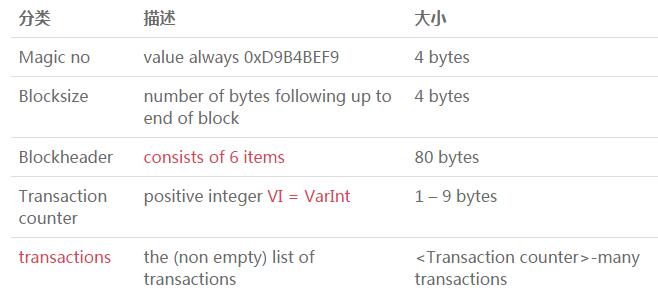
http://8btc.com/article-2238-1.html

关于区块链的常见问题

*数据通过称之为*区块的文件永久记录在比特币网络上。区块是之前没有被记录的最近所有或者部分比特币交易的记录。它们就像是一个城市记录薄的个人页面（记录了不动产所有权的变化）或者是一个股票交易账本。新的区块被添加到记录（称之为区块链）的末端，而且一旦书写（尽管如果它们是孤立的，一些软件会移除它们）无法修改或移除。每一个区块都记录了它创建之前所发生的一切。

**区块结构**

[](http://8btc.com/data/attachment/portal/201604/13/154502c5zy00qw4zh76954.jpg)

**描述**

除了其它以外，每一个区块还包含部分或所有最近的交易记录，以及在它之前所出现区块的参考。它还包括了难以解答的数学谜题答案，这个答案对每一个区块都是唯一的。没有正确的答案 – 挖矿的过程本质上就是发现解决目前区块答案的过程，新的区块就无法提交至网络。每一个区块中的数学问题都很难解答，但是一旦发现有效的解决方法，网络上的其它人就很容易确认答案是正确的。对于任意一个区块，都有很多有效的解决方案 – 只需要找到一个方案来解决这个区块。

因为解决每一个区块都会有新的比特币奖励，每一个区块也记录了哪一个[比特币地址](http://8btc.com/article-135-1.html" \t "_blank)或脚本有权获得该奖励。该记录称之为生成交易，或造币交易，而且在每一个区块中通常都会显示为第一个交易。每一个区块产生的比特币数目是从50开始，且每经过210000个区块（约4年）减半。

比特币交易由发送者广播至网络，其它所有节点则收集交易并将它们添加至他们正在解决的区块中。因为受到附加交易费的激励，矿工会把交易包含到他们的区块中。

数学问题的难度也是由网络自动调节的，目的是为了实现平均每个小时能解决6个区块的目标。每隔2016个区块（需要两周来解决），所有比特币客户端对比真实创建的数目与该目标值，并以百分比的变化来调整该目标值。网络达成一致并自动增加（或减少）生成区块的难度。

因为每一个区块包含了之前一个区块的参考，因此可以说现存的所有区块形成了一条链。然而，区块链有可能临时分叉 – 例如，如果两个互相不认识的矿工在同一时间同一区块发行了两种不同的有效解决方案。点对点的网络在很短的时间内能够解决这些分叉，因此只有一条链能够生存下来。

客户端接受最长的链作为有效链。整个区块链的长度指的是累计难度最大的链，并不是含有区块最多的链。这可以阻止一些人形成大量的低难度区块，并让网络接受作为“最长”的区块链。

区块链常见问题

**有多少区块？**

http://blockmeta.com/data/chain

**区块最大数目是多少？**

没有最大数目，区块只是按照平均每10分钟一个的速率添加至区块链的末端。

**当21000000个全部比特币产生后，发生什么？**

区块还会继续产生，因为只要有人交易比特币，就会创建区块。

**多久能生成一个区块？**

没法准确预计。有一个计算器可以告诉你大概的时间。

**如果我计算了1%的区块，那会怎么？**

没有所谓的计算了1%的区块。你并没有解决它。连续工作24小时后解决它的概率和你在刚开始或者任何时候的概率都是一样的。相信我这就是所谓的赌徒谬误。

这就好比同时抛53个硬币，同时出现头像的那一面。你每次尝试的成功概率是一样的。

**哪里能获得更多的技术细节？**

--

什么是区块链共识？

共识（Consensus）过程是一个非常有趣的过程。

在我们的日常生活中，几乎所有的事情都是达成共识的过程。

达成共识越分散的过程，其效率就越低，但满意度越高，因此也越稳定；相反，达成共识越集中的过程，效率越高，也越容易出现独裁和腐败现象。

* 达成共识常用的一种方法就是通过物质上的激励以对某个事件达成共识；但是这种共识存在的问题就是容易被外界其它更大的物质激励所破坏。
* 还有一种就是群体中的个体按照符合自身利益或整个群体利益的方向来对某个事件自发地达成共识；当然形成这种自发式的以维护群体利益为核心的共识过程还是需要时间和环境因素的，但是一旦达成这样的共识趋势，其共识结果也越稳定，越不容易被破坏。

在比特币和其它区块链币中，也存在如何达成共识的问题。或者说，**比特币或其它区块链币最核心的问题也是如何在去中心化的环境中达成共识。**

区块链是比特币背后的核心技术，也是支撑比特币的基础架构。因此在谈区块链共识，就必然要谈比特币的共识。

**比特币最核心的突破是在去中心化的情况下对交易事件达成了共识，即在没有中心组织的情况下对某个交易的有效性达成了一致。**

比特币实现这个共识的方法主要包括两个部分：

1. 激励；即通过每个区块产生一定量的新比特币来激励参与者；
2. 引入外部资源确保安全；即通过大量的外部计算来确保共识的安全性，也就是工作量证明（Proof of Power）；

这也是几乎所有PoW币种所采用的的方法。

而这套方法要能持续长期运行下去的前提就是：

1. 这种激励对参与者要有足够的吸引力；也就是说比特币要一直涨价，才能吸引参与者持续参与挖矿计算，以维护整个网络的运行；否则就会导致参与的人减少，破坏网络安全；
2. 没有外部攻击；由于比特币引入了外部计算来确保安全，因此只要有足够的挖矿算力（超过维护系统算力的51%）就能对系统成功进行攻击，这也是比特币长期存在的安全隐患之一；因为只要有钱，就能买到设备和算力。

正是由于比特币存在的问题，例如消耗大量的资源、外部51%攻击等，出现了PoS（Proof of Stake）共识机理。

总体上，PoS共识理论和实践目前仍处在探索阶段。

最原始的PoS机理就是用股权代替PoW中的挖矿算力，来模拟比特币的挖矿过程。请注意，这个过程没有引入外部资源，而是仅仅依靠自身的币种股份来维护网络安全，因此其不需要消耗大量能源来进行计算；而且由于其没有引入外部的资源，因此不会担心外部攻击，例如外界的算力攻击。

看起来PoS是很完美的，但是它存在一个严重漏洞。

PoS存在内部的Nothing-at-Stake攻击。

什么是Nothing-at-Stake（常写作N@S）攻击？

假设系统中出现了两个分支链，那么对于持有币的”挖矿者“来讲，最佳的操作策略就是同时在两个分支上进行“挖矿”，这样，无论哪个分支胜出，对币种持有者来讲，都会获得本属于他的利益，即不会有利益损失。而且由于不需要算力消耗，因此PoS中在两个分支上挖矿是可行的。

这导致的问题是，只要系统存在分叉，“矿工们”都会同时在这几个分支上挖矿；因此在某个情况下，发起攻击的分叉链是极有可能成功的，因为所有人也都在这个分叉链上达成了共识；而且甚至不用持有51%的币量，就可以成功发起分叉攻击；

而这在PoW中是不可行的，因为挖矿需要消耗算力，矿工只能在一个分支上进行挖矿。

第二个问题是重写历史攻击；即攻击者可以通过购买原始持有币种的账户来从头发起攻击，重新分叉一个区块链。因为原始的币种持有者可以将币转移至其它账户，因此他是可以在没有损失的情况下将原始账户出售给攻击者的。攻击者需要的就是有足够数量币的原始账户；当然了，这也只是概率问题，因为有可能原始账户持有者不会出售他们的账户，但是理论上确实存在这种攻击。

第三个问题是，尽管PoS中的挖矿不用消耗算力，运行成本很低，但是也存在如何激励矿工的问题。因为一般的PoS系统是没有新币产生的，矿工只能赚取交易费，而且在交易费不高的情况下，对矿工的激励也是很有限的。

当然了，也有很多PoS币种解决这个问题的办法就是持续的再产生新币来激励挖矿者，这导致的问题就是通胀。

上述3个问题是PoS要解决的，尤其是N@S的问题尤为重要，因为如果没有其它约束机制，这种攻击是完全有可能实现的

从以上可以看出，无论是PoW还是PoS机理的共识过程，其必要条件有两个：

1. 信息公开共享；
2. 个体参与；

以现实为例，事件的信息越透明、所涉及到的人员参与度越高，最终形成的共识也就越稳定、越持久。这与区块链共识是一致的。

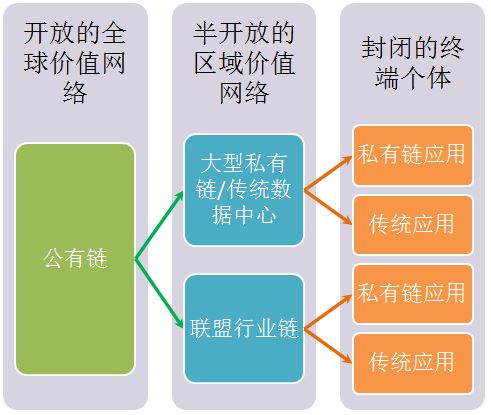
**浅谈区块链共识机制与分布式一致性算法**  
 **传统分布式一致性算法和区块链共识过程的异同点**  
相同点：

* Append only
* 强调序列化
* 少数服从多数原则
* 分离覆盖的问题：即长链覆盖短链区块，多节点覆盖少数节点日志  
  不同点：
* 传统分布式一致性算法大多不考虑拜占庭容错(Byzanetine Paxos除外)，即假设所有节点只发生宕机、网络故障等非人为问题，并不考虑恶意节点篡改数据的问题；
* 传统分布式一致性算法是面向日志（数据库）的，即更通用的情况，而区块链共识模型面向交易的，所以严格来说，传统分布式一致性算法应该处于区块链共识模型的下面一层。

**区块链共识模型与传统一致性算法的关系**  
考虑上面的不同点，结合私有链和行业链的性质，我们有：

* 私有链：封闭生态的存储网络，所有节点都是可信任的，如某大型集团内部多数公司。
* 行业链：半封闭生态的交易网络，存在对等的不信任节点，如房地产行业A、B、C、D公司。
* 公有链：开放生态的交易网络，这层主要是为行业链和私有链提供全球交易网络。

