

Research on File Security Verification Method Based on Blockchain Technology

Guo Zhiqiang

School of Computer and Information Engineering, Henan University, Kaifeng 457000, China

Email: 1036105855@qq.com

Abstract

Blockchain technology, as a popular technology at present, is practiced in various fields due to its characteristics of non-tampering, traceability, and decentralization. However, in the process of implementing the blockchain system, how to ensure that the file content is not falsified for the file data uploaded to the blockchain system becomes a key issue that restricts the development of the blockchain system. The traditional file security verification method mainly performs hash operations on user files and obtains corresponding hash values. However, in this process, the calculated hash values cannot be guaranteed to be modified. This paper proposes a file security verification method based on blockchain technology, and verifies the feasibility of the method through experiments.

Keywords: Blockchain Technique; Hash-operation; Not-tamperable

基于区块链技术的文件安全验证方法研究

郭志强

河南大学 计算机与信息工程学院 河南 开封 475000

摘要: 区块链技术作为当下流行的技术,以其内容不可篡改、可追溯、去中心化等特点,在各个领域都在进行实践落地。但是在实现区块链系统过程中,针对上传到区块链系统的文件数据怎么保证文件内容不被篡改,成为制约区块链系统发展的关键问题。传统的文件安全验证方法主要是针对用户文件进行散列运算并获取对应的哈希值,但是在此过程中无法保证计算出来的哈希值不被修改。本文基于区块链技术提出了一种文件安全验证方法,并通过实验验证了方法的可行性。

关键词: 区块链技术; 哈希运算; 不可篡改

引言

区块链技术,作为一个集分布式技术、密码学、共识机制、智能合约^[1]于一体的技术组,以其防篡改、可追溯的特性,越来越被广大学者所重视。在区块链的发展过程中,利用区块链技术特性,解决在某一领域中的问题,并设计出基于区块链网络的系统成为主流。在区块链系统中,由于区块链中形成区块的大小限制,无法把文件或文件组内容完全存入区块链中。当系统被恶意攻击时,区块链系统中文件将会存在被修改的风险。传统的文件安全校验方法,主要是在文件传递之前对文件进行哈希运算,主要选择的方法是 md5^[2,3]和 SHA-1^[4]算法生成文件的哈希值,在传递文件之后计算获取的文件是否与文件初始的哈希值有所不同。本文提出的基于区块链技术的文件安全性校验,利用散列算法和 Markle 二叉树,获取文件或者文件组的哈希值,并把生成的哈希值写入到区块链网络中,当再次使用文件或文件组时校验文件或者文件组重新计算生成的哈希值,对比两次的哈希值是否不同,保证文件或者文件组的安全性。

1 相关技术

1.1 区块链技术

区块链（Bolckchain）源自比特币^[5]的底层实现技术，它是一本分布式的去中心化账本。区块链是由一个个区块连接形成，每一个区块分成区块头部和交易记录两个部分。区块头部包含区块的版本号、时间戳、随机数、难度值、前一个区块的哈希值，和所有交易组成的 Markle 二叉树的根哈希值。

区块链技术的特征主要包括：匿名性，去中心化，去信任，防篡改。匿名性是指任意节点都可以自由的加入离开区块链网络。去中心化是指加入区块链网络的所有节点都会保存网络上账本的完整副本，任何一个节点的故障都不会导致区块链网络的瘫痪。去信任是指节点之间存在共识机制，不用依赖中心化的节点。防篡改是指在区块链形成的过程中，第 n 个区块记录着第 $n-1$ 个区块的哈希值。当修改其中某一个区块的信息时，接下来的区块，也随之改变。而且随着区块链网络中节点的增加，区块链网络将会越安全。攻击者至少需要控制网络中 51% 的节点，篡改的信息才会被允许记录到区块链账本中。

