比特币：一种对等电子现金系统

Satoshi Nakamoto
satoshin@gmx.com
www.bitcoin.com

摘要：一种电子现金完全对等的版本或将允许线上支付在不经过金融机构的条件下直接从一方送达另一方。数字签名提供了部分解决办法，但是如果受信任的第三方机构仍然被要求避免双重花费那么它最主要的优势 。我们意在找到一种利用对等网络解决双重花费问题的方法。网络时间戳交易被打乱加入基于哈希工作量证明的持续链中，格式一种除非重写否则不可修改的记录。最长链不仅可以作为一系列事件的见证，同时也是它来自最大算力池的证明。只要多数被节点控制的算力没有联合起来攻击网络，它们就会生成最长链并且超过攻击者。这个网络本身要求极小的结构。信息尽最大努力地广播，同时节点可以自由地退出和再加入网络，接受最长工作证明链作为它们离开时所发生事情的证明。

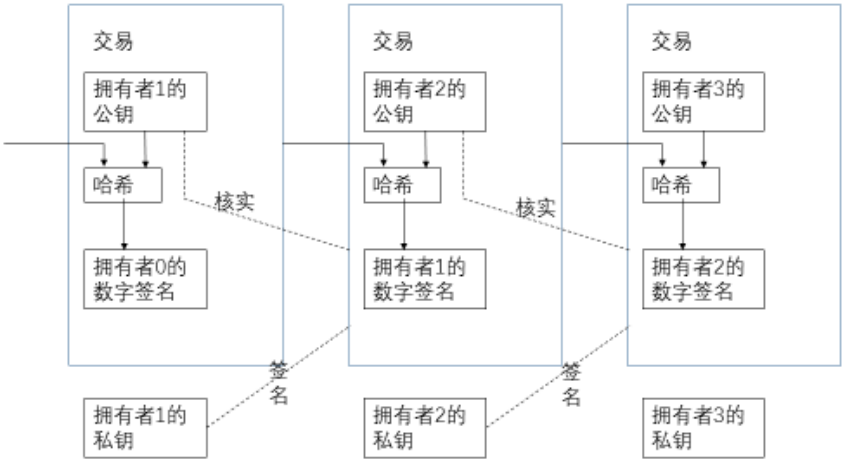
一、 介绍

互联网上的商业活动已经走向几乎完全依赖于金融机构作为受信三方来维护电子支付过程。虽然这套系统能够很好满足大多数商业活动，它仍然遭受来自于基于模型信任的脆弱性。完全不可逆的商业是真的不太可能，毕竟金融机构不可避免调解纠纷。调解费用增加了商业费用，限制了最低限度实际交易的大小并且切断了小额临时交易的可能性，并且对于达成不可逆交易能力的缺失于不可逆服务而言存在着一种更广泛的费用。可逆的可能性存在，催生了真实传播的需求。商家必须警惕他们的客户，让他们获得比他们所需要的更多的信息。相当一部分欺诈无可避免地需要被承担。这样支出的不确定性可以通过个人使用现金的方法避免，但是没有机制可以在没有受信任方的情况下通过通信渠道进行支付。

基于加密证明而不是信任的电子支付系统是我们现在所需要的，这允许任意两个有意愿的组织在不需要第三方受信机构的情况下直接相互交易。交易回溯在计算上的不切实际保护了卖方不受欺骗，同时日常托管机制可以使得保护买方很容易实施。在本文中，我们提出了一种解决双重花费问题的方法——利用分布式对等时间戳服务器来生成按时间先后顺序交易的计算证明。这个系统只要“诚实”节点比起任何组织的攻击者累计控制了大多数 CPU 就是安全的。

二、 交易

我们声明一个电子币作为数字签名的一个链条。硬币的每个主人将币传递给下一个的方法是对之前交易以及下一个币主公钥的哈希值数字签名并且把这些信息附在币的末端。收款人可以验证签名来验证链的所有权关系。

