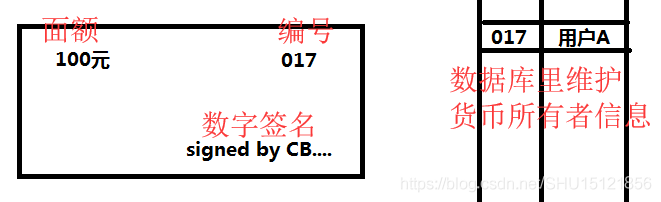
1. 数字货币与纸质货币的区别

比如央行要发行数字货币，如果直接为货币的面额等信息，用央行的私钥签名，然后使用的时候，用户直接拿央行的公钥验证签名，就这样用是行不通的。因为完全可以用这个数字货币再去买东西了，这叫**双花攻击（double spending attack）**。签名保证了面额等信息不能修改，但这个**数字文件可以复制很多份**。如果是纸质货币，花出去自己手里就没有了，这是数字货币和纸质货币的区别。

1. 中心化的数字货币

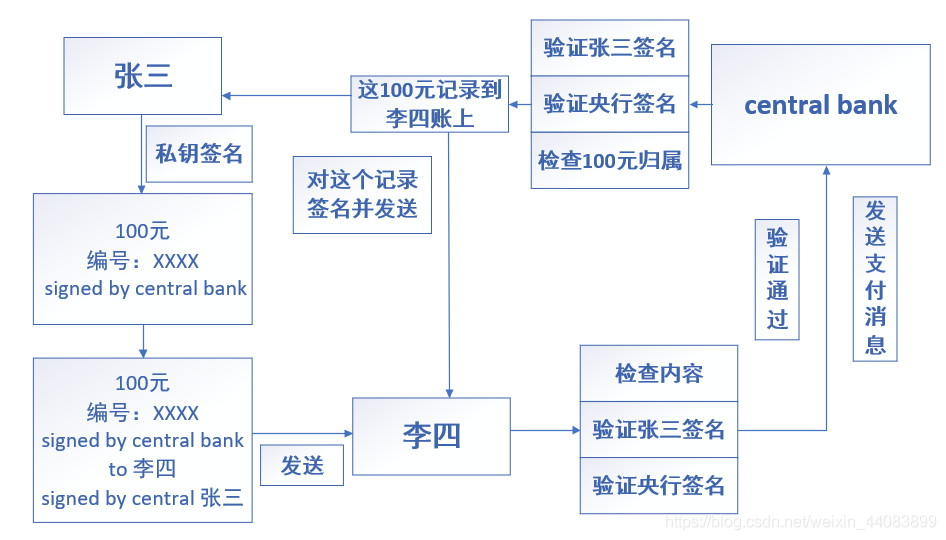
为了解决上述的**双花问题，**仅仅有面额是不够的，每个数字货币还要有编号。中心化方案中央行还要维护一个数据库，即记录每个编号的数字货币是归哪个用户所有。



在支付时，不仅要用公钥验证签名是央行签署的，还要通过央行验证该货币是归自己所有，央行再将货币所有者改成支付给的那个用户。不仅数字货币的发行是由央行来做，而且每次交易都要由央行确认其合法性，这种方案是一个**中心化**的方案。

下面是一个中心化数字货币流通的具体实现：

假设张三有100元数字货币，要支付给李四



首先张三把这100元支付给李四这个消息和对这个消息的签名发给李四，李四收到后首先用张三的公钥验证这个消息是不是张三发的(因为有签名，张三也无法抵赖)，再根据央行的签名确定这100元是不是央行发行的，如果是，那么再把整个信息发给央行，央行同样验证张三的签名和自己的签名，然后根据编号在数据库里查询这100元是否在张三的账户上，如果是，把这个编号对应的100元记录在李四的账户上，然后把这个记账的信息签名后发给李四和张三。

