1. 哈希函数

区块链使用的哈希函数(cryptographic hash function)是SHA-256。哈希函数需要满足的三个条件：

1. 可以接收任意长度的字符串作为输入；
2. 产生固定长度的输出；
3. 高效可计算的，即计算时间在合理范围内。

哈希函数的三个性质：

1. **collision resistance（抗碰撞性）**

这里的collision指**哈希碰撞**，即两个不同的输入用哈希函数算出的哈希值相等：



发生哈希碰撞是不可避免的，原因是哈希函数的输入空间是无限大的，而输出空间是有限的，以SHA-256为例，输出只有种可能性。如果当输入的可能性大于时，发生哈希碰撞就是必然的。这里所谓的**collision resistance**是指我们**没有什么高效的方法去人为的制造哈希碰撞**，即知道一个y，没有特定的找到x的方法使得H(x)=H(y)。如果一定要找的话只能用brute-force的方式，即遍历输入空间不断计算哈希值，直到找到恰好碰撞的那个x。

**collision resistance**可以用来对一个信息求digest，用来检测对这个信息的篡改。因为如果这个信息的内容被改掉了，那么哈希值必然就会改变，**没有专门的只修改信息内容而不改变其哈希值的手段**。比如使用云存储时，在上传文件之前计算一个哈希值放在本地，以后下载下来再计算一下哈希值和本地之前记录的哈希值比较一下，如果一样就说明下载下来的内容还是之前的内容，没有被篡改。

**注意：没有哪一个哈希函数已经在数学上被证明是collision resistance的，有些哈希函数已经被找到人为制造哈希碰撞的方法，比如MD5、SHA-1（**[**王小云教授**](https://baike.baidu.com/item/%E7%8E%8B%E5%B0%8F%E4%BA%91/29050?fr=aladdin)**）**

1. **hiding(单向的)**

hiding是指**哈希函数的计算过程是单向的**，是不可逆的。即可以，但无法。这表示哈希值 H ( x )没有泄露有关输入x的任何信息。

但是，因为总可以蛮力求解，所以hiding成立的前提是**输入空间足够大，使得蛮力求解在当下计算力条件下不可行**。而且**输入的分布要比较均匀，各种取值的可能性都差不多**，不然只需要在分布的峰值部分做蛮力求解就很可能能够逆推找到输入值了。实际操作时如果输入空间不够大或者分布不均匀，可以在输入后面拼接一个随机数nonce，强行增大输入空间：H(x||nonce)，这里随机数的选取要足够随机。

下面是一个利用**hiding**特性实现**digital commitment(数字承诺)**的例子：

**digital commitment也叫digital equivalent of a sealed envelop。**如一个人说自己能预测股票涨停，那么如何证明呢？如果让他提前公布自己的“预测”——明天股票的涨停，然后明天再去看股市，那么他公布的信息本身有可能就会影响到股票涨停(此人在股市的影响力可能比较大，比如巴菲特)。如果在股票涨停之后再公布，那么大家会怀疑他有没有篡改自己的“预测”。 而这里sealed envelop也就是解决这个问题的办法，让这个人先把预测结果写在纸上，放进信封里**封好**，然后交给第三方的公证机构保管。第二天股市收盘后，再让公证机构**检查信封完好**、打开信封、查看纸上的预测结果和实际涨停是不是一致即可。

而使用**digital commitment**在这个问题上的做法是把预测结果计算出一个哈希值，然后把哈希值公布出去，因为有hiding的性质，所以大家没法通过哈希值知道预测结果。第二天收盘后公布预测结果，因为有**collision resistance**的性质，所以预测结果是没有篡改过的，否则和公布的哈希值就对不上了。

1. **puzzle friendly**

比特币中使用的哈希函数需要满足前面密码学中哈希函数的两个性质，但还有一些额外的内容。

除了前面两个性质，比特币中的哈希函数还要满足**puzzle friendly**这个性质。这个性质是说哈希值的计算结果是不可预测的，也就是光是去看这个输入没法知道计算出来的哈希值可能是什么样的（具有什么特征）。

即例如，想要计算出的哈希值落在某个范围内(**系统需要一个具有指定特征的哈希值**)，那没有特定的办法去构造输入。又如，想要计算出的哈希值前面有k个连续的0，也没有特定的办法去构造输入。甚至**无法知道什么样的输入更有可能得到具有特定特征的哈希值**，要想得到这种输入就只能一个一个去试，没有捷径。

