自私挖矿

代炜琦

wqdai@hust.edu.cn
华中科技大学
Huazhong University of Science and Technology

什么是自私挖矿

和其他某些针对区块链和加密货币的攻击不同，自私挖矿不以破坏区块链网络正常运行为目的，而是更纯粹的为了获得更大利润。这一点不同非常重要，因为这个原因，诸如比特币在内的加密货币遇到自私挖矿攻击的可能性要高于其他网络攻击类型，比如常常提到的 **51% 攻击**。

什么是自私挖矿

51%攻击虽然危害巨大，但实施者会很少，除了攻击难度较高外，更主要的原因是这类攻击会直接导致该加密货币共识受损、信用崩盘，从而间接导致价格大幅下跌。这对于寄希望于加密货币价格上涨来获益的攻击者而言，反而弊大于利。

什么是自私挖矿

自私挖矿则不同。它没有破坏网络原本的共识机制，因此不会导致整个区块链网络信用受到影响而价格下跌。而且，它的攻击难度远低于 51% 攻击。自私挖矿是 “区块扣留攻击” (又称扣块攻击，Block Withholding Attack) 的一种，之所以这样命名，主要是因为这种攻击的具体实施方式就是在某一时间段内扣留新区块不公开，即所谓的 “扣块”。

什么是自私挖矿

自私挖矿主要通过扣留区块，拖延公布区块的时间来达成。- 自私挖矿的目的不是破坏加密货币的区块链网络，而是获得更大利润。- 由于攻击门槛相对较低，且收益好，理论上，这种攻击更容易出现。- 避免这种攻击需要改进区块链网络的共识机制。

自私挖矿如何操作

这种攻击主要由矿工发起。我们以比特币区块链作为例子。

简单地说，攻击者挖到新区块后藏起来不公布，其他诚实矿工因为不知道新区块的存在，还是继续在旧区块基础上挖矿。等到攻击者挖到第二枚区块后便会同时公布手中藏着的两枚区块，这时，区块链分叉就出现了。只要攻击者比诚实矿工多挖一枚区块，攻击者所在的分叉就是最长链：根据比特币的共识机制，矿工只在最长链后面挖矿。因此，原本诚实矿工们所在的那条链，因为比攻击者的分叉短，便作废了。此时此刻，攻击者因为挖到了两枚新区块而获得相应收益；而诚实矿工的分叉被废弃，他们什么也得不到。

自私挖矿

以色列学者 Ittay Eyal 在 2014 年的时候就在不同场合提到这种攻击的危险性。他通过论文、公开演讲等形式向大众解释他的理论和分析。根据 Eyal 教授的数学模型推导，进行自私挖矿的攻击者只需要拥有全网 1/3 的算力，就可以保证自己获得更多的收益了。相比 51% 攻击，自私挖矿显得更容易，更有吸引力。

自私挖矿的解决方案

已经有BIPS，以降低自私挖矿攻击的概率，例如当分叉发生时，随机分配矿工给不同的分支，或者可以提供一个矿池所能达到的阈值限制。这与政府试图停止自然垄断以允许竞争的做法一样。

另一种解决方案是根据时间戳来区分区块，这样，如果一个矿工在很短的时间内释放了很多区块，那么，网络的其余部分将根据区块被哈希的时间戳，以及区块被报告到网络上的时间戳，来对其## 标题 ##有效性进行加权。

链式结构和merkle树

代炜琦

wqdai@hust.edu.cn
华中科技大学
Huazhong University of Science and Technology

链式结构

回忆一下我们一开始对区块链的描述：

“**区块链是实现务中心分布式总账的一种技术。**除了采用块、链结构的典型区块链以外，还有其他的发方式实现分布式总账这个需求。**总账技术的基本单元是‘交易’，整个账本是由一条条的交易构成。**‘块’类似于帐本中的页，每页都记录了若干条交易，把一页一页的账页按照时间顺序装订起来，就形成了一个完整的账本--‘区块链’。‘块’是交易的容器，‘块’通过密码学算法相连接，形成了按照时间序列的‘链’。这种组织账本的好处是**由密码学算法保证了无法篡改链上的单独交易，除非整体性的篡改。**”

链式结构

根据描述，可以看出来，‘块’是交易的容器。每个交易都要放到容器里面，然后把整个容器用密码学算法进行连接，形成一个完整的链。

这种数据的组织方式最大的好处就是数据**易于保持完整，并且从密码学角度看安全性较高。**然而，这个好处是有代价的——数据一直不停的增长。

链式结构

对于比特币系统来说，这个问题并不大，因为截止目前为止，比特币仍然是每10分钟一个区块，每个区块1MB，即便到了100年后，总的数据量也不会大到单机无法处理。但是对于某些企业级应用的区块链系统来说，情况就完全不一样了。每个区块可能会非常大，生成区块的速度也会非常快。这种情况下，区块链的数据量增长是飞快的。

怎么解决数据量过大的问题呢?

在我们传统的数据系统中，也存在数据不断增长，数据量过大的问题。以传统的交易型系统为例，**由于系统中的核心设计理念是保存账户的最终状态，只需要把历史的交易过程数据移到其他专门的存储设备上，主机数据库保存账户的最新状态和最近一段时间的交易记录即可。**

（因此，我们在网银中查阅历史交易时，通常是有时间限制的。）

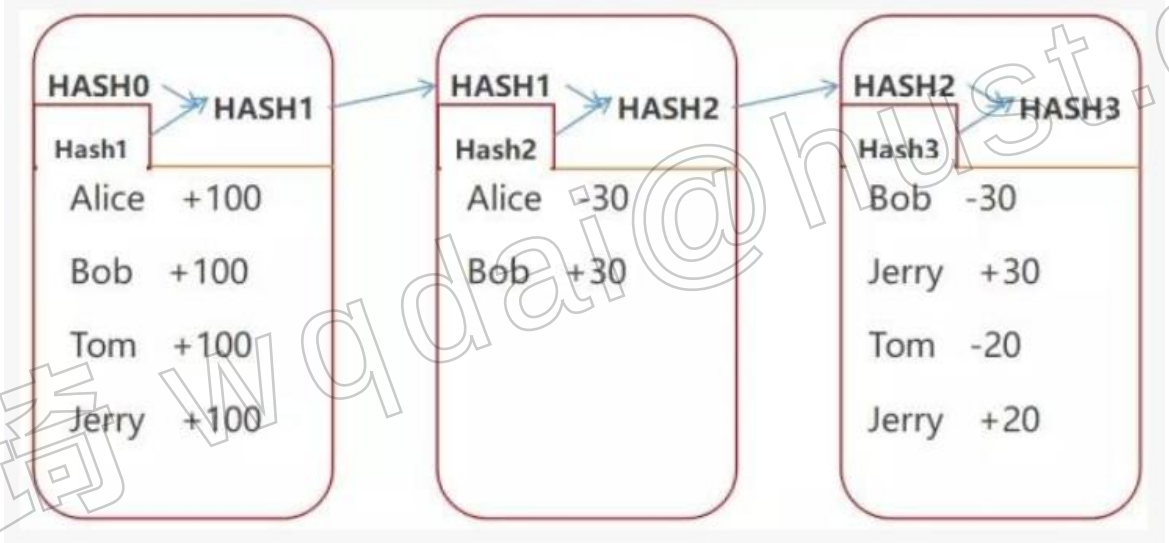
怎么解决数据量过大的问题呢?

但是在区块链系统中，尤其是使用UTXO方式存储交易的区块链系统中，保存的都是交易的过程。**如果一个账户一直没有交易，它则不会出现在最新的区块中。**

那么按照传统数据库删除历史数据的方式，只要一个区块中有一个交易一直没有后续交易（就是没有人使用这个交易的输出账户），为了维护整个区块链系统的密码学完整性和安全性，这个区块就必须保留，同时这个区块之后的所有区块也必须保留。

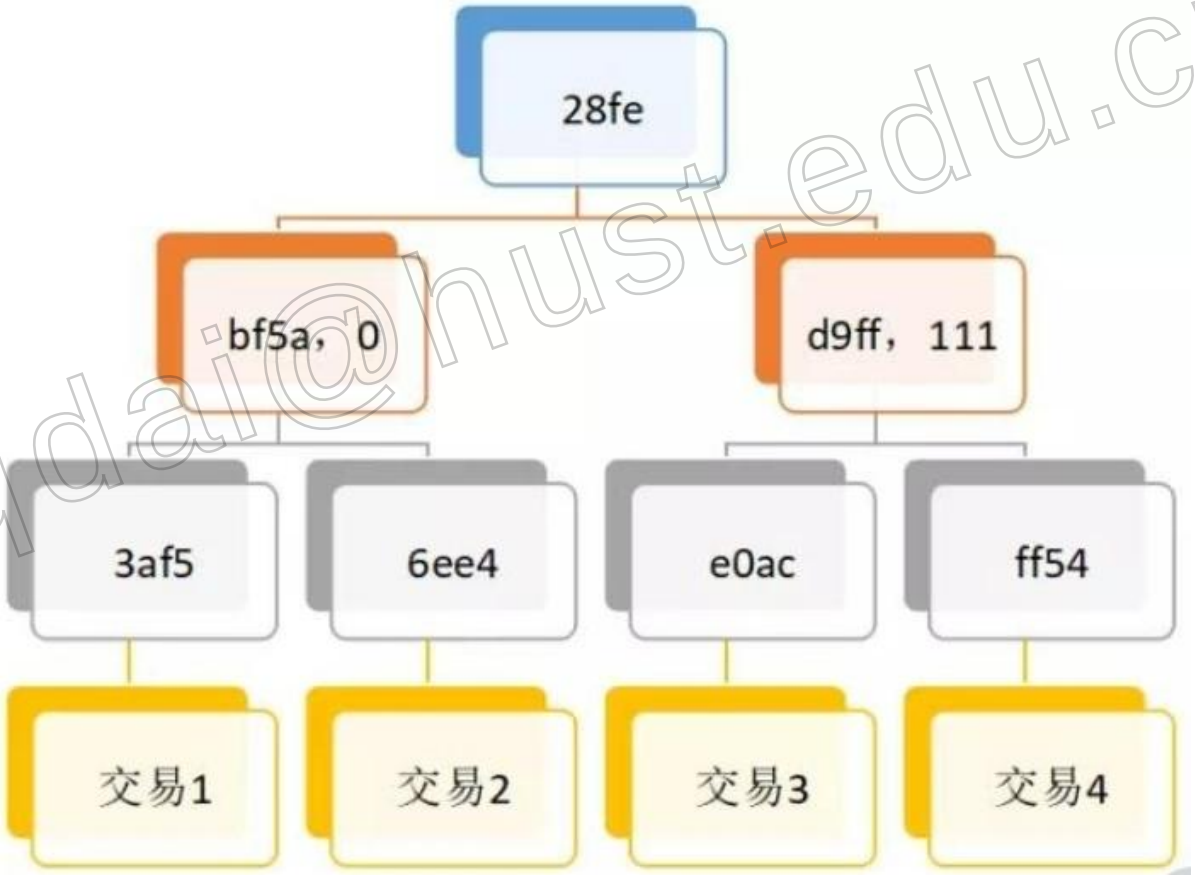
默克尔树（Merkle Tree）

在比特币区块链系统中，区块的结构如下图所示：



默克尔树（Merkle Tree）

每个区块中的Hash1就是本区块中所有交易的哈希值。但这个哈希值不是把所有交易连成一个长字符串后计算HASH值，而是使用了默克尔树（Merkle Tree）算法来计算获得这个HASH值，我们称之为Merkle根。



