

基于刚性内存的区块链协议改进

程穗^{1,2}, 林宪正^{1,2}, 俞能海^{1,2}

(1. 中国科学技术大学网络空间安全学院, 安徽 合肥 230001;
2. 中国科学院电磁空间信息重点实验室, 安徽 合肥 230001)

摘要: 首先, 提出了一种新的区块链链式协议, 该协议允许调整挖掘过程数据的读取时间。这种策略缩小了 ASIC 矿机和 CPU 挖掘新区块速度的差距。然后通过该协议, ASIC 矿机的优势将会受到限制。最后, 分析了这种新区块链结构的安全性, 并将其与原始区块链协议进行了安全性比较。

关键词: 区块链; 刚性内存函数; 多重相关协议

中图分类号: TP391

文献标识码: A

doi: 10.11959/j.issn.2096-109x.2020054

Improvement of the blockchain protocol based on memory-hard function

CHENG Sui^{1,2}, LIN Xianzheng^{1,2}, YU Nenghai^{1,2}

1. School of CyberScience, University of Science and Technology of China, Hefei 230001, China
2. CAS Key Laboratory of Electro-magnetic Space Information, Hefei 230001, China

Abstract: A new blockchain chain protocol was proposed, which allowed adjusting the loading time of the mining process. This strategy reduced the ASIC and CPU mining rate of the gap, so that the using of ASIC mining is limited. Finally, the safety of this new chain structure was analyzed and compared to the original chain protocol.

Key words: blockchain, memory-hard function, multiple related protocol

1 引言

比特币是一种去中心化的电子加密“货币”^[1]。作为一个账单系统, 比特币不依赖于中央机构发行新币和维持交易, 而是依赖分布式系统来维护

交易列表, 这个交易列表即区块链^[2]。区块链是通过密码学方法生成的一串相关数据块。新区块始终链接到其前一个区块。因此, 区块链是一种协议, 当新区块被挖掘出来时, 区块链的数据发生变化^[3]。

收稿日期: 2020-01-14; 修回日期: 2020-03-19

通信作者: 林宪正, sjlin@ustc.edu.cn

基金项目: 中科院百人计划青年俊才 C 类基金 (KJ2100330001); 安徽省自然科学基金 (BJ2100330001)

Foundation Items: The Hundred Talents Program of Chinese Academy of Sciences (KJ2100330001), The Natural Science Foundation of Anhui Province, China (BJ2100330001)

论文引用格式: 程穗, 林宪正, 俞能海. 基于刚性内存的区块链协议改进[J]. 网络与信息安全学报, 2020, 6(5): 21-26.

CHENG S, LIN X Z, YU N H. Improvement of the blockchain protocol based on memory-hard function [J]. Chinese Journal of Network and Information Security, 2020, 6(5): 21-26.

在比特币系统中,所有用户都可以参与由工作量证明(PoW)协议维护的区块链^[4-6]。PoW协议要求有效块的哈希值小于预定的阈值^[7]。每个矿工通过调整哈希函数的输入值(在区块中称为 nonce)进行有效区块的计算^[8]。获取有效区块后,矿工将广播该区块到整个网络,其他矿工在验证该区块的有效性后停止当前高度的区块挖掘^[9]。

在传统区块链中,除了创世区块外,每个区块的区块头均包含其父区块的哈希值。每个新区块的生成方式都是通过更改随机数,并不断计算其区块头的哈希值,直到其小于当前难度。因此,挖掘只是一个纯粹的计算过程。计算速度越快,挖掘出新区块的可能性越大。

普通的采矿设备经历了从 PC 到 GPU 和 FPGA 的多种变化,直到 ASIC 矿机出现。当前,比特币采矿市场已经被 ASIC 矿机所主导,并且无法通过使用其他采矿设备来营利^[10]。ASIC 矿机是致力于挖矿过程。但是,早期使用 ASIC 矿机存在一系列问题:

- 1) 随着网络的计算能力持续迅速地提高,ASIC 矿机芯片的寿命变得非常短(约 6 个月);
- 2) 投资 ASIC 采矿设备会增加成本(电力成本和冷却成本);
- 3) 由于行业的不成熟,采矿设备的延迟交付导致芯片被淘汰。

