

# 基于 Fabric 和 IPFS 文件共享系统设计与实现

杨武文<sup>1,2,3</sup>, 马玉鹏<sup>1,2,3</sup>, 王 轶<sup>1,2,3\*</sup>, 赵 凡<sup>1,2,3</sup>, 王保全<sup>1,3</sup>

(1. 中国科学院 新疆理化技术研究所, 新疆 乌鲁木齐 830011;

2. 中国科学院大学, 北京 100049;

3. 新疆理化技术研究所 新疆民族语音语言信息处理实验室, 新疆 乌鲁木齐 830011)

**摘 要:**文件的共享需求在人们生产生活中普遍存在,传统的文件存储共享系统大多以中心化技术实现,存在单点故障、文件流向不易受控且版本历史信息难以追溯、防篡改性差等问题,面临数据泄漏与滥用等潜在威胁,在特定场景下难以应用。针对这些问题,设计并实现了一种基于 Fabric 与星际文件系统 (InterPlanetary File System, IPFS) 的文件存储与共享系统。结合链上账本与链下可扩展文件系统构建了混合存储,解决了 Fabric 的存储瓶颈并去除存储冗余,实现了文件的防篡改存证、流转信息和历史版本信息的可信追溯;设计并实现了细粒度的权限控制策略,实现了文件资源的受控共享;针对 IPFS 的开放网络特征,采用加密方式,避免了 IPFS 存储中的资源暴露风险,进一步提升了存储文件的安全性;限定相同文件统一密钥,避免相同文件重复加密后存储时浪费额外空间。实验测试结果表明,该系统有效解决了传统文件共享系统中存在的问题,并满足实际应用的性能需求。

**关键词:**超级账本;文件共享;文件存储;星际文件系统;权限控制

中图分类号:TP391

文献标识码:A

文章编号:1673-629X(2023)02-0125-07

doi:10.3969/j.issn.1673-629X.2023.02.019

## Design and Implementation of File Sharing System Based on Fabric and IPFS

YANG Wu-wen<sup>1,2,3</sup>, MA Yu-peng<sup>1,2,3</sup>, WANG Yi<sup>1,2,3\*</sup>, ZHAO Fan<sup>1,2,3</sup>, WANG Bao-quan<sup>1,3</sup>

(1. The Xinjiang Technical Institute of Physics & Chemistry, Chinese Academy of Sciences, Urumqi 830011, China;

2. University of Chinese Academy of Sciences, Beijing 100049, China;

3. Xinjiang Laboratory of Minority Speech and Language Information Processing, The Xinjiang Technical Institute of Physics & Chemistry, Urumqi 830011, China)

**Abstract:** The demand for file sharing is common in people's production and life. Most of the traditional file storage and sharing systems are implemented by centralized technology, and there are some problems such as single point of failure, the file flow is not easy to control, the version history information is difficult to trace, and the poor tamper-proof. Faced with potential threats such as data leakage and abuse, it is difficult to apply in specific scenarios. To solve these problems, a file storage and sharing system based on Fabric and InterPlanetary File System (IPFS) is designed and implemented. Hybrid storage is built by combining on-chain ledger and off-chain expandable file system, which solves storage bottleneck and storage redundancy in Fabric, and implements tamper-proof storage, flow information, and reliable traceability of historical version information of files. A fine-grained access control strategy is designed and implemented to control file resource sharing. In view of the open network characteristics of IPFS, encryption is adopted to avoid the exposure risk of resources in IPFS storage and further improve the security of stored files. The unified key for the same file can avoid wasting extra space when the same file is repeatedly encrypted. The experimental results show that such system effectively solves the problems existing in the traditional file sharing system and meets the performance requirements of practical applications.

**Key words:** hyperledger fabric; file sharing; file storage; IPFS; access control

收稿日期:2022-02-28

修回日期:2022-06-29

基金项目:新疆维吾尔自治区天山青年优青项目(2019Q030);新疆维吾尔自治区重大科技专项(2020A02001);中科院青促会会员资助(2021434)

作者简介:杨武文(1993-),男,硕士研究生,CCF会员(K0807G),研究方向为区块链技术;通讯作者:王 轶(1986-),男,博士,副研究员,CCF会员(98372M),研究方向为区块链技术、数据融合。

## 0 引言

随着信息技术的普及应用,电子文件在人们生活中必不可少。协同工作、远程办公、知识传播等众多场景都对文件的网络存储和共享提出了强烈需求,各类网盘类文件共享产品随之产生并成为人们日常生活学习工作中的重要工具。