默克尔树（Merkle Tree）

默克尔树算法并不是直接计算整个字符串的Hash值，而是每个交易都计算一个Hash值，然后两两连接再次计算Hash，一直到最顶层的Merkle根。

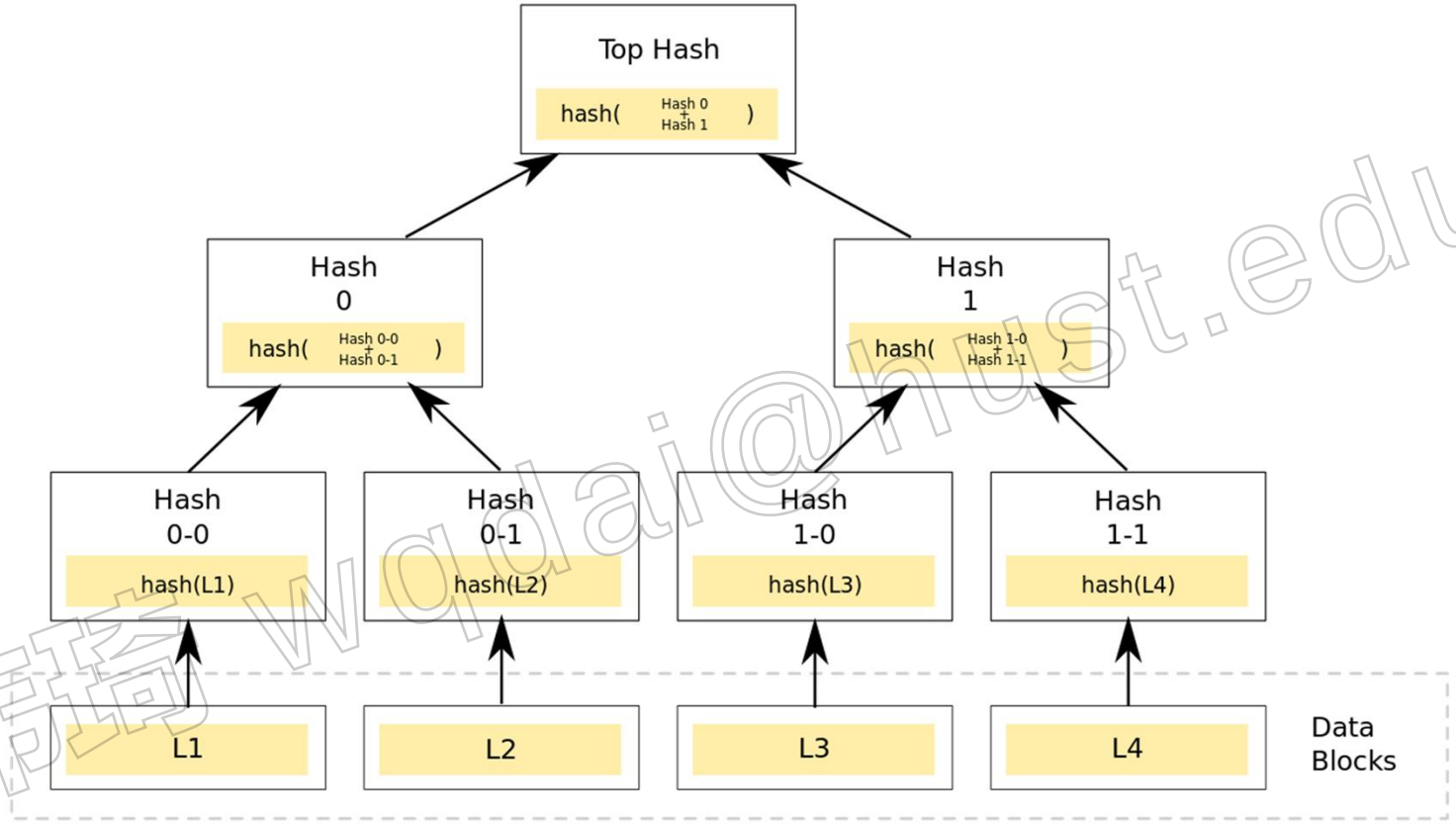
默克尔树（Merkle Tree）算法的最大好处就是，***每个交易都可以单独直接删除，只保留这个交易的Hash值即可***。这样，对整个区块来说，并没有改变他的密码学安全性和完整性，但是数据量可以大大减小。

因此，在UTXO的记账模式中，使用默克尔树结构，通常就无需担心数据量一直增长导致数据过大的问题了。

什么是Merkle树?

简单的说就是支付验证只需验证该笔交易是否被确认过了，而交易验证是需要验证该笔交易是否满足一些条件如“余额”是否足够，还有该笔交易有没有存在双花等等一些问题，只有一切都没什么问题后该笔交易才算验证通过，可以看出交易验证要比支付验证更加复杂，所以它一般是由挖矿节点来完成的，而支付验证只要普通的轻钱包就可以完成。

Merkle Tree



Merkle Tree，通常也被称作Hash Tree，顾名思义，就是存储hash值的一棵树。Merkle树的叶子是数据块(例如，文件或者文件的集合)的hash值。非叶节点是其对应子节点串联字符串的hash。

Merkle树的特点

- 首先是它的树的结构，默克尔树常见的结构是二叉树，但它也可以是多叉树，它具有树结构的全部特点。
- 默克尔树的基础数据不是固定的，想存什么数据由你说了算，因为它只要数据经过哈希运算得到的hash值。
- 默克尔树是从下往上逐层计算的，就是说每个中间节点是根据相邻的两个叶子节点组合计算得出的，而根节点是根据两个中间节点组合计算得出的，所以叶子节点是基础。

Merkle树的特点

可以知道，一颗Merkle树有如下特点：

1. 叶子结点的值是实际数据块的Hash值。

2. 每个非叶子结点的值，都是孩子结点的Hash值。根结点称为Merkle根

3. 如果树是二叉树的话，称为二叉Merkle树，且二叉Merkle树一定是满二叉树（奇数叶子凑成偶数个）

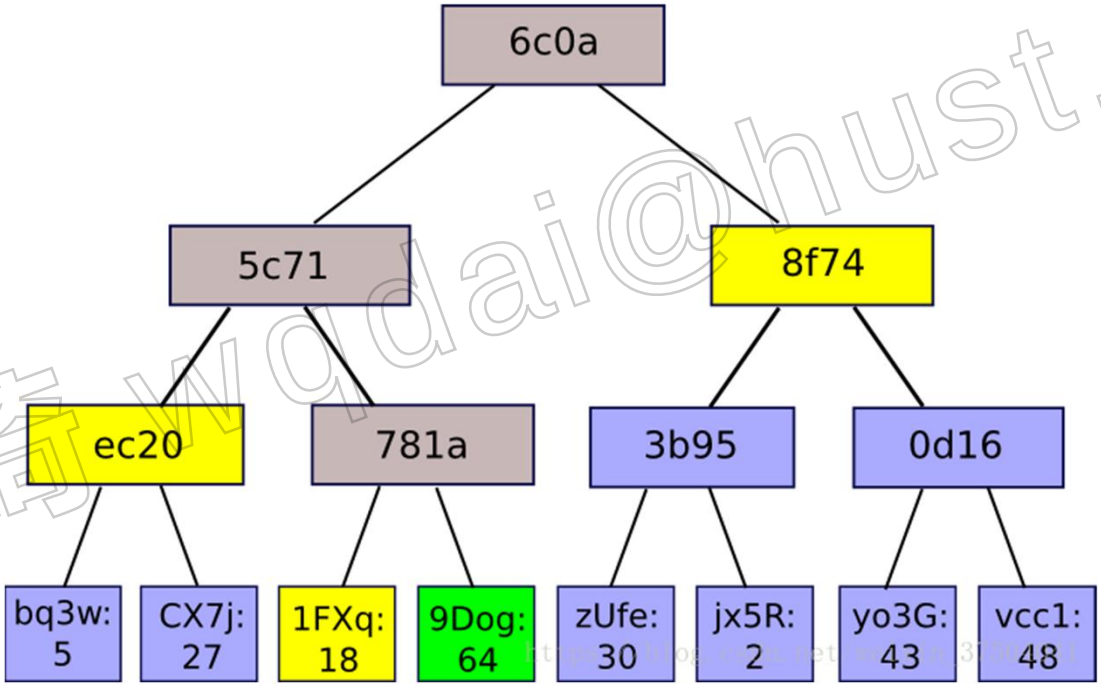
在比特币网络中，Merkle树被用来归纳一个区块中的所有交易，**同时生成整个交易集合的数字指纹**，且提供了一种校验区块是否存在某交易的高效途径。

如何通过Merkle树验证一笔交易?

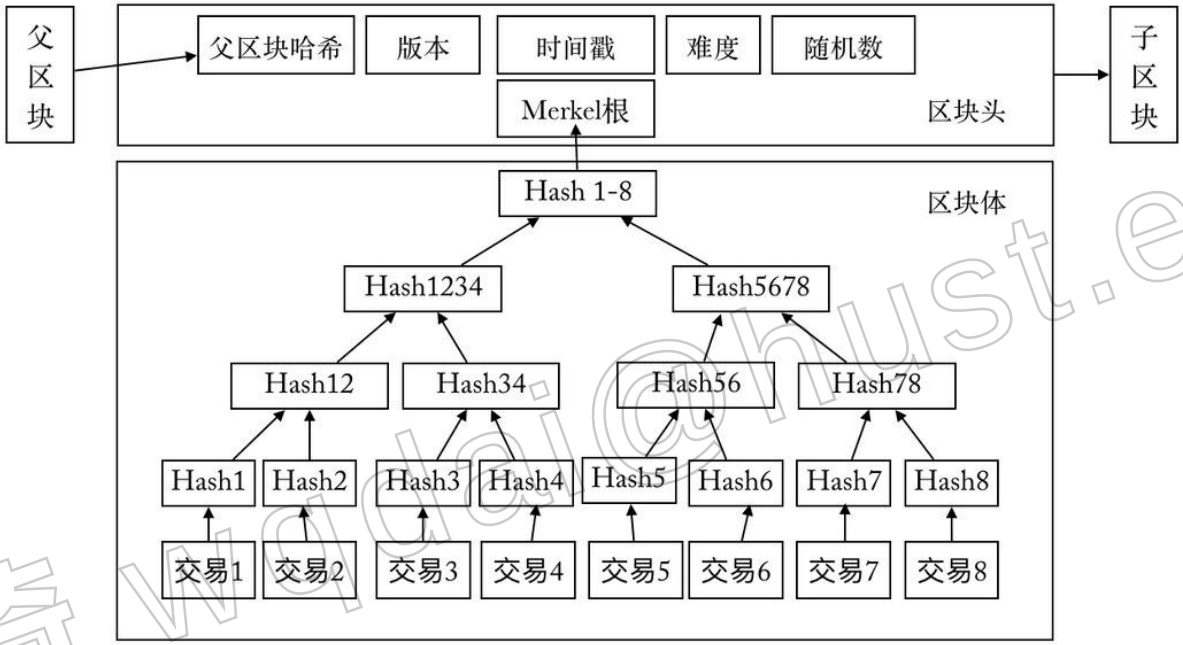
大概了解了什么是默克尔树后，可能会有一个疑问，就是默克尔树是如何验证一笔交易的？也就是SPV（支付验证）。

如何通过Merkle树验证一笔交易?

依据这两点就可以来验证一笔交易是否存在
（1）一个是默克尔树是从下往上逐层计算的
（2）根节点的hash值可以准确的作为一组交易的唯一摘要.