由于私有链是封闭生态的存储网络，也就是说使用传统分布式一致性模型应该是最优的；  
  
由于联盟行业链其半封闭半开放特性，使用Delegated Proof of XXX 是最优的，可以考虑以传统一致性算法作为基础加入拜占庭容错/安全防护机制进行改进。  
  
公有链PoW应该仍然是最优的选择。  
  
如下图所示：

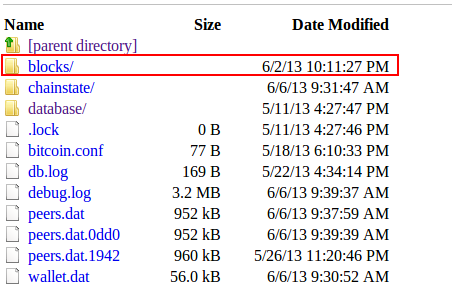
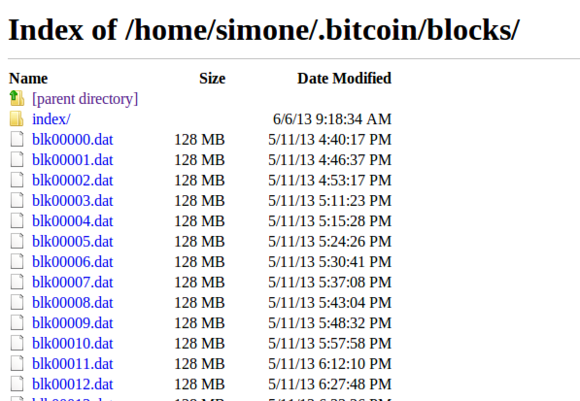
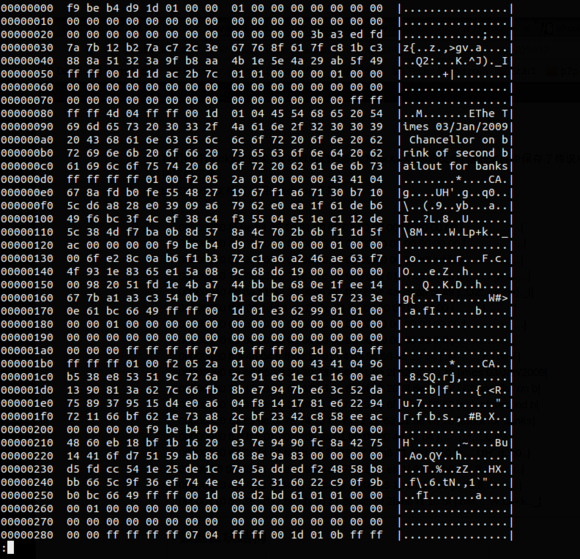
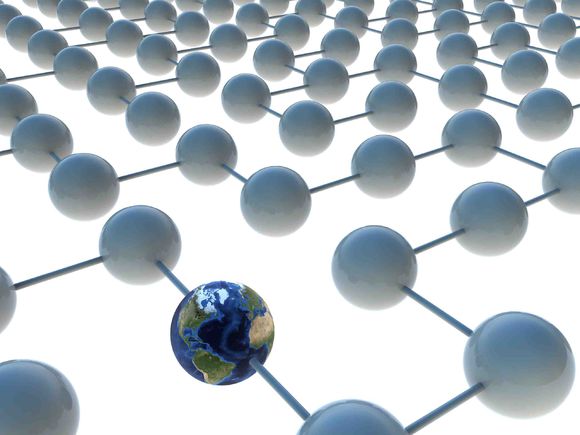


**传统分布式一致性算法介绍**  
  
本文主要讨论主流的Paxos算法家族和Raft算法，这里抛砖引玉，网络上有关两者的资料非常丰富，大家可自行搜索查阅。  
  
**1. Paxos 算法家族**  
  
1998年Lamport提出Paxos算法，后续又增添多个改进版本的Paxos形成Paxos协议家族，且Paxos都有共同点是不容易工程实现。

* **Classic Paxos** ：LeaderLess，又名Basic Paxos，以下均为Paxos的变种，基于CAP定律，侧重了不同方向。
* **Cheap Paxos**
* **Egalitarian Paxos** ： conflicts rare
* **Fast Paxos** ： Leader only when needed ，conflicts common
* **Multi-Paxos** ：Leader driven
* **Byzanetine Paxos**

"Byzantine Paxos adds an extra message (Verify) which acts to distribute knowledge and verify the actions of the other processors".Lamport 在2011年的论文《Leaderless Byzanetine Paxos》中表示不清楚实践中是否有效，考虑Paxos本身实现的难度，此方案工程角度不是最优，但是系统角度应该是最好的。  
  
本小节Ref：  
  
Wikipedia\_Paxos  
  
Leaderless Byzanetine Paxos  
  
  
  
**2 Raft 算法**  
  
这是一个非常友好的算法，容易理解、实现，不过它是Strong Leadership的，也就是说，  
  
任意包含Leader的时刻，Leader拥有完全记账权，如果此Leader节点是恶意的，后果不堪设想。  
  
且leadership的一致性算法都有个通病，吞吐量受单个节点的限制，这点在Raft身上体现尤甚。  
  
Raft算法简介：[https://raft.github.io/](https://raft.github.io/" \t "_blank)  
  
Raft算法动态演示：[http://thesecretlivesofdata.com/raft/](http://thesecretlivesofdata.com/raft/" \t "_blank)  
  
  
**3. 其他**  
  
**VRR(Viewstamped Replication Revisited)**  
这也是一个基于leadership的一致性算法，相比上述其他算法，它的优点是延迟最小。  
  
  
  
**常见区块链共识模型介绍**  
  
转载自这里：[https://bitsharestalk.org/index.php?topic=4031.0](https://bitsharestalk.org/index.php?topic=4031.0" \t "_blank)  
这是DPoS的白皮书，主要介绍了DPos，但也囊括了其他共识模型的介绍。  
授权股权证明机制白皮书  
(Delegated Proof-of-Stake ，DPOS)  
作者： Daniel Larimer  
April 3, 2014  
翻译：yidaidaxia\_郝晓曦  
比特坊数字资产研究俱乐部 翻译作品（[www.bitfarm.io](http://www.bitfarm.io/" \t "_blank)）  
  
摘要  
本白皮书介绍一种股权证明机制的新实现方式，该方式可以对交易进行秒级验证，并且能够在更短的时间内提供比现有任何股权证明系统都更好的安全性。在比特币网络产生一个区块的时间过后，一个授权股权证明系统(DPOS)能使你的交易得到20%股东的核实，而在比特币网络声明交易已几乎不可逆(6个区块，约1小时)的时间过后，在DPOS机制下，通过其代表，你的交易已经得到100%股东的核实。  
  
1.0 背景  
分布式交易总账需要在尽可能短的时间内做到安全、明确及不可逆，便于提供一个最坚实且去中心化的系统。在实践中，该流程分为两个方面：选择一个独特的节点来产生一个区块，并使得交易总账不可逆。  
  
1.1 工作量证明机制(Proof of Work, POW)  
第一个成功解决该问题的尝试是比特币系统(Bitcoin)，比特币系统使用工作量证明机制使更长总账的产生具有计算性难度。工作量证明机制就好比是乐透，平均每10分钟有一个节点找到一个区块。如果两个节点在同一个时间找到区块，那么网络将根据后续节点的决定来确定以哪个区块构建总账。从统计学角度讲，一笔交易在6个区块(约1个小时)后被认为是明确确认且不可逆的。然而，核心开发者认为，需要120个区块(约一天)，才能充分保护网络不受来自潜在更长的已将新产生的币花掉的攻击区块链的威胁。  
尽管出现更长的区块链会变得不太可能，但任何拥有巨大经济资源的人都仍有可能制造一个更长的区块链或者具备足够的哈希算力来冻结用户的账户。  
  
1.2 股权证明机制(Proof of Stake, POS)  
股权证明机制已有很多不同变种，但基本概念是产生区块的难度应该与你在网络里所占的股权(所有权占比)成比例。到目前为止，已有两个系统开始运行：点点币(Peercoin)和未来币(NXT)。点点币使用一种混合模式，用你的股权调整你的挖矿难度。未来币使用一个确定性算法以随机选择一个股东来产生下一个区块。未来币算法基于你的账户余额来调整你被选中的可能性。  
未来币和点点币都分别解决了谁来生产下一个区块的问题，但他们没有找到在适当的时间内使区块链具备不可逆的安全性的方法。根据我们能找到的信息，做到这点，点点币需要至少6个区块(约一小时)，未来币需要10个区块。我们找不到在10个区块后未来币能提供什么级别安全性的根据。  
我们之前发布了基于交易的股权证明机制(Transactions as Proof of Stake, TaPOS)的白皮书，在该机制中，每笔交易都包含区块链中前一个区块的哈希值。通过该系统，对任何人而言，网络变得越来越安全而不可逆，因为最终每个区块都经过了股东投票。TaPOS面临的挑战是它没有定义谁来产生下一个区块。  
  
1.3 瑞波共识机制(Ripple Consensus)  
瑞波共识算法，使一组节点能够基于特殊节点列表达成共识。初始特殊节点列表就像一个俱乐部，要接纳一个新成员，必须由51%的该俱乐部会员投票通过。共识遵循这核心成员的51%权力，外部人员则没有影响力。由于该俱乐部由“中心化”开始，它将一直是“中心化的”，而如果它开始腐化，股东们什么也做不了。与比特币及点点币一样，瑞波系统将股东们与其投票权隔开，并因此比其他系统更中心化。  
  
2.0 授权股权证明机制(DPOS)  
当使用去中心化自治公司(Decentralized Autonomous Company, DAC)这一说法时，去中心化表示每个股东按其持股比例拥有影响力，51%股东投票的结果将是不可逆且有约束力的。其挑战是通过及时而高效的方法达到51%批准。  
为达到这个目标，每个股东可以将其投票权授予一名代表。获票数最多的前100位代表按既定时间表轮流产生区块。每名代表分配到一个时间段来生产区块。所有的代表将收到等同于一个平均水平的区块所含交易费的10%作为报酬。如果一个平均水平的区块含有100股作为交易费，一名代表将获得1股作为报酬。  
网络延迟有可能使某些代表没能及时广播他们的区块，而这将导致区块链分叉。然而，这不太可能发生，因为制造区块的代表可以与制造前后区块的代表建立直接连接。建立这种与你之后的代表(也许也包括其后的那名代表)的直接连接是为了确保你能得到报酬。  
该模式可以每30秒产生一个新区块，并且在正常的网络条件下区块链分叉的可能性极其小，即使发生也可以在几分钟内得到解决。  
  
