

# 4. 哈希函数

特性：网上查

以下解释目的：为了向节点证明你是你自己  
用处，压缩，防对抗，减少计算成本和增加防伪

在此基础上我们来看向节点发送交易数据时，如何保证不被篡改：

数字签名

身份验证

当我们向节点网络发送交易数据时：

- 1) 首先对交易数据生成数字摘要；
- 2) 将数字摘要用私钥签名后，连同原始交易数据一起发送。

当节点们收到数据后：

- 1) 对交易数据同样使用哈希函数生成数字摘要；
- 2) 同时用我们的公钥将数字签名还原成数字摘要；
- 3) 比对 数字摘要 =? 数字摘要；
- 4) 如果两者相等则说明信息未被篡改。

如果邻村对交易数据进行篡改，则交易的数字摘要就会发生明显变化，无法通过验证。

💡 整个流程“认私钥不认人”，如果私钥丢失，攻击者就可以伪造交易与数字签名，骗过节点的验证。  
因此再次提醒私钥、助记词等打死也不能告诉别人！！

另外可能会有同学有疑问，直接将交易数据用私钥加密后生成数字签名，也能起到防伪的效果，为何还需要先进行hash？原因是节约计算成本。假设交易数据非常的长，那么使用私钥加密/公钥解密的整体计算量及耗时就会非常大，借助哈希函数“压缩”的特性，就能有效地控制计算成本。

至此，我们可以放心地发送交易数据给节点网络

这个在具体使用，原理为

将原始消息（就是交易信息）形成一个数字摘要

（哈希），再把这个数字摘要和私钥一起组成一个数字签名，这个时候把数字签名和原始消息同时发送到节点。

此时在节点内部，一边将原始信息去计算它的哈希值（图中右的下那条路），一边使用数字签名和公钥还原前面的哈希值，再将二者进行比对

