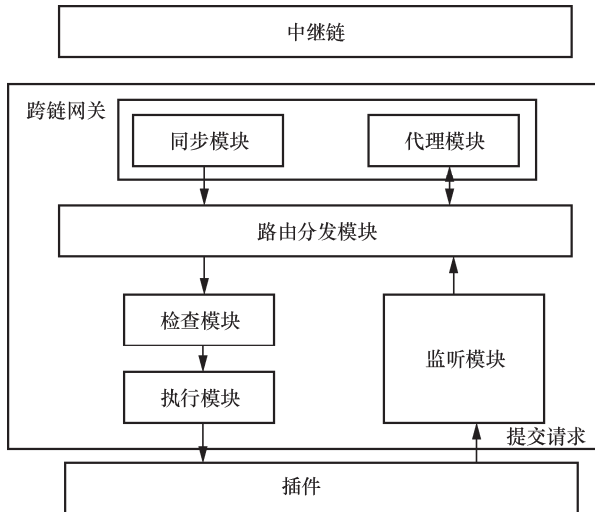


图 2 跨链网关结构
Figure 2 Cross-chain gateway structure

路由分发模块：由于跨链网关可以支持不同的跨链需求，RDM 可根据需要进行转换分发对象。在本文网关模式中，RDM 将接收到的来自 MM 的请求信息直接发送到中继链的 AM。

代理模块：相当于中继链在本网关内部的代理，负责网关内部与中继联盟链之间的跨链信息的交互与传递。

同步模块：跨链交易需要共识并且打包到区块中，在同步交易时需要中继链的轻节点不断地同步更新区块的头部信息，以此获得中继链中与本网关相关的跨链交易信息。除此之外，还需要配合轻节点对跨链交易进行简单支付验证（SPV, simplified payment verification），以此来确定跨链交易的有效性。

检查模块：跨链交易在中继联盟链上已经达成共识并获得签名，因此 CM 只需要检查 RDM 转发而来的跨链交易是否来自中继链。CM 通过对共识签名进行检验判断当前跨链交易是否正确。

执行模块：负责保证跨链交易正确地提交到应用链，并且需要返回相应的跨链交易请求回执信息。跨链交易信息通过检查后被提交到 PM。PM 向应用链插件提交交易，待 PM 获得相应的执行结果后，通过回执的形式再返回给 RDM。

2.2 数据存储模型

传统的数据信息在进行存储时，大多采用集中式的数据库。在这种存储方式中，数据拥有者可以不留痕迹地修改数据信息，数据易遭受篡改和伪造的特点，大大降低了溯源数据的可信性^[24]。此外，传统的中心化数据存储模式中，数据存在误删及泄露的风险，并且在数据发生泄露时不易察觉，通常在发现数据泄露时，已经造成大量的数据泄露。区块链作为大型分布式账本，其不易篡改、去中心化、全网数据统一等特性与数据溯源要求不谋而合。但将区块链技术用于数据存储仍存在一定问题。当数据发生多次流转时，对数据流转状态的溯源查询需要对全链遍历，这将增加溯源时延，很难满足企业溯源业务的要求。此外，区块链中提供了 Merkle 树进行区块内数据的快速验证，但对于区块的验证则需要根据区块哈希摘要逐个追溯验证。

针对用户快速溯源需求，本文将区块链技术与关系型数据库结合设计，数据存储结构如图 3 所示。每个区块包含区块头和区块体两部分内容，区块头中包含前区块哈希、Merkle 树的根哈希值 MTR 等区块信息，为实现区块的快速验证在已有的区块头信息基础上加入了 Merkle 山脉的根哈希值 MMRR。在区块体中加入了图 3 虚线框所示的 Merkle 山脉以及关系型数据库。

其中，关系型数据库用于加密存储数据流转的历史过程，区块链用于存储单次的流转交易。将每次的数据流转作为一次交易存储在区块体 Merkle 树叶节点中，同时将当前版本数据的溯源有序树（ordered tree）经加密后存储在数据库中，并将其数据库索引存储在交易中，用于溯源过程中对数据流转历史的快速查询。

考虑到当前区块链数据验证大多针对区块内数据验证，当对不同区块合法性验证时，通常需要根据区块头中的前区块哈希值进行逐一区块验证。当需要验证的区块与最新区块距离较远时，其查询复杂度将达到 $O(N)$ 。为解决区块验证时延问题，在区块体中引入 Merkle 山脉，并将 Merkle 山脉根的哈希值 MMRR 存入区块头。Merkle 山脉用来存储当前区块之前所有区块的区块头哈希值，来实现区块合法性的 SPV。