比特币中的Merkle树



【注】比特网络中的Merkle树是二叉树，所以它需要偶数个叶子节点。如果仅有奇数个交易需要归纳，那最后的交易就会被复制一份以构成偶数个叶子节点，这种偶数个叶子节点的树也被称为平衡树。

Merkle树的应用场景

默克尔树的应用场景其实很广泛，比较典型的就是P2P下载。

在点对点网络中作数据传输的时候，会同时从多个机器上下载数据，而且很多机器可以认为是不稳定或者不可信的。为了校验数据的完整性，更好的办法是把大的文件分割成小的数据块（例如，把分割成2K为单位的数据块）。这样的好处是，如果小块数据在传输过程中损坏了，那么只要重新下载这一快数据就行了，不用重新下载整个文件。

比特币挖矿

代炜琦

wqdai@hust.edu.cn
华中科技大学
Huazhong University of Science and Technology

比特币的挖矿

如何优雅地获得你人生中第一枚比特币？

挖矿对于比特币价格和走势的影响

比特币的挖矿

“挖矿”成功即是该节点成功获得当前区块记账权，也就是说其他节点就“照抄”该挖矿成功的节点的当前区块。获得记账权的节点会获取一定数量的比特币奖励，以此激励比特币网络中的所有积极参与记账工作。该奖励包含系统奖励和交易手续费两部分，系统奖励则作为比特币发行的手段。

比特币的挖矿

序号	币种	次数	减半时间	挖矿奖励
1	BTC	第一次减半	2012-11-28	50→25
		第二次减半	2016-7-10	25→12.5
		第三次减半	2020-5-7	12.5→6.25
2	BCH	第一次减半	2020-4-8	12.5→6.25
3	BSV	第一次减半	2020-4-8	12.5→6.25
4	ETC	第一次减半	2020-3-22	4→3.2
5	DASH	第一次减半	2020-4-27	3.45→3.20

币种	市值(USD)	共识机制	预计减半时间	第几次减半	当前区块挖矿奖励	减半后挖矿奖励	出块时间	每天产量
BEAM	2700万	PoW	2020.01.04	第一次	100BEAM	50BEAM	1min	144000
ETC	5.1亿	PoW	2020.03.05	第一次	4ETC	3.2ETC（减少20%，并非减半）	13.2s	24000
BCH	37亿	PoW	2020.04	第一次	12.5BCH	6.25BCH	10min	1800
BSV	17亿	PoW	2020.04	第一次	12.5BSV	6.25BSV	10min	1800
BTC	1300亿	PoW	2020.05.13	第三次	12.5BTC	6.25BTC	10min	1800
DASH	3.8亿	PoW+PoS	2020.05	第一次	3.6DASH	3.34DASH（减少7.14%，并非减半）	2.5min	550
XZC（Zcoin）	2700万	PoW+PoS	2020.09	第一次	12.5XZC	6.25XZC	10min	72000
ZEC（Zcash）	2.3亿	PoW	2020.10	第一次	12.5ZEC	6.25ZEC	8…	57…

最初每生产一个“交易记录区块”可以获得50比特币的系统奖励，为控制比特币发行数量，该奖励每4年就会减半，到2140年会基本发放完毕，最终整个系统中最多的只能有2100万个比特币。如果按照目前比特币的价格计算，比特币的总价值将在2000亿美金左右。

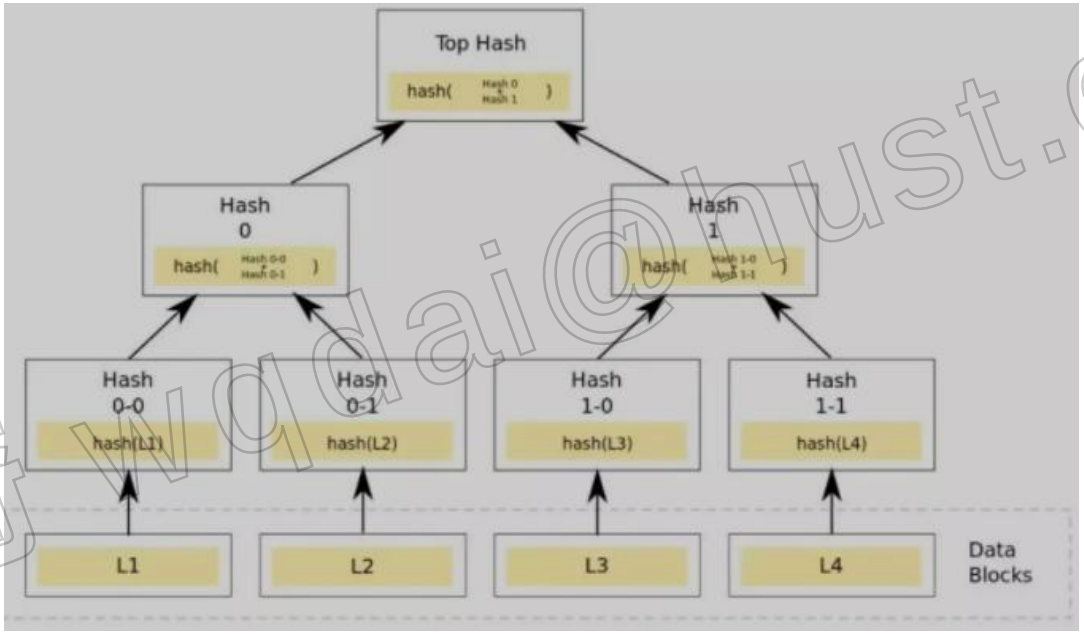
到底啥叫“挖矿”

比特币系统大约每10分钟会记录一个数据块，这个数据块里包含了这10分钟内全网待确认的部分或全部交易。所谓的“挖矿”，就是争夺将这些交易打包成“交易记录区块”的权利。比特币系统会随机生成一道数学难题，后续会详细描述该数学难题，所有参与挖矿的节点一起参与计算这道数学难题。首先算出结果的节点奖获得记账权。

每个节点会将过去一段时间内发生的、尚未经过网络公认的交易信息进行收集、检验、确认，最后打包并加签名为一个无法被篡改的“交易记录区块”，并在获得记账权后将该区块进行广播，从而让这个区块被全部节点认可，让区块中的交易成为比特币网络上公认已经完成的交易记录，永久保存。

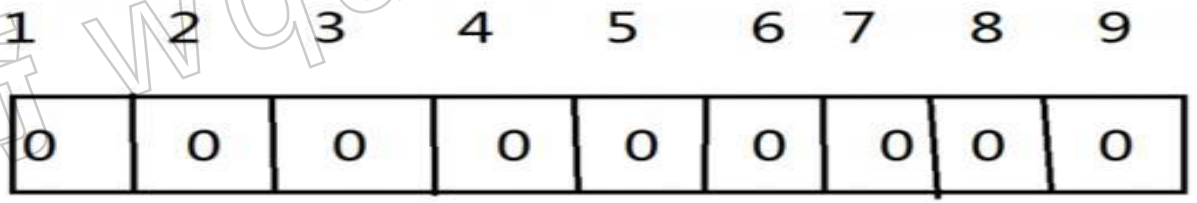
挖矿最主要的工作就是计算上文提到的数学难题，最先求出解的矿工即可获得该块的记账权。在介绍这个数学难题前，先简单介绍一下哈希算法。哈希算法的基本功能概括来说，就是把任意长度的输入值通过一定的计算，生成一个固定长度的字符串，输出的字符串即为该输入的哈希值。比特币系统中采用SHA-256算法，该算法最终输出的哈希值长度为256bit。

挖矿的原理-Merkle树



挖矿的原理–计算随机数

比特币中每个区块生成时，需要把上一个区块的哈希值、本区块的交易信息的默克尔树根、一个未知的随机数（nonce)拼在一起计算一个新的哈希值。为了保证10分钟产生一个区块，该工作必须具有一定难度，即哈希值必须以若干个0开头。哈希算法中，输入信息的任何微小改动即可引起哈希值的巨大变动，且这个变动不具有规律性。因为哈希值的位数是有限的，通过不断尝试随机数nonce，总可以计算出一个符合要求的哈希值，且该随机数无法通过寻找规律计算出来。这意味着，该随机数只能通过暴力枚举的方式获得。**挖矿中计算数学难题即为寻找该随机数的过程。**



挖矿的原理-计算随机数

十六进制	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F		
十进制	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15		

哈希值由16进制数字表示，即每一位有16种可能。根据哈希算法的特性，出现任何一个数字的概率是均等的，即每一位为“0”的概率为1/16.要求某一位为“0”平均需要16次哈希运算，要求前n位为“0”，则需要进行哈希计算的平均次数为16的n次方。矿工为了计算出该随机数，需要花费一定的时间进行大量的哈希运算。某个矿工成功计算出该随机数后，则会进行区块打包并全网广播。

其他节点收到广播后，只需对包含随机数的区块按照同样的方法进行一次哈希运算即可，若哈希值以“0”开头的个数满足要求，且通过其他合法性校验，则接受这个区块，停止本地对当前区块随机数的寻找，开始下个区块随机数的计算。

挖矿的原理-计算随机数

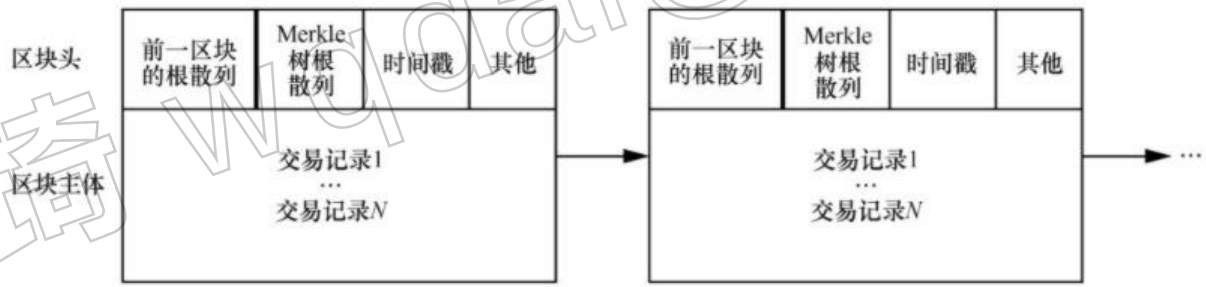
随着技术的发展，进行一次哈希计算速度越来越快，同时随着矿工的逐渐增多，算出满足哈希值以一定数量“0”开头的随机数的时间越来越短。为保证比特币始终按照平均每10分钟一个区块的速度出块，必须不断调整计算出随机哈希计算的平均次数，即调整哈希值以“0”开头的数量要求，以此调整难度。比特币中，每生成2016个区块就会调整一次难度，即调整周期大约为两周（2016x10min=14天）。也就是说，对比生成最新2016个区块花费的实际时间和按照每10分钟出一个块生成2016个块的期望时间，若实际时间大于期望时间则降低难度，若实际时间小于期望时间则增加难度。

同时，为防止难度变化波动太大，每个周期调整幅度必须小于一个因子（当前为4倍）。若幅度大于4倍，则按照4倍调整。由于按照该幅度调整，出块速度仍然不满足预期，因此会在下一个周期继续调整。

挖矿算法

上述的整个过程难就难在第四步的“链接”上面，需要大量计算，才可能成功的把你的新区快链接到整个区块链的末尾，下面就来详细讲解这个“链接”的算法。

Block的结构：区块头（Header） + 区块体（Body）。Header里面包含了上1个区块的Hash值，还包含1个随机属性nonce，Body里面包含了很多笔交易。



挖矿算法

我们要计算本区快的hash值，计算公式如下：

本区块Hash值 = SHA256D(上1个区块的Hash值 + 交易数据 + nonce)

其中SHA256D，就是计算2次SHA256

Hash值是256位的，所谓的挖矿成功，就是当算出来的Hash值符合此规则：

前48位刚好是oxOOOOOOOOFFFF

可见这个概率是非常小的。

挖矿算法

上1个区块的Hash值是固定的，交易数据也是固定的，唯独变化的就是这个nonce属性。

nonce属性是32位的，我们需要暴力破解：遍历该nonce属性0 - 2的32次方的每个值，逐个计算2次SHA256（SHA256D），直到计算出来的hash值符合某个规则---前48位为0x000000000FFFF