尽管使用 ASIC 矿机进行挖矿已经相当成熟,并且挖矿设备的寿命更长,但挖矿业已经从个人领域转移到了大型的专业挖矿中心。因此,挖矿行业对个体矿工非常不友好。此外,整个网络中的大部分计算能力集中在少数人手中,这不仅增加了资源浪费,而且增加了 51% 攻击的可能性。这与比特币^[11]最初的设计目的背道而驰。

为了打破挖矿行业中 ASIC 矿机的垄断,已经有多种方法被提出,如具有刚性内存的函数。刚性内存函数是一类很难并行处理的哈希函数。在此功能中,完成给定计算问题所需的时间主要取决于从内存中读取数据的时间,这可以减少总时间成本上的计算能力部分。因此,通过增加从内存中读取数据的次数,I/O 所占的时间将控制新区块挖掘的时间,从而达到抵抗 ASIC 矿机的目的。

现在已经有与刚性内存相关的算法被提出,如 Scrypt 和 Primecoin^[12]。这些算法可以分为两个步骤:第一步是从一个随机数种子生成一个伪随机数数组,该数组存储在主存储器中;第二步是计算新区块的哈希值,哈希函数的输入是相关区块区块头的串联,这些区块是从伪随机数数组中随机选择的。

由于必须读取几个伪随机数,数据读取将影响哈希计算的时间。然而,在 Scrypt 函数中,每个伪随机数都是通过前一个伪随机数计算出来的,ASIC 矿工可以通过仅存储具有奇数索引^[13]的伪随机数来将主存储器中使用的空间减少一半。如果哈希函数需要带有偶数索引的伪随机数,则可以从前一个随机数计算出该值。因此,对于这些刚性内存函数,计算的时间和内存属性是可以进行调节的。时间和内存的折中可以减小哈希计算中数据读取的影响,使 ASIC 矿机和通用 CPU 之间的性能差距仍然很大。

为了解决此问题,需要一种新的哈希函数,该函数不具有时间内存折中的功能。也就是说,如果存储器的大小小于阈值,则挖矿性能将显著下降,并且没有有效的方法来设计一种具有时间/内存权衡的方法。此外,希望协议中的数据读取时间是可控的,因此,本文提出一种新的区块链协议,在该协议中,每个区块除了与其父区块相关联之外,还与之前的多个区块相关联。该协议决定了主导挖矿时间的是 I/O,而不是计算速率。

2 基于刚性内存的协议模型

2.1 协议模型

本节将详细描述所提出的协议。该协议具有两个参数 (n, m) ,其中, n 表示在区块挖掘过程中可能被使用的区块的数量, m 表示在内存中选择的哈希值的数量。为了提高协议的性能,应将这 n 个哈希值存储在缓存或内存中,否则处理器必须花费大量时间在区块挖掘过程中访问这些值。在提出的协议中,区块链中最后 n 个区块的哈希值依次表示为 $\{N_i | i = 0, \dots, n-1\}$,其中 N_0 表示最新区块。 n 个哈希值中的 m 个(包含 N_0)将用于新区块的哈希计算。给定随机数的值,以下

是选择 m 个哈希值的步骤。

1) 令 $t_0 = \text{nonce}$ 和 $i=1$ ，计算 $t = \text{hash}_0(t_0)$ 。

2) 令 $t_i = N_i$ ，计算 $t = \text{hash}_0(t_i)$ 。如果 $i = m$ ，输出 $(t_0, t_1, \dots, t_m)$ ，否则 $i = i+1$ 并重复步骤 2)。

可以看到， hash_0 的共域是 $\{0, \dots, n-1\}$ ，那么有 $\text{hash}_0(x) = x \pmod n$ 。哈希值可以通过 $R = \text{hash}(t_0 \| t_{1,Y} \| \dots \| t_{m,Y} \| N_0)$ 进行计算。如果哈希值 R 满足目标难度，则挖掘过程结束，否则选择新的 nonce 并重复该过程以找出新的 m 个哈希值。在实现中，这 n 个哈希值存储在循环队列中，如图 1 所示，避免数据移动。