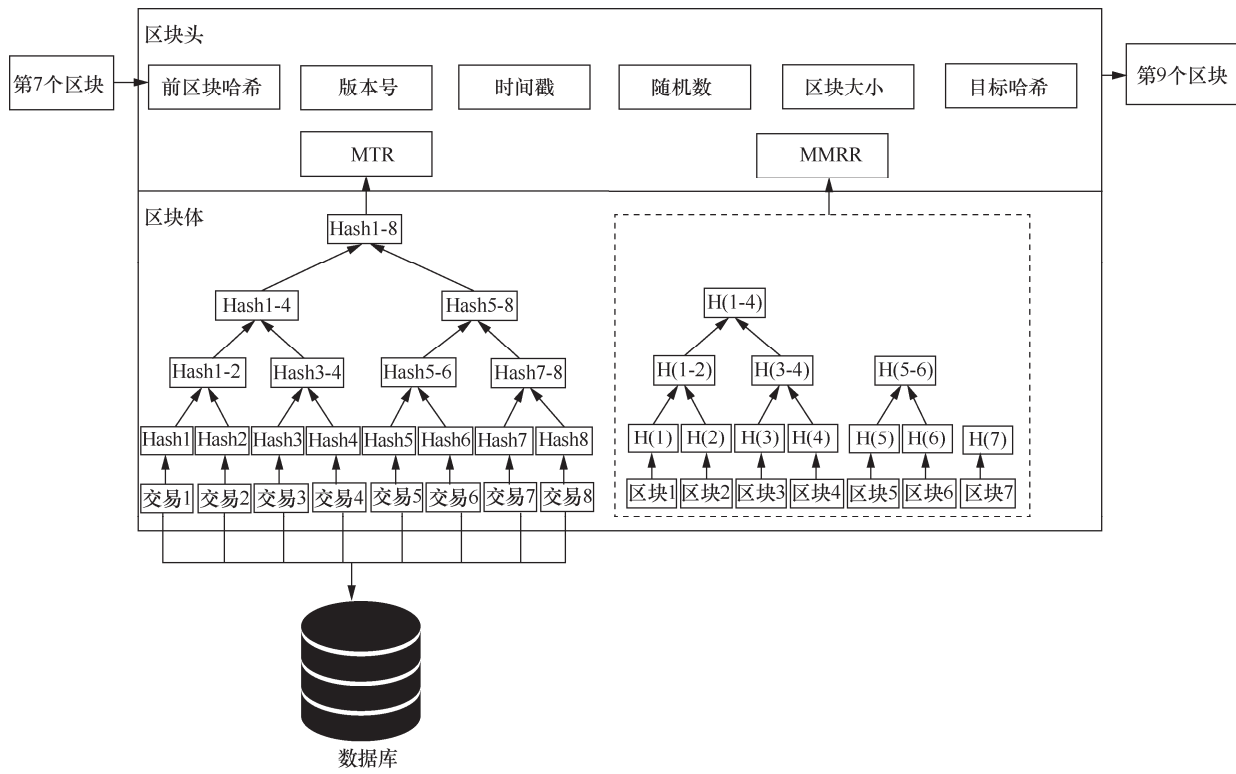


图3 数据存储结构
Figure 3 Data storage structure

2.2.1 数据存储

本文的数据存储结构中，数据流转信息以交易的形式存储在区块链中，同时将此交易之前的该数据的所有流转演变信息进行加密存储在数据库中，并将其哈希摘要索引存储到当前区块交易中。当需要溯源时，只需在当前交易索引对应数据库中读取数据流转信息，并加入当前交易时间至溯源时间内新的流转演变信息即可得到完整的流转信息。区块链交易与数据库存储一一对应，并具有数据库查询索引功能，实现一种链外状态缓存，即使数据丢失也不会对区块链数据造成影响。其加密存储与查询共包含初始化参数、生成密钥、加密及解密4个过程，具体步骤如下。

1) 初始化参数 $\text{Setup}(1^k)$ 。取安全参数 $k \in \mathbb{Z}^*$ ， G_1 是阶为素数 q 的循环群，其生成元为 P 。双线性映射 $e: G_1 \times G_1 \rightarrow G_2$ ， $g = e(P, P)$ ，哈希函数 $h_1: \{0, 1\}^* \rightarrow \mathbb{Z}_q^*$ ， $h_2: G_2 \rightarrow \{0, 1\}^*$ 。选择随机数 s_1, s_2 ，并计算其对应公钥 $P_1 = s_1 P$ ， $P_2 = s_2 P$ 。将参数 $\text{parameter} = \{q, P, G_1, G_2, e, g, h_1, h_2, P_1, P_2\}$ 对外公开。

2) 生成密钥 $\text{Keygen}(\text{parameter}, s_1, s_2) \rightarrow \text{sk}_{\text{ID}}$ 。

用户根据数据访问原则，将允许访问的用户身份存入授权日志区块链中，然后根据可访问用户身份 $\text{ID} \in \{0, 1\}^*$ ，计算出 $h_{\text{ID}} = h_1(\text{ID})$ ，并计算出对应密钥。

$$\text{sk}_{\text{ID}} = \left(\frac{1}{h_{\text{ID}} + s_1} P, \frac{1}{h_{\text{ID}} + s_2} P \right)$$

3) 加密 $\text{Encrypt}(\text{parameter}, M) \rightarrow C$ 。用户将系统中的明文数据信息 $M = \{0, 1\}^*$ 进行加密存储到数据库中，并将其哈希摘要索引存入区块链Merkel树。计算 $h_{\text{ID}} = h_1(\text{ID}) = \{0, 1\}^*$ ，选择随机数 $(r_1, r_2, r_3) \in \mathbb{Z}_p^3$ ，计算出密文 $C = (C_1, C_2, C_3, C_4)$ ，满足

$$C_1 = r_1(h_{\text{ID}}P + P_1)$$

$$C_2 = r_2P$$

$$C_3 = r_3(h_{\text{ID}}P + P_2)$$

$$C_4 = (r_2 \parallel M) \oplus h_2(g^{r_3})$$

4) 解密 $\text{Decrypt}(\text{parameter}, C, \text{sk}_{\text{ID}}) \rightarrow M$ 。溯源访问请求到达目标区块链后，智能合约通过授