一旦算出了这个值，就可以把这个区块链接到整个区块链的末尾，然后广播出去。

别的节点收到这个最新的块和对应的Hash值，就会暂时中断当前的挖矿进度。把这个块重新校验一遍，发现是符合规则的，就会认可这个区块，把这个区块加到他们的链的末尾。然后再顺着这个区块，挖一1个区块。

所以整个网络的挖矿过程，就是每个节点不断的算Hash值、不断的从别的节点同步最新的块的过程。

区块链的不可篡改性

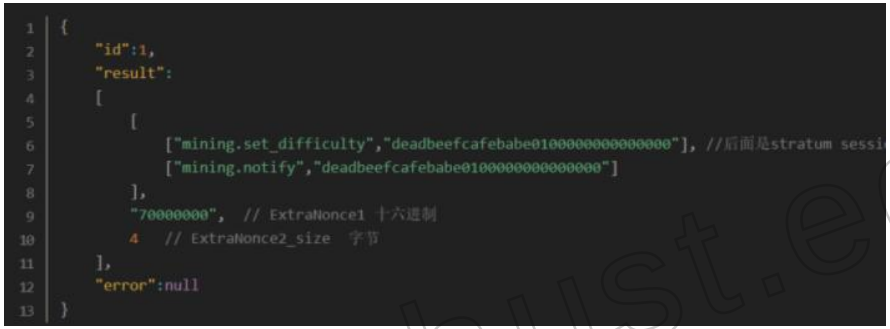
从上面的Hash的计算过程中，我们也会发现区块链的1个关键特性：数据不可篡改。

比如你改了第100个区块的交易数据，那第100个区块的Hash值也要跟着改； 导致第101个区块的Hash值也要跟着改；第102个区块的Hash值也要跟着改；意味着要把100个区块以后的整个链的数据全部篡改。

在篡改的过程中，别的节点还在一直挖新的区块，你要让所有节点都认可你篡改的数据，整个过程几乎是不可能的。

挖矿的流程

1 开始



2 客户端首先向服务器发送subscribe指令

{"id": 1, "method": "mining.subscribe", "params": ["cpuminer/2.5.0"]}
参数中指名矿工软件的名称和版本号。

3 服务器端返回信息

比特币挖矿实际上就是去寻找随机数nonce，有时所有的随机数都试遍了，仍无法满足目标，就需要用到extra nonce。

挖矿的流程

4 **客户端发送认证信息**

用户名是钱包的地址，我这里使用的是比特币测试网络的地址，并没有以1开头。

```
{
"id": 2,
"method": "mining.authorize",
"params": ["myAzQj4bH4mMF2GpoLSY2v4qVquASTpzR4", "x"]
}
```

5 **服务器返回true，表示用户验证通过**

{"id":2,"result":true,"error":null}

6 **服务器端发回难度设置消息**

{"id":null,"method":"mining.set_difficulty","params":[8]}

挖矿的流程

7 **服务器发送通知消息**

矿工可能用到了多线程，所以需要job id来区分不用的线程，后面是一堆用于区块生成的信息。

8 **客户端发现一个nonce，提交给服务器**

9 **服务器返回结果**

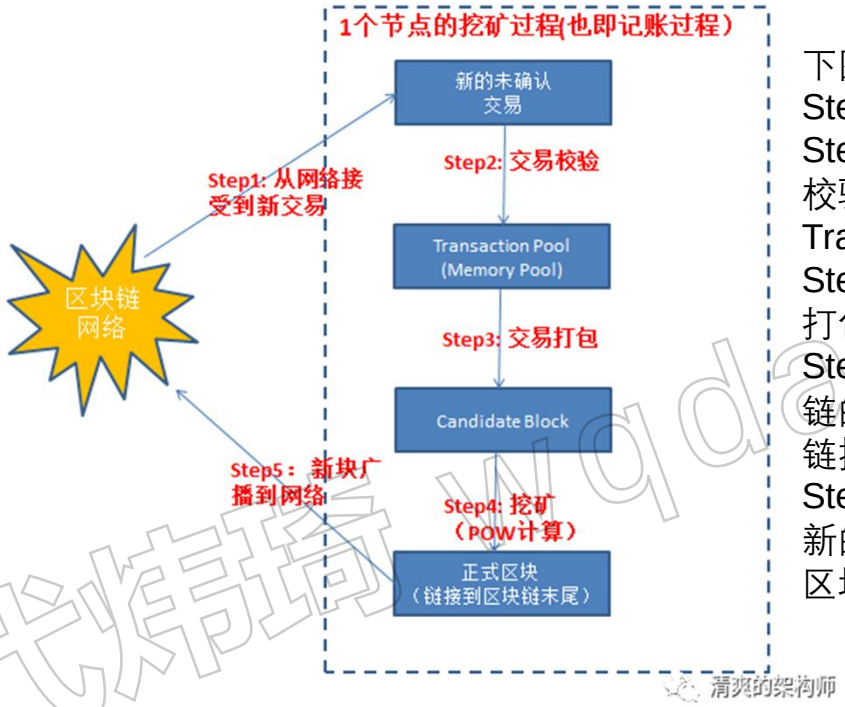
返回true表示服务器认可客户端的工作。

```
{
"method":"mining.notify",
"params":[
    "3d",  // 作业ID
    "af8cecebf98...43100000000", //previous block hash
    "010000000100000....0000000", //generation tx part1
    "0d2f6e6f646.....5b36ec88ac00000000", //generation tx part2
    ["6fb6....80cb","d4464...49a5a290", ... ,"68d803...4ec79d161"], // merkle branches 默克尔树分
    "20000000", // block version
    "1a3fffc0", // nBits encoded network difficulty
```

```
{
"method": "mining.submit",
"params": [
    "myAzQj4bH4mMF2GpoLSY2v4qVquA5TpzR4", //worker name
    "4c", //job id
    "00000000", //extranonce2
    "5a158557", //nTime
    "6ae65681"  //nonce
],
"id":4
}
```

```
{
"id":4,
"result":true,
"error":null
}
```

挖矿的流程



下图展示了1个节点挖矿的详细流程：
Step1: 节点会从区块链网络上源源不断的接受到新的未确认的交易
Step2: 对每笔交易进行校验（安全校验、双花校验等）
校验不合格的会丢弃，校验合格的会放入本机的内存池，又称作Transaction Pool。
Step3: 从这个Memory Pool里面，拿出1批交易（大概两三千个），打包放进一个新的候选块里。
Step4: 关键环节，挖矿。把这个Candidate Block链接到整个区块链的末尾。链接成功，就算是挖出了1个新区块，奖励你50比特币；链接不成功，一直尝试，挖。。
Step5: 把这个新挖出来的区块，广播到网络上。其他节点看到这个新的块，就会终止各自当前的挖矿进度，把这个区块链接到自己的区块链上，接着在这个新的区块后面挖（也就是重复Step4）。

矿机，矿池与算力

代炜琦

wqdai@hust.edu.cn
华中科技大学
Huazhong University of Science and Technology

矿机

最初呢，大家就是普通的电脑，CPU挖矿。但就像淘金一样，巨大的利益，驱动着矿机经过了好几代的巨大发展。

回顾挖矿历史，比特币挖矿总共经历了以下五个时代：CPU挖矿→GPU挖矿→FPGA挖矿→ASIC挖矿→大规模集群挖矿

挖矿芯片更新换代的同时，带来的挖矿速度的变化是：

CPU（20MHash/s）→GPU（400MHash/s）→FPGA（25GHash/s）→ASIC（3.5THash/s）→大规模集群挖矿（3.5THash/s*X）

矿池的原理

随着区块链的日渐火爆，参与挖矿的人越来越多，按照比特币原本的设计模式，只有成功打包一个区块的人才能获取奖励。如果每个矿工都独立挖矿，在如此庞大的基数下，挖矿成功的概率几乎为0，只有一个幸运儿可以获取一大笔财富，其他矿工投入的算力、电力资源就会白白亏损。或许投入一台矿机，持续挖矿好几年甚至更久才能挖到一个区块。

矿池的原理

为了降低这种不确定性，矿池应运而生。假如有10万矿工参与挖矿工作，这10万矿工的算力和占这个网络的10%，则这10万个矿工中的某个矿工成功挖到下个块的概率即为1/10。

即平均每个矿工成功挖到下个区块的概率为1/1000000，即平均每个矿工要花费19年可以成功挖到一个区块，然后获得相应的比特币奖励。

这种挖矿模式风险过大，几乎没人可以承受。但是假设这10万个矿工共同协作参与挖矿，则平均每100分钟即可成功挖到一个区块，然后按照每个矿工提供的算力分配该次收益。这10万个矿工的收益也会趋于稳定。

矿池的原理

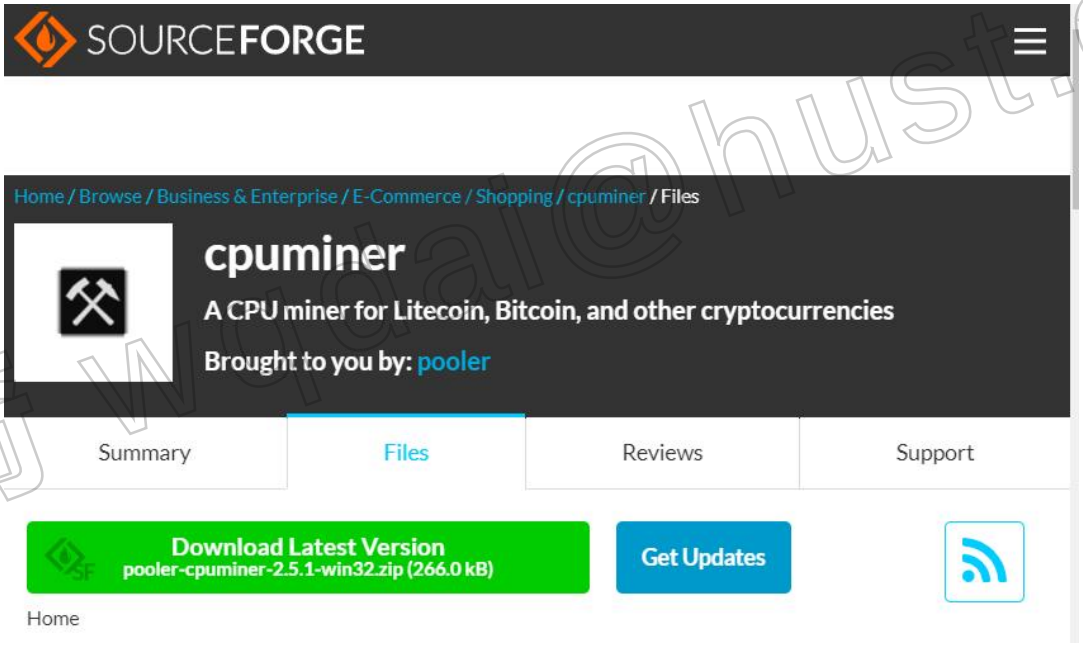
协调矿工进行计算的思路也非常简单，矿池将打包区块需要的交易等信息验证完成后发送给矿工，然后降低矿工的挖矿难度。比如某个时段比特币系统需要哈希值“ 0 ”开头的个数大于50个，矿池可以将难度降低到40个“ 0 ”开头，矿工找到一个40个“ 0 ”开头哈希值的方案后，即可提交给矿池。矿池收到一个满足哈希值“ 0 ”开头个数大于50个的方案时，即可提交至比特币网络。