**央行在货币上签名**：保证货币发行的有效性

**货币编号**：防止double-spending

**张三对支付消息签名**：确认这笔交易是张三认可的（他人无法伪造和张三无法抵赖）

**李四验证**：确认交易的内容、确认张三对交易的认可、确认这张100元是由央行发行的(但李四无法确定这张100元现在是不是张三账上的)

**央行处理交易**：确认张三对交易的认可、确认这张100元是由自己发行的、检查这100元是否在张三账上、依据交易内容修改账本(央行需要维护一个总的账本)、将账本的修改信息发送签名后发送给李四和张三，李四确定自己账户上多了这100元，张三确认自己账上少了这100元。

1. 去中心化的数字货币

将央行的职能改成由广大的用户来共同承担，也就是**去中心化**的方案，这是比特币等数字货币系统要解决的问题，也即：

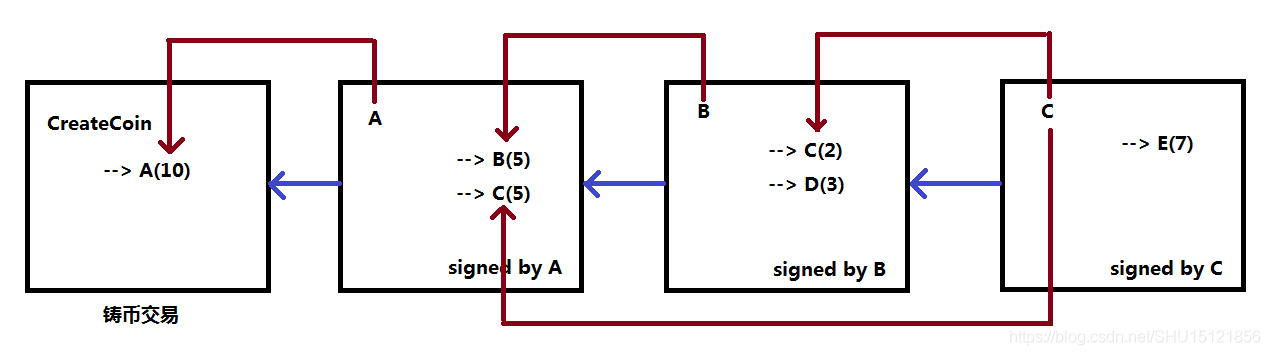
1. **怎么决定数字货币的发行及发行量**
2. **怎么验证交易的合法性，防止双花攻击**

第一个问题在比特币系统中是挖矿决定的，后面会讲到

第二个问题的解决，也是和中心化的方案一样要维护一个数据结构，但不再是由央行来维护，而是由所有用户来共同维护，这个数据结构也不再是关系表，而是区块链。

* 1. 比特币的交易过程

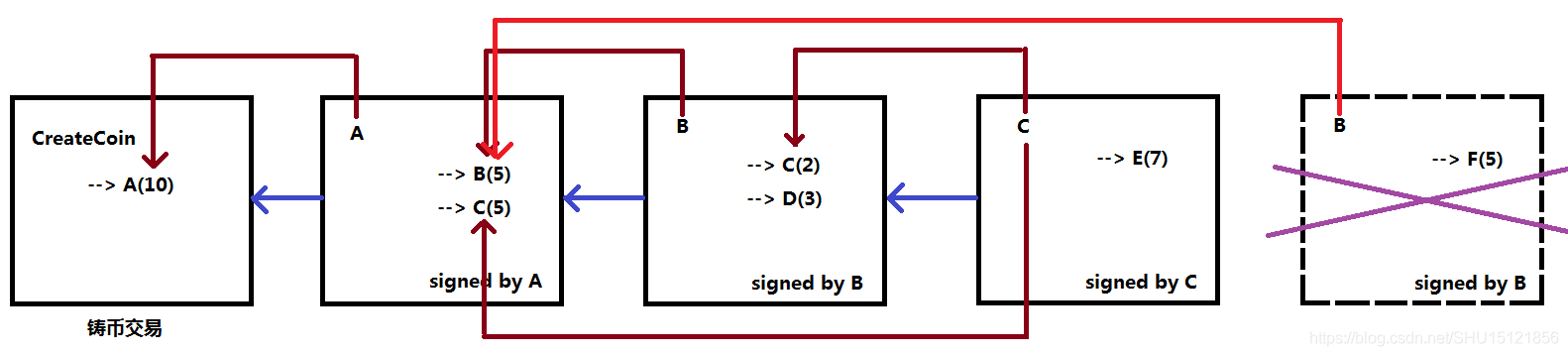
如下图，假设用户A获得了**铸币权（发行货币的权利）**，他发行了10个比特币，即自己获得了10个比特币。然后他将这10个比特币转给B和C，每个人分5个比特币。接下来B给C 2个货币，给D 3个货币。最后C将所得的7个货币全部给E。