在单位时间内，区块链中的数据以数据块的数据结构进行存储。从整体上来看，区块链所有数据以链式的数据结构进行存储，这种数据结构如图 1 所示。

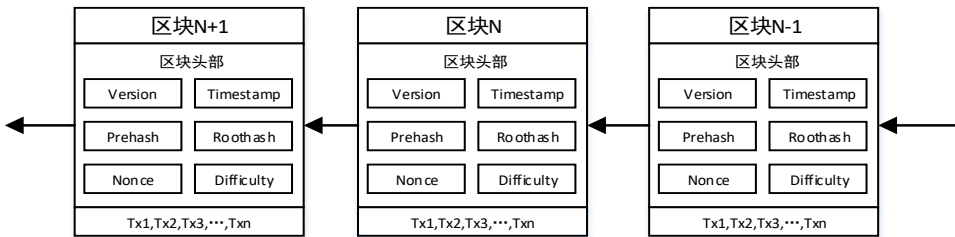


图 1 区块链结构

1.2 Markle 树

在介绍 Merkle 树之前，需要先介绍哈希表^[6]的概念。对于大块数据的检验，可以把这个大块数据等分成若干个小块数据。然后对每个小的数据块进行哈希，再将小数据的哈希值合并起来，最后将合并的结果再次进行哈希，这就是哈希表的形成过程。哈希表最终的获取的结果是根哈希值。哈希表可以理解作为一种特殊的 Merkle 树。在 Merkle 树中，将会把数据分成若干个小块数据然后放到最底层作为 Merkle 树的叶子节点。开始时，把单独的小数据块进行哈希运算处理，然后再将两个子节点的哈希值合并在一起，再次进行哈希运算形成父节点，重复上述步骤，直到只有一个根节点哈希。

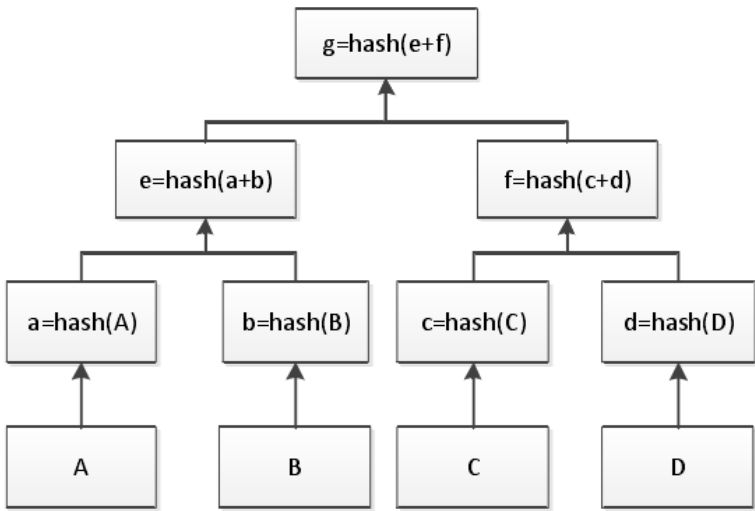


图 2 Merkle 树结构

如图 2 所示, A, B, C, D 表示四个数据块, 依次计算这四个数据块的哈希值分别为 a, b, c, d。然后将 a 和 b 组合一起继续进行哈希运算, 将会得出哈希值 e, 按照上述同样的步骤, c 和 d 组合经过哈希运算得出哈希值 f。最后, e 和 f 组合一起得到的哈希值 g 就是 Merkle 树的根节点。在验证数据时, 只需要比较数据的根节点值是否相等。如果根节点的哈希值相同, 那么两份数据就是相同的; 如果不一样, 需要从上往下进行遍历查找。假设我们另外一份数据块在数据 B 出现了问题, 并且这一份数据经过哈希运算后的哈希值依次是 a', b', c', d', e', f, g'。由于根节点哈希值 g 不等于 g', 那么开始比较根节点的两个子节点, 将会发现左子节点哈希值 e' 不等于 e, 但是右子节点哈希值 f 等于 f, 所以在下一步的过程中开始比较哈希值为 e 的节点的两个子节点的哈希值。此时, 将会发现哈希值 a' 等于 a, 但是哈希值 b' 不等于 b, 经过分析之后便发现是在数据块 B 出现了差异。