2.1 成为一名代表  
成为一名代表，你必须在网络上注册你的公钥，然后分配到一个32位的特有标识符。然后该标识符会被每笔交易数据的“头部”引用。  
  
2.2 授权你的选票  
每个钱包有一个参数设置窗口，在该窗口里用户可以选择一个或更多的代表，并将其分级。一经设定，用户所做的每笔交易将把选票从“输入代表”转移至“输出代表”。一般情况下，用户不会创建特别以投票为目的的交易，因为那将耗费他们一笔交易费。但在紧急情况下，某些用户可能觉得通过支付费用这一更积极的方式来改变他们的投票是值得的。   
  
2.3 保持代表诚实  
每个钱包将显示一个状态指示器，让用户知道他们的代表表现如何。如果他们错过了太多的区块，那么系统将会推荐用户去换一个新的代表。如果任何代表被发现签发了一个无效的区块，那么所有标准钱包将在每个钱包进行更多交易前要求选出一个新代表。  
  
2.4 解决区块链分叉  
和工作量证明系统及其他股权证明系统一样，最佳区块链是最长的有效区块链。任何时候，一名代表错过签发一个区块的机会，该区块链将比潜在竞争对手短。只要在你的交易被写入区块后的100个区块中的51%被生产出来了，那么你就可以安全地认为你在主区块链上。  
也许，在防止区块链分叉所导致的损失方面，最重要的事是在事发后第一时间得知消息。因为代表们通过生产区块得到很好的报酬，他们将保持接近100%的在线时间来防止因被投票罢免而损失收入。你可以安全地认为如果在过去的10个区块中，有一两个区块错过生产，则互联网的某些部分可能正发生连接问题，那么用户应该对此特别警觉并要求额外的确认数。如果10区块中有超过5个错过生产，那么这意味着你很可能在一条支链上，因此应该停止所有交易，直到分叉得到解决。  
以一种及时的方式(少于5分钟)简单地发现并警示用户网络分叉，是可以最小化潜在损失的非常重要的能力。而知道你是否正处在一条支链上则更为重要。  
  
2.5 100名代表是去中心化的吗？  
因为去中心化已经成为一个流行术语，所以其定义很难完全固定。我们将自由市场看作去中心化的基本形式，并将对进入自由市场设置障碍看作是所有中心化的基础。像任何事物一样，中心化有程度之分，所以我们把授权股权证明机制与其它方案的中心化程度进行对比。  
  
2.5.1 比特币  
比特币系统目前正以授权工作量证明(Delegated Proof of Work, DPOW)为基础而运行，因此有大约10名代表控制了绝大多数的哈希算力。在那些为其竞争而能使用规模经济进行无收益挖矿的人手中，哈希算力本身就是中心化的。最后，工作量证明机制为进入市场设置障碍，使得“在职”的区块制造者无法轻易被取代。与比特币系统相比，DPOS在区块生产方面至少去中西化了10倍，并且也许在市场竞争方面去中心化了无数倍。  
尽管在哈希算力方面有一定量的去中心化，当想到掌控比特币系统的股东(比特币持有者)所持股份的占比，我们认为比特币系统是最中心化的。如果你考虑使用比特币体系的用户总数，其中参与挖矿的人很可能少于百分之一。  
  
2.5.2 点点币  
点点币是一个混合系统，所以它由于工作量证明机制而是部分中心化的。和比特币系统一样，它也有矿池。与比特币相比，点点币无疑是更去中心化的，然而，因为股权证明机制矿池需要用户保持他们的电脑在线且钱包解锁，只有一小部分的股东参与了任何形式的挖矿。  
  
2.5.3 未来币  
未来币使用透明锻造，以确定的选出下一个制造节点。可以将其类比为，使用授权股权证明机制但你只能将你的投票权授予你自己，而你获得锻造区块机会的频率直接取决于你的账户余额。在这个意义上来说，未来币比点点币和比特币更为去中心化。但由于对安全风险的顾虑以及事实上大多数常规用户不会整天开启他们的电脑来籍此获得锻造机会方面的优势，它仍然遭受着少的可怜的挖矿参与度。  
从这个角度来讲，我们可以断定未来币网络是由一小部分股东来保障网络安全的。事实上，如果你不上线投票，那么你将失去你的选票。为了解决这个问题，一些未来币用户用他们的股权建立股权池，并信任第三方来为他们挖矿。这是以一种形式的授权股权证明来提高股东参与度，但这也使他们的账户余额在他们参加这些矿池时承受风险。  
  
3.0 攻击  
一般而言，网络必须抵御两种类型的攻击：拒绝服务攻击和双重支付攻击。一个攻击者通过不把一些或全部的交易加入总账来进行拒绝服务攻击。这种攻击可以由任何拥有51%网络(无论比特币、未来币或其它)的人进行。而利用在网络正试图达成共识时的短期优势，可以进行双重支付攻击。  
为抵御这些攻击，网络必须使51%的股东尽快达成协议。  
  
3.1 防止排除交易  
拥有全部经股东投票选出的100名代表，并且按要求轮流生产区块，意味着任何一笔由至少1%的股东批准的交易能够在30分钟内加入总账。这意味着没有代表可以通过将投票支持其他代表的交易排除在外来获取利益。  
  
3.2 将一些代表的权力中心化  
与其所被授权的投票权无关，这前100人所获得的权力权重是相同的，每名代表都有一份相等的投票权。因此，无法通过获得超过1%的选票而将权力集中到一个单一代表手上。  
个人或者组织控制区块链的多名代表是有可能的。但是这个过程将需要欺骗很大比例的股东数去支持“傀儡”。  
即使可以建立这51%傀儡，他们扰乱网络的能力仍将是有限的、能够被快速识别快速纠正的。没有工作量证明机制设置的进入障碍，占据多数的诚实用户会把攻击鉴别出来，然后将代码分叉并无视攻击者生产的区块。这种攻击可以扰乱网络，但不会是致命的。  
  
3.3 针对代表的分布式拒绝服务攻击(DDOS)  
因为只有100名代表，   可以想象一个攻击者对每名轮到生产区块的代表依次进行拒绝服务攻击。幸运的是，由于事实上每名代表的标识是其公钥而非IP地址，这种特定攻击的威胁很容易被减轻。这将使确定DDOS攻击目标更为困难。而代表之间的潜在直接连接，将使妨碍他们生产区块变得更为困难。  
  
4.0 基于交易的股权证明机制(TaPOS)  
代表制是一个短时间内达成坚固共识的高效方式，而TaPOS为股东们提供了一个长效机制来直接批准他们的代表的行为。平均而言，51%的股东在6个月内会直接确认每个区块。而取决于活跃流通的股份所占的比例，差不多10%的股东可以在几天内确认区块链。这种直接确认保障了网络的长期安全，并使所有的攻击尝试变得极度清晰易见。  
  
5.0 高质量的服务  
假设一个DPOS系统拥有100亿美元的市场总量，平均每年的交易费为0.25%，代表们合计获得所有交易费的10%，那么每名代表每年能获得25,000美元以使其节点保持在线。  
这是一个利润可观的角色，许多人将为获取它持续竞争。这意味着每个想要获得这份工作的人都会想方设法从拥有这份工作的人那里把它“偷走”。为做到这点，他们将对代表行为进行统计学分析，以找到对于标准算法的任何偏离行为。一旦找到这种偏离，他们就能有希望赢得一些选票。  
那些拥有这份工作的人，可能会全力以赴地证明他们正在按标准软件运行。他们越有效地证明其对区块生产的正直性，越有可能保住他们的工作。你可以想象开发者会很快制作出系统，代表们可以通过这些系统快速证明哪些交易得到了广泛的散播。  
事实上，市场竞争将产生用以证明代表们的正直性与可靠性的最具创造性的解决方案。让网络变得更安全的工作可以获得很多收益，而尝试绕轮网络则得不到什么好处。  
  
6.0 结论  
DPOS流程与TaPOS结合所产生的网络，其网络共识的可证明性将至少3倍于比特币、点点币及未来币网络。DPOS能够更快地达成共识，同时消除随机小股东带来小规模干扰的可能性。经济激励确保了代表们致力于证明他们有良好行为，并可能采用类似于瑞波系统的共识算法(来实现这种证明)。DPOS，事实上，是一种通过无网络分叉之虞的去中心化方式来产生瑞波特殊节点列表的方法。  
  
来自：[blockchaindev](http://blockchaindev.org/article/consensus_introduction.html" \t "_blank)

比特币区块链的数据结构解析

区块链（数据区块、数据块、数据块链）作为比特币的核心概念，对于理解比特币结构起着至关重要的作用。在《[什么是数据区块](http://p2pbucks.com/?p=3015)》中，编者从数据区块分布式交易记录的角度简单的介绍了数据区块的概念。今天，编者从数据区块的程序结构上来详细的分析数据区块里面到底记录了什么信息。  
  
**数据区块文件的位置**  
如果你用的是Bitcoin-Qt客户端，那么数据区块的信息就存在你的电脑里面。每次当你打开Bitcoin-Qt，数据区块都会跟整个P2P网络分布式数据库同步。根据操作系统的不同，数据区块目录blocks一般存放路径为：  
  
**Windows**: %APPDATA%\Bitcoin\   
**Linux**:~/.bitcoin/   
**Mac** **OS**: ~/Library/Application Support/Bitcoin/   
  
**打开数据区块文件**我们打开blocks文件夹，便可以看到很多名为blkXXX.dat的文件，这些文件中保存了传说中的数据区块记录。  
  
  
blkXXX.dat文件在终端下面可以用hexdump打开。  
hexdump -n 10000 -C blk00000.dat  
  
**数据区块结构**hexdump程序把数据区块文件转化为十六进制+ASCII表示，编者解释一下：每一个数据区块记录了六个内容：神奇数、区块大小、数据区块头部信息、交易计数、交易详情。在这当中，**数据区块头部信息**的HASH值是下一个新区块的HASH值的参考目标数，最后一项**交易详情**记录了该区块中所有的交易记录（编者会在以后详细说明交易记录的结构）。  
数据区块结构如下图：  
  
**数据区块头部结构**  
数据区块头部结构中记录了：版本号、前一个区块的记录、Merkle树的根值、[时间戳](http://8btc.com/article-165-1.html" \t "_blank)、目标特征值、随机数。  
在比特币矿工挖矿的过程就是产生新的数据区块的过程（编者以后会详细说明），这个过程需要对比前一个数据区块头部的HASH值和随机数，如果满足一定条件则生成新的区块。**Merkle树**的根值为该区块中所有被记录交易的根节点HASH值，[中本聪](http://8btc.com/article-25-1.html" \t "_blank)用一个HASH树来对每一比交易进行数字签名，以确保每一比交易都不可伪造和没有重复交易，Merkle树就是HASH树的一种。数据区块头部结构如下图：  
  