权日志查看用户是否拥有访问权限,待验证通过后获得解密密钥 sk_{id} ,并计算

$$C_4 \oplus h_2\left[\frac{1}{h_{id} + s_2} P, C_3\right] = r_2 \parallel M$$

最后验证 $C_2 = r_2 P$,若成立则获取明文 M 。

2.2.2 数据验证

在已有的区块结构中,人们通过Merkel树来实现对区块中已存储交易的合法性快速验证。然而,对于区块的合法性验证,则需要从当前最新区块逐步溯源至创世区块才能验证区块链中的每一个区块是否合法,造成了极大的时间浪费。为实现数据的快速验证,Bunz等^[25]提出采用基于Merkel树改进的数据结构(即Merkel山脉)来进行数据的高效验证。Merkel山脉中数据结构和Merkel树中数据结构类似,每一个非叶子节点的值都是左右两个非叶子节点值之和。其区别为Merkel树为一棵完美的二叉树,而Merkel山脉是若干棵完美二叉树的组合。在Merkel山脉中追加数据只需在已有数据的基础上进行添加,而无须对已有的数据进行修改,从而实现数据添加的便利性。利用Merkel山脉特点,将区块链中的每一个区块作为叶子节点并计算其哈希值,采用当前区块之前的所有区块信息构建Merkel山脉,可以有效提高区块链验证时间。

Merkel山脉结构如图4所示,每个区块中的Merkel山脉需要存储此区块之前的所有区块的哈希值信息。

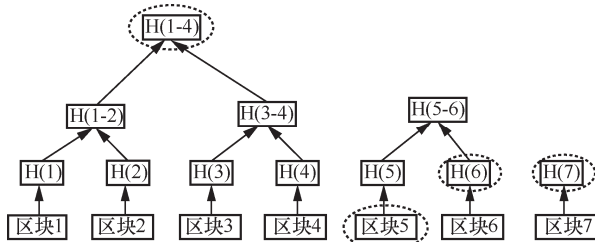


图4 Merkel山脉结构

Figure 4 Structure of the Merkel mountains

例如,当需要验证第5个区块是否合法时,采取以下步骤。

Step 1: 根据第5个区块中的数据信息计算出区块哈希值 $H(5)$ 。

Step 2: 获取当前最新区块中区块5的邻居节

点 $H(6)$,并计算出 $H(5-6) = \text{hash}(H(5) \& H(6))$ 。

Step 3: 计算Merkel山脉根节点 $MMRR' = \text{hash}(H(1-4) \& H(5-6) \& H(7))$ 。

Step 4: 比较 $MMRR'$ 与当前最新区块中Merkel根的 $MMRR$ 是否相等,若相等则区块合法,否则不合法。

3 数据溯源

3.1 私有数据管道

针对敏感数据在联盟网络中传播带来的安全性问题,在物理数据通道的基础上提出私有数据管道,用来满足更小范围内有数据流转交易的隐私需求。在通道内部发送的交易数据信息只有属于通道内的成员才能够查看,可以将通道看作联盟链网络内部的一个私有通信子网。然而,在由各应用链系统组成的联盟链网络中,不同应用链之间均可能存在数据交互,若每次交易均建立一条通道,则不仅为网络管理带来负担,而且通道性能无法得到充分利用。为此,在数据通道的基础上,设立私有数据管道机制。私有数据管道存在于通道内部,作为一个更小的数据集传递通路,它通常负责资产转移或重要敏感数据的传递。数据通道结构如图5所示。

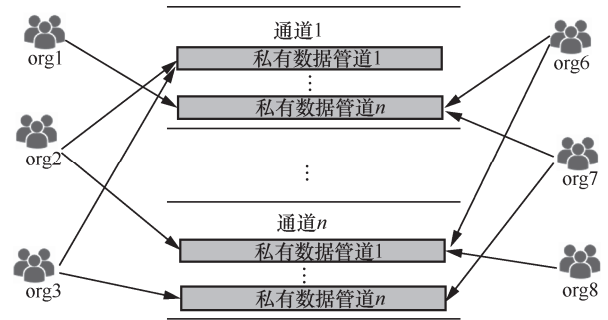


图5 数据通道结构

Figure 5 Data channel structure diagram

其中,每个组织中包含多个节点用户,不同通道之间彼此物理隔离,避免了数据信息在整个联盟网络中的公开透明,在物理逻辑上实现数据隔离,不同应用链间溯源数据通过通道进行传递。私有数据管道能够实现同一通道中不同节点对溯源数据的隐私性要求,保证不同应用链、不同节点的溯源数据在通道中传递的安全性。

同时,通道数量并非越多越好,引入通道策

包含数据管道创建阶段和跨链数据溯源阶段。各阶段步骤如下。

第一阶段为数据管道创建阶段。

Step 1: 溯源请求用户通过所在区块链发起跨链请求, 其消息格式为 $\langle \langle \text{request}, \text{UID}, \text{SBID}, \text{TBID}, H, \text{DID}, \text{timestamp} \rangle, \sigma_{\text{UID}} \rangle$ 。其中, UID 为请求用户的身份信息, SBID 为请求用户所在区块链编号, TBID 为目标区块链编号, H 为溯源数据所在区块高度, DID 为溯源数据索引, timestamp 为时间戳, σ_{UID} 表示请求者的签名。

Step 2: 请求用户所在区块链对应的插件将用户请求消息进行解封装, 将消息封装为联盟链系统能够识别的消息格式。

Step 3: 跨链网关解析请求消息, 经检验签名等信息无误后, 转发至联盟链网络。

Step 4: 联盟链收到消息后, 将请求消息解封装后, 构建权限验证消息, 其格式为 $\langle \langle \text{verify}, \text{UID}, \text{SBID}, \text{TBID}, H, \text{DID}, \text{timestamp} \rangle, \sigma \rangle$ 。其中, σ 为联盟链网络成员共识签名。