1.3 SHA-256 算法

SHA-256 散列算法是美国国家标准技术研究所 NIST 在 2002 年推出 SHA-2 系列哈希算法, 输入的最大长度是 2^{64} 位的消息, 输出是 256 位的消息摘要。其算法的主要流程如下所示:

1. 进行比特填充。对初始报文进行填充, 保证报文长度与 448 模 512 同余, 即

$$\text{Length} = 448 \bmod 512$$

对缓存进行初始化。使用大小为 256 位的缓存区存放散列函数的中间结果和最终结果。256 位的缓存区分 8 个 32 位的寄存器, 用 A, B, C, D, E, F, G, H 表示。寄存器存放对自然数中前 8 个质数 (2, 3, 5, 7, 11, 13, 17, 19) 的平方根的小数部分取前 32 位而来。

2. 处理主循环模块。报文消息以 512 位分组为单位进行处理后, 将进行 64 步循环操作。每一轮的输入均为当前处理的消息分组和得到的上一轮输出的 256 位缓冲区 A, B, C, D, E, F, G, H 的值。
3. 获取最终的 256 位的 hash 值。所有的 512 位的报文消息块都处理完分组之后, 最后一个分组的处理得到的结果即为最终输出结果。

2 基于区块链技术的文件安全验证方法设计

基于区块链技术的文件安全验证方法的流程图 3 所示:

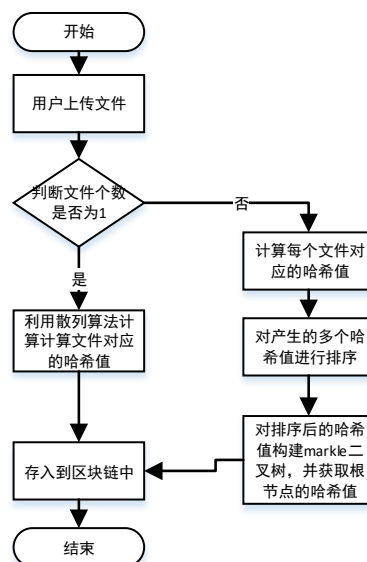


图 3 方法流程图

用户上传文件到区块链系统后, 首先会判断上传文件的个数。如果上传文件的个数为一个时, 则直接

利用 SHA-256 算法计算出文件对应的哈希值，并放入到区块链中。如果上传文件的个数大于一个时，首先对文件组进行哈希计算，并记录对应的文件的哈希值，再对生成的哈希值进行大小排序并作为 Markle 二叉树的叶子节点。最后利用 Markle 二叉树计算出二叉树的根节点哈希值，并把根节点哈希值记录到区块链网络中。

区块链网络主要是使用 Hyperledger fabric^{[7][8][9]}区块链架构搭建。用户首先需要搭建 Hyperledger fabric 区块链网络架构所需要的支持环境。首先是 Docker 环境，Go 语言的环境，Java 语言的开发环境。在所有的支撑环境搭建完全后，搭建基于双组织的区块链节点，每个组织包含两个节点，分为锚节点和记账节点。然后创建频道，双组织同时加入该频道。创建相应的提案提交到网络上并存储到网络中。

3 方法实现及实验设计

3.1 基于区块链技术的文件安全验证方法的实现

1. 用 SHA-256 算法计算文件的哈希值核心代码如下所示：

```
public String applySha256(String input) {
    try {
        MessageDigest digest = MessageDigest.getInstance("SHA-256");
        byte[] hash = digest.digest(input.getBytes("UTF-8"));
        StringBuffer hexString = new StringBuffer();
        for (int i = 0; i < hash.length; i++) {
            String hex = Integer.toHexString(0xff & hash[i]);
            if (hex.length() == 1)
                hexString.append('0');
            hexString.append(hex);
        }
        return hexString.toString();
    } catch (Exception e) {
        throw new RuntimeException(e);
    }
}
```

该代码主要使用 MessageDigest 类的 getInstance 方法调用 SHA-256 算法，对传递到函数的字符串转换成比特数据并进行运算，获取最终的 Hash 值。