当然，你也许会想：如果矿工计算得到一个“ 0 ”开头个数大于50的哈希值后，则直接提交给比特币网络，独享该区块的收益；

如果计算得到一个“ 0 ”开头数在40到50之间的则提交到矿池，享受整个矿池分配的收益。该方案当然是行不通的，因为区块内容是由矿池发送给矿工的，即受益者地址已经包含在该区块中了，即使直接提交，最终受益的也是矿池。如果修改该地址，即意味着区块内容改变，则前面计算的哈希值也无效了。最后矿池按照矿工提交方案数量计算贡献的算力，最后根据算力分配收益。

矿池的原理

当前的主流挖矿协议是stratum,以前还有GBT(*getblocktemplate*)、getwork等几种协议，它们都过时了。可以用免费的Cpuminer软件把协议调通。



算力

算力是衡量在一定的网络消耗下生成新块的单位的总计算能力。每个硬币的单个区块链随生成新的交易块所需的时间而变化。

算力： 一个挖矿机每秒钟能做多少次hash碰撞，就是其“算力”的代表，单位写成hash/s。

算的越快，在和其他矿工的竞争中越具有优势，所以所有的矿工都在拼命的提高自己的算力。

现在整个比特币网络的所有算力的加总，已经进入P时代。什么意思呢？

1P = 1024T;
1T = 1024 G;
1G = 1024M;

也就是说，整个网络每秒能做的Hash运算的总次数，是1024 * 1024 * 1024 * 1024 * 1024 这个量级。

算力的升级-BCT和BCH

在2013年之前BCT和BCH的限制对系统完全没有影响，因为没有那么多人使用比特币。但随着比特币交易的火热，越来越来多的人开始投身于此，问题也就随之而来了。每秒7笔交易的效率实在太低了，所以通过区块大小升级来提升交易速度的问题提上了日程。

直到2017年8月1日，比特币发生分差，在一个“硬叉”的事件中，诞生了一种被称为比特币现金（BCH）的新数字货币。由于版本切换，比特币区块链被分叉至两条独立的区块链。在分叉前拥有比特币的所有人都有权获得相同数量的“比特币现金”代币，类似于股票中的股息派发。

自此比特币就被分叉为BTC和BCH两种货币，当然它们也代表着两种方向。

算力的升级-BCT和BCH

- 比特币（BTC）是目前世界上最受追捧的数字货币，2017年8月1日发生分差，在一个“硬叉”的事件中，诞生了一种被称为比特币现金（BCH）的新数字货币。由于版本切换，比特币区块链被分叉至两条独立的区块链。在分叉前拥有比特币的所有人都有权获得相同数量的“比特币现金”代币，类似于股票中的股息派发。

- 比特币现金（BCH）是由一小部分比特币开发者推出的不同配置的新版比特币，是一种新型的区块链资产。在2017年8月1日，比特币现金开始挖矿，每个比特币投资者的账户上将出现与比特币数量等量的比特币现金（BCH）。

新生代BCH(比特现金）

比特币现金（BCH）是由一小部分比特币开发者推出的不同配置的新版比特币，是一种新型的区块链资产。

比特币可以说是BCH的前世，但是分叉以后，BCH一直被认为是新方向的比特币，BCH有更大的区块，能够处理更多交易，不会出现像BTC那样由于区块太小，处理速度跟不上交易数量的情况，从而造成拥堵的现象。***这也是BCH与BTC最明细的区别。***

这也直接造成了BCH的手续费用比BTC要低的多，BTC由于缓慢的交易处理速度，出现了谁的手续费价格高谁的交易先处理的规则，而BCH是不可能出现这种情况的。由于BCH社区从理念上与现在BTC社区core团队不一样，简单的说BCH与BTC没有多少关联了，成为一种全新的币种。

新生代BCH(比特现金）

BCH可以算是比特币中的“革新派”，他们认同比特币通过提升区块大小，认同有更多的项目能够落实，应该运用最新的技术在比特币上，但是他们的激进也让他们受到非议，说他们是逐利的矿工，不尊重中本聪的意志，他们就如同历史上的革新派一般，充满激情但又太过激进。

币圈“黄金”BTC（比特币）

比特币由中本聪提出到现在已有9年时间了，可以说它的历史地位毋庸置疑。在区块链市场中，“共识”是最为重要的，作为最早的货币，比特币所获得的“共识”理所当然是最多的，所以BTC就是是加密货币中的黄金，它如同黄金一样是“硬通货”，所以在BTC团队看来强行升级区块，风险太大，颇有”祖宗之法不可动“的意思。

当然这并不是说他们保守，相反他们注意到比特币交易速度缓慢的问题。他们同样希望比特币能够快速发展，而不是被缓慢的交易速度所拖累。现如今BTC也在通过诸如侧链和闪电网络等方法提升自身的交易速度。

币圈“黄金”BTC（比特币）

BTC就是比特币中的“保守派”

他们认同比特币就是比特币、且只有唯一比特币（BTC）的理念，他们认为如此重要的货币应该循序渐进，慢慢完善，而不能“用力过猛”。

不管是“革新派”还是“保守派”，二者虽然理念差异不同，但是对于虚拟货币的发展都是看好的。老一辈的BTC开始完善自身，自立门户的BCH也开始新的征程，他们都希望在未来区块链市场继续发光发热！

哈希函数在区块链中的应用

代炜琦

wqdai@hust.edu.cn
华中科技大学
Huazhong University of Science and Technology

哈希的应用-以太坊用户地址的生成

第一步：生成私钥 (private key)

产生的256比特随机数作为私钥(256比特 16进制32字节)：

18e14a7b 6a307f42 6a94f811 4701e7c8 e774e7f9 a47e2c20 35db29a2 06321725

哈希的应用-以太坊用户地址的生成

第二步：生成公钥 (public key)

利用将私钥(32字节)和椭圆曲线ECDSA-secp256k1计算公钥(65字节)(前缀04||X公钥||Y公钥)：

04 ||50863ad6 4a87ae8a 2fe83c1a f1a8403c b53f53e4 86d8511d ad8a0488 7e5b2352 || 2cd47024 3453a299 fa9e7723 7716103a bc11a1df 38855ed6 f2ee187e 9c582ba6

利用Keccak-256算法计算公钥的哈希值(32bytes)：

fc12ad81 4631ba68 9f7abe67 1016f75c 54c607f0 82ae6b08 81fac0ab eda21781

取上一步结果取后20bytes即以太坊地址：

1016f75c54c607f082ae6b0881fac0abeda21781

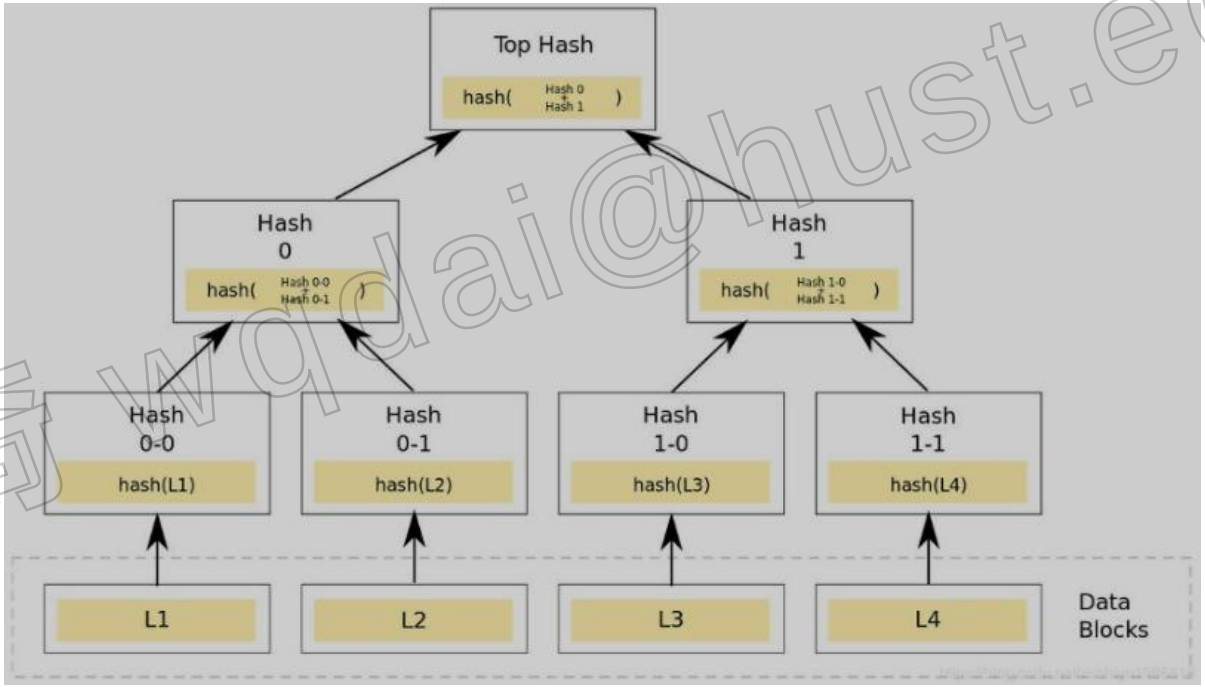
哈希的应用-以太坊用户地址的生成

第三步：输地址 (address)

0x1016f75c54c607f082ae6b0881fac0abeda21781

哈希的应用-默克尔哈希树

比特币区块包含了区块头部和一些比特币交易。一个区块上所有交易的哈希值构成了该区块Merkle哈希树的叶子节点，Merkle哈希树的根节点保存在区块头里面，因此所有交易与区块头部绑定在了一起。



哈希的应用-挖矿难度的设置

比特币难度是对挖矿困难程度的度量，即指：**计算符合给定目标的一个哈希值的困难程度。**

difficulty = difficulty_1_target / current_target

difficulty_1_target的长度为256比特， 前32位为0，后面全部为1 ， 一般显示为哈希值，

0x00000000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

difficulty_1_target表示btc网络最初的目标哈希。current_target是当前块的目标哈希，先经过压缩然后存储在区块中，区块的哈希值必须小于给定的目标哈希值, 表示挖矿成功。

哈希的应用-数字签名

比特币需要利用公钥进行加锁，利用私钥签名进行解锁，从而实现数字货币的交易。解锁过程实际上是利用ECDSA算法的产生数字签名。给定交易信息m，签名过程如下：

•选择一个随机数k

•计算点$R = k*G = (x_R, y_R)$，计算$r = x_R \bmod n$

•利用私钥d计算$s = k^{-1}*((\boldsymbol{H(m)} - d*r)) \bmod n$

输入签名(r, s)

哈希的应用-数字签名

问题的提出

- **手写签名**:传统的确认方式，如书信、签约、支付、批复等
- 在网络时代，人们通过网络支付费用、买卖股票，为了保证网上商务活动的安全，需要一个很重要的安全机制——**数字签名**

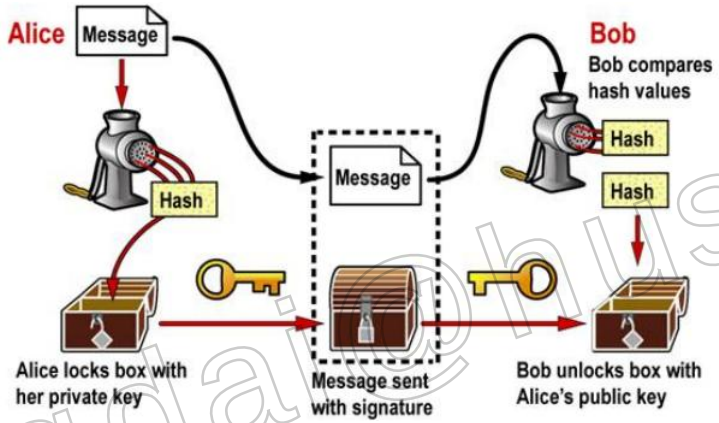
图2.3. 传统合同示意图

哈希的应用-数字签名

数字签名是手写签名数字化的产物，但又有着显著区别

- 不同消息的签名不同，甚至相同的消息也有不同的签名，否则签名就会被获取并复制到另外的文件中；
- 数字签名的基础是公钥密码学；

哈希的应用-数字签名



数字签名方案一般包括三个过程：

- 系统初始化过程：产生数字签名方案中的所有系统和用户参数（公开的+秘密的）
- 签名过程：用户利用给定的签名算法对消息签名，签名过程可以公开也可以不公开，但一定包含仅签名者才能拥有的秘密信息（签名密钥）
- 验证过程：验证者利用公开的验证方法对给定消息的签名进行验证

数字签名的作用有哪些?