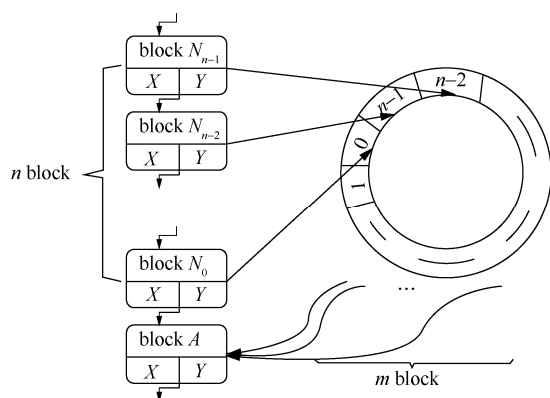


图1 环队列存储方式
Figure 1 Circular Queue

2.2 协议分析

2.2.1 时间/内存属性权衡

作为内存依赖型的 PoW 算法，Scrypt 算法包含两个步骤。第一步，依次生成 n 个伪随机数，其中每个随机数都是由其前一个随机数计算而来。第二步，通过使用伪随机顺序选择的多个伪随机数来生成哈希值。通过在存储器中存储偶数位索引的 $\frac{n}{2}$ 个伪随机数，可以进行时间内存的权

衡。由于在哈希运算中，ASIC 矿机的计算速率比通用 CPU 快得多，ASIC 矿机可以牺牲部分计算能力来减少所需的内存。因此，这种刚性内存算法无法达到较好的抗 ASIC 效果。

在本文提出的协议中，每个哈希值不能直接通过其前一个哈希值计算出来，因此该协议不具有时间/内存权衡的属性，这个属性被称为 Memory-hard。因此，如果 n 个哈希值不能完全存储在内存中，那么应当存储在辅助存储设备（如

硬盘）中。当哈希计算过程需要使用对应的哈希值时，处理器必须访问硬盘读取数据，这将减慢整个挖掘速度。

2.2.2 加载时间分析

在本文提出的协议中，对内存是有限制的，因此当内存大小小于 n 时，区块的挖掘速度将显著降低。 s 表示存储在内存中的哈希值的数量，当 $s < n$ 时，本节提供了计算所需的哈希值的平均读取时间。令 t_0 表示从内存中读取数据的时间， t_1 表示从辅助存储设备中读取数据的时间。

当 $s \geq n$ 时，哈希计算所需数据的平均读取时间为 mt_0 。当 $s < n$ 时，哈希计算所需数据的平均读取时间为 $m \left(\frac{s}{n} t_0 + \frac{n-s}{nt_1} \right)$ 。通常， t_1 是 t_0 的数万倍，因此，合适的 n 值可以限制 ASIC 矿机在挖矿哈希计算过程中的优势。

3 安全性分析

本节将通过对区块链交易进行篡改攻击来分析区块链的安全性。在每个区块的区块头，令 Y 表示区块中交易的哈希值，令 X 表示区块的其余部分。本文将从单个区块的篡改攻击和连续两个区块的篡改攻击来分析协议的安全性。

(1) 单个区块篡改攻击

首先分析对区块链中一个区块进行篡改的难度。对于如图 2 所示的传统区块链，区块 A 是区块 B 的父区块。攻击者篡改了区块 A 中的交易，这将改变区块 A 中 Y 的值。区块 A' 表示被篡改后的区块， Y' 表示篡改后交易的哈希值。如果修改了区块 A 中的交易，则区块 A' 必须满足

$$\text{hash}(A_x \| A_y) = \text{hash}(A_x \| A'_y) \quad (1)$$

假设找到合适的 A'_y 满足式(1)的概率为 P 。

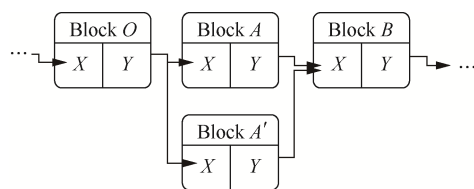


图2 对传统区块链的攻击
Figure 2 Attack on traditional blockchain

对于本文提出的协议,如图 3 所示,每个区块都与其父区块以外的 m 个区块相关联,当区块 N_0 被篡改时,应满足

$$\begin{aligned} \text{hash}(N_{d_1,Y} \parallel N_{d_2,Y} \parallel \cdots \parallel N_{d_i,Y} \parallel \cdots \parallel N_{d_m,Y} \parallel N_{0,X} \parallel N_{0,Y}) = \\ \text{hash}(N_{d_1,Y} \parallel N_{d_2,Y} \parallel \cdots \parallel N_{d_i,Y} \parallel \cdots \parallel N_{d_m,Y} \parallel N_{0,X} \parallel N_{0,Y'}) \end{aligned} \quad (2)$$