传统的网络文件存储与共享系统大多采用中心化服务方式,由第三方平台提供文件存储与共享服务。虽然已有较多成熟的产品并被广泛使用,但在数据安全方面仍存在一些问题。一方面,将个人数据交予第三方机构集中式存储,面临数据泄露和被滥用的风险。另一方面,文件资源在共享过程中不易受控,即使通过日志等方式记录共享过程,仍存在中心节点作恶或日志被篡改的风险,导致一些敏感的资源只能在专网或局域网内传输共享,应用场景受限。另有部分依托于点对点(Peer-to-Peer, P2P)去中心化网络的文件共享方式,在数据安全性方面并无改善,且存在稳定性差、难以保证持久化存储等问题,无法大规模应用。

## 1 相关工作

区块链技术有着不可篡改、可追溯、去中心化<sup>[1]</sup>等特点,已经在金融<sup>[2]</sup>、供应链<sup>[3-5]</sup>、多媒体<sup>[6-7]</sup>、医疗<sup>[8-9]</sup>等领域得到了广泛的应用。利用区块链的特性,将区块链技术应用到文件共享系统中,可以有效改善传统文件共享系统中所面临的一系列问题:区块链的防篡改与可追溯为文件共享记录追溯提供了可信保障,去中心化可以有效防止中心节点故障。但区块链面对海量数据存在存储瓶颈,多节点同步的共享账本将极大地消耗存储资源。针对这一问题,采用链上链下混合的存储架构逐渐成为主流解决方案。

在区块链存储扩展方面, Ali M<sup>[10]</sup>提出了一种将云存储和区块链相结合的方法解决区块链存储问题,将数据主体储到云端,数据的索引存储在区块链,但云端存储仍存在着信任问题。 Saqib Ali 等人<sup>[11]</sup>为 Ping ER (Ping End-to-end Reporting) 项目设计了一个分布式存储架构,架构使用分布式哈希表 (Distribute Hash Table, DHT) 拓展联盟链的存储,增加了该项目的可持续性。而随着 IPFS<sup>[12]</sup>的兴起,与区块链技术相结合已逐步成为拓展区块链存储的重要技术手段<sup>[13]</sup>。 Rafi 等人<sup>[14]</sup>利用区块链和 IPFS 来存储物联网 (IoT) 设备产生的数据,解决了 IoT 设备数量巨大时数据安全性存储的问题,但其仍选取采用工作量证明 (Proof Of Work, POW) 作为共识机制,会造成资源的浪费。

在基于区块链技术的文件存储共享应用方面,也有研究人员做了大量工作。 Muqaddas 等人<sup>[15]</sup>提出了一种基于区块链的数字资产安全数据共享与交付框

架,利用激励机制促进资源拥有者为客户提供有质量的数据,但框架基于以太坊进行构建,存在使用代价高昂的缺点。谭海波等人<sup>[16]</sup>针对档案数据管理中存在的问题,提出了一种基于联盟链和公有链的档案数据保护方法,实现了档案馆联盟的数字档案的保护、验证、恢复与共享,联盟链和以太坊结合,一定程度上减少了系统使用的代价。 Khatal<sup>[17]</sup>提出了一种基于区块链文件共享框架,实现了数据安全共享和版权保护,但其为了实现版权保护,限制了用户只能用其 DApp 才能创建和使用文件数据,用户不能通过该 App 存储分享自己现有的文件和下载文件到本地系统使用。他们大多基于以太坊公链进行实现,使用代价较为高昂,或文件被限制不能脱离平台使用。

该文设计并实现了一个基于 Fabric 的文件存储共享系统,有如下特点:

(1) 把区块链与 IPFS 结合,拓展了区块链的存储;结合 IPFS 的内容寻址特性和加密技术,实现了文件的防篡改、去冗余持久化存储。

(2) 基于属性的控制策略方便用户对文件的操作;权限上链和基于细粒度的权限控制,实现了文件的受控共享。

(3) 系统实现了对操作历史信息 and 文件历史版本的溯源。对相同文件用同一密钥加密处理,在提高文件在 IPFS 中的隐私安全的同时,又不产生额外的存储空间浪费。

## 2 相关技术

### 2.1 超级账本

超级账本 (Hyperledger)<sup>[18]</sup>是由 Linux 基金会创建和管理的一个开源项目。 Hyperledger Fabric 是一个开源企业级许可的分布式账本框架,用于开发解决方案和应用程序,其模块化和多功能设计可满足广泛的行业用例。 Hyperledger Fabric 具有良好的性能、可伸缩性和信任水平。 Fabric 可支持多种键值数据库,如 levelDB 和 CouchDB<sup>[19]</sup>等,该系统选择使用 CouchDB 作为状态数据库,以提供丰富的查询功能。