|  |
| --- |
| 比特币系统中每个交易都分为输入部分和输出部分，**输入部分要给出这笔交易的比特币的来源以及付款方的公钥，输出部分要给出收款人的公钥的哈希值**。比特币系统中的收款地址就是收款人的公钥取哈希再经过一些转换得到的。 |

这里涉及两种哈希指针，一种就是图中蓝色的，将各个区块串起来的哈希指针；另一种是图中深红色的，是为了说明币的来源是从哪个交易来的。

这里“说明币的来源”也就**防止了双花攻击**，如在下图中，B已经将自己的5个比特币花掉了，假设B尝试再花一次，将5个比特币转给F。这时**顺着区块链去检查这个区块到来源交易之间的区块**，发现B已经花了来源区块的比特币，说明这新个交易是不合法的，也就不会接受这个区块进入区块链。

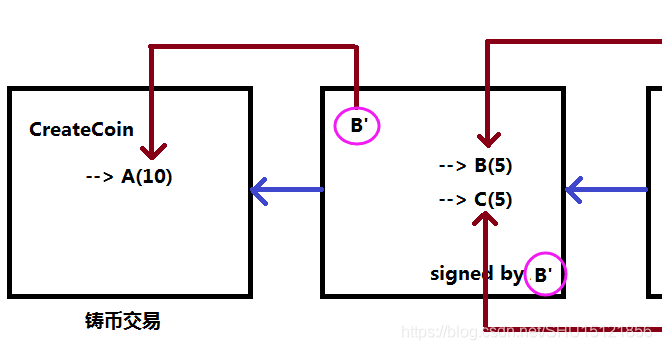


类似于银行没有提供查询某用户的银行账号的功能一样，**比特币系统也没有提供查询某用户的公钥或账户地址的功能**，要向某用户转账，就需要对方提供公钥或账户地址。这种情况收款方可以把**公钥公布在网站上(一般会做成二维码形式)**。

然而A向B转账，除了A需要知道B的地址，B也需要知道A的公钥。因为一方面A的公钥代表A的身份，B要知道转账的是谁，另一方面是为了验证比特币交易中A的签名（私钥签名公钥验证），也就是说所有结点都需要知道A的公钥才行。

|  |
| --- |
| 区块链上每个结点都需要独立验证，即使是一个和交易无关的旁观者也要验证这笔交易的合法性。一般来说，在比特币中由**矿工节点**验证交易的合法性，用A的公钥验证交易的[数字签名](https://so.csdn.net/so/search?q=%E6%95%B0%E5%AD%97%E7%AD%BE%E5%90%8D&spm=1001.2101.3001.7020" \t "_blank)(确认这笔交易是A是同意的)，同时用A的公钥验证币的来源是A（即前面一笔交易的收款地址是A）。矿工节点通过记账可以获得此次交易的交易费，**交易费的好处在于激励矿工打包此次交易。** |

在这种情况下，如何知道A的公钥？这里如前面所学A的公钥是A自己写在这笔交易的输入部分里，即在交易中付款方自己宣称的。但这样是否会造成其他人可以伪造成A来发起交易？如B的同伙B’说自己是A，然后用自己的私钥签名，将自己的公钥说是A的公钥放在交易输入部分里，尝试将A账户上的比特币转走。



