比特币：一种P2P的电子现金系统

**摘要**：一个纯粹的P2P的电子现金系统可以使在线支付直接在发起方和接收方之间传递，无需经由某金融机构。数字签名可以提供这个系统的部分功能，但是为了防止重复付款，还是需要一个可信的第三方参与，与预期相距甚远。我们提出一个利用P2P网络解决重复付款问题的方案。网络通过将交易加入到一个不断延伸的链，来为交易加上时间戳，这个链是基于哈希的工作量证明（POW）的链，交易在其中是一条条的记录，除非重新进行工作量证明，否则不能修改记录。最长的链不仅作为发生的事件的证明，同时也是最大CPU功率池的证明。只要控制大部分CPU功率的节点间没有共同来攻击网络，那么他们会一起生成最长的链来阻止攻击。这个网络本身需要最小的结构。消息基于“尽力而为”的方式广播，节点可以任意的离开和重新加入网络，离开的节点需要接受最长的工作量链作为在他们离开期间发生的交易的有效证明。

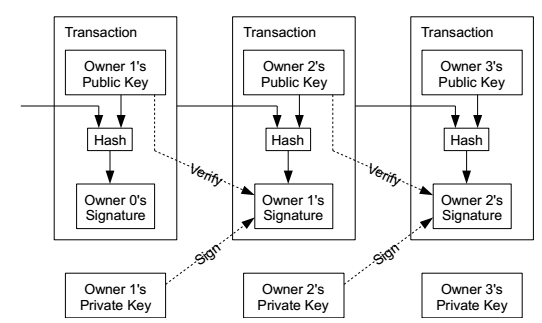
1. 介绍

因特网商业只能依赖金融机构作为可信的第三方来处理电子支付。虽然对于大多数交易来说，这样的系统足够有效，但是这样的信任模型与生俱来的弱点其实还是困扰着整个系统。因为完全的不可逆交易其实是不可能的，所以金融机构不能避免要去调解纠纷。调解的成本增加了交易成本，限制了最小实际交易大小，并且阻断了小额偶发交易的可能性，不能够为不可逆业务进行不可逆支付，又有进步一的增大了成本。因为存在可逆性，增加了对信任的需求。商人必须警惕他的客户，因此需要很麻烦的处理额外的信息。一定比例的欺诈行为会成功，这是不可避免的。这些成本和支付不确定性可以通过个人使用物理货币来避免，但是目前还没有绕开可信的第三方在通信通道进行支付的机制。

我们需要的是一个基于密码学证明的电子支付系统，而不是基于信任的系统，这样交易双方可以直接进行交易而不需要可信的第三方。在计算上不可逆的交易会保护售卖方免于欺诈，routine escrow机制很容易实现来保护购买方。本文，我们提出一个方案使用P2P分布式时间戳服务器来生成按照时间顺序的所有交易的计算证据，来解决重复付款问题。这个系统只要诚实的节点合加起来控制了比任何单一攻击者组织节点更多的CPU能力，就能保证安全。

1. 交易

我们定义一个电子币是一个数字签名的链。前一次交易和下一个拥有者的公钥进行散列，得到的散列值经过当前拥有者的签名，再将签名加入到电子币的末端，表明当前拥有者将这个电子币转移给下一个拥有者。接受支付的人可以检查签名，来验证所有权链。

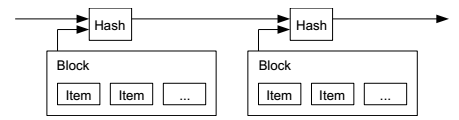


这个过程的问题是接受支付方并不能验证货币的一个拥有者没有进行双重支付。一般的方案是引入一个可信的中央权威机构或者铸币厂来检查每笔交易是否有双重支付。每次交易以后，货币必须返回铸币厂然后生产一个新的货币，只有从铸币厂直接发出的货币才是可信的，不会被双重支付。这个方案的问题是整个货币系统都依赖于运作铸币厂的公司，因为每一笔交易都需要讲过他们，就像银行一样。

我们需要一个方法让收款方知道前一个拥有者没有在更早的时候签署别的交易。为了这个目的，最早的交易是有效的交易，后续的多重支付尝试我们不去管它。确认不存在某个交易的唯一办法是能够知道所有的交易。在基于铸币厂的模型中，铸币厂知道所有的交易并且决定哪个是先到的。为了在没有可信第三方的情况下实现无效交易的确认，所有的交易必须公开声明，然后我们需要参与者都针对他们接收到的交易顺序达成共识。收款方需要在每笔交易发生时，多数的节点同意这笔交易是首先收到的。