其中,每个 d_i 由 2.1 节中的算法计算。也就是说,对于 $i=0, \dots, m$, 有 $d_1 = \text{hash}_0(A_{\text{nonce}})$ 和 $d_i = \text{hash}_0(N_{d_{i-1}})$ 。假设找到合适的 $N_{0,Y'}$ 满足式(2)的概率表示为 q 。在最坏的情况下,如果 $d_i \neq 0$, 则对于 $i=0, \dots, m$, 在式(1)和式(2)中更改的长度是相同的。由生日攻击^[14]可知,有 $q = p$ 。

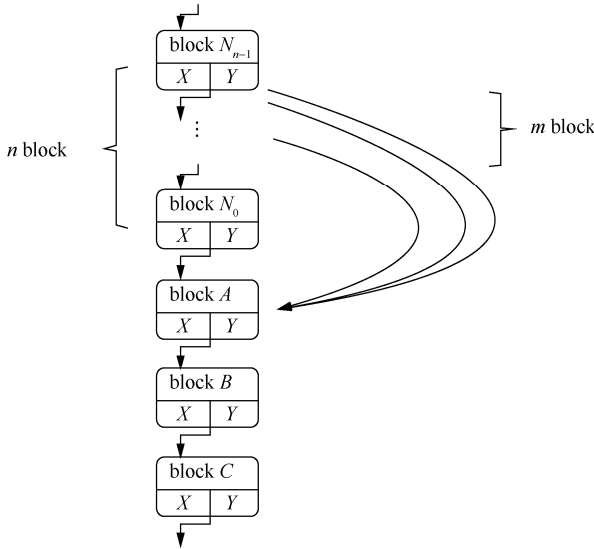


图 3 新的区块链结构
Figure 3 New blockchain structure

攻击者将用于区块 A 的哈希计算中使用到的区块 N_0 篡改为 N'_0 。此外, N_0 用于其他 t 个表示为 $G_1, G_2, \dots, G_t$ 的区块的哈希计算中。令 U_i 表示用于区块 G_i 哈希计算中使用的 $m+1$ 个哈希值,令 U_0 表示用于区块 A 哈希计算中使用的 $m+1$ 个哈希值的集合。对于 $i=0, \dots, t$, $\text{hash}(N_0) \in U_i$, 有

$$\begin{aligned} \text{hash}(u_{i,0} \parallel \cdots \parallel u_{i,k} \parallel \cdots \parallel u_{i,m-1} \parallel u_{i,m}) = \\ \text{hash}(u_{i,0} \parallel \cdots \parallel u'_{i,k} \parallel \cdots \parallel u_{i,m-1} \parallel u_{i,m}) \end{aligned} \quad (3)$$

其中,每个 $u_{i,j} \in U_i$, $j=0, \dots, m$ 。值得注意的是,对于 $i=0, \dots, t$, $u_{i,k} = \text{hash}(N_{0,Y}) \in U_i$ 和 $u'_{i,k} =$

$\text{hash}(N_{0,Y'})$ 表示替换后的哈希值。

这 $t+1$ 个方程是独立的。在最坏的情况下,只有 $u_{i,k} = \text{hash}(N_0) \in U_i$, 因此概率为 $q^{t+1} = p^{t+1} \leq p$ 。这表明,当 $t \geq 1$ 时,所提出的协议比传统的区块链协议更安全。