### 2.2 链码

Fabric 中智能合约<sup>[20]</sup>称为链码 (chaincode),链码用来控制区块链网络中的不同实体或相关方如何相互交互或交易的业务逻辑。链码是独立可运行的应用程序,运行在基于 Docker<sup>[21]</sup>的安全容器中。链码支持多种语言进行编写,如 Go、Java、NodeJS 等。

### 2.3 IPFS

IPFS 集合 BitTorrent、DHT、Git、自验证文件系统 (Self-Certifying File System, SFS) 和密码学等已有技术,提供了一种更加便宜、安全、可快速集成的存储解

决方案。IPFS 采用为数据块内容建立哈希去重的方式存储数据,相同内容的数据块具有同样的哈希,数据的存储成本将会显著下降。IPFS 分为公有集群和私有集群,具有相同的 swarm. key 才能加入同一个私有集群里。通过 IPFS 集群,IPFS 以可持续的速率固定数据,并在出现故障时重试固定,保证数据持久化

存储。

### 3 系统设计

### 3.1 系统架构

如图 1 所示,所设计的文件共享系统包括四部分:客户端、系统服务、Fabric 联盟链和 IPFS 私有集群。

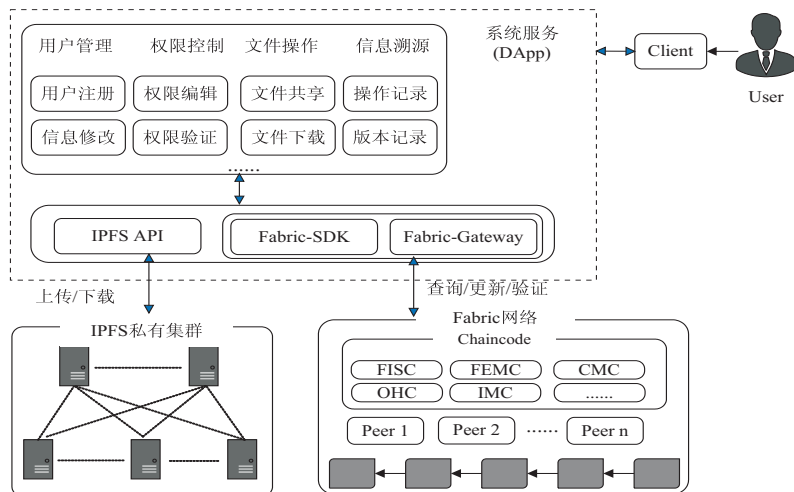


图 1 系统架构

客户端通过良好的界面交互直接给用户提供服务,通过系统服务提供的 RESTful 接口进行访问;系统服务是一种去中心化应用(DApp),通过调用 Fabric-SDK 和 IPFS API,分别与 Fabric 系统和 IPFS 集群进行交互配合,实现用户管理、权限控制、文件操作和信息溯源等系统功能;Fabric 存储用户、部分文件和控制策略等信息,提供链上数据信息的查询、更新以及验证等

功能;IPFS 私有集群则用来存储文件原始数据,保证持久化存储,提供稳定的数据支持。

### 3.2 链码架构

系统主要链码包括文件管理链码 (File Manage Chaincode, FMC)、文件授权链码 (File Authority Chaincode, FAC)、身份管理链码 (Identity Manage Chaincode, IMC) 等, 如图 2 所示。