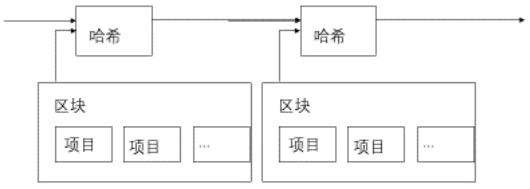


这个过程存在的问题是收款人不能核实其中一个拥有者没有对这枚货币进行双重花费。一个普遍的解决方式是引入受信的中央机构，或者造币厂，用于每一笔交易的双重花费问题进行检查。在每一笔交易之后，货币必须回到造币厂来发行新货币，而且只有直接发行于造币厂的货币才是可以被确信没有双重花费的。这个解决方法的问题是整个金融系统的命运依赖于造币厂的运行，每一笔交易都要经过它们，就像是银行一样。

我们需要一种方式让收款方知道之前拥有者没有签署任何更早的交易。我们的目的，最早的交易是被算作交易的那一个，所以我们不关心后续企图的双重花费。唯一确认交易缺席的方法是知晓所有的交易。在基于模型的造币厂里，造币厂知晓所有的交易并且决定哪一笔最先到达。为了在没有受信机构情况下完成这件事，交易必须被公开广播，同时我们需要一个我们需要一个系统用于参与者就单笔交易抵达次序达成共识。收款者需要证明每笔交易在时间上，多数节点认可它是第一笔达到的。

三、 时间戳服务器

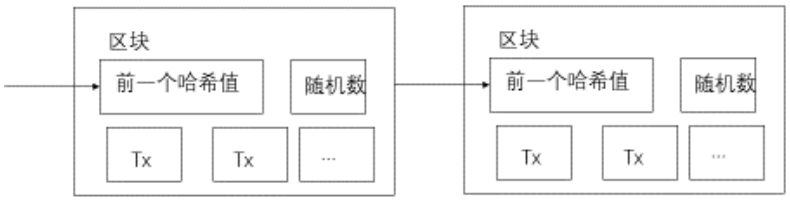
我们所提出的解决方案以时间戳服务器开始。一个时间戳服务器的工作原理是获取要打上时间戳项目块的哈希值以及在新闻或新闻组帖子上广播该哈希值。这个时间戳证明数字在某时产生，显然，才能进入哈希。每个时间戳在它的哈希值蕴含信息包括前一个时间戳，形成一个链，每一个增加的时间戳都会对之前时间戳进行加强。



四、 工作量证明

为了实现一个基于对等网络的分散式时间戳服务器，我们将需要使用类似于 Adam Back 的 Hashcash 的工作量证明系统，而不是新闻或者新闻组帖子。工作量证明涉及扫描一个值，该值在散列时（例如使用 SHA-256）时，散列存在前导零。所需的平均工作在所需的前导零上呈指数级增长，可以通过执行单个哈希来验证。

对于我们时间戳网络，我们实现工作量证明的方式是在区块中递增随机数直至满足区块哈希前导零的书被发现。一旦 CPU 花费精力满足了工作量证明，除非重做否则区块不可更改。当下一个区块介入链条，改变之前区块的工作就包括了重做要更改区块之后所有的区块。



工作量证明还解决了确定大多数决定代表的问题。如果大多数基于一 IP 地址一票的话，那么任意一个拥有大量 IP 地址的人都可以颠覆它。工作量证明本质上是一 CPU 一票。大多数决定由最长链代表，在这条链上倾注了最大的工作量证明投资。如果大多数 CPU 算力都被诚实节点所控制，那么诚实的区块链会增长最快并且超过其他竞争的链条。要修改过去的区块，攻击者不得不重做翻之前工作量证明的区块和所有在要更改区块之后的区块，并且要追上并超过城市结点的工作。我们稍后将展示，随着后续块的添加，较慢的攻击者赶上的可能性呈指数级下降。

