**区块链学习笔记**

——我是一个区块链小白，从今天起我要坚持学习区块链知识，争取早日捐给国家。每天的知识尽可能的用自己的语言写成一小段学习笔记，又描述不准确或者不正确的地方，我会在以后的笔记中更正，也欢迎各路大神前来吐槽。每篇笔记后面会附有一些重要知识来源的链接，供大家参考。每天学习的东西基本上都跟区块链相关，但不要指望有什么逻辑性，有时候我可能会在我感兴趣的某个点研究很深，也有可能会根据我的项目需要针对性的学习一些内容。

# 关于比特币你所不知道的事？

在中本聪的白皮书中，比特币的学名是一种点对点的电子现金系统。在比特币出现的前10年，一个叫做戴维的人在一个名叫密码朋克的邮件群中发表一篇名为B-money的文章，这篇文章开头从无政府主义引入了两个问题，第一个问题就是如果没有政府，谁来发行钱。另外一个问题是谁来保证合约的执行，也就是说如果两个人订立了合约，如果由于一方反悔，给另一方造成了损失，谁来惩罚违约方，或者谁来保证合约强制执行。

基于这两个问题，戴维提出了两个想法，一个是基于工作量证明的电子现金系统，另外一个就是智能合约，但是戴维只是描述了一个大概的概念，并没有设计出完整可行的体系。直到十年之后，中本聪精巧的设计出了比特币系统。