  
**交易记录**  
  
交易记录就像一个记账本一样，记录了所有比特币的交易信息，每一个比特币玩家的比特币收支情况都被永久的嵌入了数据区块中以供别人查询，这也就是为什么每一比比特币交易都是可以查询到的。这个这些数据区块中的交易数据存放在每一个比特币玩家的客户端节点中，所有的这些节点则组成了比特币那及其坚韧的分布式数据库系统。任何一个节点的数据被破坏都不会影响整个数据库的正常运转，因为其它的健康节点中都保存了完整的数据库。  
在本文中，编者将详细解析数据区块中交易记录的结构。  
  
**交易记录结构**数据区块的交易记录中，详细记载了比特币的交易记录和相关细节。其中在比特币收支详情里面，记录了收支的[比特币地址](http://8btc.com/article-135-1.html" \t "_blank)和merkle节点值等情况，因此收支详情是交易记录中最重要的部分。  
完整的交易记录结构如下表格：  
  
  
**比特币支出和接收交易的结构**在数据块中，比特币支出和接收交易是写在一起的，整个收支记录包括很多比记录。每一比记录都有自己的索引编号以供查询。  
  
每一比记录中包括了生成时间、引用交易的哈希值、交易记录索引编号、比特币支出地址、支出地址数量等细节。每一比收支交易记录都有一个Merkle节点值，这个hash节点值是整个Merkle树的一部分，决定了每一个地址都不能重复交易和被伪造。  
收支交易结构图如下：  


比特币地址格式是怎样的？

比特币对于初学者来说最大的困难就是[比特币地址](http://8btc.com/article-135-1.html" \t "_blank)那一串长长的字符串，各种格式的比特币地址更是让外行人看的迷迷糊糊。就算是比特币的老玩家也未必能准确的说出各种比特币地址格式之间的关系。



我们常用的比特币地址格式有四种：

**1、BASE58格式**

就是人们常说的比特币地址，由1开头的，例如：1PyMi4EYzGZKoxK7DozMMuoQ91EdrMMkBP

**2、HASH160格式**

Tab content 由RIPEMD160算法对130位公钥的[SHA256](http://8btc.com/article-136-1.html" \t "_blank)签名进行计算的结果，如：fbfb58defc272942fc31d00c007b59aa4cb5087a

**3、WIF压缩格式**

即钱包输入格式，是将BASE58格式进行压缩后的结果130位公钥格式 这是最原始的由[ECDSA](http://8btc.com/article-140-1.html" \t "_blank)算法计算出来的比特币公钥，如：0469B0E479C9A358908DB9CF4628BDD643C3F8  
1C4F0096AAD442DA6CA8BCC4FD86A8D47D7A865E178B6D062CC9B70290  
8973952062A1D767DA9B2BD2095D5CCF6E

**4、60位公钥格式**

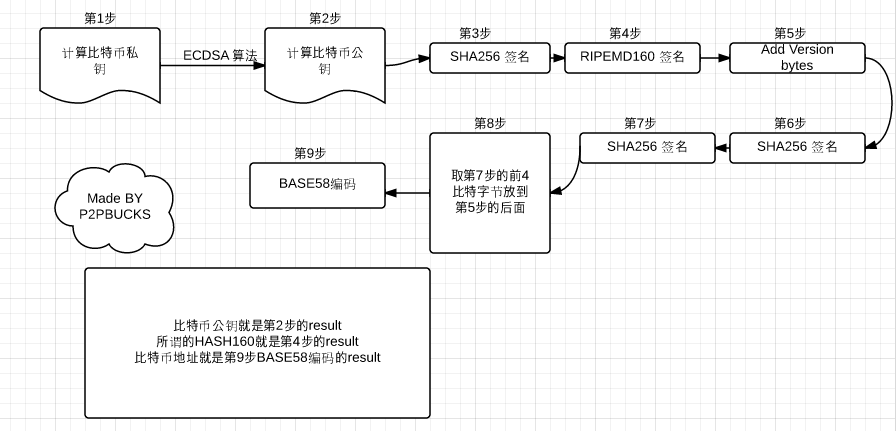
130位公钥进行压缩后的结果，如：0269B0E479C9A358908DB9CF4628BDD643  
C3F81C4F0096AAD442DA6CA8BCC4FD86

比特币地址的生成算法是怎样的？

首先，让我们先简单的说说[比特币地址](http://8btc.com/article-135-1.html" \t "_blank)是怎么算出来的。比特币是建立在数学加密学基础上的，[中本聪](http://8btc.com/article-25-1.html" \t "_blank)大神用了椭圆加密算法（[ECDSA](http://8btc.com/article-140-1.html" \t "_blank)）来产生比特币的私钥和公钥。由私钥是可以计算出公钥的，公钥的值经过一系列数字签名运算会得到比特币地址。

需要说明的是：因为由公钥可以算出比特币地址，所以我们经常把公钥和比特币地址的说法相混淆，但是他们都是指的一个概念。比特币地址只是另一种格式的公钥。

从比特币私钥得到我们所用的比特币地址需要九个步骤。中间要用到[SHA256](http://8btc.com/article-136-1.html" \t "_blank)加密、RIPEMD160加密和BASE58编码。我们最终用到的比特币私钥和比特币地址一般都是经过BASE58编码的结果。整个流程如下图所示。



第一步，随机选取一个32字节的数、大小介于1 ~ 0xFFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFE BAAE DCE6 AF48 A03B BFD2 5E8C D036 4141之间，作为私钥。

18E14A7B6A307F426A94F8114701E7C8E774E7F9A47E2C2035DB29A206321725

第二步，使用椭圆曲线加密算法（ECDSA-secp256k1）计算私钥所对应的非压缩公钥。 (共65字节， 1字节 0x04, 32字节为x坐标，32字节为y坐标）关于公钥压缩、非压缩的问题另文说明。

0450863AD64A87AE8A2FE83C1AF1A8403CB53F53E486D8511DAD8A04887E5B

23522CD470243453A299FA9E77237716103ABC11A1DF38855ED6F2EE187E9C582BA6

第三步，计算公钥的 SHA-256 哈希值

600FFE422B4E00731A59557A5CCA46CC183944191006324A447BDB2D98D4B408

第四步，取上一步结果，计算 RIPEMD-160 哈希值

010966776006953D5567439E5E39F86A0D273BEE

第五步，取上一步结果，前面加入地址版本号（比特币主网版本号“0x00”）

00010966776006953D5567439E5E39F86A0D273BEE

第六步，取上一步结果，计算 SHA-256 哈希值

445C7A8007A93D8733188288BB320A8FE2DEBD2AE1B47F0F50BC10BAE845C094

第七步，取上一步结果，再计算一下 SHA-256 哈希值（哈哈）

D61967F63C7DD183914A4AE452C9F6AD5D462CE3D277798075B107615C1A8A30

第八步，取上一步结果的前4个字节（8位十六进制）

D61967F6

第九步，把这4个字节加在第五步的结果后面，作为校验（这就是比特币地址的16进制形态）。

00010966776006953D5567439E5E39F86A0D273BEED61967F6

第十步，用base58表示法变换一下地址（这就是最常见的比特币地址形态）。

16UwLL9Risc3QfPqBUvKofHmBQ7wMtjvM

（说明: 有些数字以"0x"开头，意思是此数字使用十六进制表示法。"0x"本身没有任何含义，它是C语言流传下来的，约定俗成的写法，比如0xA就是十进制的10。另外，1个字节 = 8位二进制 = 2位十六进制）。

我们经常说的比特币公钥就是指的图中第二步所产生的结果。而HASH160指的是第四步RIPEMD160签名所产生的结果，由于RIPEMD也是一种HASH算法所以就统称为HASH160了。而我们常用的比特币地址就是经过BASE58编码后的结果。

比特币客户端和钱包也接受各种[比特币地址格式](http://8btc.com/article-1928-1.html)，常用的格式有BASE58格式、WIF压缩格式、130位和66位公钥(Public key)格式。进入比巴克的钱包工具点击钱包详情查询页面，如下图，红色框中130位的字符串就是比特币ECDSA算出的原始公钥，下面是公钥压缩成66位的格式，上面两个分别是我们常用的比特币BASE58地址格式和比特币地址WIF（wallet import format）压缩格式。



其实现在各种格式的地址基本上都是通用的，在学习了相关知识后，我们会对比特币体系有更加深入的理解。

[Gobittest网站(国外可访问)](http://gobittest.appspot.com/Address)提供了从比特币原始私钥到公钥再到HASH160和BASE58的计算程序，读者可以参考。

比特币交易的数据结构与签名类型

简介

交易(Transaction)是比特币系统的信息载体，最小单元。而块(Block)就是将这些基础单元打包装箱，贴上封条，并串联起来。巨大算力保障了块的安全，也就保障了单个交易的安全。

类型

交易有三种常见类型：产量交易(Generation)，合成地址交易(Script Hash)，通用地址交易(Pubkey Hash)。该分类并非严格意义的，只是根据交易的输入输出做的简单区分。

Generation TX

每个Block都对应一个产量交易(Generation TX)，该类交易是没有输入交易的，挖出的新币是所有币的源头。

Script Hash TX

该类交易目前不是很常见，大部分人可能没有听说过，但是非常有意义。未来应该会在某些场合频繁使用。该类交易的接受地址不是通常意义的地址，而是一个合成地址，以3开头（对，以3开头的也是[比特币地址](http://8btc.com/article-135-1.html" \t "_blank)！）。三对公私钥，可以生成一个合成地址。在生成过程时指定n of 3中的n，n范围是[1, 3]，若n=1，则仅需一个私钥签名即可花费该地址的币，若n=3，则需要三把私钥依次签名才可以。

Pubkey Hash TX

该类是最常见的交易类型，由N个输入、M个输出构成。

数据结构

交易中存放的是货币所有权的流转信息，所有权登记在比特币地址上(Public Key)。这些信息是全网公开的，以明文形式存储（比特币系统里的所有数据都是明文的），只有当需要转移货币所有权时，才需要用私钥签名来验证。

[](http://8btc.com/data/attachment/portal/201508/13/184109jjr6n99cqo7c761c.jpg)

字段大小-描述-数据类型-解释

4-version, 版本-uint32\_t-交易数据结构的版本号

1+-tx\_in count, 输入数量-var\_int-输入交易的数量

41+-tx\_in-tx\_in[]-输入交易的数组，每个输入>=41字节

1+-tx\_out count, 输出数量-var\_int-输出地址的数量

9+-tx\_out-tx\_out[]-输入地址的数组，每个输入>=9字节

4-lock\_time, 锁定时间-uint32\_t-见下方解释

lock\_time是一个多意字段，表示在某个高度的Block之前或某个时间点之前该交易处于锁定态，无法收录进Block。

值-含义

0-立即生效

< 500000000-含义为Block高度，处于该Block之前为锁定（不生效）

>= 500000000-含义为Unix[时间戳](http://8btc.com/article-165-1.html" \t "_blank)，处于该时刻之前为锁定（不生效）