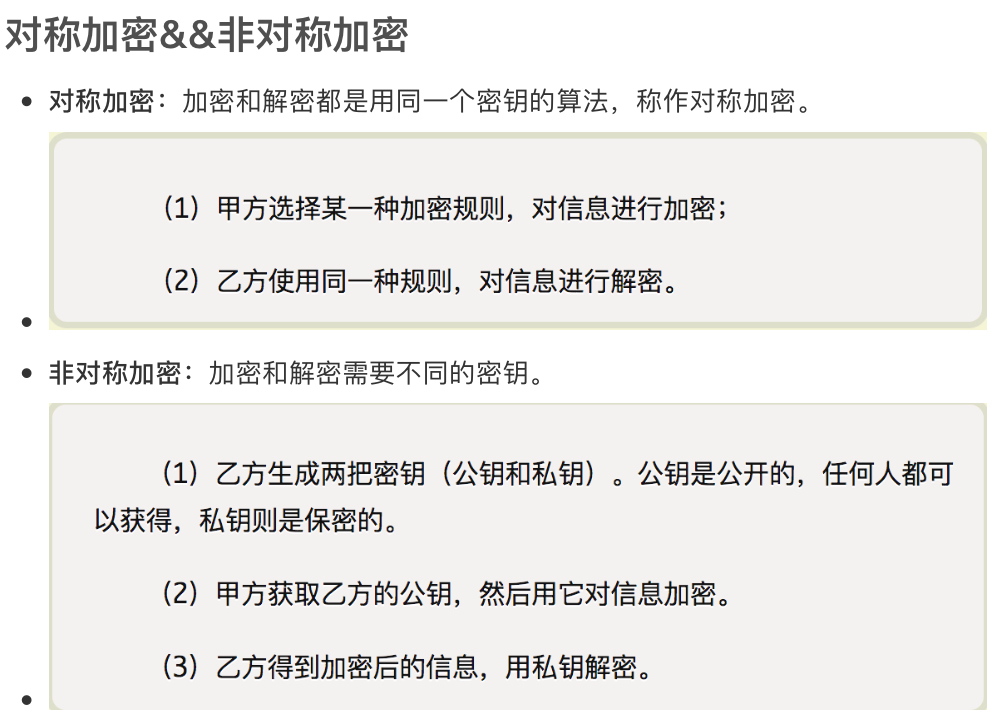
比特币挖矿的过程就是去找一个随机数nonce，这个nonce和区块的块头里的其它信息合并在一起作为输入，取哈希值，**所得哈希值要小于等于某个指定的阈值**：



注意，nonce不是区块块头以外的东西，它是其中的一个域，但是可以人为设置。挖矿的过程就是不停的去试nonce，使得整个块头部分取哈希小于等于target。有了**puzzle friendly**这个性质，使得比特币挖矿的过程没有捷径可言，只能不停的去试大量的nonce才能找到符合要求的解，所以才能用来用作**工作量证明**（proof of work，POW）。

这个puzzle friendly和最前面的collision resistance有一定的联系，但不是完全一样。前者是无法去人为制造哈希碰撞，后者是无法去人为构造符合特定特征的哈希值。

1. 数字签名
2. **两种加密体系—对称加密与非对称加密**



在数学上可以用d(c(x))=x，来表示加密解密过程。

1. **非对称加密算法 –公钥与私钥的产生**

**私钥**： 私钥就是随机选取的一个数而已，可以使用随机数生成器生成，也可以采用手工生成的方式。比如需要一个256位的私钥，那你可以掷硬币256次，记录0和1，将这256位随机得到的二进制数字作为你的私钥。但注意，私钥一定要保持机密，而且最好备份防止意外丢失。

**公钥** ：私钥通过一些算法(比如椭圆曲线算法)可以生成公钥，公钥可以公开，实际中无法从公钥中逆向算出私钥。

1. **常见的非对称加密算法**

1. RSA：名字来源于三位科学家的名字，目前应用非常广泛、历史也比较悠久的数字加密和签名技术，支付宝就是通过RSA算法(2048位)来进行签名验证，安全性的保证基于大整数的质因数分解十分困难的原理。

2. DSA：即Digital Signature Algorithm，数字签名算法，DSA 仅能用于数字签名，不能进行数据加密解密，其安全性和RSA相当，但其性能要比RSA快。

3. ECC：Elliptic Curve Cryptography，椭圆曲线加密算法，是基于椭圆曲线数学理论，相比RSA，ECC优势是可以使用更短的密钥，来实现与RSA相当或更高的安全。据研究，160位ECC加密安全性相当于1024位RSA加密，210位ECC加密安全性相当于2048位RSA加密。