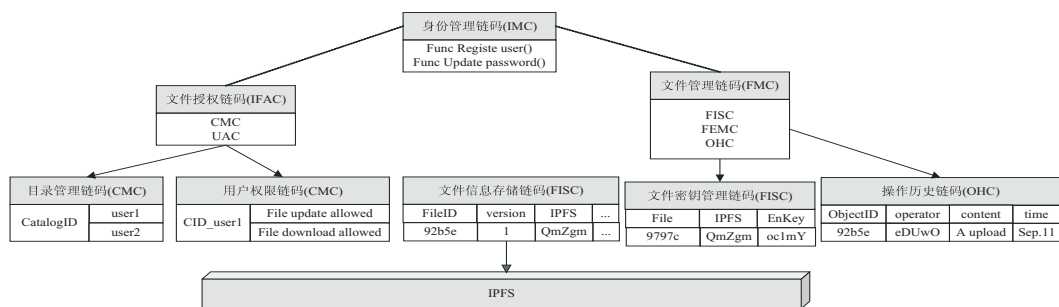


图 2 链码架构

FMC 用于实现文件的保护、验证、溯源、下载等业务逻辑,包括文件信息存储链码 (File Information Storage Chaincode, FISC)、文件密钥管理链码 (File Encryption Key Manage Chaincode, FEMC)、操作历史链码 (Operation History Chaincode, OHC)。FISC 链码用于存储文件的关键信息,包括文件名、摘要信息、类型、属性、上传者、版本等信息,执行文件的上传、下载、更新、共享等操作;FEMC 链码用来管理文件加密密钥,保持与文件加密前后文件指纹的映射关系;OHC 链码用于记录用户的操作,记录了操作目标、操作用户、目标用户、操作具体内容等信息。

FAC 用来实现对文件的细粒度控制,包括目录管理链码(Catalog Manage Chaincode,CMC)和用户权限链码(User Auth Chaincode,UAC)。CMC 链码对目录树进行管理,包括对目录的创建、修改和删除等操作;UAC 链码控制用户对目录及其包含的文件操作权限。

IMC 管理整个链中所有用户的身份信息,包括其通行证书 ID、用户名等对应的信息。

## 4 方法设计与实现

#### 4.1 细粒度权限控制

文件共享系统既要能够实现文件受控共享,又要

有良好的操作性,需要有合理的权限控制策略来保证。

#### 4.1.1 基于属性的文件访问控制策略

根据文件所需共享范围的差异,分别对文件赋予不同的属性。基于属性差别,通过链码进行预处理规则控制,分别执行不同的共享方式和权限访问控制策略。

如图 3 所示,根据业务需求,给文件赋予 public、shared 和 private 三类属性。基于 UAC 链码,以文件属性作为输入,属性为 public 的文件由所有用户任意访问,属性为 shared 的文件根据权限约束进行更加细粒度的控制访问,属性为 private 的文件则只能由文件拥有者进行操作,其他用户不可见。

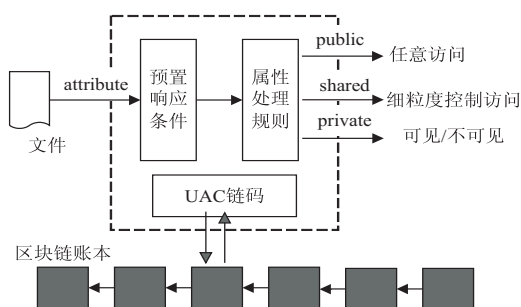


图 3 基于属性访问控制

#### 4.1.2 基于合约的细粒度权限控制

针对需要共享的数据资源,执行细粒度的权限控制与验证,如图 4 所示。

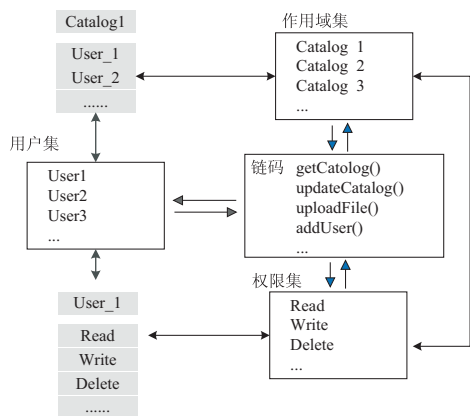


图 4 细粒度权限控制示意图

(1) 规约系统的数据集和文件浏览、下载、删除和共享等动作,将每项操作均内化成一个权限点,UAC 链码为每一个权限点进行编码注册。

(2) 将权限点、作用域和用户之间的关系用数据集上链存储,基于 CMC、IMC 和 UAC 链码进行细粒度的控制约束。

(3) 权限的验证以权限控制链码为核心,以用户操作动作作为权限点进行输入,以用户和权限点的关联作为授权条件。

#### 4.1.3 面向层级的权限赋予与验证

为便于进行批量文件资源的共享管理,以文件夹

为对象,采用面向层级的作用域权限管理。

依托于文件夹的层级目录结构,将作用域的根目录域用户进行权限绑定,下级目录自动继承相应权限。若需要在整体作用域下对部分文件资源权限进行调整,可以以子目录方式,新建一个作用域,重新关联用户和相应权限点,兄弟节点间权限互不影响。

进行权限验证时,根据文件资源所在层级位置,进行权限查询与验证。验证流程如算法 1 所示。

算法 1:面向层级权限验证算法。

输入 (nodeId, userId)