补偿硬件速度的提高和对运行节点的利息随时间推移而变化，工作量证明难度由针对每小时平均区块数的移动平均值确定。如果它们增长得太快了，难度就会提升。

五、 网络

运行网络的步骤如下：

1）新交易被广播至所有节点

2）每个节点收集新交易加入区块

3）每个节点运行为它的区块寻找一个困难工作量证明

4）当一个节点发现了工作量证明，它会向所有节点广播这个区块

5）只有所有在节点中的交易是有效且没有已经花点时节点才会接受区块

6）节点表达它们对于区块接受的方法为：在链之后创造新的区块，使用接受了的区块哈希值作为之前区块的哈希值

节点总是认为最长链是正确的并且会继续运行延伸这条链。如果两个同时节点广播不同版本的下一个区块，有些节点或许会第一时间收到其中的一个。在这种情况下，他们会为首先接受到的区块工作，但是保存另一个分支以防它变得更

长。这个结会被打破当下一个工作量证明被发现且一条支链变得更长时；为另一条链工作的节点会转而在更长链上工作。

新的交易广播并不一定需要达到所有节点。只要它们达到多数节点，它们就会很快进入区块。区块广播同样有容错能力。如果一个节点没有接受到区块，它将在接受下一个区块的时候发现自己缺漏一个区块并发出请求。

六、 激励

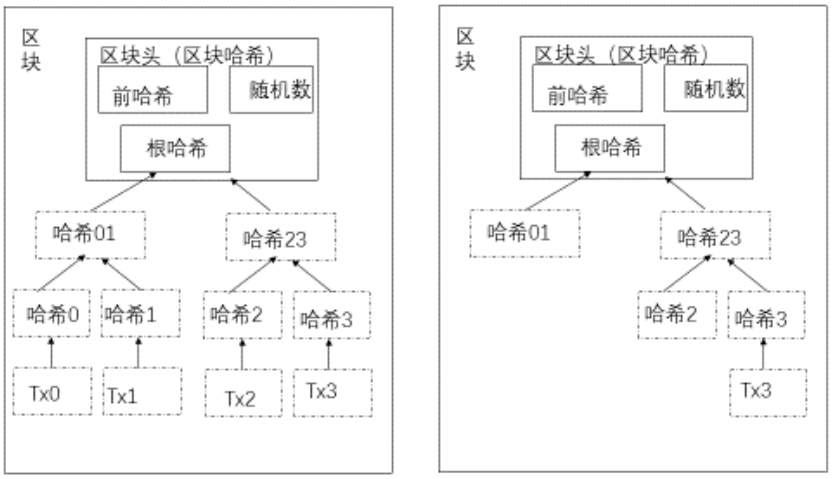
在合约中，区块的第一笔交易是特殊的交易——区块的创造者开始了一个新的币。这加入了一个对节点支持网络的激励，并且提供一种方式将最初分配的货币转移到流通中去，当这里没有中心权威机构来发行他们的时候。平稳补充常量新的货币类似于金矿工花费资源在流通领域补充黄金。在我们的情境下，是 CPU 消耗时间和电力。

激励同样可以通过交易费用获得资金。如果输出交易的价值要少于它输入的价值，差额是添加到包含交易的区块的激励值的交易费用。一旦预定数量的硬币进入流通，激励可以完全过渡到交易费用，并且完全没有通货膨胀。

激励或许能帮助鼓励节点保持诚实。如果贪婪的攻击者有能力聚集比所有诚实节点更多的 CPU 算力，它就不得不在其中做出选择——用它来通过偷回他的付款来欺骗人们，或是使用它来生成新的币。他应该发现，比破坏系统和他自己财富的有效性，遵守规则更有利可图，因为这些规则使得他的收益比其他人加起来多。

七、 回收磁盘空间

一旦某个货币的最新交易已经被足够多的区块覆盖，它之前的交易就可以被丢弃来节省磁盘空间。在不损伤区块哈希的情况下促进这个，交易被哈希化在 Merkle Tree 之中[7][2][5]，只有根节点被纳入了区块的哈希值。老的区块可通过剪除树枝的方式被压缩。树枝内部的哈希不需要被保存。