1. 时间戳服务器

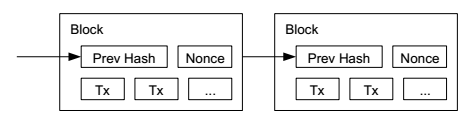
我们提出的解决方案始于一个时间戳服务器。若干事项组成一个块，将这个块进行散列，得到的结果经由时间戳服务器加盖时间戳，然后广泛的发布出去，就像在报纸上登发和Usenet发帖一样[2-5]。这个时间戳能够证明此时这个数据一定是存在的，因为很明显只有这样才能进行散列运算。每个时间戳都在它的散列中包含上一次的时间戳，这样形成了一个链，每一环都会在它的上一个时间戳的基础上增加新的时间戳，以加强之前形成的所有的时间戳。



1. 工作量证明

为了在P2P网络中实现分布式的时间戳服务器，我们需要使用类似于Adam Back的Hashcash[6]的工作量证明（pow），而不是真的登报或者在Usenet上发帖。工作量证明需要搜索用于散列的一个数值，比如用SHA-256进行散列，使用这样的数值散列以后得到的结果以若干个0比特开头。所需的平均工作量通过散列结果开头的0比特的个数调整，可以通过一次散列运算来验证。

针对我们的时间戳网络，我们通过在块中进行nonce递增，找到符合要求的nonce值，能够使块散列结果以给定数量的0开头。一旦CPU的工作能够满足工作量证明，除非重做一遍，不然这个块就不能修改了。因为后续还有其他的块会链接到这个块，修改这个块需要重新生成它之后的所有的块。



工作量证明也解决了多数人所做的决定如何表示的问题。如果多数人是基于一个IP一票的方式确定，那么只要有人能够拥有大量的IP就能颠覆系统。工作量证明本质上是一个CPU一票。多数人的决定是通过最长链来体现的，最长链意味着最多的工作量在里边。如果多数CPU能力由诚实的节点掌控，那么诚实的链会获得最快的增长，超过任何竞争的链。如果要修改之前生成过的块，攻击者必须重做这个块以及它后续所有块的工作量证明，才能赶上并超过诚实节点的工作。我们将会展示速度不够快的攻击者会造成链分叉的可能性。

为了平衡日益增长的硬件速度，为了调整运行节点获得收益，工作量证明的难度需要根据每个小时产生的块的目标平均数量来调整。如果生成的太快了，那么需要增加难度。

1. 网络

运行这个网络的步骤如下：

1. 新的交易广播到所有的节点
2. 每个节点搜集新的交易到一个块里
3. 每个节点根据他们搜集到的块进行工作量证明
4. 当一个节点找到了工作量证明，它将这个块向所有的节点广播
5. 其他节点决定接受这个块，仅当这个块内部的所有交易经过验证，并没有多重支付过
6. 其他节点通过将这个块作为前一个块，在后面继续生成新的块，来体现对这个块的承认

各节点总是认为最长的链是正确的链，并且在最长的链后面继续扩展它。如果2个节点同时广播了新的块，某些节点会首先收到其中一个块，某些节点首先收到另一个。这样的情况下，它们对最先收到的块进行工作，但是也会保存另一个分支以备分支会成为最长的链。一旦下一个工作量证明被找到了，那么更长的链就确定了，节点会切换到更长的链上继续工作。

新交易的广播不需要传达到所有的节点。只要这个交易广播到一些节点，那么不久之后它就会添加到块中。块广播也是可以容忍消息丢失的。如果一个节点没有收到一个块，那么当这个节点收到下一个块的时候，它就会意识到丢失了一个块。

1. 激励

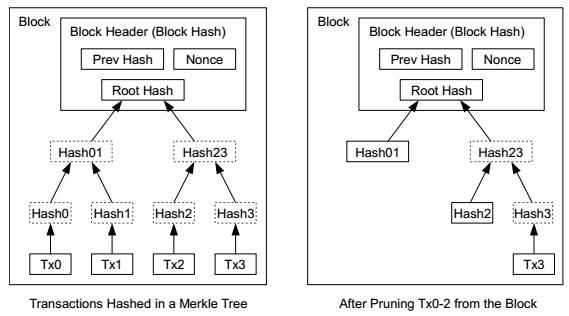
根据惯例，每个块的第一笔交易是一个特殊的交易，会将新的币给予这个块的创建者。这对支撑网络的节点是个激励，并且为货币流通提供了货币来源，因为并没有中央权威机构来发行货币。新货币的稳定增加，可以类比挖金矿，为金属货币的流通提供黄金。这里需要付出的是CPU时间和电力。

激励也可以通过交易费来获取。如果交易的输出值小于输入值，那么之间的差值就是对包含这个交易的块支付的交易费。一旦预先定义的所有货币都进入了流通领域，那么激励就完全转换为交易手续费，完全不会有通货膨胀。