(2) 连续两个区块篡改攻击

如图 4 所示,如果将区块 A 篡改为 A' , 并且将区块 B 篡改为 $B' = B'_{X'} \parallel B'_{Y'}$, 则在传统区块链协议中,当区块 A 被篡改为区块 A' 时,成功的攻击必须满足

$$B'_{Y'} = \text{hash}(A') \quad (4)$$

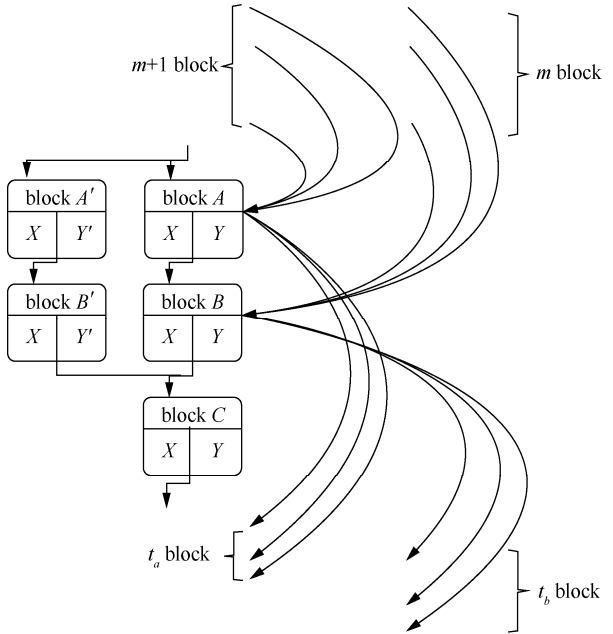


图 4 对新区链的相邻篡改攻击
Figure 4 Adjacent tamper attack on new blockchain

当区块 B 被篡改为区块 B' 时,成功的攻击必须满足

$$\text{hash}(\text{hash}(A') \parallel B_{Y'}) = \text{hash}(\text{hash}(A) \parallel B_Y) \quad (5)$$

假设满足式(5)的概率表示为 p_1 。

在本文提出的协议中,区块 B 和区块 C 都与 $m+1$ 个之前的区块相关联。对于 $i=0, \dots, m$, 令 $(B_i)_{i=0}^m$ 表示区块 B 哈希计算过程中使用的 $m+1$ 个哈希值组成的向量,令 $(C_i)_{i=0}^m$ 表示区块 C 哈希计算过程中使用的 $m+1$ 个哈希值组成的向量。值得注意的是,有 $B_0 = \text{hash}(A)$ 和 $C_0 = \text{hash}(B)$ 。那么

必须满足

$$B_{X'} = \text{hash}(B_1 \parallel \cdots \parallel B_m \parallel B'_0) \quad (6)$$

其中, $B'_0 = \text{hash}(A')$ 。当区块 B 被篡改为区块 B' 时, 必须满足

$$\text{hash}(C_1 \parallel \cdots \parallel C_m \parallel C_0) = \text{hash}(C_1 \parallel \cdots \parallel C_m \parallel C'_0) \quad (7)$$

其中, $C'_0 = \text{hash}(B')$, $B' = B_{X'} \parallel B_{Y'}$ 。假设满足式(7)的概率表示为 q_1 。比较式(5)和式(7), Y' 的长度和哈希函数的输出长度相同。因此, 通过生日攻击, 有 $p_1 = q_1$ 。

如果区块 B 的哈希计算过程使用了区块 A 的哈希值, 则将其他 t_a 个区块定义为 $GA_1, GA_2, \dots, GA_{t_a}$ 。同样地, 如果区块 C 的哈希计算过程使用了区块 B 的哈希值, 则将其他 t_b 个区块定义为 $GB_1, GB_2, \dots, GB_{t_b}$ 。

对于 $i=1, 2, \dots, t_a$, 令 UA_i 表示参与区块 GA_i 的哈希计算的 $m+1$ 个哈希值组成的向量, 令 UB_i 表示参与区块 GB_i 的哈希计算的 $m+1$ 个哈希值组成的向量。特别地, UA_0 对应区块 B 哈希计算的 $m+1$ 个哈希值向量, UB_0 对应区块 C 哈希计算的 $m+1$ 个哈希值向量。对于 $i=0, 1, \dots, t_a$, 有 $\text{hash}(A) \in UA_i$ 。对于 $i=0, 1, \dots, t_b$, 有 $\text{hash}(B) \in UB_i$, 那么可以得到

$$\begin{aligned} & \text{hash}(ua_{i,0} \parallel \cdots \parallel ua_{i,k} \parallel \cdots \parallel ua_{i,m-1} \parallel ua_{i,m}) = \\ & \text{hash}(ua'_{i,0} \parallel \cdots \parallel ua'_{i,k} \parallel \cdots \parallel ua_{i,m-1} \parallel ua_{i,m}) \end{aligned} \quad (8)$$