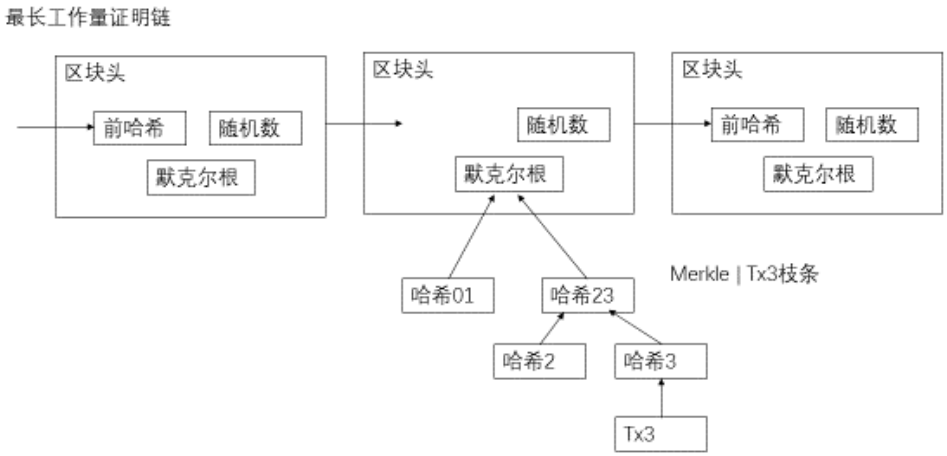


没有交易的区块头大概 80 字节。如果我们认为每十分钟生成一个区块，80 字节*6*24*365=4.2MB/年。早在 2008 年电脑系统经典售卖款都是 2G 随机处理

器，而且根据 Moore’s Law 预测每年还有 1.2G 的增速，即使区块头必须储存内存短缺也不会是个问题。

八、 简化的支付验证

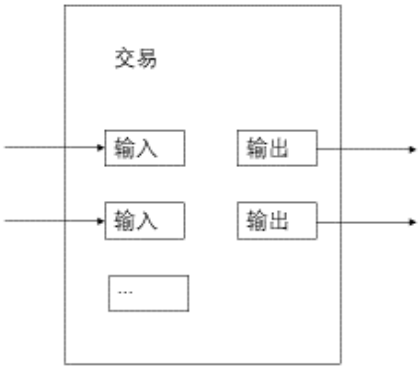
在没有运行全网络节点上验证支付是非常可能的事情。一个使用者仅仅需要保存最长工作证明链的区块头副本即可。他可以通过像其他网络节点查询以确认他拥有了最长的链，并获取链接交易到给交易盖时间戳区块的默克尔分支。虽然她不能自我核实交易，但通过链接它到链的某个位置，他可以查看网络节点已经接受了它，而且在它确认之后的区块加入也接受了它。



像这样，只要诚实节点控制着网络证明就是可信的，但如果网络被攻击者压倒性攻击就会更加脆弱。当网络节点可以为自身交易背书，简化模式可能被攻击者捏造的交易愚弄主要攻击者可以持续压倒。一个策略来保护避免这种情况时接受来自网络节点的警告当它们侦察到一个无效的区块时，提醒用户软件来下载整个区块和被警告交易来检查一致性。为了更加独立的安全性以及更快的支付确认，收款频繁的公司可能仍需运行他们自己的节点

九、 合并和分割交易额

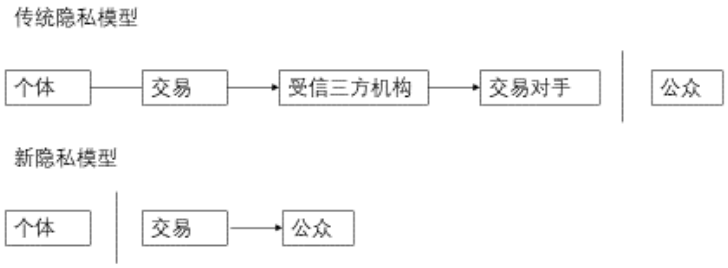
尽管单独处理每一个币是可能的，但将交易拆成每一分来处理显然是不明智的。为了允许价值分割与合并，交易包括多个输入值和输出值。通常这里会有或是一个从之前更大交易中得到的单独的输入，或是多个输入合并更小的账户，而以及至多两个输出：一个给支付，另一个找零，如果存在的话，回退到发送者那里。