这样的激励可以鼓励节点保持诚实。如果一个贪婪的攻击者能够组装超过所有诚实节点的CPU算力，那么这个攻击者需要在两者之间权衡，一是通过收回它所付货币实施诈骗，二是用它的算力生成新的货币。它会发现遵守规则是更有利可图的，生成新币比破坏系统能够带来更大的收益。

1. 磁盘空间回收

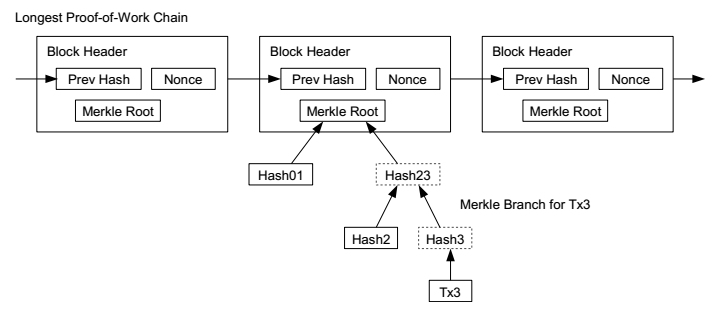
一旦最新的交易通过足够的块来确认了， 那么这个交易之前的支付交易就可以丢掉来节省磁盘空间。为了实现这个磁盘空间的回收同时不破坏块的散列，交易按照Merkle树[7]来进行散列。老的块可以通过砍掉树的分支来压缩。内部的那些散列结果无需继续保存。



一个不包含交易信息的块头约含80个字节。如果我们假定每10分钟生成一个块，那么一年回产生80字节\*6\*24\*365=4.2MB的数据。根据2008年售卖的电脑的主流配置，2GB内存，根据摩尔定律的预测每年增加1.2GB的内存，即使块头都保存在内存中也不是个问题。

1. 简化支付验证

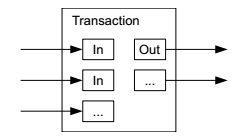
没有完整的网络运行节点也是可以验证支付的。用户只需要保留最长工作量证明链的一份拷贝，可以通过向网络节点查询来获取，确信是最长的链后在其中找到链接了目标交易信息的Merkle分支，然后他可以看到网络节点已经接受了这个交易，再之后增加的块更进一步的确认了网络已经接受了这个交易。

1. 

如上所述，只要诚实节点还控制了网络，那么这样的认证就是可靠的，但是如果网络被攻击者攻陷了，那么这样的认证就是不可靠的。当网络节点能够验证他们自身的交易时，这个简化的方式可以被攻击者编造的交易来愚弄，因为攻击者能够继续控制网络。防范此类攻击的一个策略是从网络节点那里接收警告，当它们检测到无效的块，提示用户软件下载完整的块，下载被警告的交易来确认不一致性。频分收到支付交易的商家可能仍然想要运行它们自己的节点获取更独立的安全性和更便捷的确认。

1. 现金的合并和分离

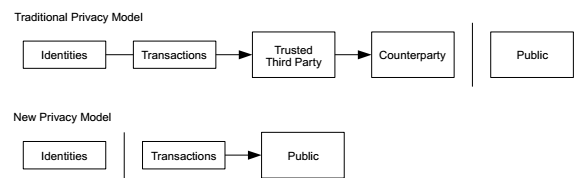
尽管可以对币进行拆分处理，但是将一次转账中的每一分都拆成不同的交易也是比较笨拙的方式。为了允许转账数值的拆分和合并，交易可以包含多个输入和输出。通常要么是一个从上一个更大的交易带来的单一的输入，或者更小一些数额的多个输入，然后至多有2个输出：一个用于支付，一个返还找零给支付方，如果有的话。



需要注意，这里扇出问题，也就是一个交易依赖于若干个交易，这些交易又依赖于更多的交易的问题，在这里不是个问题。这里完全不需要将一个完整的交易历史拷贝进行拆包提取的需要。

1. 隐私

传统的银行模型通过限制参与方获取信息，选择相信可信第三方的方式保证一定程度的隐私性。本文需要将所有交易信息公开的方法显然不适用于上述方式，但是隐私性可以通过另一类的信息截留来维护：保持公钥的匿名性。公众能够看到某人向另一个人转账了若干货币，但是并不能将交易和人关联起来。这类似于股票交易提供的信息等级，每次交易的时间和交易额是公开的，谁进行的交易是不知道的。