2. 利用 Markle 二叉树获取根节点的 Hash 值的代码如下所示：

```
public void rootMarkle(List<String> list) {
    if (list.size() != 1) {
        List<String> listTemp = new ArrayList<String>();
        for (int i = 0; i < list.size(); i++) {
            if (i % 2 == 0) {
                if ((i + 1) != list.size()) {
                    String temp1 = list.get(i);
                    String temp2 = list.get(i + 1);
                    listTemp.add(applySha256(temp1 + temp2));
                } else {
                    listTemp.add(applySha256(list.get(i)));
                }
            }
        }
    }
}
```

```

    }
}
listAll.add(listTemp);
rootMarkle(listTemp);
}
}

```

该代码主要是用来计算 Markle 二叉树的根节点 Hash 值。首先新建一个顺序表，通过迭代，调用函数本身计算二叉树的父节点，直至顺序表中只有一个 Hash 值，则此时这个 Hash 值即为 Markle 二叉树的根节点 Hash 值。

3. 存入区块链网络

```

ChannelClient channelClient = fabClient.createChannelClient(FabricConfig.CHANNEL_NAME);

Channel channel = channelClient.getChannel();

Peer peer = fabClient.getInstance().newPeer(FabricConfig.ORG1_PEER_0, FabricConfig.ORG1_PEER_0_URL);
EventHub eventHub = fabClient.getInstance().newEventHub("eventhub01", "grpc://10.12.26.174:7053");
Orderer orderer = fabClient.getInstance().newOrderer(FabricConfig.ORDERER_NAME,
FabricConfig.ORDERER_URL);

channel.addPeer(peer);
channel.addEventHub(eventHub);
channel.addOrderer(orderer);
channel.initialize();

TransactionProposalRequest request = fabClient.getInstance().newTransactionProposalRequest();
ChaincodeID ccid = ChaincodeID.newBuilder().setName(FabricConfig.CHAINCODE_1_NAME).build();
request.setChaincodeID(ccid);
request.setFcn("initPatent");
request.setArgs(args);
request.setProposalWaitTime(1000);

Map<String, byte[]> tm2 = new HashMap<>();
tm2.put("HyperLedgerFabric", "TransactionProposalRequest:JavaSDK".getBytes(UTF_8));

tm2.put("method", "TransactionProposalRequest".getBytes(UTF_8));
tm2.put("result", ":").getBytes(UTF_8));
tm2.put(EXPECTED_EVENT_NAME, EXPECTED_EVENT_DATA);
request.setTransientMap(tm2);
Collection<ProposalResponse> responses = channelClient.sendTransactionProposal(request);

```

在存入区块链网络时，首先需要创建频道并对其进行初始化；其次操作需要加入频道的节点并且设置对应的监听事件。最后编写程序创提案并绑定需要提交的节点，调用需要执行此操作的链码。在节点根据背书策略验证完全提案之后，将会转发到排序节点进行排序生成区块并广播至网络中。

3.2 实验设计

实验中客户端将会在 Windows 系统中搭建，服务器端在 ubuntu 系统中搭建，利用网络实现客户端和服务器的通信。在具体实现的过程中，区块链网络我们将搭建 Hyperledger fabric 区块链环境。我们利用 web 系统进行文件的上传，计算并得到文件的散列值并存储到区块链网络中。

客户端配置方案如表 1 所示：

表 1 客户端实验环境配置方案

类别	详细配置
操作系统	Windows 10
开发环境	JDK1.8
客户端开发工具	Eclipse
开发语言	Java、html
其他软件	Tomcat

服务器端配置方案如表 2 所示：

表 2 客户端实验环境配置方案

类别	详细配置
操作系统	Ubuntu 17.10
开发环境	Go1.6
开发语言	JDK1.8
其他软件	docker
开发语言	Java
其他软件	Couchdb

首先进行区块链环境的搭建，主要包含以下几个步骤：

- 1、在 Ubuntu 系统下 Go 语言环境和 Java 语言环境的搭建；
- 2、在 Ubuntu 系统下 Docker 的安装；
- 3、利用 Cryptogen 工具生成 Hyperledger fabric 区块链节点对应的证书；
- 4、利用 Cryptogen 工具生成频道文件和创始块；
- 5、编写区块链网络启动程序 Build.sh。