若该笔交易的所有输入交易的sequence字段，均为INT32最大值(0xffffffff)，则忽略lock\_time字段。否则，该交易在未达到Block高度或达到某个时刻之前，是不会被收录进Block中的。

示例

为了演示方便，我们读取稍早期的块数据，以高度116219 Block为例。

# ~ bitcoind getblock 0000000000007c639f2cbb23e4606a1d022fa4206353b9d92e99f5144bd74611   
{  
“hash” : “0000000000007c639f2cbb23e4606a1d022fa4206353b9d92e99f5144bd74611″,  
“confirmations” : 144667,  
“size” : 1536,  
“height” : 116219,  
“version” : 1,  
“merkleroot” : “587fefd748f899f84d0fa1d8a3876fdb406a4bb8f54a31445cb72564701daea6″,  
“tx” : [  
"be8f08d7f519eb863a68cf292ca51dbab7c9b49f50a96d13f2db32e432db363e",  
"a387039eca66297ba51ef2da3dcc8a0fc745bcb511e20ed9505cc6762be037bb",  
"2bd83162e264abf59f9124ca517050065f8c8eed2a21fbf85d454ee4e0e4c267",  
"028cfae228f8a4b0caee9c566bd41aed36bcd237cdc0eb18f0331d1e87111743",  
"3a06b6615756dc3363a8567fbfa8fe978ee0ba06eb33fd844886a0f01149ad62"  
],  
“time” : 1301705313,  
“nonce” : 1826107553,  
“bits” : “1b00f339″,  
“difficulty” : 68977.78463021,  
“previousblockhash” : “00000000000010d549135eb39bd3bbb1047df8e1512357216e8a85c57a1efbfb”,  
“nextblockhash” : “000000000000e9fcc59a6850f64a94476a30f5fe35d6d8c4b4ce0b1b04103a77″  
}  
该Block里面有5笔交易，第一笔为Generation TX，解析出来看一下具体内容：

# ~ bitcoind getrawtransaction be8f08d7f519eb863a68cf292ca51dbab7c9b49f50a96d13f2db32e432db363e 1  
{  
“hex” : “01000000010000000000000000000000000000000000000000000000000000000000000000ffffffff070439f3001b0134ffffffff014034152a010000004341045b3aaa284d169c5ae2d20d0b0673468ed3506aa8fea5976eacaf1ff304456f6522fbce1a646a24005b8b8e771a671f564ca6c03e484a1c394bf96e2a4ad01dceac00000000″,  
“txid” : “be8f08d7f519eb863a68cf292ca51dbab7c9b49f50a96d13f2db32e432db363e”,  
“version” : 1,  
“locktime” : 0,  
“vin” : [  
{  
"[coinbase](http://8btc.com/article-108-1.html" \t "_blank)" : "0439f3001b0134",  
"sequence" : 4294967295  
}  
],  
“vout” : [  
{  
"value" : 50.01000000,  
"n" : 0,  
"scriptPubKey" : {  
"asm" : "045b3aaa284d169c5ae2d20d0b0673468ed3506aa8fea5976eacaf1ff304456f6522fbce1a646a24005b8b8e771a671f564ca6c03e484a1c394bf96e2a4ad01dce OP\_CHECKSIG",  
"hex" : "41045b3aaa284d169c5ae2d20d0b0673468ed3506aa8fea5976eacaf1ff304456f6522fbce1a646a24005b8b8e771a671f564ca6c03e484a1c394bf96e2a4ad01dceac",  
"reqSigs" : 1,  
"type" : "pubkey",  
"addresses" : [  
"1LgZTvoTJ6quJNCURmBUaJJkWWQZXkQnDn"  
]  
}  
}  
],  
“blockhash” : “0000000000007c639f2cbb23e4606a1d022fa4206353b9d92e99f5144bd74611″,  
“confirmations” : 145029,  
“time” : 1301705313,  
“blocktime” : 1301705313  
}  
Generation TX的输入不是一个交易，而带有coinbase字段的结构。该字段的值由挖出此Block的人填写，这是一种“特权”：可以把信息写入货币系统（大家很喜欢用系统中的数据结构字段名来命名站点，例如blockchain、coinbase等，这些词的各种后缀域名都被抢注一空）。[中本聪](http://8btc.com/article-25-1.html" \t "_blank)在比特币的第一个交易中的写入的coinbase值是：

“coinbase”:”04ffff001d0104455468652054696d65732030332f4a616e2f32303039204368616e63656c6c6f72206f6e206272696e6b206f66207365636f6e64206261696c6f757420666f722062616e6b73″  
将该段16进制转换为ASCII字符，就是那段著名的创世块留言：

The Times 03/Jan/2009 Chancellor on brink of second bailout for banks  
接下来展示的是一个三个输入、两个输出的普通交易：  
# ~ bitcoind getrawtransaction 028cfae228f8a4b0caee9c566bd41aed36bcd237cdc0eb18f0331d1e87111743 1  
{  
“hex” : “″,  
“txid” : “028cfae228f8a4b0caee9c566bd41aed36bcd237cdc0eb18f0331d1e87111743″,  
“version” : 1,  
“locktime” : 0,  
“vin” : [  
{  
"txid" : "b79a4803014177ea0139c95f09c6903c01bb8f80d039631afd68cabf7eb0f3c9",  
"vout" : 0,  
"scriptSig" : {  
"asm" : "3044022055bac1856ecbc377dd5e869b1a84ed1d5228c987b098c095030c12431a4d5249022055523130a9d0af5fc27828aba43b464ecb1991172ba2a509b5fbd6cac97ff3af01 048aefd78bba80e2d1686225b755dacea890c9ca1be10ec98173d7d5f2fefbbf881a6e918f3b051f8aaaa3fcc18bbf65097ce8d30d5a7e5ef8d1005eaafd4b3fbe",  
"hex" : "473044022055bac1856ecbc377dd5e869b1a84ed1d5228c987b098c095030c12431a4d5249022055523130a9d0af5fc27828aba43b464ecb1991172ba2a509b5fbd6cac97ff3af0141048aefd78bba80e2d1686225b755dacea890c9ca1be10ec98173d7d5f2fefbbf881a6e918f3b051f8aaaa3fcc18bbf65097ce8d30d5a7e5ef8d1005eaafd4b3fbe"  
},  
"sequence" : 4294967295  
},  
{  
"txid" : "b79a4803014177ea0139c95f09c6903c01bb8f80d039631afd68cabf7eb0f3c9",  
"vout" : 1,  
"scriptSig" : {  
"asm" : "304402206b993231adec55e6085e75f7dc5ca6c19e42e744cd60abaff957b1c352b3ef9a022022a22fec37dfa2c646c78d9a0753d56cb4393e8d0b22dc580ef1aa6cccef208d01 042ff65bd6b3ef04253225405ccc3ab2dd926ff2ee48aac210819698440f35d785ec3cec92a51330eb0c76cf49e9e474fb9159ab41653a9c1725c031449d31026a",  
"hex" : "47304402206b993231adec55e6085e75f7dc5ca6c19e42e744cd60abaff957b1c352b3ef9a022022a22fec37dfa2c646c78d9a0753d56cb4393e8d0b22dc580ef1aa6cccef208d0141042ff65bd6b3ef04253225405ccc3ab2dd926ff2ee48aac210819698440f35d785ec3cec92a51330eb0c76cf49e9e474fb9159ab41653a9c1725c031449d31026a"  
},  
"sequence" : 4294967295  
},  
{  
"txid" : "da30b272fb73d78108ff80ddd1ac2f76419533af79ad06a5b3c70fc4a62086c9",  
"vout" : 1,  
"scriptSig" : {  
"asm" : "3045022040a5d957e087ed61e80f1110bcaf4901b5317c257711a6cbc54d6b98b6a8563f02210081e3697031fe82774b8f44dd3660901e61ac5a99bff2d0efc83ad261da5b4f1d01 04a7d1a57e650613d3414ebd59e3192229dc09d3613e547bdd1f83435cc4ca0a11c679d96456cae75b1f5563728ec7da1c1f42606db15bf554dbe8a829f3a8fe2f",  
"hex" : "483045022040a5d957e087ed61e80f1110bcaf4901b5317c257711a6cbc54d6b98b6a8563f02210081e3697031fe82774b8f44dd3660901e61ac5a99bff2d0efc83ad261da5b4f1d014104a7d1a57e650613d3414ebd59e3192229dc09d3613e547bdd1f83435cc4ca0a11c679d96456cae75b1f5563728ec7da1c1f42606db15bf554dbe8a829f3a8fe2f"  
},  
"sequence" : 4294967295  
}  
],  
“vout” : [  
{  
"value" : 0.84000000,  
"n" : 0,  
"scriptPubKey" : {  
"asm" : "OP\_DUP OP\_HASH160 634228c26cf40a02a05db93f2f98b768a8e0e61b OP\_EQUALVERIFY OP\_CHECKSIG",  
"hex" : "76a914634228c26cf40a02a05db93f2f98b768a8e0e61b88ac",  
"reqSigs" : 1,  
"type" : "pubkeyhash",  
"addresses" : [  
"1A3q9pDtR4h8wpvyb8SVpiNPpT8ZNbHY8h"  
]  
}  
},  
{  
“value” : 156.83000000,  
“n” : 1,  
“scriptPubKey” : {  
“asm” : “OP\_DUP OP\_HASH160 7514080ab2fcac0764de3a77d10cb790c71c74c2 OP\_EQUALVERIFY OP\_CHECKSIG”,  
“hex” : “76a9147514080ab2fcac0764de3a77d10cb790c71c74c288ac”,  
“reqSigs” : 1,  
“type” : “pubkeyhash”,  
“addresses” : [  
"1Bg44FZsoTeYteRykC1XHz8facWYKhGvQ8"  
]  
}  
}  
],  
“blockhash” : “0000000000007c639f2cbb23e4606a1d022fa4206353b9d92e99f5144bd74611″,  
“confirmations” : 147751,  
“time” : 1301705313,  
“blocktime” : 1301705313  
}  
字段hex记录了所有相关信息，后面显示的是hex解析出来的各类字段信息。下面把逐个分解hex内容（hex可以从上面的直接看到）：  
01000000 // 版本号，UINT32  
03 // Tx输入数量，变长INT。3个输入。