# 什么是侧链？

概念：

可以让比特币安全地从比特币主链转移到其他区块链，又可以从其他区块链安全地返回比特币主链的一种协议。

用途：

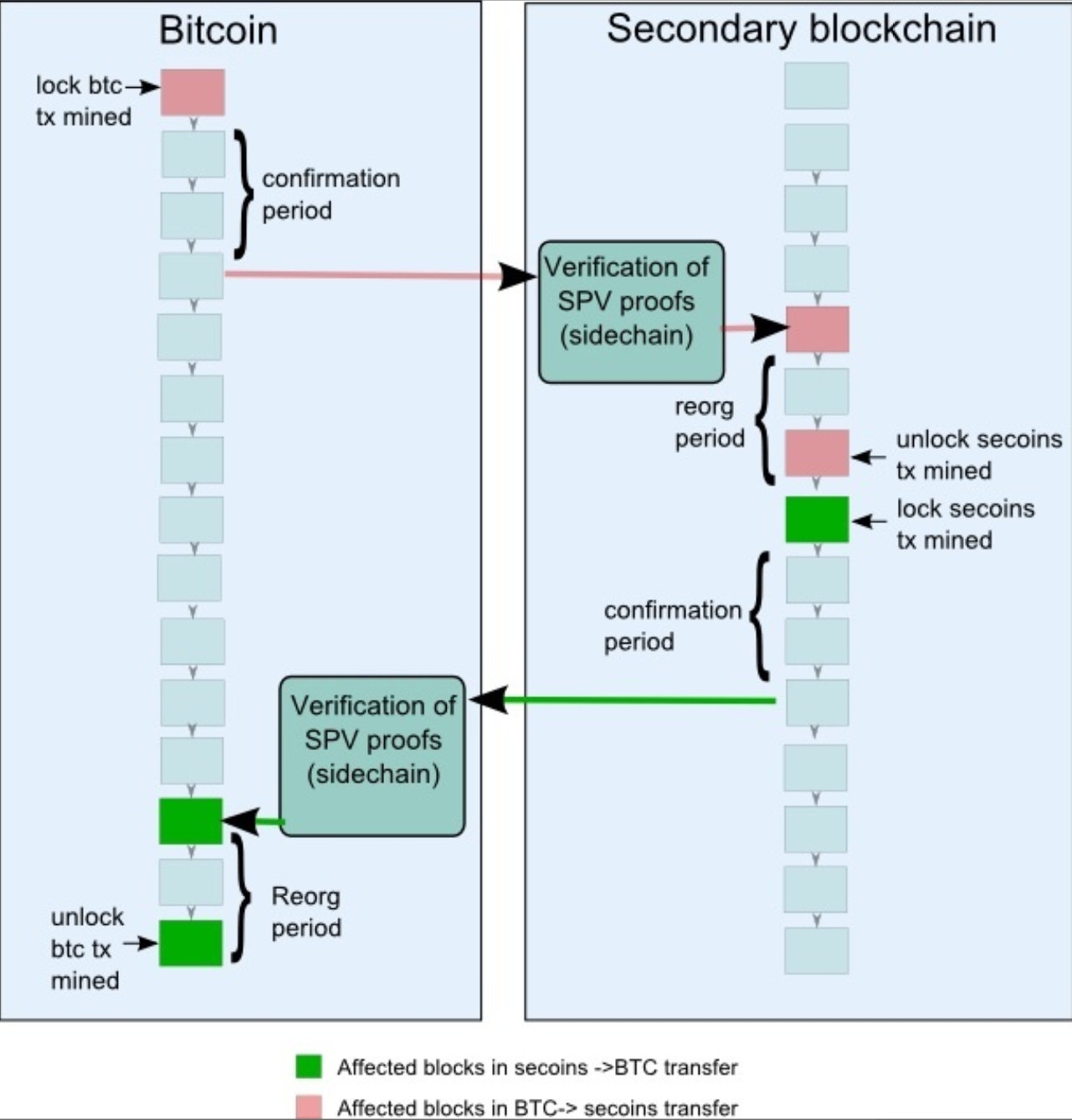
侧链的意义在于比特币不仅可以在比特币区块链上流通，还可以再其他区块链上流通。

产生的原因：

（1）应对其他区块链的创新威胁  
以太坊（Ethereum）区块链、比特股（Bitshares）区块链后来居上，对比特币区块链产生相当大的威胁，智能合约和各种去中心化应用在以上两个区块链上兴起，受到人们欢迎；而基于比特币的应用则因为开发难度大，项目不多。

（2）比特币核心开发组不欢迎附生链  
基于比特币区块链也有合约币（Counterparty）、万事达币（Mastercoin）和彩色币（ColoredCoin）等附生链，但是比特币核心开发组并不欢迎，觉得它们降低了比特币区块链的安全性。他们曾经一度把OP\_RETURN的数据区减少到40字节，逼迫合约币开发团队改用其他方式在比特币交易中附带数据。

（3）BlockStream商业化考虑  
2014年7月份以太坊众筹时，获得了价值1.4亿人民币的比特币，还有20%的以太币，开发团队获得了巨大的回报。但是比特币核心开发组并没有因为他们辛勤工作获得可观回报，因而他们成立了BlockStream，拟实现商业化价值。



# DAG（有向无环图），异步处理

以交易为单位的异步记账技术，DAG中没有区块概念，不用把数据打包成区块，而是每个用户可以提交一个数据单元，这个数据单元里可以有很多东西，比如交易，消息等等。数据单元间通过引用关系链接起来，从而形成具有半序关系的DAG（有向无环图）。DAG的特点是把数据单元的写入操作异步化，大量的钱包客户端可以自主异步地把交易数据写入DAG，从而可以支持极大的并发量和极高的速度。