作为一个额外的防火墙，对每一笔交易使用一对新的密钥对可以防止将交易关联到所有者那里。对于多输入的交易，有些关联还是不可避免的，因为这样的交易需要确认多个输入时来自同一个拥有者的。风险是如果一个秘钥的所有者被发现了身份，那么这个所有者的其他交易也有可能被发现。

1. 计算

我们考虑攻击者尝试比诚实链更快的生成另一条链的情景。即使攻击者成功了，也不会将系统带入任人修改的地步，比如无中生有的创造币值或者拿到从来没有属于过攻击者的钱。其他节点不会同意一个无效的支付交易，诚实节点永远不会包含这样的信息的块。攻击者只能尝试改变他自己的交易，将他最近付出的钱再拿回来。

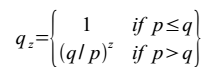
诚实链和攻击者链之间的竞赛可以用Binalmial Random Walk表征。成功事件是诚实节点扩展了一个块，领先+1，失败事件是攻击者节点扩展了一个块，减少差距-1.

攻击者从给定差距追赶上来的概率可以类比于赌徒破产问题。假定一个赌徒可以随便欠账，从某一个赤字开始进行无限数量的赌博争取消除赤字。我们可以计算他回本的概率，也就是攻击者能够追上诚实链的概率，如下：

P = 诚实节点找到下一个块的概率

q = 攻击者找到下一个块的概率

qz = 攻击者从落后Z块开始赶上来的概率



按照我们的假设，p>q，攻击者越落后，追上来的概率越低。如果不是他运气逆天，攻击者成功的概率会随着落后越来越多而变得越来越渺茫。

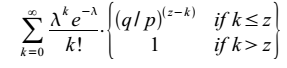
我们考虑一下一个新的交易的接受者需要等多久才能确保发送者不能够改变这个交易了。假定发送者是一个攻击者，他会想要让接收方在一定时间内以为他已经发出了交易，然后再把支付的钱转回来。接收方在这种情况下会收到警告，但是发出方会希望这个警告发出的非常晚。

接收方可以在发送方即将签署前生成一个新的密钥对，将公钥交给发送方。防止发送方提前准备好块链，持续增加这个块链然后成功的领先足够远，在那个时候执行交易。一旦交易发出，不诚实的发送者开始秘密的在一个并行的链上开始工作，其上有他的另一个版本交易。

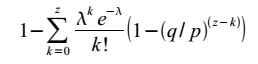
接受者等待交易被加入到一个块中，然后Z个块又被加入到这个块之后。他并不知道攻击者所进行的小动作，但是假定诚实块按照每个块的平均时间进行了工作，攻击者的潜在的进展是一个期望值的泊松分布：



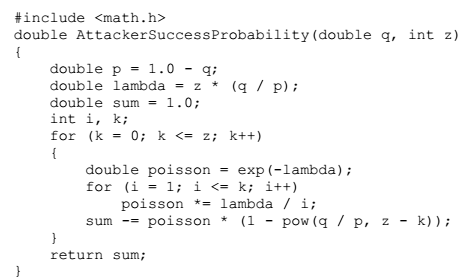
为了得到攻击者现在仍然能够跟上的概率，我们将他能够取得的进展都乘以泊松密度，乘以他能够从这一点跟上的概率：



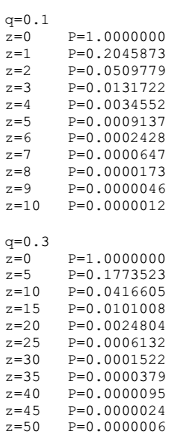
调整算式避免对分布的无限尾部进行求和：



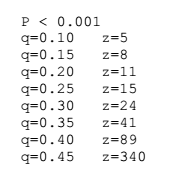
转换成C代码：



运行得到一些结果，我们能够看到随着Z的增加，概率不断降低。



P小于0.1%的情况：



1. 结论

我们提出了一个不依赖于信任的电子交易系统。我们从数字签名形成的货币框架开始，数字签名提供了很强的所有权控制，但是并不完整，不能抵御重复付款。为了解决这个问题，我们提出了一个使用工作量证明记录交易历史的P2P网络，很快就可以成为一个在计算上攻击者不能够攻破的系统，只要诚实节点控制了更多的CPU算力。这个网络因为它无结构的简单性，因此是健壮的。所有的节点通过少量的协调就可以一起工作。节点不需要都识别出来，因为消息并不需要传递到所有节点，只需要尽力而为的传递即可。节点可以任意离开和重新加入网络，接受离开期间的工作量证明可以对此期间发生的业务进行证明。他们通过CPU算力投票，通过拒绝无效的块，而在有效的块上继续工作，来体现对有效块的接受。任何所需的规则和激励都可以添加到这个共识机制中。

参考