其中, 当 $j=0, \dots, m$ 时, 有 $ua_{i,j} \in UA_i$ 。式(8)中的 $ua_{i,k} = \text{hash}(A_j) \in UA_i$, 并且当 $i=0, \dots, t$ 时, 将 $ua'_{i,k} = \text{hash}(A'_{j'})$ 定义为篡改后的哈希值。同样地, 必须满足

$$\begin{aligned} & \text{hash}(ub_{i,0} \parallel \cdots \parallel ub_{i,k} \parallel \cdots \parallel ub_{i,m-1} \parallel ub_{i,m}) = \\ & \text{hash}(ub_{i,0} \parallel \cdots \parallel ub'_{i,k} \parallel \cdots \parallel ub_{i,m-1} \parallel ub_{i,m}) \end{aligned} \quad (9)$$

其中, 当 $j=0, \dots, m$ 时, 有 $ub_{i,j} \in UB_i$ 。式(9)中的 $ub_{i,k} = \text{hash}(B_j) \in UB_i$, 并且当 $i=0, \dots, t$ 时, 将 $ub'_{i,k} = \text{hash}(B'_{j'})$ 定义为篡改后的哈希值。

可以看出, 这 t_a+1 和 t_b+1 个方程都是独立的。因此, 当 $t_a \geq 0$ 时, 将区块 A 篡改为区块 A' 并满足式(8)的概率为 $P_1 = q_1^{t_a+1}$; 当 $t_b \geq 0$ 时, 将区块 B 篡改为区块 B' 并满足式(9)的概率为

$P_2 = q_1^{t_b+1}$ 。因此, 攻击成功的概率为 $P = P_1 P_2 = q_1^{t_a+t_b+2} < p_1$ 。可以看出, P 比在传统的区块链协议中攻击成功的概率低。

4 协议改进

为了提高协议实现的效率与稳固该协议的刚性属性, 本节从相关联的区块数 m 的确定与新区块计算使用的哈希函数的输入两方面进行分析。

(1) 相关联的区块数的确定

由 2.1 节中的协议步骤可知, 本文提出的协议在新区块的挖掘过程中, 每选择一个随机值 nonce , 就需要重新从内存中读取 m 个区块的哈希值。令 x 表示比特币网络中计算一个新区块平均所需的计算次数, t_2 表示 ASIC 矿机进行一次哈希计算所需的时间, t_3 表示普通 CPU 进行一次哈希计算所需的时间。由 2.2.2 节可知, 当 $s \geq n$ 时, ASIC 矿机进行一轮挖矿计算所需的时间为 $t_2 + mt_0$, CPU 矿机进行一轮挖矿计算所需的时间为 $t_3 + mt_0$; 当 $s < n$ 时, ASIC 矿机进行一轮挖矿计算所需的时间为 $t_2 + m\left(\frac{s}{n}t_0 + \frac{n-s}{n}t_1\right)$, CPU 矿机进行一轮挖矿计算所需的时间为 $t_3 + mt_0$ 。可以看出, 哈希计算的时间在 ASIC 矿机和 CPU 矿机进行一轮挖矿计算的时间中所占的比例可忽略不计。因此, 为了达到抗 ASIC 矿机的效果, 应增大 n 的取值, 而 m 的取值大小在抗 ASIC 方面没有突出的作用。增大 m 的值仅仅增强了区块链稳定性, 但同时导致节点在进行区块验证时面临更大的压力, 新区块的验证过程也变得更加烦琐。因此, 为了减轻验证节点的负担, 取 $m=1$ 。

(2) 哈希函数的输入

在传统区块链的区块挖掘过程中, 新区块仅与其前一个区块相关联, 因此其哈希计算过程中, 哈希函数的输入仅包含其前一个区块的哈希值。在本文提出的协议中, 新区块除了与其前一个区块相关, 还与 m 个之前的区块相关。假设 $m=2$, 令 T_0 表示当前区块, T_1 、 T_2 和 T_3 表示 T_0 的父区块。 T_1 为 T_0 的前一个区块, T_2 和 T_3 为随机选择的父区块, T_3 的区块高度小于 T_2 。按照区块的形成时间, 区块 T_3 和 T_2 比区块 T_1 更早出现。因此, 在对区块 T_0 进行哈希计算的过程中, 需要用到其