DAG不是一条链，发送每笔交易需要验证之前的至少一笔交易，相当于每个人都是矿工，只有少量的工作量证明来防止SPAM。

交易越多，速度越快

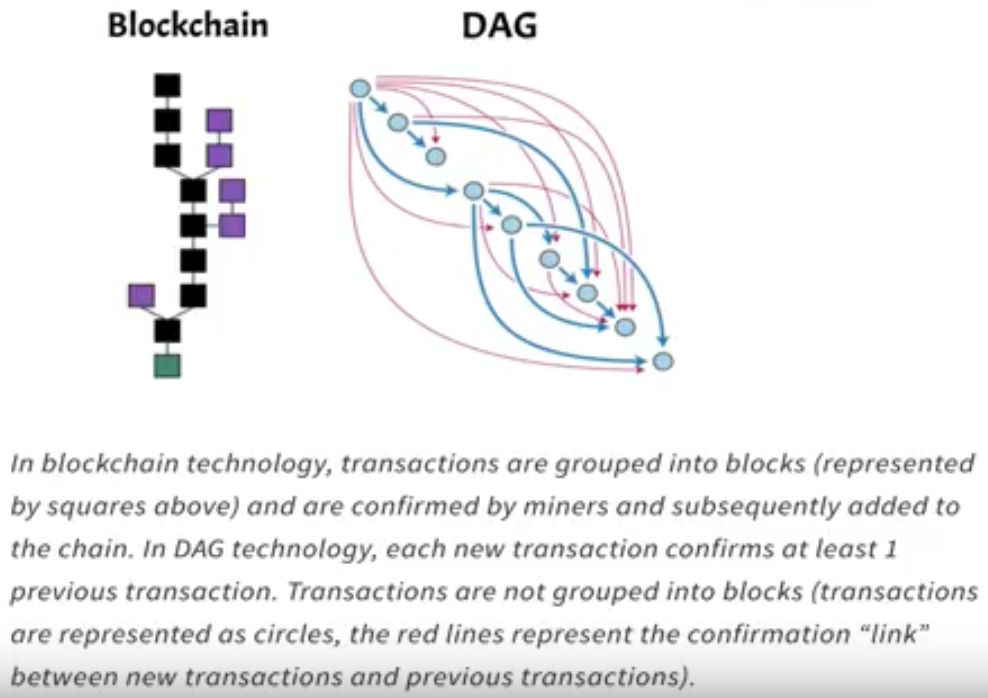
目前使用中心节点验证所有交易

掌握百分之33%的节点就能对系统构成

数据如何存储，每个节点都要存储，数据会不会很大？

节点的概念？

每笔交易被验证几次才可成为不可逆交易



在区块链中，交易被分组存放在区块中，区块经过矿工确认后按顺序添加到链上。在DAG中每一个新的交易确认最少一个之前的交易。交易并不会被分组打包到区块中。

# 闪电网络

参考资料

<http://www.8btc.com/ln-rn-corda>

再比特币链外开通小额支付通道，双方需要锁定一定数量的比特币用于支付，如果A，B之间没有开通小额支付通道，也可以通过一个或多个中间人C，即利用A-B，B-C之间的通道建立间接通道。以此为基础拓展，可以形成一个巨大的网络，即闪电网络，闪电网络中可能会形成若干个中间商，他们与大量的用户之间存在通道，由此建立用户之间的联系，中间商收取一定的通道费用。

闪电网络提供了一个可扩展的微支付通道网络。交易双方若在区块链上预先设有支付通道，就可以多次、高频、双向地通过轧差方式实现瞬间确认的微支付；双方若无直接的点对点支付通道，只要网络中存在一条连通双方的、由多个支付通道构成的支付路径，闪电网络也可以利用这条支付路径实现资金在双方之间的可靠转移。

# 雷电网络

相当与基于以太坊建立的闪电网络

# SPV

<http://8btc.com/article-2002-1.html>

简单支付确认，在不运行全节点的情况下，也能够对支付进行检验的方法。用户只需要保本区块头。需要验证某个交易是否真实存在时，

计算该交易的哈希值，

然后搜索包含该哈希值的区块并下载该区块的信息，根据区块的信息计算哈希值，如果与区块头一致，则确认该交易存在。

门罗币 环形签名

抗量子加密算法

# NEO

NEO 身份系统

NEO 分布式存储技术 减小区块大小

ETH

# 币乎里KEY分配机制

币乎是一个基于区块链技术的知识分享平台，在平台上我们发表的文章可以通过被用户点赞的方式获得收益，KEY，同时点赞的用户也能够获得收益。根据币乎的分配机制，每篇文章获得的奖励，45%归作者，45%按权重分配给投票的用户，

能量

币乎里每人每天100点能量，每次点赞或者踩都需要消耗10点能量，消耗的能量线性恢复，每天100点。因此能量是币乎社区里的稀缺资源。

# Scrypt算法，MD5，SHA256，FPGA矿机

区块链常用的三种哈希算法，其中使用的MD5，SHA256没有内存依赖，可以通过CPU以及ASIC芯片挖矿。Scrypt算法是一种内存依赖型算法，比较适合通过显卡挖矿。

FPGA称为现场可编程门阵列，一般来说币ASIC（专用集成电路）的速度慢，实现同样功能币ASIC电路面积要大。但是FGPA板每块有高达24.8GB/s或以上的内存带宽，属于专用可编程芯片，所以FPGA矿机可以有效运行scrpyt算法进行大规模挖矿。

# P2P网络

比特币每个节点最多连接125个节点，比特币钱包第一次启动时，会试图连接写在钱包里的一个DNS种子文件中的一个或多个IP地址。

在钱包首次使用，或者区块高度落后144个块，或者落后24小时，使用IBD(Initial Block Download)方法下载区块

# 基于MAC地址的物理加密

量子计算机的技术还需要一定时间，并且一定是一部分人先掌握技术，从而对整个现有的加密体系构成挑战，这不仅包含区块链和数字资产，也包括银行在内的各大金融机构。为避免私钥被大规模破解，FTU增加了物理双重加密的方式，来确保量子计算机不能够对整个系统造成毁灭性打击。

每块网卡都有一个全球唯一的48位的MAC地址，在通信网络中起到区分每个节点的作用，正是由于MAC地址数据的这个性质特点，使他成了不少应用软件实时保护的“指纹”，或用作安全方面的身份标识。大多数应用读取的是注册表中的MAC数据，对于这类应用，只要对其注册表中的MAC数据进行相应变更，或对其网卡IO空间的寄存器进行相应改写，就可以实现MAC地址的伪装。即使在应用程序中通过NetBIOS提供的API接口函数获取MAC地址，同样也是不可靠的。而FTU会从ROM存储器中读取网卡的MAC地址，从而保证了其准确可靠性。