/\*\*\* 第一组Input Tx \*\*\*/  
// Tx Hash，固定32字节  
c9f3b07ebfca68fd1a6339d0808fbb013c90c6095fc93901ea77410103489ab7  
00000000 // 消费的Tx位于前向交易输出的第0个，UINT32，固定4字节  
8a // 签名的长度, 0x8A = 138字节  
// 138字节长度的签名，含有两个部分：公钥+签名  
47 // 公钥长度，0×47 = 71字节  
3044022055bac1856ecbc377dd5e869b1a84ed1d5228c987b098c095030c12431a4d5249022055523130a9d0af5fc27828aba43b464ecb1991172ba2a509b5fbd6cac97ff3af01  
41 // 签名长度，0×41 = 65字节  
048aefd78bba80e2d1686225b755dacea890c9ca1be10ec98173d7d5f2fefbbf881a6e918f3b051f8aaaa3fcc18bbf65097ce8d30d5a7e5ef8d1005eaafd4b3fbe  
ffffffff // sequence，0xffffffff = 4294967295， UINT32, 固定4字节

/\*\*\* 第二组Input Tx。与上同理，省略分解 \*\*\*/  
c9f3b07ebfca68fd1a6339d0808fbb013c90c6095fc93901ea77410103489ab7010000008a47304402206b993231adec55e6085e75f7dc5ca6c19e42e744cd60abaff957b1c352b3ef9a022022a22fec37dfa2c646c78d9a0753d56cb4393e8d0b22dc580ef1aa6cccef208d0141042ff65bd6b3ef04253225405ccc3ab2dd926ff2ee48aac210819698440f35d785ec3cec92a51330eb0c76cf49e9e474fb9159ab41653a9c1725c031449d31026affffffff

/\*\*\* 第三组Input Tx \*\*\*/  
c98620a6c40fc7b3a506ad79af339541762facd1dd80ff0881d773fb72b230da010000008b483045022040a5d957e087ed61e80f1110bcaf4901b5317c257711a6cbc54d6b98b6a8563f02210081e3697031fe82774b8f44dd3660901e61ac5a99bff2d0efc83ad261da5b4f1d014104a7d1a57e650613d3414ebd59e3192229dc09d3613e547bdd1f83435cc4ca0a11c679d96456cae75b1f5563728ec7da1c1f42606db15bf554dbe8a829f3a8fe2fffffffff

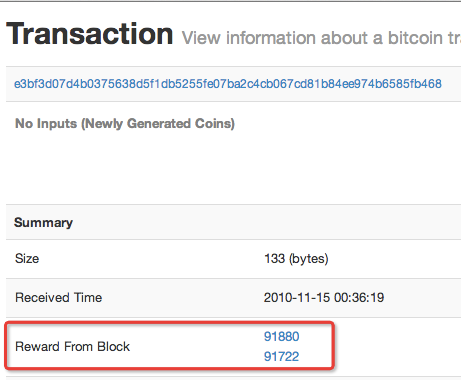
02 // Tx输出数量，变长INT。两个输出。

/\*\*\* 第一组输出 \*\*\*/  
00bd010500000000 // 输出的币值，UINT64，8个字节。字节序需翻转，~= 0x000000000501bd00 = 84000000 satoshi  
19 // 输出目的地址字节数, 0×19 = 25字节，由一些操作码与数值构成  
// 目标地址  
// 0×76 -> OP\_DUP(stack ops)  
// 0xa9 -> OP\_HASH160(crypto)  
// 0×14 -> 长度，0×14 = 20字节  
76 a9 14  
// 地址的HASH160值，20字节  
634228c26cf40a02a05db93f2f98b768a8e0e61b  
// 0×88 -> OP\_EQUALVERIFY(bit logic)  
// 0xac -> OP\_CHECKSIG(crypto)  
88 ac

/\*\*\* 第二组输出 \*\*\*/  
c096c7a603000000  
19  
76 a9 14 7514080ab2fcac0764de3a77d10cb790c71c74c2 88 ac

00000000 // lock\_time，UINT32，固定4字节  
Tx Hash，俗称交易ID，由hex得出：Tx Hash = [SHA256](http://8btc.com/article-136-1.html" \t "_blank)(SHA256(hex))。由于每个交易只能成为下一个的输入，有且仅有一次，那么不存在输入完全相同的交易，那么就不存在相同的Tx Hash（SHA256碰撞概率极小，所以无需考虑Hash碰撞的问题，就像无需考虑地址私钥被别人撞到一样）。

即便如此，在系统里依然产生了相同的Tx Hash，是某位矿工兄弟挖出Block后，打包Block时忘记修改Generation Tx coinbase字段的值，币量相同且输出至相同的地址，那么就构造了两个完全一模一样的交易，分别位于两个Block的第一个位置。这个对系统不会产生什么问题，但只要花费其中一笔，另一个也被花费了。相同的Generation Tx相当于覆盖了另一个，白白损失了挖出的币。该交易ID为e3bf3d07d4b0375638d5f1db5255fe07ba2c4cb067cd81b84ee974b6585fb468，第一次出现在#91722，第二次出现在#91880。



交易签名

签名是对所有权的验证，节点收到交易广播后，会对交易进行验证，通过后则收录进内存、打包进Block，否则，丢弃之。签名就类似传统纸质合同盖章、签字过程，合法转移所有权的保证手段。

签名类型

由于一个交易的输入、输出都可能具有多个，那么签名也具有多种类型，目前共三类：SIGHASH\_ALL, SIGHASH\_NONE, SIGHASH\_SINGLE。

SIGHASH\_ALL

该签名类型为默认类型，也是目前绝大部分交易采用的，顾名思义即签名整单交易。首先，组织所有输出、输入，就像上文分解Hex过程一样，每个输入都对应一个签名，暂时留空，其他包括sequence等字段均须填写，这样就形成了一个完整的交易Hex（只缺签名字段）。然后，每一个输入均需使用私钥对该段数据进行签名，签名完成后各自填入相应的位置，N个输入N个签名。简单理解就是：对于该笔单子，认可且只认可的这些输入、输出，并同意花费我的那笔输入。

SIGHASH\_NONE

该签名类型是最自由松散的，仅对输入签名，不对输出签名，输出可以任意指定。某人对某笔币签名后交给你，你可以在任意时刻填入任意接受地址，广播出去令其生效。简单理解就是：我同意花费我的那笔钱，至于给谁，我不关心。

SIGHASH\_SINGLE

该签名类型其次自由松散，仅对自己的输入、输出签名，并留空sequence字段。其输入的次序对应其输出的次序，比如输入是第3个，那么签名的输出也是第三个。简单理解就是：我同意花费我的那笔钱，且只能花费到我认可的输出，至于单子里的其他输入、输出，我不关心。

交易的构造、签名与广播

上篇介绍了交易结构、签名等，为了更直观的认识比特币，借助bitcoind演示手动构造并广播交易的完整过程。

普通交易

1. 1. 找出未花费的币（unspent output）

通过命令：listunspent [minconf=1] [maxconf=9999999] ["address",...]列出某个地址未花费的币(交易)，minconf/maxconf表示该笔收入交易的确认数范围，如果需要列出还未确认的交易，需将minconf设置为0。

执行：

bitcoind listunspent 0 100 ‘["1Lab618UuWjLmVA1Q64tHZXcLoc4397ZX3"]‘  
输出：

[  
{  
"txid" : "296ea7bf981b44999d689853d17fe0ceb852a8a34e68fcd19f0a41e589132156",  
"vout" : 0,  
"address" : "1Lab618UuWjLmVA1Q64tHZXcLoc4397ZX3",  
"account" : "",  
"scriptPubKey" : "76a914d6c492056f3f99692b56967a42b8ad44ce76b67a88ac",  
"amount" : 0.19900000,  
"confirmations" : 1  
}  
]  
我们找到该地址的一个未花费交易，位于交易296ea7bf981b4499…9f0a41e589132156的第0个位置。

1. 2. 创建待发送交易

创建待发送交易，由命令：createrawtransaction [{"txid":txid,"vout":n},...] {address:amount,…}来完成。我们将 0.1 BTC发送至 1Q8s4qDRbCbFypG5AFNR9tFC57PStkPX1x ，并支付 0.0001 BTC做为矿工费。输入交易的额度为 0.199 ，输出为 0.1 + 0.0001 = 0.1001 ，那么还剩余： 0.199 – 0.1001 = 0.0989 ，将此作为找零发回给自己。

执行：

bitcoind createrawtransaction \  
‘[{"txid":"296ea7bf981b44999d689853d17fe0ceb852a8a34e68fcd19f0a41e589132156","vout":0}]‘ \  
‘{“1Q8s4qDRbCbFypG5AFNR9tFC57PStkPX1x”:0.1, “1Lab618UuWjLmVA1Q64tHZXcLoc4397ZX3″:0.0989}’  
输出：

010000000156211389e5410a9fd1fc684ea3a852b8cee07fd15398689d99441b98bfa76e290000000000ffffffff0280969800000000001976a914fdc7990956642433ea75cabdcc0a9447c5d2b4ee88acd0e89600000000001976a914d6c492056f3f99692b56967a42b8ad44ce76b67a88ac00000000  
通过命令：decoderawtransaction ，可以将此段十六进制字符串解码。

执行：

bitcoind decoderawtransaction ’010000000156211389e5410a9fd1fc684ea3a852b8cee07fd15398689d99441b98bfa76e290000000000ffffffff0280969800000000001976a914fdc7990956642433ea75cabdcc0a9447c5d2b4ee88acd0e89600000000001976a914d6c492056f3f99692b56967a42b8ad44ce76b67a88ac00000000′  
输出：  
{  
“txid” : “54f773a3fdf7cb3292fc76b46c97e536348b3a0715886dbfd2f60e115fb3a8f0″,  
“version” : 1,  
“locktime” : 0,  
“vin” : [  
{  
"txid" : "296ea7bf981b44999d689853d17fe0ceb852a8a34e68fcd19f0a41e589132156",  
"vout" : 0,  
"scriptSig" : {  
"asm" : "",  
"hex" : ""  
},  
"sequence" : 4294967295  
}  
],  
“vout” : [  
{  
"value" : 0.10000000,  
"n" : 0,  
"scriptPubKey" : {  
"asm" : "OP\_DUP OP\_HASH160 fdc7990956642433ea75cabdcc0a9447c5d2b4ee OP\_EQUALVERIFY OP\_CHECKSIG",  
"hex" : "76a914fdc7990956642433ea75cabdcc0a9447c5d2b4ee88ac",  
"reqSigs" : 1,  
"type" : "pubkeyhash",  
"addresses" : [  
"1Q8s4qDRbCbFypG5AFNR9tFC57PStkPX1x"  
]  
}  
},  
{  
“value” : 0.09890000,  
“n” : 1,  
“scriptPubKey” : {  
“asm” : “OP\_DUP OP\_HASH160 d6c492056f3f99692b56967a42b8ad44ce76b67a OP\_EQUALVERIFY OP\_CHECKSIG”,  
“hex” : “76a914d6c492056f3f99692b56967a42b8ad44ce76b67a88ac”,  
“reqSigs” : 1,  
“type” : “pubkeyhash”,  
“addresses” : [  
"1Lab618UuWjLmVA1Q64tHZXcLoc4397ZX3"  
]  
}  
}  
]  
}  
至此，一个“空白交易”就构造好了，尚未使用私钥对交易进行签名，字段scriptSig是留空的，无签名的交易是无效的。此时的Tx ID并不是最终的Tx ID，填入签名后Tx ID会发生变化。