Step 5: 目标链网关监听到发给该链的消息并检查后, 将验证消息转发给目标链插件。

Step 6: 应用链插件收到验证消息后, 将消息解封装成目标链能够识别的格式。

Step 7: 通过查看目标链中的授权日志, 检查请求用户对请求数据是否拥有查询权限, 并将结果返回给联盟链网络。

Step 8: 若拥有权限, 则根据请求者和数据所有者所处通道情况, 创建私有数据传递管道, 并将管道信息发送给对应用户, 消息格式为 $\langle \langle \text{result}, \text{SUID}, \text{SBID}, \text{TUID}, \text{TBID}, \text{channel}, \text{pipe}, \text{timestamp} \rangle, \sigma \rangle$ 。若无权限, 则返回给请求者, 溯源结束。其中, SUID 表示请求者身份, SBID 表示请求者所在区块链编号, TUID 表示数据所有者身份, TBID 表示数据所有者所在区块链编号, channel 表示私有管道所在通道, pipe 表示新建的私有数据管道。

第二阶段为跨链数据溯源阶段。

Step 9: 溯源请求用户通过私有数据管道发起溯源请求, 消息格式为 $\langle \text{Query}, \text{UID}, \text{SBID}, \text{TUID}, \text{TBID}, \text{DID}, \text{channel}, \text{pipe}, \text{timestamp} \rangle, \sigma_{\text{UID}} \rangle$ 。接下来, 类似第一阶段, 经应用链插件及跨链网关转

发请求后到达目标区块链。

Step 10: 请求用户首先对溯源数据的最新交易进行验证, 并获取最新交易; 然后根据最新交易对应区块交易中存储的哈希索引查找到数据库中的数据流转信息; 最后将当前区块中该数据的最新流转信息加入已有的流转信息中, 得到最终的完整流转溯源信息。

Step 11: 将完整的溯源数据流转信息通过私有数据管道返回给溯源请求者, 溯源结束。

3.3 溯源算法

根据数据存储模型, 在每次数据交易时, 与此交易索引对应的数据库中存储此交易前的数据溯源有序树。数据溯源有序树结构如图 7 所示。树中每个节点代表一种数据状态, 数据状态中包含数据的授权、访问及所在区块等信息。树根节点表示原始数据状态, 其余每个节点都表示由父节点经某种变化演变而来的数据状态。同一父节点的子节点之间, 按照数据变更交易时间从左至右进行排序。

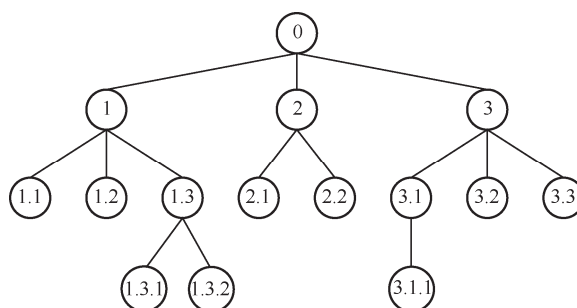


图 7 数据溯源有序树结构
Figure 7 Structure of data traceability ordered tree

图 7 中节点 1.3.2 表示由 1.3 数据状态发生某种数据变更交易后得到的新的数据状态, 在此交易对应的数据库中存储此交易前的数据状态有序树。进行数据溯源时, 只需在区块链交易对应数据库中存储的有序树基础上, 从根节点开始依次查询添加 1.3.2 数据状态交易时间至溯源时间之间新增加的授权、访问及数据变更等事务。采用广度优先的方式在已有的有序树上添加新的交易事务, 得到当前时期数据的完整溯源结果。数据溯源如算法 2 所示。

算法 2 数据溯源

输入 区块链账本, 溯源数据 D

输出 溯源结果 orderedTree

```

1) state←getLatestTx(D)//查询有关数据 D 的
最新状态信息
2) orderedTree←query from state.Database//获
取数据流转有序树
3) treeRootState=orderedTree.rootState// 获取
原始数据
4) queue.enqueue(treeRootState)//将原始数据
状态加入队列
5) while(!queue.isEmpty)
6) temState=queue.dequeue()
7) If temState.newAuthorizeSet≠∅//存在新授
权信息
8)      AddnewAuthorizeSet into Authori-
zeSet
9)      Set newAuthorizeSet=∅
10) End if
11) If temState.newAccessSet≠∅//存在新访
问信息
12)      AddnewAccessSet into AccessSet
13)      Set newAccessSet =∅
14) End if
15) If temState.newTX≠∅//存在新交易
16)      Foreach newState in temState.newTX
17)          Insert(temState→rightChild, newS-
tate) //将新交易加入有序树
18)          Set temState.newTX=∅
19)      End foreach
20) End if
21) Foreach childState in temState
22) Queue.enqueue(childState)
23) End foreach
24) End while
25) Return orderedTree

```

4 实验分析

4.1 安全评估

(1) 可信性分析

利用区块链对数据进行存储, 区块链中的 Merkel 树用来保证数据信息的可信性。首先锁定交易所在区块, 在区块体中获取交易 Merkel

树, 当需要验证某一交易是否被篡改时, 只需根据该交易中存储的交易信息计算哈希值, 然后与相邻子树或叶子节点的哈希值进行合并得到上一层新的子树哈希值。重复步骤最终得到所有交易的根哈希值 (即 Merkel 树根哈希), 存入区块头中, 一旦交易内容发生改变, 则所有上层子树的哈希值必将改变, 最终导致根哈希值改变, 验证失败。因此, 用户可以轻松验证数据信息是否发生了恶意篡改, 利用区块链技术可进一步提高溯源数据的可信性和真实性。