输出 (authCode)

```
authCode = getAuth (nodeId, userId); // 查询当前节点用户权限
if authCode exist then // 如果权限存在
    return authCode; // 返回权限码
else
    while (authCode = null and nodeId.father != null) // 进入循环
        nodeId = nodeId.father; // 转到父节点
        authCode = getAuth (nodeId, userId); // 查询父节点权限
    end while // 退出循环
return authCode; // 返回查询到的权限点
end if
```

用户在子目录权限验证时,若未查询到当前目录用户对对应权限点,则向上递归查询验证,直到达到边界条件。

## 4.2 存储策略

#### 4.2.1 基于 IPFS 的链下存储扩展

区块链由于全节点备份和只可追加的特性,本身面临较大的存储压力,文件本身并不适合写入区块。为了对区块链的存储进行扩展,把文件内容本身存入链下 IPFS,将文件的摘要信息和 IPFS 返回的指纹写入区块作为链上存储。链上文件存储结构定义如下:

```
fileStore = { file_id, cid, name, ipfs_hash, size, abstract, version, time, user }
```

其中, file\_id 是用户在当前目录下首次创建时生成的唯一识别码 (Universally Unique Identifier, UUID), 作为文件唯一识别 id, 也作为查询键值; cid 是文件当前所属文件夹 id; name, ipfs\_hash, size, abstract, version, time, user 分别对应文件名, 文件 IPFS 哈希, 大小, 摘要信息, 版本号, 上传时间, 文件拥有者。

#### 4.2.2 基于内容寻址的去冗余存储和版本追溯

IPFS 依据内容寻址, 存储时将文件数据划分为 256K 的多个块, 根据块内容计算 hash, 组合后再次哈希计算构造出整个文件的地址哈希。两个相同内容的文件只需要存储一次, 达到去冗余效果。若文件部分



变动产生变化仅变动内容所在块的 hash 也会同步产生变化,此时文件同步会产生新的 hash,如图 5 所示。

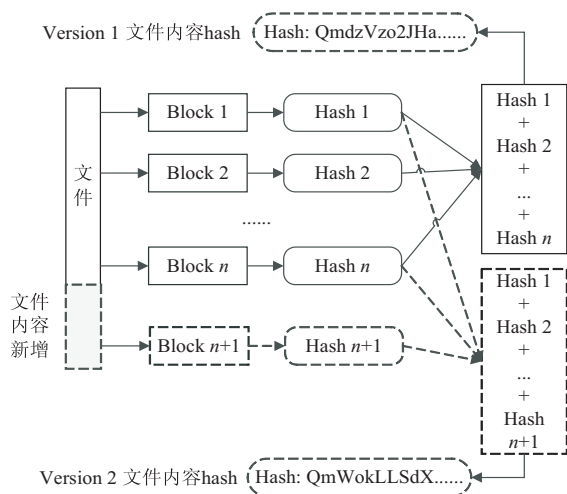


图5 IPFS 文件更新示意图

Fabric 维护了一个历史索引数据库,可以快速实现对区块链账本历史数据的查询。系统调用 `GetHistoryForKey()` 接口可查询键值对应的历史变化记录。

结合 Fabric 和 IPFS 的特性,以文件的 `file_id` 为键值,在历史数据库中检索 `ipfs_hash` 的变动记录,以及相应的文件修改信息,从 IPFS 中获取历史文件数据,实现了对文件历史版本的追溯。

#### 4.2.3 文件加密存储

IPFS 是一个内容寻址的开放分布式动态共享网络,在应用与 IPFS 进行交互时不可避免地需要进行文件内容 hash 的明文传输,处于集群中的任一节点在获得文件 hash 的情况下均能下载文件,这导致一定的文件泄露风险。

针对这一问题,该系统采用加密存储的方式进一步提升文件的安全性,实现对非法访问的可见不可读。其中针对 IPFS 内容寻址特性,对相同内容的文件采用同一密钥进行加密,使用 FEMC 链码对密钥进行统一管理,避免了存储冗余。

加密存储流程为:

(1) 计算文件 hash,调用 FEMC 链码判断系统是否已经存储过相同文件;

(2) 若存在,则调用 FISC 查询该文件其相关信息,然后在当前用户目录下创建该文件,上传成功;

(3) 若不存在,则需要加密后再进行存储,将新的密钥及其对应加密前后的 hash 信息交由 FEMC 进行管理。关键代码如下所示。

算法 2:文件存储加密算法。

输入:(file, userId)