•**防篡改**：通过对数字签名的验证，可以保证信息在传输过程中未被篡改。

•**验证数据的完整性**：与防篡改同理，如果信息发生丢失，签名将不完整，解开数字签名和之前的比较就会出现不一致，因而可保证文件的完整。

•**仲裁机制**：数字签名也可以认为是一个数字身份，通过唯一私钥生成，在网络上交易时要求收到一个数字签名的回文，保证过程的完整。如果对交易过程出现抵赖，那么用数字便于仲裁。

•**保密性**：对于全级别要求较高的数据，数字签名加密后传输，保证数据在被中途截取后无法获得其真实内容；有利于保证数据的安全性。

•**防重放**：在数字签名中，如果采用了对签名报文添加流水号、时戳等技术，可以有效防止重放攻击。

哈希的应用-数字签名

经典的数字签名算法

- RSA数字签名
- DSA数字签名
- ECDSA数字签名
- ＳＭ２数字签名
- .....

其他数字签名算法

- 盲签名
- 群签名
- 环签名
- .....

钱包、非对称加密与交易

代炜琦

wqdai@hust.edu.cn
华中科技大学
Huazhong University of Science and Technology

钱包地址及私钥

通过一个对比来理解：

- 地址 = 银行卡卡号
- 密码 = 银行卡密码
- keystore = 银行卡
- 助记词 = 私钥 = 银行卡 + 银行卡密码
- Keystore + 密码 = 银行卡 + 银行卡密码

什么是钱包地址

钱包地址是我们最常见的，它们是一串数字和字母的组合，看起来有点像乱码。钱包地址就像银行卡号，代表了你的比特币账户。通过交易所、比特币客户端和在线钱包都可以获得钱包地址。

- **钱包地址可以看成是银行卡账号**
- **不会重复**
- **不会反推出私钥**

什么是私钥/助记词

•助记词 = 私钥 = 银行卡 + 密码
•助记词 = 用人类语言描述的私钥

这两者都是最高权限，任何人拿到，就可以直接转走你钱包里的一切财产，就像拿着你的银行卡和密码直接去atm取钱一样。

什么是私钥/助记词

举个形象的例子：

私钥：x12Nedx3edsrEdfh

助记词：love can play games tomorrow money

私钥完全没有意义，而助记词则是由一个个单词组成，虽然意义也不大，但记忆难度小了很多，且不容易错。

什么是KEYSTORE/钱包密码

因为对于数字货币的钱包来说，地址和keystore职责分开了，地址只负责转账，keystore则负责安全。

因此，只有keystore需要配合密码使用，而助记词和私钥都不需要密码，这是和银行卡在安全性方面极大不同。

- Keystore = 银行卡
- keystore = 加密的私钥
- keystore+密码 = 银行卡+银行卡密码 = 私钥/助记词

什么是KEYSTORE/钱包密码

从技术上说，keystore是一段结构化的内容，里面包含了非常多的信息，例如地址、密码、id、编码、加密方式等等，和银行卡比较类似。拿着银行卡，没有密码也是用处不大的。

关于信息泄露

•地址泄露，完全无影响。

•钱包密码泄露，只要手机不丢失，无影响。（keystore存储在手机上）

•地址+密码，只要手机不丢，无影响。（keystore存储在手机上）

•keystore泄露，只要密码不丢，无影响。（密码存储在用户脑子里）

•keystore+**密码泄露**，赶快把资产转到其他钱包，并重新生成keystore+密码，原先的账户不再使用。（钱包地址还可以使用）

•**助记词、私钥泄露**，同上面，转移资产，并不再使用泄露的钱包。（钱包地址还可以使用）

什么是非对称加密技术?

对称加密算法在加密和解密时使用的是同一个秘钥;

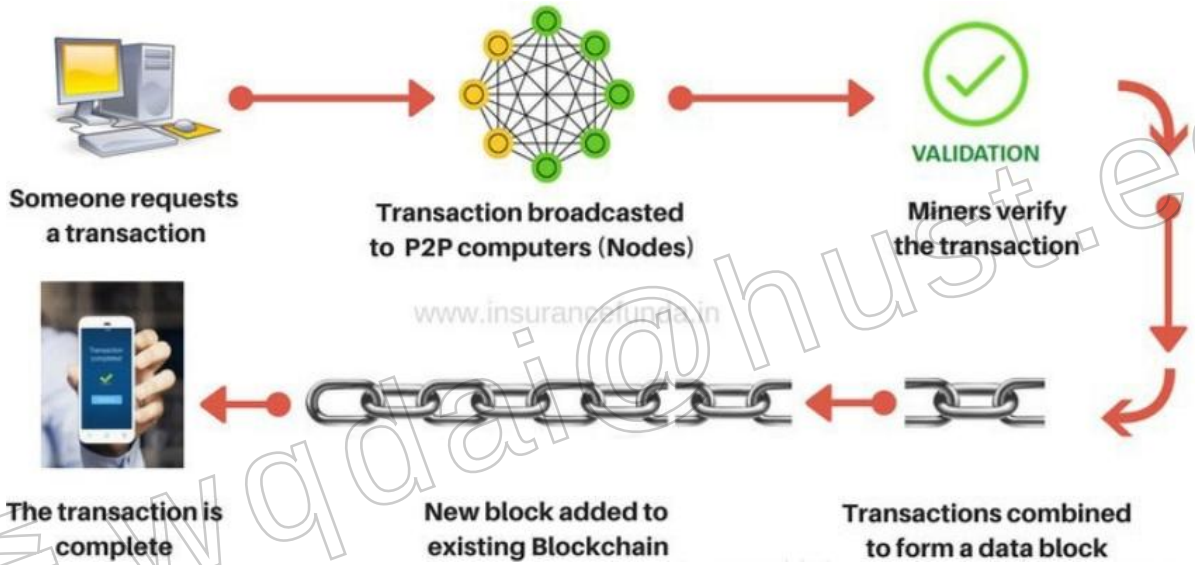
而非对称加密算法需要两个密钥来进行加密和解密，这两个秘钥是公开密钥（public key，简称公钥）和私有密钥（private key，简称私钥）。

什么是非对称加密技术?

公钥与私钥是一对，如果用公钥对数据进行加密，只有用对应的私钥才能解密；如果用私钥对数据进行加密，那么只有用对应的公钥才能解密，因为加密和解密使用的是两个不同的密钥，所以这种算法叫作非对称加密算法。

在比特币转账的过程中数字签名加密的私钥和解密的公钥就是不一致。

区块链交易



一个Transaction的生命周期简化为6个步骤：

1.某人发出交易请求
2.广播交易请求到 P2P 网络
3.验证，miners验证交易正确性
4.多个交易组成一个区块
5.新的区块加入到一个已经存在区块链中
6.交易完成

区块链交易过程

要实现交易首先提出几个问题：

1.如何证明你有钱?
2.如何证明你的身份?
3.如何防止你的钱被使用两次?

如何证明你有钱?

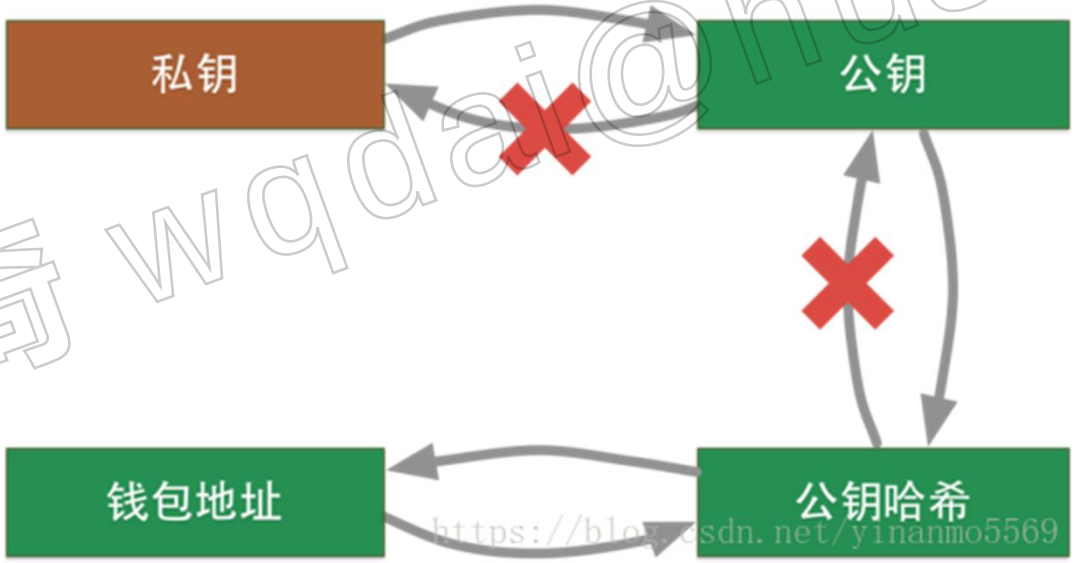
根据过往的交易记录，看是否有 UTXO 的公钥与交易发起人的公钥是一致的 .

如果遍历整个去区块链来判断你是否有钱，那么效率将会及其低下 . 所以一般会维护一个 UTXO**集合** .

如何证明你是你?

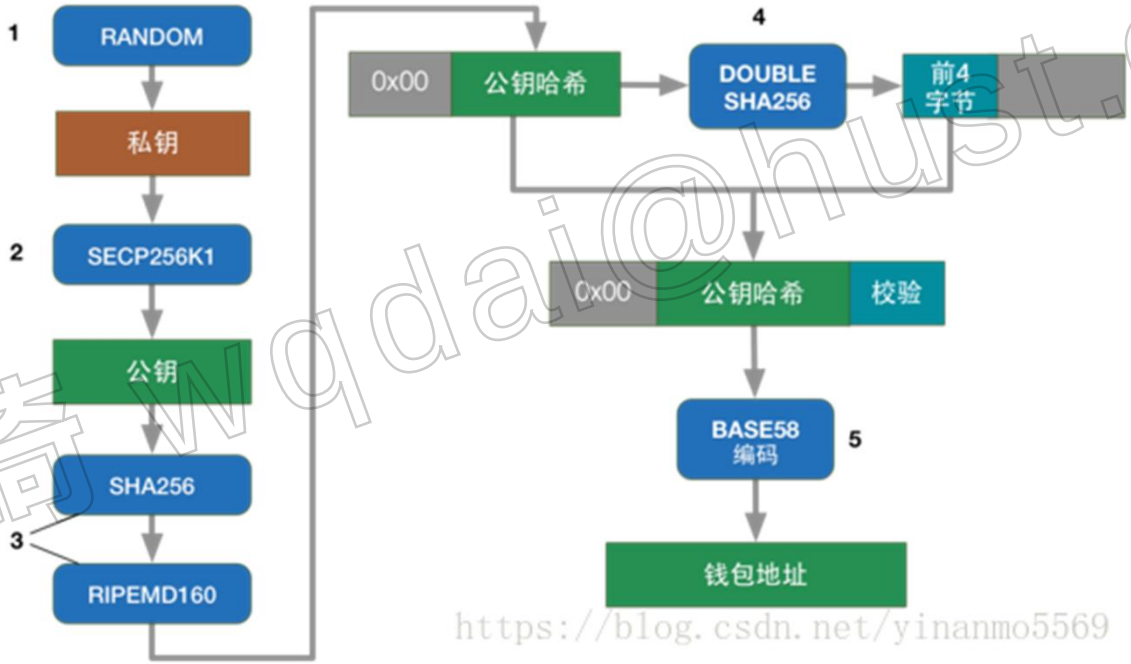
这就涉及前面的密码学问题了．先需要弄清楚**私钥，公钥，钱包地址**之间的关系.