1. **比特币系统的账户管理**

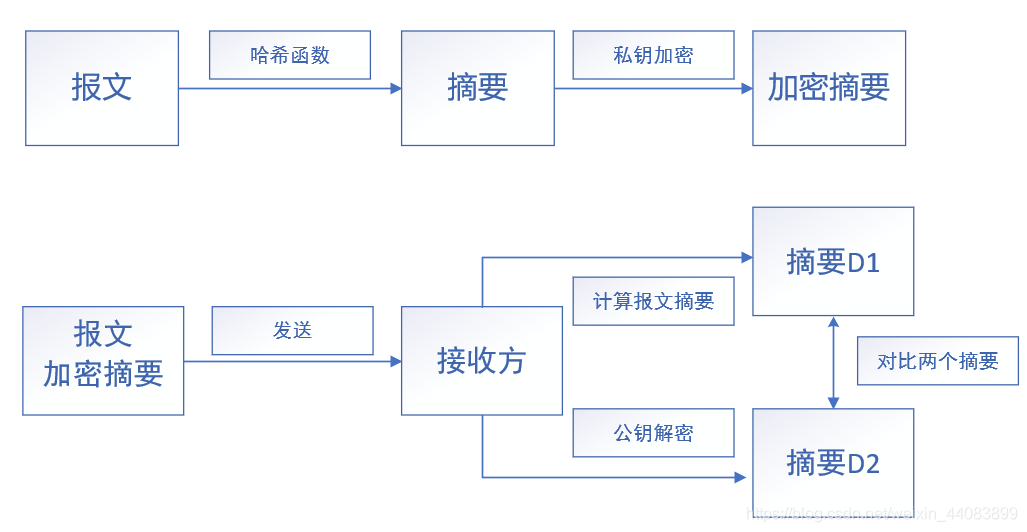
日常生活中想要开个账户，那就是带上证件去银行办理，这种就是**中心化**的账户管理方式。在比特币系统中是**去中心化**的，也就是没有银行之类的机构。每个用户自己决定开户，不需要任何人批准，**开户的过程就是在本地创立一对公钥和私钥的过程**：



公钥就相当于用户的银行账户，转账只需要知道公钥就可以了。私钥相当于账户密码，知道了私钥就可以将账户上的钱转走。

1. **数字签名过程**

发送报文时，发送方用一个**哈希函数**从报文文本中生成**报文摘要**，然后用发送方的**私钥**对这个摘要进行加密，这个**加密后的摘要将作为报文的数字签名和报文一起发送给接收方**，接收方首先用与发送方一样的哈希函数从接收到的原始报文中计算出报文摘要，接着再**公钥**来对报文附加的数字签名进行解密，如果这两个摘要相同、那么接收方就能确认该报文是发送方的。



1. **数字签名的作用**
2. **确定消息的不可抵赖性**

因为他人假冒不了发送方的私钥签名。发送方是用自己的私钥对信息进行加密的，只有使用发送方的公钥才能解密。

1. **数字签名能保障消息的完整性**

一次数字签名采用一个特定的[哈希函数](https://www.zhihu.com/search?q=%E5%93%88%E5%B8%8C%E5%87%BD%E6%95%B0&search_source=Entity&hybrid_search_source=Entity&hybrid_search_extra=%7B%22sourceType%22%3A%22article%22%2C%22sourceId%22%3A%2225162134%22%7D)，它对不同文件产生的数字摘要的值也是不相同的。如果在传输中途 报文+加密摘要 的组合消息被修改了，那么接收方最终计算出的两个摘要D1和D2必定是不相等的。

1. **公钥私钥对重复问题**

既然公钥私钥对始终是在本地产生，那么产生重复怎么办？

如大量生成公钥私钥对，然后去对比产生的公钥是不是和区块链上已有的某个公钥相同。如果相同，那么就可以用对应的私钥把账户上的钱转走。

这种方式理论上可行，但是实际上没法做，因为产生重复的公钥私钥对的概率非常小，可以忽略不计。到目前为止还没有能用这种方法攻击成功的先例。

生成公钥私钥的过程是随机的，但要求选取一个好的**随机源**，否则前面的分析就不成立了（就有不足够小的可能生成重复的公钥私钥对）。

比特币中使用的签名算法不仅在生成公钥私钥对时有好的随机源，在之后每次签名的时候也要有好的随机源。如果签名时使用的随机源不好，就有可能泄露私钥。