3个父区块的哈希值 $\text{hash}(T_3||T_2||T_1)$ 。然而,当按照区块高度由低到高对区块哈希值进行串联作为哈希函数的输入时,ASIC矿机可能不需要对所有 n 个区块进行存储,这种情况是由某些哈希函数是线性输入导致的。因此,顺序串联可能导致本协议不是完全刚性的,这与本文提出的完全刚性的属性不符,对哈希函数输入的串联方式进行了更改。本文将新区块的前一个区块的哈希值放在首位,然后按照区块高度由低到高进行串联。对于 T_1 、 T_2 和 T_3 ,计算新区块 T_0 时使用的哈希函数输入的串联方式为 $\text{hash}(T_1||T_3||T_2)$ 。

5 结束语

本文提出了一种在区块链上的完全刚性内存的协议。该协议是一种无记忆功能,没有时间/内存折中的协议。因此,内存中必须存储区块链的最后 n 个区块的哈希值。然后,随机选择 n 个哈希值中的 m 个和最新的哈希值串联来计算新区块的哈希值。因此,每个区块不仅与其父区块相关联,而且与 n 个先前的区块中的 m 个相关联。分析表明,本文提出的协议是一种完全刚性的存储协议,并且证明了所提出协议的安全性优于常规区块链协议。

参考文献:

- [1] SWAN M. Blockchain: blueprint for a new economy[M]. O'ReillyMedia, Inc., 2015.
- [2] ZYSKIND G, NATHAN O. Decentralizing privacy: using blockchain to protect personal data[C]//2015 IEEE Security and Privacy Workshops (SPW). 2015: 180-184.
- [3] NARAYANAN A, BONNEAU J, FELTEN E W. et al. Bitcoin and cryptocurrency technologies: a comprehensive introduction[P]. Princeton University Press, 2016.
- [4] VUKOLI M. The quest for scalable blockchain fabric: proof-of-work vs. BFT replication[C]//International Workshop on Open Problems in Network Security. 2015: 112-125.
- [5] ANTONOPOULOS A M. Mastering bitcoin: unlocking digital cryptocurrencies[M]. O'Reilly Media, Inc., 2014.
- [6] BÖHME R, CHRISTIN N, EDELMAN B, et al. Bitcoin: economics, technology, and governance[J]. Journal of Economic Perspectives, 2015, 29(2): 213-238.
- [7] UNDERWOOD S. Blockchain beyond bitcoin[J]. Communications of the ACM, 2016, 59(1): 15-17.
- [8] MARCO I, LAKHANI K R. The truth about blockchain[J]. Harvard Business Review 2017, 95(1): 213-238.
- [9] CROSBY M. Blockchain technology: beyond bitcoin[J]. Applied Innovation Review, 2016(7): 6-10.
- [10] TAPSCOTT D, TAPSCOTT A. Blockchain revolution: how the technology behind bitcoin is changing money, business, and the world[M]. Penguin Group, 2016.
- [11] NAKAMOTO S. Bitcoin: a peer-to-peer electronic cash system[EB].
- [12] ALWEN J, CHEN B, PIETRZAK K, et al. Script is maximally memory-hard[C]//Annual International Conference on the Theory and Applications of Cryptographic Techniques. 2017: 33-62.
- [13] PERCIVAL C, JOSEFSSON S. The script password-based key derivation function[S]. RFC 7914. 2016.
- [14] BELLARE M, KOHNO T. Function balance and its impact on birthday attacks[C]//International Conference on the Theory and Applications of Cryptographic Techniques. 2004: 401-418.

[作者简介]



程穗(1995-),男,湖北黄冈人,中国科学技术大学硕士生,主要研究方向为区块链和矿策略。



林宪正(1981-),男,中国台湾人,中国科学技术大学特任研究员、博士生导师,主要研究方向为信道编码算法、应用于存储系统的纠删码设计。



俞能海(1964-),男,安徽无为,中国科学技术大学教授、博士生导师,主要研究方向为多媒体信息检索、图像处理与视频通信、数字媒体内容安全。