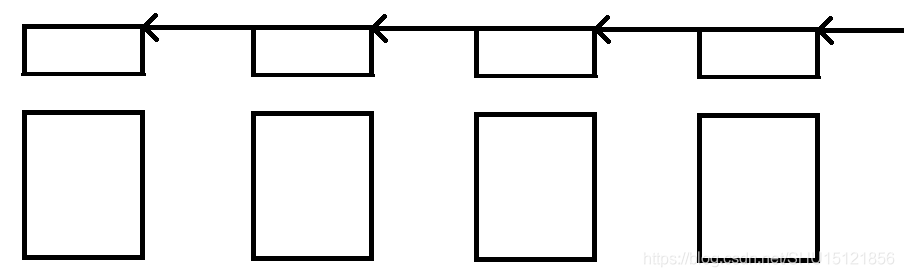
但因为**币的来源（图中铸币交易）中交易的输出部分有收款人A的公钥的哈希值**，这时B’伪造的公钥的哈希就和A的公钥的哈希对不上了，所以可以防止这种攻击。倘若B’直接使用A的公钥写上去，但因为没有A的私钥，这时签名就无法用A的公钥验证了，显然也是不行的。

|  |
| --- |
| 交易的输入部分和输出部分实际上都是脚本，A的公钥也是写在这笔交易的输入脚本里面。对公钥的验证过程，实际上就是把这笔交易的输入脚本，和币的来源的交易的输出脚本拼在一起，看看能不能顺利执行。因此**每笔交易的输出(收款)，只能单独一次作为下一笔交易的输入(币的来源）。如果币来源的数量多于本次交易的数量，A需要在收款方中添加自己的收款地址作为找零账户。** |

|  |
| --- |
| 在前面的几张图里，每个区块里只有一个交易，实际系统中每个区块中可以有很多交易，这些交易就组成了上节课学习的Merkle Tree。 |

* 1. **区块结构**

这里补充了上节学习的区块链中结点的构造，详述一下块头和块身。



上图中连起来的是块头，块身挂在区块上，哈希指针和块身没有直接联系（间接联系就是通过Merkle Tree的根哈希建立的）。

|  |
| --- |
| 注意全结点（fully validation node）是有块身的，需要验证所有交易的合法性；轻结点（light node）是没有块身的，没有办法独立验证交易的合法性。 轻结点没有参与区块链的构造和维护，只是利用了区块链中的部分信息。 系统中大部分结点是轻结点，全结点不是很多。 |

1. **块头**

块头里保存的是区块的宏观的信息：

* 1. 用的是比特币的哪个版本的协议 version
  2. 指向前一个区块块头的哈希指针（注意！这里的哈希值只计算前一个区块的块头，块头保存的Merkle Tree的根哈希就已经可以保证区块中保存的所有交易没有被篡改了）
  3. 整棵Merkle Tree的根哈希值
  4. 挖矿的难度目标阈值target
  5. 挖矿用的随机数nonce，要使得 H ( b l o c k h e a d e r ) ≤ t a r g e t

1. **块身**

包含交易列表

* 1. **比特币的共识协议**

每个账户都可以发布交易，区块链可以看做账本，那么**发布的交易应该写在哪个区块里**呢？交易广播给每个区块，每个人都在自己本地的区块链上写入交易，如何**保证写入后的一致性**？也就是说**账本的内容要取得分布式的共识（distributed consensus）**。