在手动创建交易时，务必注意输入、输出的值，非常容易犯错的是忘记构造找零输出（如非必要勿手动构造交易）。曾经有人构造交易时忘记找零，发生了支付 200 BTC 的矿工费的人间惨剧，所幸的是收录该笔交易的Block由著名挖矿团队“烤猫（Friedcat）”挖得，该团队非常厚道的退回了多余费用。

1. 3. 签名

交易签名使用命令：

signrawtransaction \  
[{"txid":txid,"vout":n,"scriptPubKey":hex,"redeemScript":hex},...] [,...] \  
[sighashtype="ALL"]  
第一个参数是创建的待签名交易的十六进制字符串；  
第二个参数有点类似创建交易时的参数，不过需要多出一个公钥字段scriptPubKey，其他节点验证交易时是通过公钥和签名来完成的，所以要提供公钥；如果是合成地址，则需要提供redeemScript；  
第三个参数是即将花费的币所在地址的私钥，用来对交易进行签名，如果该地址私钥已经导入至bitcoind中，则无需显式提供；  
最后一个参数表示签名类型，在上一篇里，介绍了三种交易签名类型；  
签名之前需要找到scriptPubKey，提取输入交易信息即可获取(也可以根据其公钥自行计算)，由命令：getrawtransaction [verbose=0]完成。

执行：

bitcoind getrawtransaction 296ea7bf981b44999d689853d17fe0ceb852a8a34e68fcd19f0a41e589132156 1  
输出：

{  
“hex” : “01000000010511331f639e974283d3909496787a660583dc88f41598d177e225b5f352314a000000006c493046022100be8c796122ec598295e6dfd6664a20a7e20704a17f76d3d925c9ec421ca60bc1022100cf9f2d7b9f24285f7c119c91f24521e5483f6b141de6ee55658fa70116ee04d4012103cad07f6de0b181891b5291a5bc82b228fe6509699648b0b53556dc0057eeb5a4ffffffff0160a62f01000000001976a914d6c492056f3f99692b56967a42b8ad44ce76b67a88ac00000000″,  
“txid” : “296ea7bf981b44999d689853d17fe0ceb852a8a34e68fcd19f0a41e589132156″,  
“version” : 1,  
“locktime” : 0,  
“vin” : [  
{  
"txid" : "4a3152f3b525e277d19815f488dc8305667a78969490d38342979e631f331105",  
"vout" : 0,  
"scriptSig" : {  
"asm" : "3046022100be8c796122ec598295e6dfd6664a20a7e20704a17f76d3d925c9ec421ca60bc1022100cf9f2d7b9f24285f7c119c91f24521e5483f6b141de6ee55658fa70116ee04d401 03cad07f6de0b181891b5291a5bc82b228fe6509699648b0b53556dc0057eeb5a4",  
"hex" : "493046022100be8c796122ec598295e6dfd6664a20a7e20704a17f76d3d925c9ec421ca60bc1022100cf9f2d7b9f24285f7c119c91f24521e5483f6b141de6ee55658fa70116ee04d4012103cad07f6de0b181891b5291a5bc82b228fe6509699648b0b53556dc0057eeb5a4"  
},  
"sequence" : 4294967295  
}  
],  
“vout” : [  
{  
"value" : 0.19900000,  
"n" : 0,  
"scriptPubKey" : {  
"asm" : "OP\_DUP OP\_HASH160 d6c492056f3f99692b56967a42b8ad44ce76b67a OP\_EQUALVERIFY OP\_CHECKSIG",  
"hex" : "76a914d6c492056f3f99692b56967a42b8ad44ce76b67a88ac",  
"reqSigs" : 1,  
"type" : "pubkeyhash",  
"addresses" : [  
"1Lab618UuWjLmVA1Q64tHZXcLoc4397ZX3"  
]  
}  
}  
],  
“blockhash” : “000000000000000488f18f7659acd85b2bd06a5ed2c4439eea74a8b968d16656″,  
“confirmations” : 19,  
“time” : 1383235737,  
“blocktime” : 1383235737  
}  
scriptPubKey位于”vout”[0]–>“scriptPubKey”–>“hex”，即： 76a914d6c492056f3f99692b56967a42b8ad44ce76b67a88ac 。

签名使用[ECDSA](http://8btc.com/article-140-1.html" \t "_blank)算法，对其，“空白交易”签名之，执行：

bitcoind signrawtransaction \  
“010000000156211389e5410a9fd1fc684ea3a852b8cee07fd15398689d99441b98bfa76e290000000000ffffffff0280969800000000001976a914fdc7990956642433ea75cabdcc0a9447c5d2b4ee88acd0e89600000000001976a914d6c492056f3f99692b56967a42b8ad44ce76b67a88ac00000000″ \  
‘[{"txid":"296ea7bf981b44999d689853d17fe0ceb852a8a34e68fcd19f0a41e589132156","vout":0,"scriptPubKey":"76a914d6c492056f3f99692b56967a42b8ad44ce76b67a88ac"}]‘  
输出：

{  
“hex” : “″,  
“complete” : true  
}  
签名后，签名值会填入上文所述的空字段中，从而得到一个完整的交易。可通过上文介绍的命令decoderawtransaction 解码查看之。

最后一步，就是将其广播出去，等待网络传播至所有节点，约10~60秒广播至全球节点，取决与你的节点的网络连接状况。稍后一些时刻，就会进入Block中。广播由命令sendrawtransaction 来完成。如果没有运行节点，可以通过公共节点的API进行广播，例如：blockchain.info/pushtx。

执行：

bitcoind sendrawtransaction \  
“″  
输出：

b5f8da1ea9e02ec3cc0765f9600f49945e94ed4b0c88ed0648896bf3e213205d  
返回的是Transaction Hash值，即该交易的ID。至此，交易构造、签名、发送的完整过程完成了。

合成地址交易

合成地址以3开头，可以实现多方管理资产，极大提高安全性，也可以轻松实现基于比特币原生的三方交易担保支付。一个M-of-N的模式：

m {pubkey}…{pubkey} n OP\_CHECKMULTISIG  
M和N需满足：

1<=N<=3  
1<=M<=N  
可以是1 of 1，1 of 2，2 of 3等组合，通常选择N=3：

1 of 3，最大程度私钥冗余。防丢私钥损失，3把私钥中任意一把即可签名发币，即使丢失2把都可以保障不受损失；  
2 of 3，提高私钥冗余度的同时解决单点信任问题。3把私钥任意2把私钥可签名发币，三方不完全信任的情形，即中介交易中，非常适用；  
3 of 3，最大程度解决资金信任问题，无私钥冗余。必须3把私钥全部签名才能发币，适用多方共同管理重要资产，但任何一方遗失私钥均造成严重损失；  
合成地址的交易构造、签名、发送过程与普通交易类似，这里只介绍如何创建一个合成地址。大神Gavin Andresen已经演示过，下面内容摘自其gist.

首先，需要三对公钥、私钥。公钥创建地址、私钥用于签名。

# No.1  
0491bba2510912a5bd37da1fb5b1673010e43d2c6d812c514e91bfa9f2eb129e1c183329db55bd868e209aac2fbc02cb33d98fe74bf23f0c235d6126b1d8334f86 / 5JaTXbAUmfPYZFRwrYaALK48fN6sFJp4rHqq2QSXs8ucfpE4yQU  
# No.2  
04865c40293a680cb9c020e7b1e106d8c1916d3cef99aa431a56d253e69256dac09ef122b1a986818a7cb624532f062c1d1f8722084861c5c3291ccffef4ec6874 / 5Jb7fCeh1Wtm4yBBg3q3XbT6B525i17kVhy3vMC9AqfR6FH2qGk  
# No.3  
048d2455d2403e08708fc1f556002f1b6cd83f992d085097f9974ab08a28838f07896fbab08f39495e15fa6fad6edbfb1e754e35fa1c7844c41f322a1863d46213 / 5JFjmGo5Fww9p8gvx48qBYDJNAzR9pmH5S389axMtDyPT8ddqmw  
使用命令：createmultisig <’["key","key"]‘>来合成，其中key为公钥，创建地址时仅需公钥。创建类型是2 of 3.

输入：

bitcoind createmultisig 2 \  
‘["0491bba2510912a5bd37da1fb5b1673010e43d2c6d812c514e91bfa9f2eb129e1c183329db55bd868e209aac2fbc02cb33d98fe74bf23f0c235d6126b1d8334f86","04865c40293a680cb9c020e7b1e106d8c1916d3cef99aa431a56d253e69256dac09ef122b1a986818a7cb624532f062c1d1f8722084861c5c3291ccffef4ec6874","048d2455d2403e08708fc1f556002f1b6cd83f992d085097f9974ab08a28838f07896fbab08f39495e15fa6fad6edbfb1e754e35fa1c7844c41f322a1863d46213"]‘  
输出：

{  
“address” : “3QJmV3qfvL9SuYo34YihAf3sRCW3qSinyC”,  
“redeemScript” : “52410491bba2510912a5bd37da1fb5b1673010e43d2c6d812c514e91bfa9f2eb129e1c183329db55bd868e209aac2fbc02cb33d98fe74bf23f0c235d6126b1d8334f864104865c40293a680cb9c020e7b1e106d8c1916d3cef99aa431a56d253e69256dac09ef122b1a986818a7cb624532f062c1d1f8722084861c5c3291ccffef4ec687441048d2455d2403e08708fc1f556002f1b6cd83f992d085097f9974ab08a28838f07896fbab08f39495e15fa6fad6edbfb1e754e35fa1c7844c41f322a1863d4621353ae”  
}  
得到的合成地址是：3QJmV3qfvL9SuYo34YihAf3sRCW3qSinyC，该地址没有公钥，仅有redeemScript，作用与公钥相同。后续的构造、签名、发送过程与上文普通地址交易类似，略去。

[白皮书翻译]

Delegated Proof-of-Stake (DPOS)  
by Daniel Larimer

摘要

这份白皮书介绍了一种新的POS实作方式，能够在几秒之内验证转帐，并在短时间内提供比现有的POS更大的安全性.