输出(status)

fileHash = SHA256(file); // 计算文件 hash

```

if fileExist(fileHash) then //如果文件存在
    fileInfo = getFileInfo(fileHash) //获取文件信息
    status = createNewFile(fileInfo, fileHash, userId); //在用户
    目录下创建文件
    return status; //返回成功创建结果
else
    key = randomKey(); //产生随机密钥
    encryptedFile = AESencrypt(file, key) //文件加密得到
    密文
    ipfsHash = ipfs.add(encryptedFile); //上传至 IPFS
    keyManage(fileHash, ipfsHash, key); //管理文件 hash 与
    密钥
    status = createNewFile(fileInfo, fileHash, user.id) //根据
    文件信息创建文件
    return status; //返回创建成功状态
end if

```

用户下载文件时,系统只会给授权的合法用户进行下载解密,整个过程用户本身不能拿到密钥。非法用户在使用泄露的 IPFS hash 下载文件后,并不能解密使用,从而实现了文件的隐私安全保护。

## 5 系统测试与分析

系统实现使用 Fabric-2.2.0 作为区块链框架,链码开发使用 Go 语言进行编写;IPFS 则使用 go-ipfs v0.8.0 客户端进行本地私有集群模式的搭建;客户端主要通过 angular 8 框架、Ant Design 框架进行构建;系统服务使用了 SpringBoot 框架,通过 Fabric-SDK-Java 调用 Fabric 链码,通过 java-ipfs-api 实现了与 IPFS 的接口调用。在文件加解密部分,为了提高处理速率,采用了加解密速度较快的 AES 对称加密算法。

核心功能实现了用户管理、权限编辑、文件操作和信息溯源等功能。用户管理主要包括用户注册、用户信息编辑等功能;权限编辑包括了权限的赋予、编辑和申请验证等功能;文件操作包括了文件的上传、共享、下载、更新和检索等常用功能;信息溯源包括了对文件历史信息等溯源,以及用户的操作记录和共享记录等。

### 5.1 系统性能测试

实验设备为 1 台笔记本电脑,配置为 16 GB 内存, Intel Core i5 处理器,两台服务器搭载了 CentOS 7.9 系统,16 GB 内存, E5620 处理器。使用 Fabric 自带的 etcdRaft 共识协议,共两个组织,每个组织一个证书颁发机构(Certificate Authority, CA),两个 peer 节点,一个排序节点。其中 Fabric 初始参数配置为: BatchTimeout: 2 s, MaxMessageCount: 500, AbsoluteMaxBytes: 10 MB, PreferredMaxBytes: 2 MB。

系统中对文件的操作以上传下载的时间开销最大,主要包含文件到 IPFS 网络的上传下载,文件本身

的加解密和相关信息的上链存储三个过程。为了探究各个阶段处理性能,以四组不同尺寸大小的文件做上传下载测试。文件平均尺寸分别为 0.5 M,4 M,10 M 和 80 M,每组 50 个测试样例。测试结果取平均值,实验数据如表 1 所示。

表 1 系统性能测试结果 (BatchTimeout = 2)

	文件大小 /M	总耗时 /ms	交易耗时 /ms	IPFS 耗时 /ms	加密耗时 /ms
上传	0.5	4 589	4 291	292	2
	4	4 990	4 268	691	9
	10	5 671	4 273	1 305	22
	80	13 352	4 375	8 478	176
下载	0.5	2 571	2 335	232	1
	4	3 175	2 368	800	4
	10	4 118	2 306	1 794	10
	80	14 934	2 332	12 478	59

由表 1 测试结果可以看出,系统总耗时,IPFS 耗时和加解密耗时均随着文件尺寸增大而增大,其中加解密过程所占耗时对于整个文件上传下载处理速度影响几乎可以忽略不计。区块链系统交易耗时相对稳定,且上传交易耗时是下载交易的两倍,这是由于上传操作 Fabric 系统需要处理两笔数据写入交易:一笔是创建文件信息,另一笔是文件密钥的管理。而下载操作,只是调用了查询交易,没有账本数据的写入操作。

表 2 系统性能测试结果 (BatchTimeout = 0.2)

	文件大小 /M	总耗时 /ms	交易耗时 /ms	IPFS 耗时 /ms	加密耗时 /ms
上传	0.5	979	676	294	2
	4	1 372	636	707	10
	10	1 967	674	1 226	22
	80	9 925	672	8 710	233
下载	0.5	624	389	232	1
	4	1 317	391	734	4
	10	1 924	396	1 498	10
	80	11 325	386	10 719	55