比特币系统中的共识要考虑到有些结点是有恶意的，假设系统中大部分结点是友好的，有恶意的结点占少数。

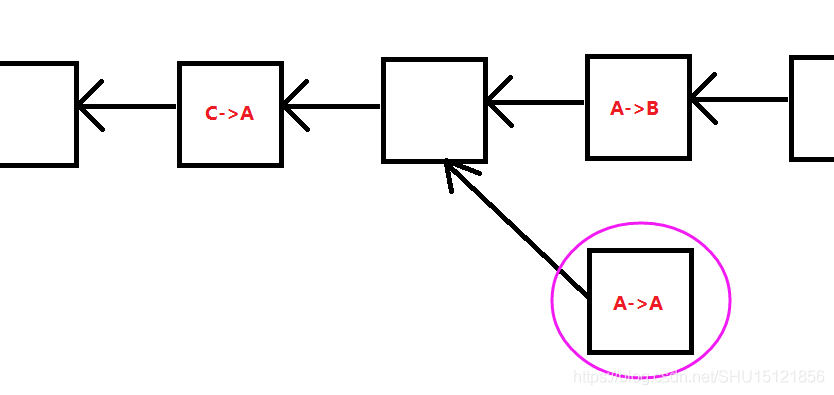
一种思路是用投票的方式，将所有交易写入一个候选区块，然后发给所有结点，大家验证这个区块中的交易是不是都是合法的，然后投赞成和反对票，按一定票比通过后将候选区块写入区块链中。这种思路的问题是一个membership的问题（任何基于投票的系统都要考虑谁有投票权，例如hyperledger fabric就是一个联盟链的协议，规定了谁可以参加）

因为比特币系统中要产生账户只要在本地生成公私钥对就可以了，所以如果用这种方式，那么有恶意的人就可以进行**女巫攻击（sybil attack）**，只要不断产生账户，然后获取大量的投票权就能控制整个区块链了。

**比特币系统中不是用账户来投票，而是用计算力来投票**，每个结点都可以在本地组装出一个候选区块，把它认为合法的区块放在这个区块里，然后就开始尝试各种nonce值（4 byte），使得H(block header)≤target。如果某个结点找到了符合要求的nonce，也就获得了**记账权——往比特币去中心化的账本（区块链）里写入下一个区块的权力**，其它结点收到这个区块之后，要验证这个区块的合法性（如检查target的编码nBits域设置的是不是符合比特币协议规定的难度要求、检查带nonce的块头哈希值是不是小于target、检查块身中的每个交易是否都有合法的签名、检查每个交易都没有双花等）。

**最长链原则：**

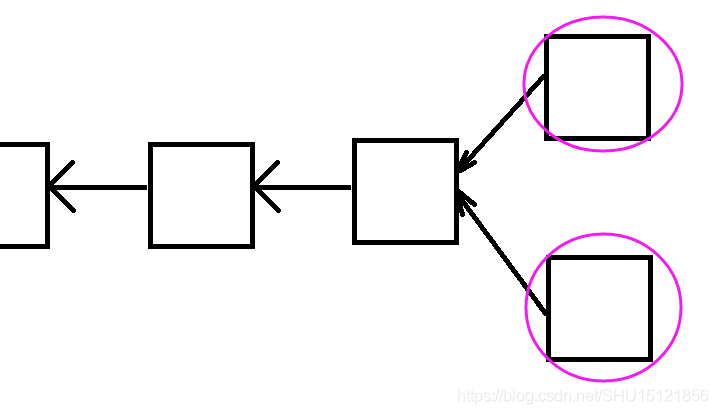
根据上述的计算力投票方式，如果检查出的候选区块完全合法，就应当接受吗？如下图，C转账给A，A又将这些比特币转给B，然后A又发起了一次交易把比特币转给自己，这个交易的候选区块在下面，它希望挂在如图的位置上（分支上）。这里其实A所做的相当于将A转给B这个交易回滚了(因为这两个区块不是先后顺序的，而是同时接在C->A这个区块之后，同时将C->A区块的输出脚本接到自己的输入脚本，也就是说分别对C->A区块的输入进行了各自的改动，整体上等于A的金额没有变化)，仅仅做双花的检查会发现这两个分支都是没问题的，但如果接收了，那么在这个分支上A又获得了转给B的那些比特币。