(2) 数据安全性分析

首先, 在联盟链通道中引入了私有数据管道, 不同通道之间彼此物理隔离, 避免了数据信息在整个联盟网络中的公开透明, 在物理逻辑上实现了数据隔离。其次, 在通道内部划分私有数据管道, 提供更小范围内数据的共享, 用户可在私有数据管道内进行数据的加密传输, 保证信息仅在交易用户双方之间实现共享, 可灵活地根据用户的溯源需求对私有数据管道设定权限, 有效地保证数据在传递过程中的隐私与安全。此外, 通过为每条区块链设置授权日志, 可对用户的访问行为做出限制, 防止非法用户获取密钥查询数据造成隐私泄露。同时通过访问日志记录的数据, 对用户的访问行为进行分析, 当发生数据泄露时, 通过溯源查询, 可及时锁定恶意用户, 并进行追责。

(3) 可用性分析

不同应用链的结构通常不相同, 为简化不同结构应用链间适配问题, 本文引入了应用链插件。将需要在应用链上具体操作的部分全部封装到应用链插件中, 满足不同架构应用链交互需求。在区块体中加入 Merkel 山脉, 使得数据溯源过程中不必遍历所有的区块验证是否遭到篡改, 只需要根据 Merkel 山脉中各完美二叉树的哈希值及山脉根哈希值即可验证, 有效缩短了区块链数据验证时间。

4.2 应用测试

为测试本文提出的溯源机制性能表现, 本节主要在联盟链跨链溯源可行性、区块构建、溯源效率以及联盟链通道的交易吞吐量等方面对溯源机制性能进行实验分析。实验环境: 处理器为 Intel

Core i7-7700H @ 3.60 GHz, 内存为 16 GB, 以及 64 位 CentOS Linux 7 操作系统。

本节基于 Fabric 2.0 搭建了联盟链跨链溯源实验环境, 设置 3 个组织, 每个组织包含 5 个节点, 共 15 个节点。节点参数配置如表 2 所示。

表 2 节点参数配置

Table 2 Node parameter configuration table

组织	节点	端口号	加入通道
org1	peer0.org1	7051	channel1
	peer1.org1	7052	channel1
	peer2.org1	7053	channel1
	peer3.org1	7054	channel1
	peer4.org1	7055	channel1
org2	peer0.org2	9051	channel2
	peer1.org2	9052	channel2
	peer2.org2	9053	channel2
	peer3.org2	9054	channel2
	peer4.org2	9055	channel2
org3	peer0.org3	11051	channel1 & channel2
	peer1.org3	11052	channel1 & channel2
	peer2.org3	11053	channel1 & channel2
	peer3.org3	11054	channel1 & channel2
	peer4.org3	11055	channel1 & channel2

在 Fabric 中, 每个通道维护一条分布式区块链账本。根据需要创建通道文件 channel1.tx 和 channel2.tx, 并在对应的通道文件中创建命名通道 channel1 和 channel2, 创建通道的执行结果如图 8 所示。

```
2022-09-21 12:30:47.013 UTC [channelCmd] InitCmdFactory -> INFO 00f Endorser and
orderer connections initialized
2022-09-21 12:30:47.216 UTC [cli.common] readBlock -> INFO 010 Received block: 0
bash-5.0# ls
channel-artifacts channel2.block log.txt mychannel.block
channel1.block crypto mycc.tar.gz scripts
bash-5.0#
```

图 8 创建通道的执行结果

Figure 8 Execution result of creating channels diagram

按照节点参数配置表将 org1 中节点加入 channel1 通道, 将 org2 中节点加入 channel2 通道, 将 org3 中节点同时加入两个通道中。节点 peer0.org1 加入 channel1 过程如图 9 所示。

```
bash-5.0# export CORE_PEER_LOCALMSPID=org1MSP
bash-5.0# CORE_PEER_TLS_CERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org
1.example.com/peers/peer0.org1.example.com/tls/server.crt
bash-5.0# CORE_PEER_TLS_KEY_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org
1.example.com/peers/peer0.org1.example.com/tls/server.key
bash-5.0# export CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrgani
zations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt
bash-5.0# export CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganiza
tions/org1.example.com/users/Admin@org1.example.com/msp
bash-5.0# export CORE_PEER_ADDRESS=peer0.org1.example.com:7051
bash-5.0# peer channel join -b channel1.block
2022-09-21 12:33:01.778 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
2022-09-21 12:33:01.925 UTC [channelCmd] executeJoin -> INFO 002 Successfully submitted proposal to join channel
bash-5.0#
```

图 9 节点 peer0.org1 加入 channel1 过程

Figure 9 Node peer0.org1 join channel1 process

节点按参数配置条件加入对应通道并部署通道链码后, 进行跨区块链的数据查询操作。首先由节点 peer3.org1 在其通道 channel1 中的分布式区块链账本上存储数据信息, 并在 channel1 的智能合约上为 peer0.org2 进行访问授权, 第一个区块链上存储信息初始化如图 10 所示。

```
bash-5.0# peer chaincode invoke -o orderer.example.com:7050 --tls true --cafile /opt/gopath/src/github.com/hyperl
edger/fabric/peer/crypto/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlsacerts/tlsca.examp
le.com-cert.pem -C mychannel -n mycc --peerAddresses peer3.org1.example.com:7054 --tlsRootCertFiles /opt/gopath/s
rc/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org1.example.com/peers/peer3.org1.example.com/tls
ca.crt --isInit -c '{"Args":["Init","ID","002","name","zhangsan","tel","17345678911"]}'
# set tx
2022-09-21 12:37:24.211 UTC [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 001 Chaincode invoke successful. result
status:200
Invoke transaction successful on peer3.org1 on channel 'channel1'
```