可以注意到,在文件尺寸较小时,相关信息的上链交易是主要的性能瓶颈。分析其原因,主要在于 Fabric 出块速度受四个参数共同影响,需满足任意一个参数才能完成区块的上链写入。在本实验中由于测试输入并发不够,上链信息并不满足其他参数条件,只能等待时间达到 BatchTimeout 设定的时间条件才能触发数据打包,出块完成交易。针对这一问题,调整 Fabric 出块配置参数,将 BatchTimeout 时间设为 0.2 s,其他参数保持不变,验证交易耗时的提升。实验结

果如表 2 所示。

可以看出,调整后系统信息上链耗时有了明显的降低,在应用中可以根据实际的并发压力,调整 Fabric 的网络参数以提升系统性能。综合而言,系统的整体性能可以接受,能够满足用户的实际使用需求。

## 5.2 安全性分析

该系统采用的许可型区块链 Fabric,结合 IPFS 实现文件资源的安全存储与共享。系统安全性从以下三个方面来保证:

**Fabric 安全性:**利用 Fabric 认证机制保护,只有经过系统认证的合法用户才能进入区块链进行操作。区块链节点间通信可以通过 TLS 来保证传输层的安全。Fabric 系统会区分参与的节点和应用程序角色,为访问资源的操作设置严格的权限控制。

**IPFS 安全性:**IPFS 对数据进行分布式存储,通过集群可把整个系统存储统一调度,能够实现资源自动修复和去冗余,自带容灾备份功能。通过 swarm. key 组成私有集群,可以与外部数据有效隔离,保证数据仅在 IPFS 私有集群中流转。IPFS 存储文件时将文件分割成数据块存储,并通过计算数据块的哈希值来进行唯一标识该文件,只有通过文件的 CID 才能从 IPFS 网络中获取到该文件。

**文件流转安全:**细粒度权限控制链码,可以有效控制文件的合法流转。随机生成密钥对文件进行加密,密钥与文件本身分别存储在区块链系统和 IPFS 中,在整个文件的流转过程中,密钥由系统进行管理,其他任何人无权单独取出密钥,即使非法用户通过 IPFS 集群中节点获取到了加密文件,也无法获取文件解密对应的密钥,从而保证了文件的流转安全。

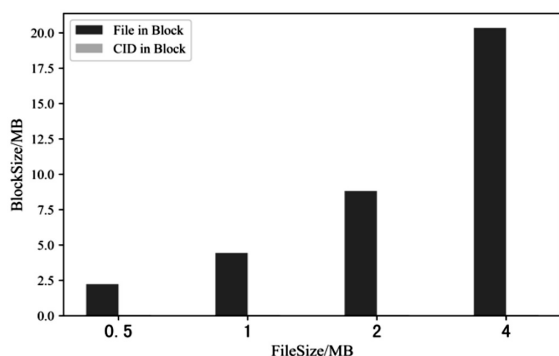
通过模拟权限受限用户,解析客户端与服务端通信的数据,可以获得当前用户可见但无权操作的列表文件对应的 CID。对获取到的 CID 进行下载验证,通过文件 CID 可从 IPFS 集群中的节点获取到文件,但由于无法获取到对应的文件密钥,无法对获取到的文件进行解密。因此系统安全性得到保障。

## 5.3 存储空间分析

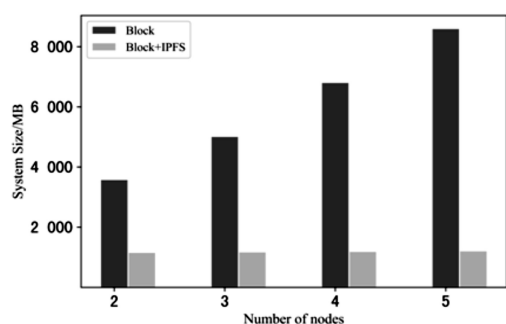
分别用区块链、区块链+IPFS 的方式存储文件,对比两种方式的单个区块平均大小和系统整体存储消耗。测试样例为四组不同大小的文件,文件平均尺寸分别为 0.5 M,1 M,2 M,4 M,每组 50 个测试样例进行存储测试,测试结果如图 6 所示。

图 6(a) 所示为区块存储文件和区块存储文件 CID 两种方案的单个区块大小,由于存储 CID 时区块平均大小仅为 60 KB,在图中所占比例太小几乎不可见。而采用区块存储文件时,区块的平均大小约为所存文件的 4.8 倍。在存储超过 10 M 的文件测试时,会