应该被注意的是这里的扇出，一笔交易依赖多笔交易，而且那些交易依赖更多的交易，是不存在问题的。永远不会需要获取一笔交易历史的完整独立副本。

十、 隐私

传统的银行模型通过限制信息流向机构和三方受信企业渠道的方式来保证一定水平的私密性。公开所有交易的必要性就排除了这种方式，但是私密性仍然可以保证——通过截断另一处的信息流——保证公钥匿名。公众可以看见某人向另外一个人打款，但没有信息可以将交易关联到确定的人。这有些类似于通过股票交易的形式降低信息水平，时间和规模独立于交易存在，行情是公开的，但不知道所属机构。



作为增加的防火墙，一个新的钥匙对需要被应用在每笔交易中用以保证它们不被普通拥有者所关联。有些关联在多重输入交易中仍然是不可避免的，这重要揭示了它们的输入被同一个拥有者所拥有。风险在于如果一把钥匙的拥有者被暴露，关联可能会暴露其他属于这名拥有者的交易。

十一、 计算

我们认为攻击者尽自己最大努力生成可选择链的速度快过诚实节点的场景。即使这已经被完成了，也不会使得系统可以任意修改，比如凭空创建货币或者拿走不属于他的钱。节点不会接受无效的支付交易，而且诚实节点永远不会接受包含它们的区块。一个攻击者仅仅可以修改他交易中的一笔来退回他不久前花费的金钱。

诚实链条和攻击者链条之间的竞赛可以是被定性为二项随机漫步。成功事件

是诚实节点延伸一个区块，两条链差距+1，失败事件是攻击者的链延长一个区块，两条链差距-1。

攻击者追赶上给定的赤字差距的可能性就像是 Gambler's Ruin 问题。设定一个赌徒拥有不加限制的信用从一个赤字开始，然后进行无限次赌博试图达到盈亏平衡。我们可以计算他达到盈亏平衡的可能性，或者说是攻击者追上诚实链条的可能，如下[8]：

p = 诚实节点发现下一个区块

q = 攻击者发现下一个区块

q_z = 攻击者将追平 z 个区块的差距

$$q = \begin{cases} 1 & \text{if } p \leq q \\ (q/p)^z & \text{if } p > q \end{cases}$$

给出我们的假设：p>q，当攻击者需要追赶弥补的区块数增加，可能性呈指数递减。由于形势对他不利，如果他没有在早期幸运地快速赶上，他落下越多赶上的希望越渺茫。

我们现在考虑一个新交易的收款人在需要等待多久才可以确保付款人不能再改变这个交易。我们假设付款人是一个的攻击者——想要让收款方相信付款人已经支付给他货币一段时间，然后过一段时间再转换它收回货币。当这件事发生时接受者会得到警告，但是发送者希望警告会太迟了。

接受者生产一个新的钥匙对并将在签名前将公钥短暂地交给发送者。这可以防止发送者通过持续工作指导足够幸运获得大幅领先的方式预先准备好一个区块链，然后执行交易。一旦交易被发出，不诚实的发送者开始秘密的用一条平行链替换他现有交易版本的链条。

接收者一直等到交易被加入区块并且 z 个区块链接在该区块之后。他不知道攻击者所做的额外的工作，但是假设诚实节点每区块花费平均期望的时间，攻击者的潜在过程将符合 Poisson 分布，期望如下：

$$\lambda = z\frac{q}{p}$$

为了得到攻击者仍能够追上的可能性，我们对每个他可能达到进度的 Poisson 密度乘以他在该进度上能赶上诚实链的概率：

$$\sum_{k=0}^{\infty} \frac{\lambda^k e^{-\lambda}}{k!} \cdot \begin{cases} (q/p)^{(z-k)} & if\ k \leq z \\ 1 & if\ k > z \end{cases}$$