图 10 存储信息初始化

Figure 10 Store information initialization

待第一条区块链上部署完成数据信息后, 由第二条区块链上的用户 peer0.org2 进行跨区块链账本的数据溯源操作, 结果如图 11 所示。

```
bash-5.0# env|grep CORE
CORE_PEER_LOCALMSPID=org2MSP
CORE_PEER_ID=cli
CORE_PEER_ADDRESS=peer0.org2.example.com:9051
CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
CORE_PEER_TLS_CERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org2.example.
com/peers/peer0.org2.example.com/tls/server.crt
CORE_PEER_TLS_KEY_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org2.example.
com/peers/peer0.org2.example.com/tls/server.key
CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org2.exa
mple.com/peers/peer0.org2.example.com/tls/ca.crt
CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org2.example.
com/users/Admin@org2.example.com/msp
CORE_PEER_TLS_ENABLED=true
bash-5.0# peer chaincode query -C channel2 -n mycode -c '{"Args":["query","ID","name","tel"]}'
002
zhangsan
17345678911
bash-5.0#
```

图 11 数据溯源结果

Figure 11 Result of data traceability

为方便溯源数据查看, 以电商数据溯源为背景, 将区块链溯源数据进行可视化展示。系统后端采用 Spring Boot 框架, 前端采用渐进式的 Vue 框架, 溯源页面展示如图 12 所示。溯源时主要依据数据编号查询, 为验证溯源用户是否拥有访问权限, 需用户在溯源时输入身份证明。在溯源过程中, 同时展示获得当前数据查看权限的用户, 以及已经对当前数据进行访问的用户, 以便数据所有者进行授权与访问的联合分析, 判断用户的非法访问行为, 防止数据泄露。



图 12 溯源页面展示

Figure 12 Display of the traceability page

4.3 性能分析

在传统区块中,只构建Merkel树用来对区块内的交易合法性进行验证,而对区块的验证则需要按照区块哈希摘要的链式链接逐个查询,造成验证时间浪费。由于在区块中加入了Merkel山脉,因此在构造区块时,时间开销增大。通过Go语言模拟在不同交易数量及不同区块高度下,区块构建时间对比,其结果如图13所示。

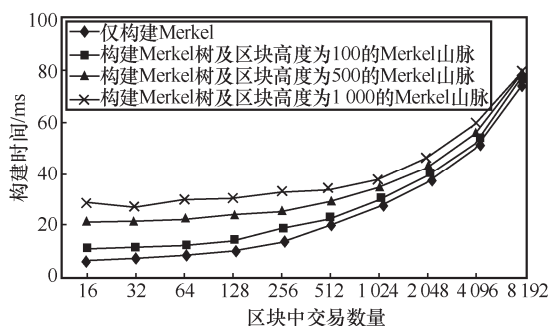


图13 区块构建时间对比
Figure 13 Block build time comparison

图13结果显示,引入Merkel山脉,造成构建区块的时间消耗增加,在同一区块中存储相同数量交易的情况下,随着区块高度增加,构建区块消耗的时间增多。但随着同一区块中存储交易数量的增加,发现无论是否在区块中构建Merkel山脉,构建区块消耗的时间变化趋势均相同,并且区块存储交易数量越多,区块高度对构建区块的时间影响越小。

虽然在区块中引入Merkel山脉造成了区块构建时间增加,但通过观察发现随着交易数量增加,其影响越来越小,并且在区块验证上,能够显著提高区块的验证效率。相比区块验证中节省的时间,构建区块时的额外时间开销是可以接受的。引入Merkel山脉后,区块中数据验证时间对比如图14所示。

当联盟链网络中节点数量、数据流转次数不同时,对数据溯源时间将产生不同的影响,本文用Go语言模拟不同节点数量、不同流转次数情况下的溯源查询时间开销。溯源查询时间对比如图15所示。