私钥可以生成公钥，公钥可以生成公钥哈希，公钥哈希又可以生成钱包地址．整个过程中除了公钥哈希生成钱包地址是可逆的，其他都是不可逆的．



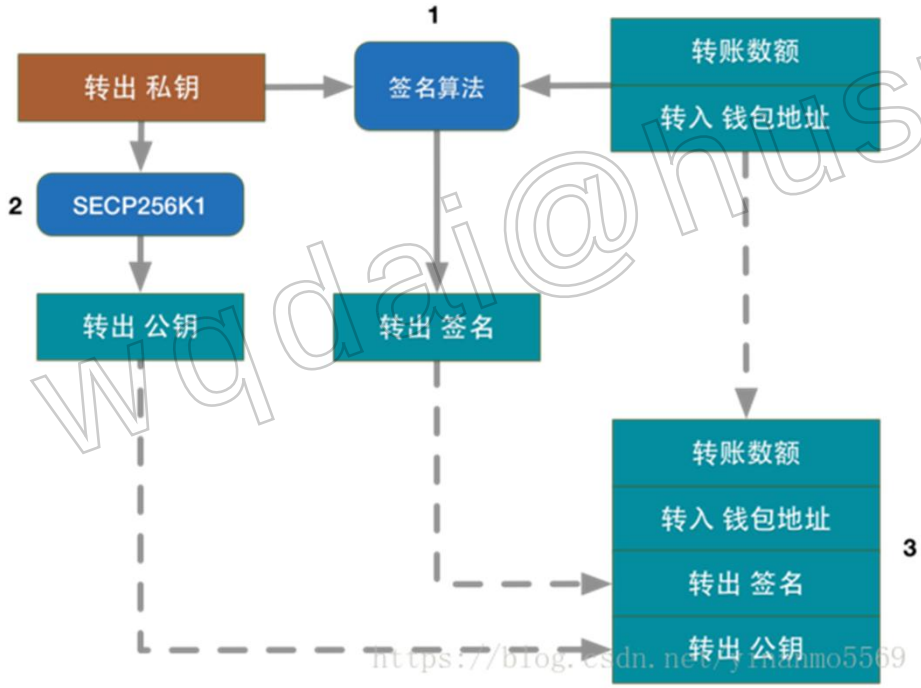
如何证明你是你?

详细的钱包抵制生成过程如下:



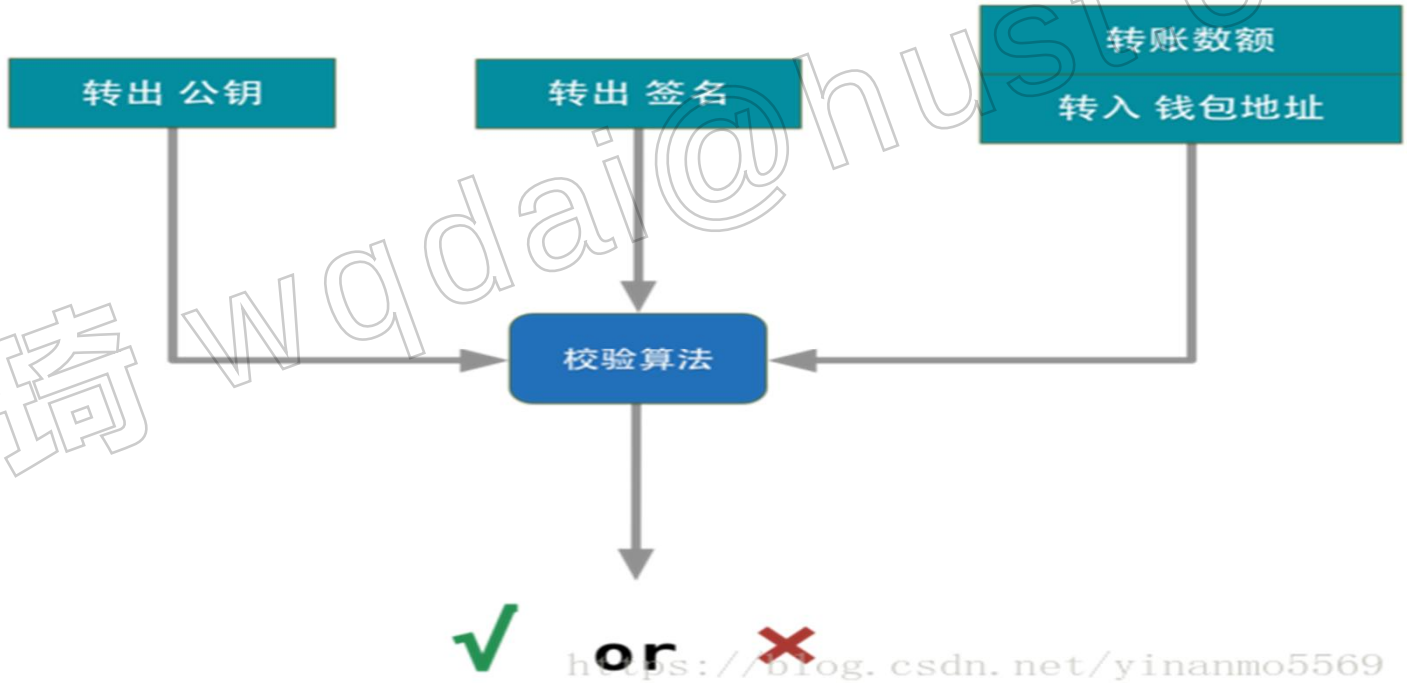
如何证明你是你?

在发起交易时，交易发起人需要使用【私钥】对交易进行签名，生成交易的过程如下:



如何防止你的钱被使用两次?

这个就涉及到比较复杂的共识算法了，比特币是使用的PoW(Proof of Work)算法．每个交易都需要广播请求到P2P网络．经过大家验证，大家验证就需要验证你是否有钱，验证你是不是真正钱包的主人，智能合约整个验证过程还是比较复杂的，之后单独再写一篇博客来讲．



ABE及变色龙哈希

代炜琦

wqdai@hust.edu.cn
华中科技大学
Huazhong University of Science and Technology

什么是ABE?

ABE是一个**公钥加密算法。**

- 既然是公钥加密算法，就有公钥和私钥的概念

ABE是**基于属性的加密**，又称模糊的基于属性的加密，被看作是最具前景的支持细粒度访问的加密原语。

- 设计者将**属性集合**与**策略**嵌入到了用户私钥与密文中，这样一来，私钥与密文输入解密算法尝试解密的过程，实际也就是属性集合与策略相匹配的过程，倘若能够匹配成功，则算法顺利完成解密操作，用户可成功恢复出明文数据。倘若匹配失败，则用户无法恢复明文，解密失败。

ABE根据嵌入对象的不同分为KP-ABE**和**CP-ABE。

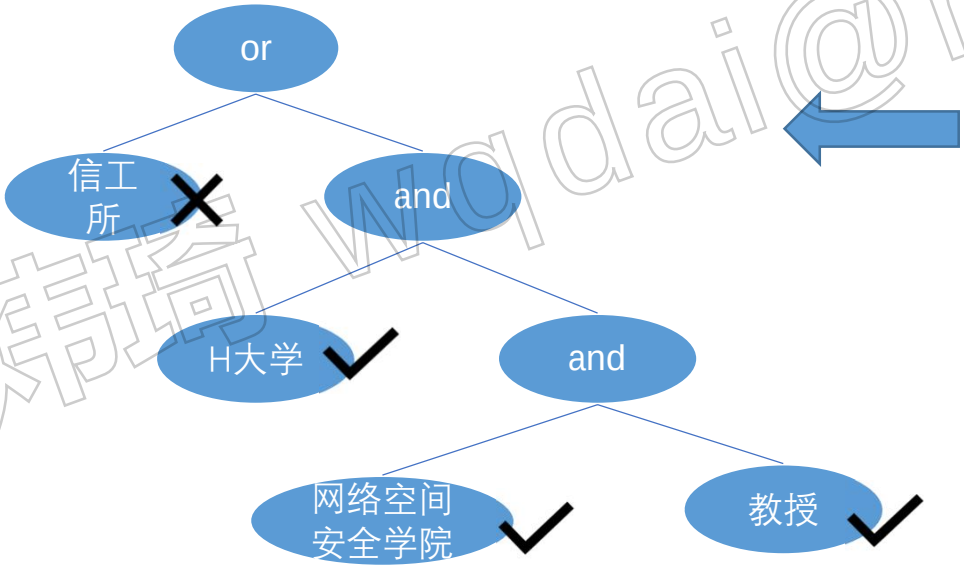
属性

举个例子，邹教授是H大学网络空间安全学院的教授，那么“H大学”，“网络空间安全学院”以及“教授”就是ABE中刻画邹教授这个人的多个属性，这些属性可以构成一个**属性集合**。