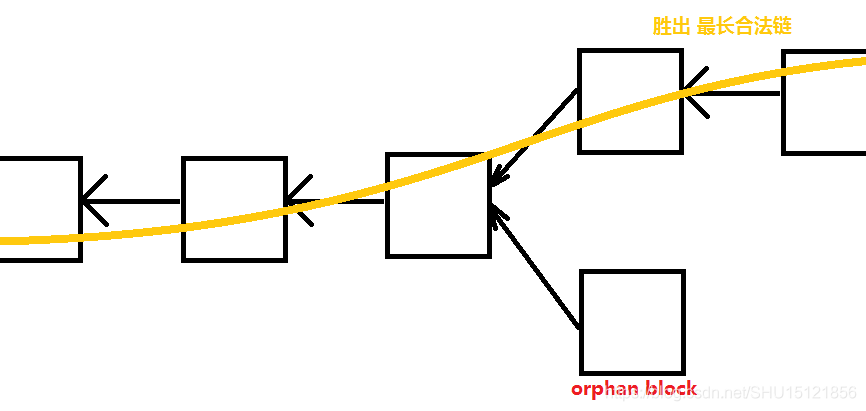
这是一个**分叉攻击（forking attack）** 的例子，通过往区块链中间位置插入区块，来回滚某个已经发生了的交易。这样一个候选区块检测其内容是合法的，但不应当被接受。因此，为了防止这种攻击，比特币协议中规定，**接受的区块应当是在扩展最长合法链**。这样一来，作为分叉攻击的区块在进行插入时，A->B路线上的区块链已经很长了，因此矿工节点不会认为A->A这个区块应该上链。

|  |
| --- |
| 注：如何知道某个区块是插入在哪的？块头有指向前一个区块块头的哈希指针，通过这个哈希指针就能判断了。 |

还需要注意一点，区块链在正常运行下也会出现分叉。如两个矿工都发布了在扩展最长合法链的合法区块，并且都想挂在同一个区块上。这种情况下**不同结点的抉择可能不一样，就看分布式的不同的结点先接收到哪一个区块了**，先接收到的那个区块就会被这个结点接受。



这种等长的，多条最长合法链（分支）的情况会维持一段时间，直到某个分支胜出。假如上面的先找到下一个区块，那么下面这个区块就成为了orphan block被丢弃掉。



这里，部分结点接受上面那个区块，部分结点接受下面那个区块，然后他们竞争，谁先找到下一个结点，最终形成一个共识的最长合法链。

|  |
| --- |
| 什么是接受一个区块？**如果一个结点收到一个区块后，沿着这个区块继续往下扩展，那么就算该结点接受了这个区块**。 |

* 1. **比特币的奖励机制**

让大家去竞争记账权的动力是什么？获得记账权的结点本身有一定的权力，如**可以决定哪些交易被写入下一个区块中**，但这不应当成为竞争记账权的主要动力，因为比特币系统设计来希望所有交易都能被公平写入账本。出块奖励机制解决了这个问题。

比特币协议中规定，**获得记账权的结点，在发布的区块里可以有一个特殊的交易——铸币交易，在这个交易中可以发布一定数量的比特币，这是发行比特币（产生新的比特币）的唯一方法，不必指定币的来源**。这也就解决了前面谈及的去中心化系统中的第一个问题——谁来发行比特币和发行多少比特币。

|  |
| --- |
| 比特币刚出现时，出块奖励是50个BTC，比特币协议规定**每21万个区块之后，出块奖励就要减半**，也就变成25个BTC。如今的情况是每个区块中能产生12.5个BTC。  因为orphan block不在最长合法链里，所以里面的出块奖励的铸币交易也就无效了。 |

* 1. **一些问题**

1. **共识协议取得的共识到底是什么？**

在普通的分布式系统中，如分布式哈希表里，取得的共识就是哈希表中的内容。比特币系统中，**共识协议取得的共识是去中心化的账本里的交易**。

只有获得记账权的结点可以往区块里写交易，而获得记账权的途径就是解那个不等式puzzle，根据第一节课学习的哈希函数puzzle friendly的性质，求解这个puzzle的过程没有捷径，只能一个一个nonce去尝试，所以可以作为工作量的证明，算力越强得到出块奖励的概率也就越大，所以才说比特币系统中是靠算力来投票的。

1. **靠算力投票是如何避免女巫攻击的？**

因为投票是靠算力的，创建多少个账户都没有影响，创建很多账户并不会使每秒尝试的nonce数目增加。