导致区块链系统崩溃,这是由于存储文件时是作为单笔区块交易进行,单笔交易数据过大导致系统产生错误,所以 Fabric 中单笔交易数据不能过大。



(a) 区块大小对比图



(b) 系统总空间消耗图

图6 存储性能测试

使用区块存储文件时会导致区块大小尺寸五倍增大,对区块数据进行了多次测试和解析。区块在存储<key,value>数据时,Fabric 为了安全性进行数据处理,导致写入区块数据增大。如利用链码存储数据<Demon,good>,解析对应交易区块时数据存储为{"key": "Demon", "value": "eyJmaWxlIjoRGVtb24iLCJmaWxlX2NvbnRlbnQiOiJnb29kIn0="},从而导致数据增大。而当<key,value>中value数据大小为0.5 M时,解析出value对应的数据大小为2.3 M。

图6(b)展示的为不同节点数量存储相同文件时的系统整体存储空间消耗,其中IPFS的备份数量设定为3。采用区块链账本存储文件导致整个系统的存储空间消耗极具增大,且随着节点数量增加线性增长。

综上所述,该系统所采用的方式具有极大的空间节省性能。

## 6 结束语

系统利用区块链技术克服了传统文件存储共享系统存在的不可溯源和防篡改性差的问题;通过细粒度的权限控制实现了文件受控共享;文件采用先验证是否存在,然后确定是否加密存储的方法,在提高隐私安全的同时,没有带来额外的空间浪费。通过实验测试,系统的整体性能可以满足日常生活需要。在未来工作

中,将会探究对已存储文件的内容进行分布式检索等功能,探索新的功能和方法,进一步完善系统。

### 参考文献:

- [1] 张志威,王国仁,徐建良,等. 区块链的数据管理技术综述[J]. 软件学报,2020,31(9):2903-2925.
- [2] NAKAMOTO S. Bitcoin: a peer-to-peer electronic cash system[EB/OL]. 2009. <https://bitcoin.org/en/bitcoin-paper>.
- [3] PEÑA M, LLIVISACA J, SIGUENZA-GUZMAN L. Blockchain and its potential applications in food supply chain management in Ecuador[C]//The international conference on advances in emerging trends and technologies. [s. l.]: Springer, 2019: 101-112.
- [4] MALIK S, DEDEOGLU V, KANHERE S S, et al. Trust-chain: trust management in blockchain and iot supported supply chains[C]//2019 IEEE international conference on blockchain (Blockchain). Atlanta: IEEE, 2019: 184-193.
- [5] LENG K, BI Y, JING L, et al. Research on agricultural supply chain system with double chain architecture based on blockchain technology[J]. Future Generation Computer Systems, 2018, 86: 641-649.
- [6] 朱彦霞,张雪萍,华南,等. 基于IPFS及区块链的互联网融媒体中心平台设计[J]. 电子设计工程, 2021, 29(18): 10-16.
- [7] 张国潮,唐华云,陈建海,等. 基于区块链的数字音乐版权管理系统[J]. 计算机应用, 2021, 41(4): 945-955.
- [8] TARALUNGA D D, FLOREA B C. A blockchain-enabled framework for mHealth systems[J]. Sensors, 2021, 21(8): 2828.
- [9] 张超,李强,陈子豪,等. Medical Chain: 联盟式医疗区块链系统[J]. 自动化学报, 2019, 45(8): 1495-1510.
- [10] ALI M. Trust-to-trust design of a new Internet[D]. Princeton: Princeton University, 2017.
- [11] ALI S, WANG G, WHITE B, et al. A blockchain-based decentralized data storage and access framework for pinger[C]//2018 17th IEEE international conference on trust, security and privacy in computing and communications/12th IEEE international conference on big data science and engineering (TrustCom/BigDataSE). New York: IEEE, 2018: 1303-1308.
- [12] PRÜNSTER B, MARSALEK A, ZEFFERER T. Total eclipse of the heart - disrupting the InterPlanetary file system[J]. arXiv; 2011. 00874, 2020.
- [13] 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): 1-20.
- [14] RIFI N, RACHKIDI E, AGOULMINE N, et al. Towards using blockchain technology for IoT data access protection[C]//2017 IEEE 17th international conference on ubiquitous wireless broadband (ICUWB). Salamanca: IEEE, 2017: 1-5.
- [15] NAZ M, AL-ZAHRANI F A, KHALID R, et al. A secure

(下转第152页)