策略

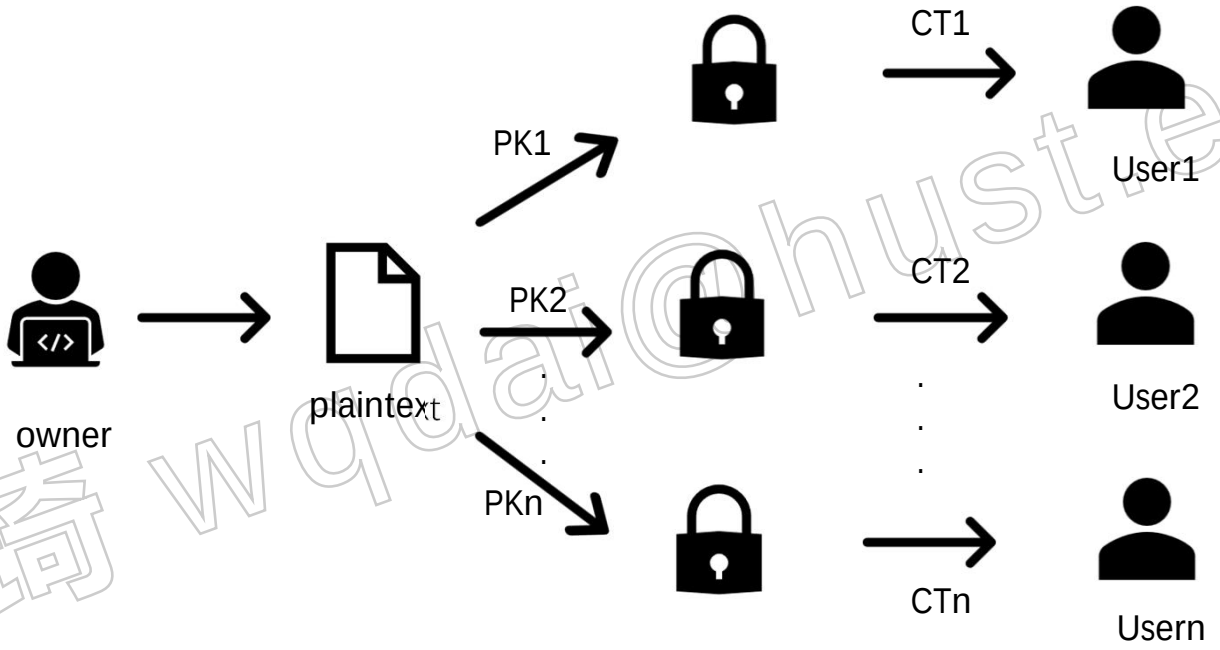
策略实际是由属性及它们间关系所组成的一个逻辑表达式。

例如，信工所 or（H大学 and 网络空间安全学院 and 教授）

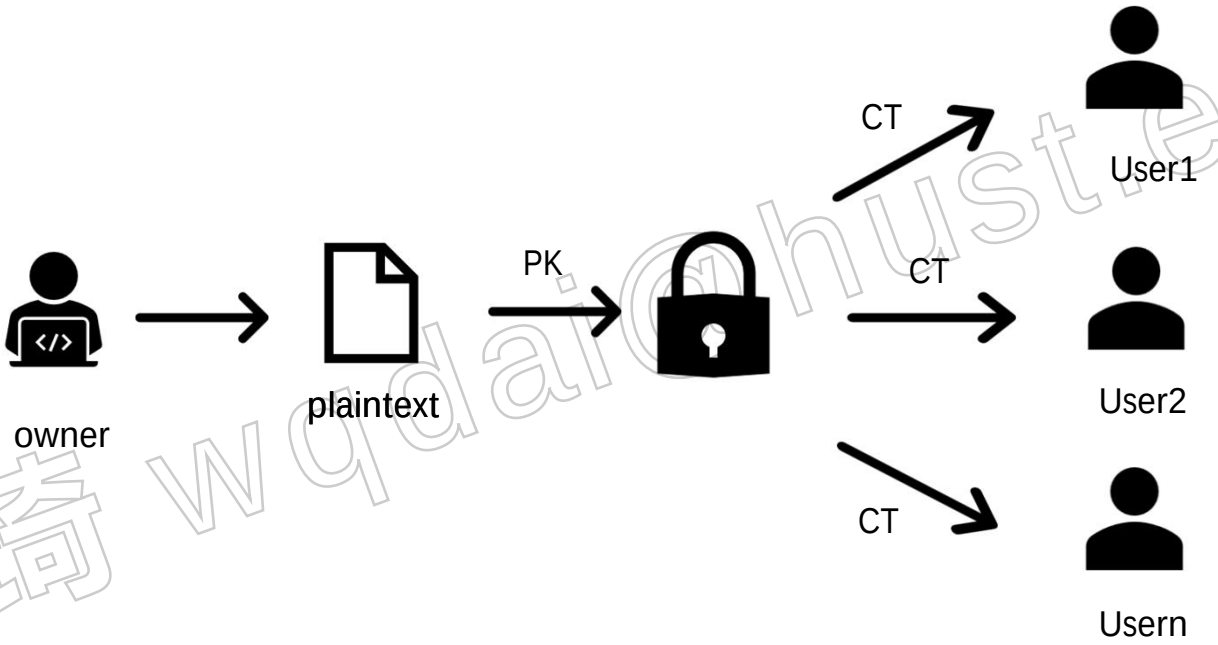


邹教授：
H大学
网络空间安全学院
教授

传统公钥加密算法



ABE加密算法



KP-ABE

- KP-ABE（基于密钥策略的属性加密）是将**策略**嵌入到**密钥**中，**属性**嵌入到**密文**中。
- 密钥对应于一个访问结构而密文对应于一个属性集合，解密当且仅当属性集合中的属性能够满足此访问策略。
- KP-ABE的应用场景则更加偏向于**付费视频网站**、**日志加密管理**等等。

CP-ABE

- CP-ABE（基于密文策略的属性加密）是将**策略**嵌入到**密文**中，**属性**嵌入到**密钥**中。
- 密文对应于一个访问结构而密钥对应于一个属性集合，解密当且仅当属性集合中的属性能够满足此访问结构。
- CP-ABE的应用场景一般是**公有云上的数据加密存储与细粒度共享**。

CP-ABE与KP-ABE

	CP-ABE	KP-ABE
策略相关	密文	密钥
数据拥有者是否有权决定数据能被谁访问	能	否
属性集合相关	密钥	密文
属性集合中属性来源	用户属性	数据/文件属性
适用的场景	云环境、区块链数据加密共享 P2P社交网络	付费视频网站 日志管理

CP-ABE实现流程

① **设置**：除了隐藏的安全参数不接受其他输入，输出公开参数PK和主密钥MK。

② **加密**：该算法以公共参数PK、明文数据M和访问结构A为输入，加密M并产生密文CT，使得只有具有满足访问结构的一组属性的用户才能解密消息。

③ **密钥生成**：密钥生成算法以主密钥MK和描述密钥的一组属性S作为输入，输出一个私钥SK。

④ **解密**：解密算法以公共参数PK、包含访问结构A的密文CT和一组属性的私钥SK作为输入。如果属性集S满足访问结构A，则算法将解密密文并返回明文数据M。

基于CP-ABE的访问控制

一般形式包括四个阶段，即系统参数的生成、私钥的提取、用公钥加密得到密文和访问控制树、用私钥解密得到明文。

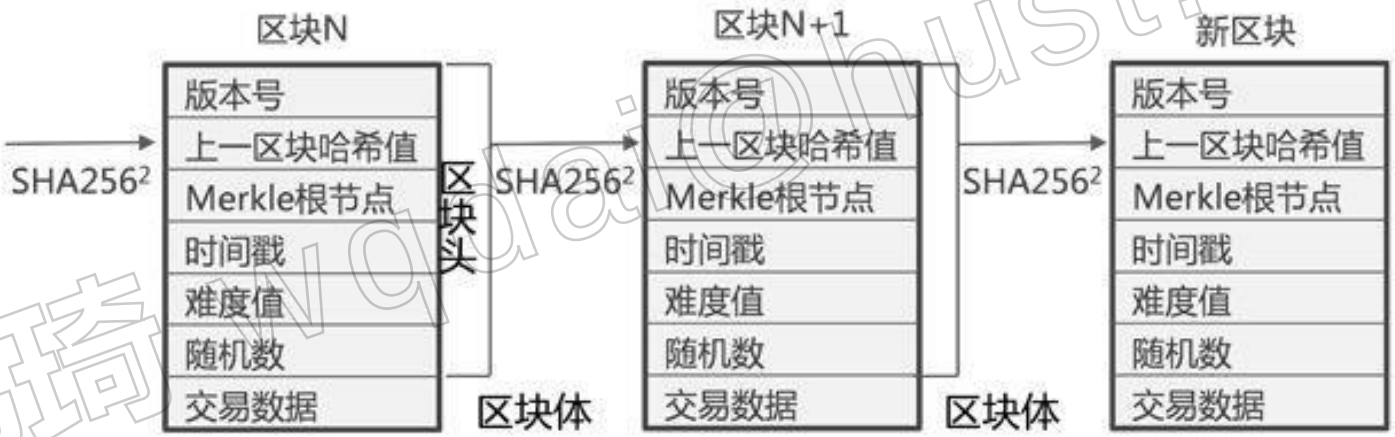
- **初始化**：服务器端生成公钥PK和主密钥MK
- **加密**：对服务器端的资源服务M（例如一份文件）进行加密生成密文CT，并生成由属性资源组成的访问控制树T，其中,CT中暗含T
  - A. 访问控制树T中的每一个节点x（包括叶节点）都定义一个多项式Qx。结点的遍历方式为由根结点R开始从上往下，从左往右的先序遍历方法。
  - B. 根结点R随机选取s，满足$Q_R(0)=s$；并选取其他各点完成对树中的每个节点多项式Qx的完整定义。
  - C. 运用公式CT对M进行加密

基于CP-ABE的访问控制

- **提取**：客户端用户输入自己的属性集S，生成用户的私钥SK
- **解密**：利用递归算法从T的叶子节点到根结点R开始递归。对树中的每个结点通过执行判决条件来计算Fx，直至计算出s。根据s运用对称密钥算法AES对密文进行解密。
  a) 利用结点函数来表示T中任意结点与用户提供的属性的访问判定，通过则得到最后的密钥s
  b) 由对称密钥算法AES对密文进行解密。

变色龙哈希-区块链的问题

区块通过**哈希函数**链接区块。其不可篡改的特性也源自于此。



变色龙哈希-区块链的问题

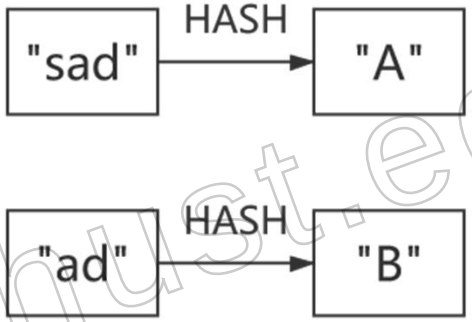
问题：

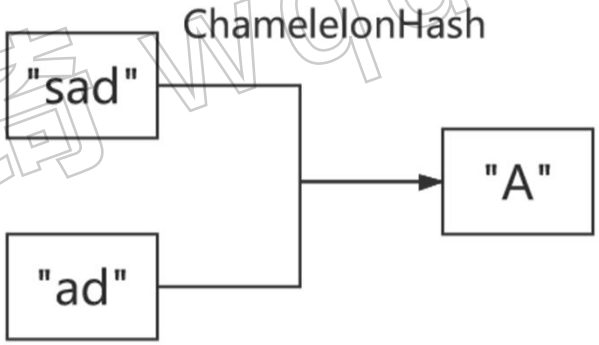
当区块信息需要修改的时候应该怎么办？

比如：上传交易记录时操作错误需要修改错误记录等

变色龙哈希-简介

传统哈希函数，一个输入对应一个输出，具有**单向性**和**抗碰撞性**。





变色龙哈希函数，在使用**陷门**的情况下，可以**人为制造哈希碰撞**。

变色龙哈希-原理介绍

变色龙哈希函数基于指数的两个特性被设计出来：

指数特性一： g^a*g^b = g^(a+b)

指数特性二： (g^a)^b = g^(ab)

变色龙哈希-原理介绍

假设一个区块**原内容**为m，小明拥有变色龙哈希的**陷门**为x，这个**陷门**对应的公钥为h=g^x。

生成变色龙哈希对应的**随机数**为r，则此时该区块的哈希值为：

H(m)=g^m*h^r

变色龙哈希-原理介绍

现在将内容m篡改为m'，现在希望找到一个随机数r'，使得H(m')=H(m)。

r'的求解过程是：

H(m)=(g^m)*(h^r)=g^m*g^(xr)=g^(m+xr)

H(m')=(g^m')*(h^r')=g^m'*g^(xr')=g^(m'+xr')

所以，已知m,m',x,r,则r'=(m+xr-m')/x

变色龙哈希-原理介绍

问题：

小明拥有该哈希函数的陷门可以随意修改区块的内容，他是否可以在区块链的系统中为所欲为？

变色龙哈希-原理介绍

那当然是不可能的。

首先，区块链都是存在本地的，如果有修改可以有记录。

其次，如果持有后门x者乱修改，或者不承认修改，拿着记录与被修改后的区块进行碰撞即可证明其被修改，因为对于没有后门x的人来说，要在2^256中找到碰撞几乎是件不可能的事情。

变色龙哈希-应用

基于变色龙哈希的可编辑区块链平台可以作为一种新型区块链广泛应用于有数据交换、数据记录和数据屏蔽的场景中。

例如：结合门限签名控制陷门所有权将其应用于多银行共同维护的联盟链中。