在比特币產生单一区块的时间之内DPOS能够让你的转帐被20%的持股人验证，而在比特币体制内当你的转帐已经几乎不可逆时  
(六个区块， 一个小时)，DPOS已经让持股人经由代表完成100%验证.

1.0 背景

分散式交易总帐必须要是安全的， 清楚的，并且要在尽可能短的时间之内不可逆转，以使的系统更加健壮以及去中心化.

实务上有两个面向必须考虑:

选取唯一的节点来產生区块，并使的总帐不可逆转.

1.1 工作量证明机制 (POW)

史上第一个成功解决这个问题的, 是比特币使用的POW机制，POW使得產生新的区块具有计算机层次的难度.

POW的运作就像是乐透彩一般，平均每十分鐘才有一个节点开採出新的区块. 如果有两个节点同时发现一个区块，那麼网络就会基於后来的节点的决议，来抉择要採纳那一个区块.

统计上来说，在经过个区块(大约1小时)之后，一笔转帐就可以被视為明确而不可逆转；然而，核心开发者需要经过120个区块(大约一天)，才能够确定网络免於延长区块链的攻击.

儘管不太可能產生延长区块链，实际上任何具有经济上强大资源的人都有可能產生延长区块链或是囊括足够的算力来冻结使用者帐户.

1.2 权益证明机制(POS)

POS机制有许许多多的变体，不过基本概念就是说，產生区块的难易度应该和你的持股成比例(占全网络的%)有关联.

直至今日有两个POS体系存在：Peercoin以及Nxt.

Peercoin使用混合机制，当中你的持股将会影响你的挖矿难度，而Nxt使用决定性的演算法来决定產生下一个区块的随机持股人.Nxt的演算法基於你的餘额来调整你被选上的机率.

当Peercoun以及Nxt各有各的方法来解决如何產生下一个区块的问题时，他们并没有提出一种有效率的方法来在有限时间内使得区块链无可逆转地安全.

Peercoin需要至少6个区块(1小时)，而依据我们找到的资讯Nxt需要10个区块.

我们无法找到10个区块之后Nxt如何保证安全性.

我们之前发表过TaPOS的白皮书，当中听到每一笔转帐包含了上一个区块所產生的hash值.

在这样的系统裡系统越来越安全，并且由於最终区块都会被每一个持股人投票，因此无可逆转.

TaPOS所面临的挑战是在其中并没有指定谁来產生下一个区块.

1.3 Ripple共识制度

Rippke的共识演算法基於了一个独特的节点清单来允许了群节点同意达成协议.

这一份创始的节点清单就像是一个社团一样，如果要纳入新的节点，就必须要囊括51%的旧成员选票.

核心的51%决定了共识為何，而圈外人无能為力.

由於这样的社团以中心化开始，也将维持中心化，而当持股人无事可做的时候体系便要崩坏.

如同比特币以及Peercoin一样，Ripple剥夺了持股人的投票权，因此比其他的体系更加中心化.

2.0 代表制的POS

套用DAC的比喻来说，去中心化意味著每个持股人依照持股比例而有相当的影响力，而超过了51%的持股人选票则意味著牢不可破.

困难在於在有限时间内达成51%相当不易.

為了达成这个目的，每个持股人可以委任他的投票权给一位代表.

前100强的代表可以排表轮流產生区块.

每个代表都会被分配一个时段来產生区块.

如果代表人无法在时段之内產生区块，那麼他会被跳过，下一个代表就会递补上来.

所有的代表会收到转帐手续费总额平均的10%.

例如平均来说每个区块含有100share手续费，那麼每个代表会收到1 share.

理论上来说，可能会由於某些代表可能无法在时间内传送区块并造成块链分岔.   
然而这实际上是不可能的，因為每个代表可以直接和他之前以及之后的代表连线.   
这样的直接连线能够确保支付被处理.   
  
在这样的模式底下，区块每30秒就会生成，而在正常的网络情况下块链分岔是几乎不可能的，或者说可以在几分鐘之内被解决.

2.1 成為代表

要成為代表你必须用你的公钥在网络上註册，并且签署一个32bit的唯一识别码.

这一段识别码就可以在每一笔转帐的header当中引用.

2.1 委任你的投票权

每个钱包都有偏好设定的项目，当中你可以选取一个或更多的代表并加以评比.   
设定完成之后，使用者的转帐将会从当前的代表中做為选票选给特定的代表.   
正常情况下使用者并不会為了投票权而转帐，因為这是要扣手续费的，但是紧急情况下某些使用者可能愿意支付来达成主动投票的效果.

2.3 让代表们保持诚实

每个钱包都会有一个指示器来显示他们的代表的绩效.

如果他遗失了太多区块那麼就建议使用者换新的代表.

如果任何代表被发现签署了无效区块，那麼钱包将强制需要在新的转帐之前选取新的代表.

2.4 解决块链分岔

如同POW以及其他POS体制，最佳的区块链就是含有最长的有效区块链.

任何时候当一个代表错失了签署区块的时候，那麼区块链就比潜在竞争者少了一个区块.

只要排定在你的转帐之后的100个区块的51%能够被发佈并產生，那麼你就不用担心你的转帐位在块链分岔当中,或许避免块链分岔最重要的就是在第一时间察觉.

由於代表们经由產生区块获得支付，并因而尽可能保持100%上线时间以避免被投票出局，损失收入。你可以安全地假设如果过去10个区块当中有1到2个被遗失那麼网络上的一部分可能经歷了连线的问题，使用者应该保持警觉， 并要求更多确认.

如果过去10个区块当中有5个遗失，那表示你可能位在块链分岔上面，停止交易并等待分岔问题被解决.

能够在短时间之内(五分鐘之内)侦测到并警告使用者网络分裂，对於减少潜在的损失至為重要.

更重要的是明白您是否正处於块链分岔上面.

2.5 100个代表足够去中心化吗？

去中心化的定义是由於广泛流行，因此难以被牵制.

我们认為自由市场将是去中心化的终极样貌，而进场阻碍则是中心化的土壤.

正如同许多事物，中心化的程度可以被量化，因此我们将比较DPOS和其他的方案的中心化程度.

2.3.1 比特币

比特币目前以DPOW运作，因此大约有10个代表掌握了算力的主流.

算力本身是中心化的，因為经济实力强的人可以佈署相当的规模，以使的和他竞争无利可图.

最终POW使得现任者难以被换下来.

相较於比特币，DPOS至少在產生区块上面10倍地更加去中心化，而在市场竞争上面或许无限地更加去中心化.

除了在算力上面有限度地去中心化之外，当提到比特币持有者的比例分配时，我们认為比特币很清楚的更加去中心化.   
如果你考虑到比特币的使用者数量，当中只有很小一部分参与了挖矿.

2.3.2 Peercoin

Peercoin使用复合式的体制，因此会因為POW造成中心化. 如同比特币有矿池一样.

相较於比特币，Peercoin当然更加去中心化；然而POS挖矿需要使用者将它的钱包维持解锁状态，只有一小部分的持股人参与了任何形式的挖矿.

2.3.3 Nxt

Nxt使用透明锻造，当中下一个节点是被预选的.

以DPOS来说就像是你只能委任给自己，而你有机会锻造区块的机会和你的持股成正比.

在这种情况下，Nxt比比特币和Peerxoin更加去中心化，但仍然苦於大多数使用者不太参与，因為大部份的使用者不会整天打开电脑并参与整个过程.

从这个角度来说我们可以说Nxt是经由一小群持股人来达成安全性.

事实上，如果你不现身投票，你就丧失了投票权.

為了解决这个问题一些Nxt使用者将他们的持股託付给一些第三方来代替他们行使职权.

这其实是经由某种的DPOS来增加持股人参与，但也增加了託付第三方的风险.

3.0 攻击

一般来说有两种攻击方式不得不防:

DOS以及双重支付.   
任何持有51%的人(在比特币，Nxt以及其他的网络上面)都有可能藉由不将部份或是全部的转帐纳入到总帐当中执行DOS攻击.

双重支付是在当网络试图达成协议时获取短期的好处.

為了避免这两种攻击，网络必须儘速达成51%持股人共识.

3.1避免交易被排除

选出100个代表并轮值，就代表著转帐即使只被1%持股人认可就可以在30分鐘之内生效.

表示没有代表可以排除投给其他代表的选票以及交易.

3.2 少数代表的集权

由於代表是在前100强当中彼此对等，因此取得大於1%的选票没有意义。

或许同一个人或组织可能掌控多於一个代表席次，但是这个过程将会牵涉到欺骗许多持股人成為傀儡.

即使可以设立51个傀儡来崩坏网络，他们的威力仍然是相当有限的，并将会被快速辨认出来，并修复错误.

没有POW造成的进场阻碍，诚实的大多数将能够快速找出攻击的源头，并作出分岔来忽略攻击者產生的区块.

这将会是破坏性的，但并不是致命的.   
  
3.3 代表的DDOS攻击  
由於只有100个代表，可以想见攻击者可以在轮到某个代表的时候攻击之.

幸运的是每个代表都只会显示出它的公钥而不是IP.

这将会使DDOS更加困难. 考虑到代表之间会有直接连线，区块的传送将更加难以阻挡.   
  
4.0 TaPOS

代表制意味著能够在短时间之内更有效率地取得共识，而TaPOS代表著使用者将会随时间验证代表的行為.

平均来说51%的持股人将会在6个月之内验证每个区块.

取决於流通中的股份数量，大约10%的持股人将会在几天之内验证区块链.

这样的直接验证长期来说将确保网络安全，并使的所有的攻击无所遁形.

5.0 高品质服务

假设DPOS体制将能够有10亿美金的市场，年平均转帐费用是0.25%,代表赚取10%的转帐费用，那麼每个全勤的代表将每年能够赚25000美金.

这是一个很有赚头的位置，每个人都会挤破头.

这代表每个求职者将会试图降低代表的支持基础, 经由作统计分析来找出代表裡面任何背离标準演算法的行為.

一旦某些乖离被发现，检举者就能赢得选票.

代表们将竭尽全力来证明他们是运行著标準软体，他们对於產生区块的正直性的证明将能够帮助他们保住工作.

可以想见开发者将快速做出能够让代表快速担保转帐能够广為传播的系统.

事实上市场竞争将能够產生最有创意的方案来确保代表的正直以及可靠性.

确保网络安全有利可图，试图崩坏网络却无利可图.

6.0 结论

DPOS背后的机制混合了TaPOS将產生一个具有比比特币， Peercoin，Nxt三倍以上的更能够被验证的网络.

DPOS将能够在更快的达成共识的同时，消除少数随机持股人造成小规模破坏的可能性.

经济上的诱因将会使代表们证明他们的良好行為，并将套用类似於Ripple的共识机制.

DPOS实质上以去中心化的方式產生了一个Ripple的节点清单，而不必担心网络分裂.