结果显示,在数据流转次数相同的情况下,节点数量越多,在数据查询过程中的共识时间开销越大,导致查询时间越长。在节点数量相同的

情况下,随着数据流转次数增加,溯源查询时间逐渐增长,但由于采用了链下缓存的溯源机制,溯源时间增加非常缓慢。

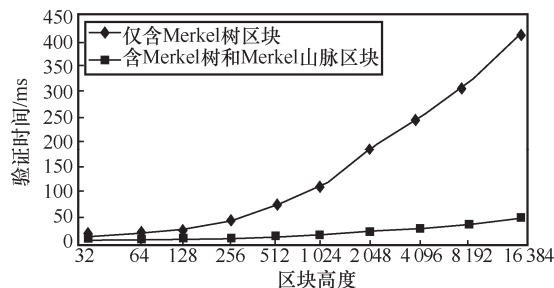


图14 数据验证时间对比
Figure 14 Data validation time comparison

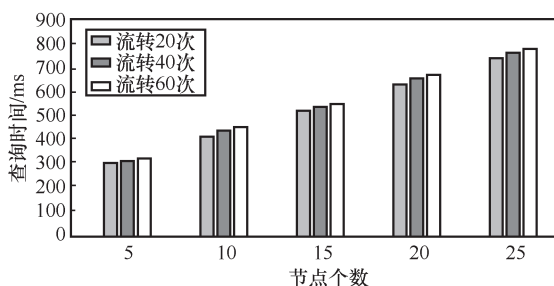


图15 溯源查询时间对比
Figure 15 Traceability query time comparison

在一定范围内,通道数量的增加将提高联盟链系统中的交易吞吐量,但通道数量并非越多越好,引入通道策略,在方便数据传递的同时,增加了联盟链网络的管理成本,当通道数量过多时,必将导致中继联盟链网络因管理成本过高而性能下降。本节通过谷歌远程过程调用向网络注入10万条交易测试联盟网络中通道的处理性能,加入私有数据通道前后,通道处理性能对比如图16所示。

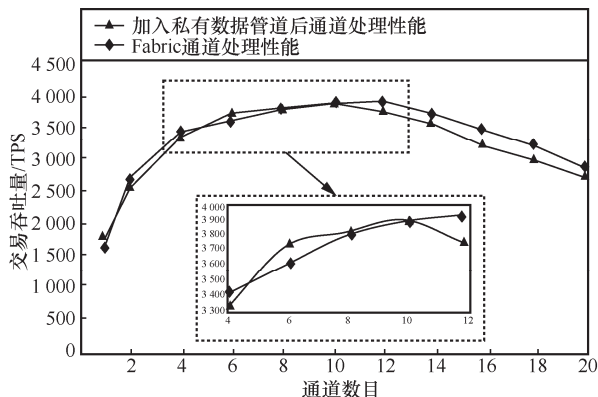


图16 通道处理性能对比
Figure 16 Channel processing performance comparison

结果显示,在通道处理性能达到最大值之前,随着通道数量的增加,加入私有数据管道前后,通道的处理能力并没有明显变化,其处理能力均随着通道数量的增加逐渐增大;添加私有数据管道后,在通道数量为10时交易吞吐率达到最大,在未添加私有数据管道的Fabric通道中,在通道数量为12时吞吐率达到最大,两者的吞吐率最大值均在3 900 TPS左右,没有明显区别。在通道数量达到最佳值之后,两者的吞吐率呈现相似的下降趋势,说明加入私有数据管道前后,对Fabric的通道处理性能没有明显影响。但引入私有数据管道后,实现了同一通道中不同区块链节点之间溯源数据传递的隐私性,在没有明显降低交易吞吐量的同时,提高了溯源数据传递的安全性。

5 结束语

本文针对数据在不同系统间跨域流转带来的跨链数据交互及溯源需求,设计了基于联盟链的数据溯源机制。针对区块链中数据存储及验证存在的不足,将Merkel山脉引入区块链结构中,并给出了数据存储模型,提高了区块验证及数据流转状态溯源效率。面对数据跨域流转中的安全性需求,在设计的联盟链数据通道中引入私有数据管道,通过私有数据管道保障了数据在跨区块链传递过程中的安全性和隐私性。通过跨链溯源场景测试及对比实验,验证了本文溯源机制的可行性及性能优势。针对大量数据并发请求问题,如何提高通道处理能力,有待进一步研究。

参考文献:

- [1] 田有亮, 杨科迪, 王纘, 等. 基于属性加密的区块链数据溯源算法[J]. 通信学报, 2019, 40(11): 101-111.
TIAN Y L, YANG K D, WANG Z, et al. Algorithm of blockchain data provenance based on ABE[J]. Journal on Communications, 2019, 40(11): 101-111.
- [2] 石丽波, 孙连山, 王艺星. 数据起源安全研究综述[J]. 计算机应用研究, 2017, 34(1): 1-7.
SHI L B, SUN L S, WANG Y X. Survey of data provenance security [J]. Application Research of Computers, 2017, 34(1): 1-7.
- [3] 张学旺, 冯家琦, 殷梓杰, 等. 基于区块链的数据溯源可信查询方法[J]. 应用科学学报, 2021, 39(1): 42-54.
ZHANG X W, FENG J Q, YIN Z J, et al. Trusted query method for data provenance based on blockchain[J]. Journal of Applied Sciences, 2021, 39(1): 42-54.
- [4] 谢绒娜. 面向数据跨域流转的访问控制关键技术研究[D]. 西安: 西安电子科技大学, 2020.
- XIE R N. Research on key technologies of access control for cross-domain data exchange[D]. Xi'an: Xidian University, 2020.
- [5] TOSH D K, SHETTY S, LIANG X, et al. Consensus protocols for blockchain-based data provenance: challenges and opportunities[C]//Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON). 2017: 469-474.
- [6] 于戈, 聂铁铮, 李晓华, 等. 区块链系统中的分布式数据管理技术——挑战与展望[J]. 计算机学报, 2021, 44(1): 28-54.
YU G, NIE T Z, LI X H, et al. The challenge and prospect of distributed data management techniques in blockchain systems[J]. Chinese Journal of Computers, 2021, 44(1): 28-54.
- [7] 孙知信, 张鑫, 相峰, 等. 区块链存储可扩展性研究进展[J]. 软件学报, 2021, 32(1): 20.
SUN Z X, ZHANG X, XIANG F, et al. Survey of storage scalability on blockchain[J]. Journal of Software, 2021, 32(1): 20.
- [8] 蔡维德, 郁莲, 王荣, 等. 基于区块链的应用系统开发方法研究[J]. 软件学报, 2017, 28(6): 1474-1487.
CAI W D, YU L, WANG R, et al. Blockchain application development techniques[J]. Journal of Software, 2017, 28(6): 1474-1487.
- [9] 韩妍妍, 张齐, 闫晓璇, 等. 基于区块链的电子文件流转设计与实现[J]. 计算机应用, 2020, 40(11): 3357-3365.
HAN Y Y, ZHANG Q, YAN X X, et al. Design and implementation of electronic file circulation based on blockchain[J]. Journal of Computer Applications, 2020, 40(11): 3357-3365.
- [10] 冯家琦. 基于区块链的数据溯源方法研究与应用[D]. 重庆: 重庆邮电大学, 2021.
FENG J Q. Research and application of data provenance method blockchain-based[D]. Chongqing: Chongqing University of Posts and Telecommunications, 2021.
- [11] 樊玉琦, 盛东, 王伦飞. 基于纠错码的区块链存储优化[J]. 计算机学报, 2022, 45(4): 858-876.
FAN Y Q, SHENG D, WANG L F. Blockchain storage optimization based on erasure code[J]. Chinese Journal of Computers, 2022, 45(4): 858-876.
- [12] 乔蕊, 曹琰, 王清贤. 基于联盟链的物联网动态数据溯源机制[J]. 软件学报, 2019, 30(6): 1614-1631.
QIAO R, CAO Y, WANG Q X. Traceability mechanism of dynamic data in Internet of things based on consortium blockchain[J]. Journal of Software, 2019, 30(6): 1614-1631.
- [13] SUN J, YAO X M, WANG S P, et al. Blockchain-based secure storage and access scheme for electronic medical records in IPFS[J]. IEEE Access, 2020(8): 59389-59401.
- [14] 刘炜, 李阳, 田钊, 等. IDDS: 一种双链结构传染病数据共享区块链模型[J]. 计算机应用研究, 2020(1): 1-6.
LIU W, LI Y, TIAN Z, et al. IDDS: double-chain structure infectious disease data sharing blockchain model[J]. Application Research of Computers, 2020(1): 1-6.
- [15] 张国潮, 王瑞锦. 基于门限秘密共享的区块链分片存储模型[J]. 计算机应用, 2019, 39(9): 2617-2622.
ZHANG G C, WANG J R. Blockchain shard storage model based on threshold secret sharing[J]. Journal of Computer Applications, 2019, 39(9): 2617-2622.
- [16] 贾大宇, 信俊昌, 王之琮, 等. 存储容量可扩展区块链系统的高

- 效查询模型[J]. 软件学报, 2019, 30(9): 2655-2670.
- JIA D Y, XIN J C, WANG Z Q, et al. Efficient query model for storage capacity scalable blockchain system[J]. Journal of Software, 2019, 30(9): 2655-2670.
- [17] 刘耀宗, 刘云恒. 基于区块链 RFI 大数据安全溯源模型[J]. 计算机科学, 2018, 45(S2): 367-368, 381.
- LIU Y Z, LIU Y H. Security provenance model for RFID big data based on blockchain[J]. Computer Science, 2018, 45(S2): 367-368, 381.
- [18] LI X, MA Z, LUO S. Blockchain-oriented privacy protection with online and offline verification in cross-chain system[C]//2022 International Conference on Blockchain Technology and Information Security (ICBCTIS). 2022: 177-181.
- [19] NEISSE R, STERI G, NAI-FOVINO I. 2017. A blockchain-based approach for data accountability and provenance tracking[C]//Proceedings of the 12th International Conference on Availability, Reliability and Security (ARES '17). 2017: 1-10.
- [20] 余涛, 牛保宁, 樊星. FabricSQL: 区块链数据的关系查询[J]. 计算机工程与设计, 2020, 41(10): 2988-2995.
- YU T, NIU B N, FAN X. FabricSQL: relational query of blockchain data[J]. Computer Engineering and Design, 2020, 41(10): 2988-2995.
- [21] LAI C, WANG Y. Achieving efficient and secure query in blockchain-based traceability systems[C]//2022 19th Annual International Conference on Privacy, Security & Trust (PST). 2022: 1-5.
- [22] 刘炜, 王栋, 余维, 等. 一种面向区块链溯源的高效查询方法[J]. 应用科学学报, 2022, 40(4): 623-638.
- LIU W, WANG D, SHE W, et al. An efficient query method for blockchain traceability[J]. Journal of Applied Sciences, 2022, 40(4): 623-638.
- [23] PENG Z, WU H, XIAO B, et al. VQL: providing query efficiency and data authenticity in blockchain systems [C]//2019 IEEE 35th International Conference on Data Engineering Workshops. 2019: 1-6.
- [24] 韩宝富. 基于区块链的云数据完整性审计机制研究[D]. 沈阳: 沈阳工业大学, 2021.
- HAN B F. Research on cloud data integrity audit mechanism based on blockchain[D]. Shenyang: Shenyang University of Technology, 2021.
- [25] BUNZ B, KIFFER L, LU L, et al. FlyClient: super-light clients for cryptocurrencies[C]//2020 IEEE Symposium on Security and Privacy. 2020.

[作者简介]



赵守才（1997- ），男，河南商丘人，信息工程大学硕士生，主要研究方向为区块链安全、信息安全。



曹利峰（1981- ），男，河南禹州人，信息工程大学副教授、博士生导师，主要研究方向为网络安全、信息安全。



杜学绘（1968- ），女，河南新乡人，信息工程大学教授、博士生导师，主要研究方向为信息系统安全、大数据和区块链安全。