整理以避免对分布的无限尾部求和…

$$1 - \sum_{k=0}^{z} \frac{\lambda^k e^{-\lambda}}{k!} \left(1 - (q/p)^{(z-k)}\right)$$

转换为 C 语言：

```
#include <math.h>
double AttackerSuccessProbability(double q, int z)
{
    double p = 1.0 - q;
    double lambda = z * (q / p);
    double sum = 1.0;
    int i, k;
    for (k = 0; k <= z; k++)
    {
        double poisson = exp(-lambda);
        for (i = 1; i <= k; i++)
            poisson *= lambda / i;
        sum -= poisson * (1 - pow(q / p, z - k));
    }
    return sum;
}
```

运行一些结果，我们可以看到可能性随 Z 增大指数下降。

```
q=0.1
z=0    P=1.0000000
z=1    P=0.2045873
z=2    P=0.0509779
z=3    P=0.0131722
z=4    P=0.0034552
z=5    P=0.0009137
z=6    P=0.0002428
z=7    P=0.0000647
z=8    P=0.0000173
z=9    P=0.0000046
z=10   P=0.0000012

q=0.3
z=0    P=1.0000000
z=5    P=0.1773523
z=10   P=0.0416605
z=15   P=0.0101008
z=20   P=0.0024804
z=25   P=0.0006132
z=30   P=0.0001522
z=35   P=0.0000379
z=40   P=0.0000095
z=45   P=0.0000024
z=50   P=0.0000006
```

P 小余 0.01%的解：

```
P < 0.001
q=0.10   z=5
q=0.15   z=8
q=0.20   z=11
q=0.25   z=15
q=0.30   z=24
q=0.35   z=41
q=0.40   z=89
q=0.45   z=340
```

十二、 总结

我们已经提供了一种不依赖信任的电子交易系统。我们从通用的数字签名货币框架开始，这套系统提供了强有力的对于拥有者的控制，但是没有一种方法防止双重花费是不完善的。为了解决这个问题，我们提出了对等网络下使用工作量证明来记录交易的公共历史。在这套方法中，对于攻击者而言进行如果诚实节点掌握了大多数 CPU 算力那么算力上的攻击就很快变得不切实际。这套网络系统在它的非结构化的简易性上是稳健的。节点同时工作只要很少的协调。它们不需要被认证，因为信息不会被发送到某个特定的文职，只需要被尽力传播。节点可以自愿离开和再次加入网络，接受工作量证明链条作为它们离开时发生事情的证明。它们通过各自的 CPU 算力投票，表达他们对于有效区块的接受，以在链条上工作并延伸的方式；同时拒绝无效的区块，以拒绝在该链条上工作的方式。任何需要的规则和激励都可通过这个共识机制来加强。

参考文献

[1] W. Dai, "b-money," http://www.weidai.com/bmoney.txt, 1998.

[2] H. Massias, X.S. Avila, and J.-J. Quisquater, "Design of a secure timestamping service with minimal
trust requirements," In 20th Symposium on Information Theory in the Benelux, May 1999.

[3] S. Haber, W.S. Stornetta, "How to time-stamp a digital document," In Journal of Cryptology, vol 3, no
2, pages 99-111, 1991.

[4] D. Bayer, S. Haber, W.S. Stornetta, "Improving the efficiency and reliability of digital timestamping,"
In Sequences II: Methods in Communication, Security and Computer Science, pages 329-
334, 1993.

[5] S. Haber, W.S. Stornetta, "Secure names for bit-strings," In Proceedings of the 4th ACM Conference
on Computer and Communications Security, pages 28-35, April 1997.

[6] A. Back, "Hashcash - a denial of service counter-measure,"
http://www.hashcash.org/papers/hashcash.pdf, 2002.

[7] R.C. Merkle, "Protocols for public key cryptosystems," In Proc. 1980 Symposium on Security and
Privacy, IEEE Computer Society, pages 122-133, April 1980.

[8] W. Feller, "An introduction to probability theory and its applications," 1957.