其次，客户端编写程序获取文件或者文件组的哈希值，并和区块链网络通信，存入区块链网络主要包含以下几个步骤：

- 1、编写 Java 程序利用 Fabric-sdk-java 创建频道；
- 2、编写 Java 程序使得节点加入频道；
- 3、频道初始化；
- 4、创建提案并发送至节点；
- 5、获取提案返回状态。

3.3 实验分析

本实验针给出三个大小为 10MB，20 MB，50 MB 的文件进行实验，分别为 A 文件、B 文件和 C 文件。利用上述方法进行，当只上传 A 文件时，记录到区块链网络的该文件的哈希值如下图 4 所示：

74be79789c01c3eefa6f54e85be2e3f7d657a183b16495591c40d54dde396091

图4 文件A 哈希值

当修改文件A的内容时，此时计算出来的哈希值为df251740f6307520440425b02419a6f33cbf5d6a53c9e7b310883f4233095cdf。因此可以验证出上传到区块链网络的文件A内容已经被修改。

当同时上传三个文件时针对文件的A、B、C时，得到的哈希值如图5所示：

1b1838e4a5995f6d112742218201852778f12b59792722d78e3c75d8cea7bfe7|

图5 A、B和C三个文件经过运算的哈希值

当修改其中B文件时，对应的三个文件经过运算产生的哈希值为1b1838e4a5995f6d112742218201852778f12b59792722d78e3c75d8cea7bfe7。此时将会发现，修改了其中的一个文件，其最终计算出来的哈希值都是不同的。

4 结论

本文主要研究了区块链技术在实践过程中针对文件数据如何保证其安全性的问题，提出了针对文件的区块链处理方法。通过验证，得到了较好的结果，但在此过程中也存在着一些问题。接下来的研究工作将从以下几个方面进行研究：

- (1) 区块链技术具有分布式数据库的技术特性，加入网络中的每一个节点都将会存储网络中账本完整副本，但是随着区块链网络中账本数据的增加，每个节点都将面临着数据存储的压力。在接下来的工作中，这将是区块链技术着重需要解决的问题。
- (2) 在本文中，虽然提出了针对文件数据的安全验证方法，保证了内容的原始性。但是无法保证上传到区块链系统中的文件数据被恶意删除。如何利用区块链技术的理念，保证上传到区块链系统中的文件数据安全完整的存放在系统中，这将是区块链技术在应用过程中研究重点之一。

参考文献

- [1] 冯翔，刘涛，吴寿鹤，周广益.区块链开发实战 Hyperledger fabric 关键技术与案例分析 [M]. 北京：机械工业出版社，2018：2 - 12.
- [2] “Java 防止文件篡改之文件校验和”. March 16,2018. <https://blog.csdn.net/JavaBuilt/article/details/79583921>.
- [3] “md5sum 对下载传输的文件进行安全性校验”. April 7,2018. https://blog.csdn.net/qq_26442553/article/details/79841969.
- [4] 申屠青春，宋波，张鹏，王晓明，季宙栋，左川民.区块链开发指南 [M]. 北京：机械工业出版社，2017：54 - 57.
- [5] Nakamoto S. Bitcoin: A peer-to-peer electronic cash system[J]. Consulted, 2008.
- [6] 马如林，蒋华，张庆霞. 一种哈希表快速查找的改进方法[J]. 计算机工程与科学, 2008, 30(9).
- [7] Androulaki E, Barger A, Bortnikov V, et al. Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains[J]. 2018
- [8] Sousa, João, Bessani A, Vukolić, Marko. A Byzantine Fault-Tolerant Ordering Service for the Hyperledger Fabric Blockchain Platform[J]. 2017
- [9] Benhamouda F, Halevi S, Halevi T. Supporting Private Data on Hyperledger Fabric with Secure Multiparty Computation[C]// 2018 IEEE International Conference on Cloud Engineering (IC2E). 2018.