每位用户可以选择绑定最多3个MAC地址，绑定的MAC经过SHA256加密后存放在用户账户中，用户使用私钥（或助记词）通过绑定的设备登陆客户端，客户端会自动读取设备的MAC地址并经过哈希运算后与用户账户中存储的哈希值进行比对，只有私钥和地址都正确，才能进行转账等关键操作。如果只有私钥而地址不匹配则只能查看账户信息。为防止设备丢失导致的资金损失，建议绑定3个设备，通过其中一个设备和私钥，可以操作绑定和解绑其他设备。

# FUSION

FUSION可以实现不同资产在同一条链上的交易，跨期交易，所有权和使用权分离。

如果需要把10个BTC放到链上，通过在BTC上创建一个新的地址，把10个比特币赚到新的地址中锁定，在FUSION确认了这笔交易后，就会在fusion链上产生10个BTC，由于fusion支持几乎所有的代币，因此可以通过只能合约实现所有币的转换。

比特币

区块确认并不需要等到10分钟以后，

# Paxos

Paxos是一种用于分布式系统中的保持系统内信息一致性的算法。

产生背景：

在分布式系统中，总会发生宕机或者网络异常等情况，因此需要一套算法保证在出现网络问题时，仍然能够保证系统内数据传输的一致性。

Paxos中的约束规则：

* **P1：一个Acceptor必须接受它收到的第一个提案。**
* **规定：一个提案被选定需要被半数以上的Acceptor接受**
* **P2：如果某个value为v的提案被选定了，那么每个编号更高的被选定提案的value必须也是v。**
* **P2a：如果某个value为v的提案被选定了，那么每个编号更高的被Acceptor接受的提案的value必须也是v。**
* **P2b：如果某个value为v的提案被选定了，那么之后任何Proposer提出的编号更高的提案的value必须也是v。**
* **P2c：对于任意的N和V，如果提案[N, V]被提出，那么存在一个半数以上的Acceptor组成的集合S，满足以下两个条件中的任意一个：**

S中每个Acceptor都没有接受过编号小于N的提案。

S中Acceptor接受过的最大编号的提案的value为V。

* **P1a：一个Acceptor只要尚未响应过任何编号大于N的Prepare请求，那么他就可以接受这个编号为N的提案。**

算法描述：

Paxos算法分为两个阶段：

阶段一：

* **Proposer选择一个提案编号N，然后向半数以上的Acceptor发送编号为N的Prepare请求。（此阶段只发送编号，不发送Value）**
* **如果Acceptor收到一个编号为N的Prepare请求，并且N大于该Acceptor已经响应过的所有Prepare请求的编号，那么它就会将它已经接受过的编号最大的天作为响应反馈给Proposer，同时该Acceptor承诺不再接受任何编号小于N的提案（如果没有此承诺，Acceptor接受更小编号的方案，就会导致当前proposer的提案被忽略）**

阶段二：

* **如果Proposer收到半数以上Acceptor对其发出的编号为N的Prepare请求的响应，那么它就会发送一个针对[N,V]提案的Accept请求给半数以上的Acceptor。注意：V就是收到的响应中编号最大的提案的value，如果响应中不包含任何提案，那么V就由Proposer自己决定。**
* **如果Acceptor收到一个针对编号为N的提案的Accept请求，只要该Acceptor没有对编号大于N的Prepare请求做出过响应，它就接受该提案。**

**Paxos算法存在活锁的问题，当多个proposer交错提交时，有可能相互排斥导致没有一个proposer能够提交成功，Fast Paxos做了一些优化，通过选举一个leader，只有leader才能提交提案**

参考文献：

<https://www.cnblogs.com/linbingdong/p/6253479.html>

# RAFT

Raft的工作模式是一个leader和多个follow模式，初始状态都是follow状态，并且所有server随机睡眠随机睡眠一段时间，这个时间在0~1000ms之间，最先醒来的server进入Candidate状态，Candidate有权发起投票，向其它所有server发出request\_vote请求，请求其他server给它投票成为leader。当其他server收到请求后，将自己仅有的一票投给A，同时继续保持Follower状态并重置选举计时器。

通信时有leader向follower发送信息，follower接收信息。如果follower宕机，在恢复之后会同步leader的信息。

Follower会向leader发送心跳，以确保leader在线，如果leader宕机，无法收到心跳，则该follower发起竞选leader的投票。

参考文献：

<https://www.jianshu.com/p/4711c4c32aab>

# 预言机Oracles

Oracle的核心功能时链外数据的上链服务，链外数据，例如温度天气等，智能合约是无法直接读取的，预言机将数据抓取下来，然后向特定的钱包地址进行转账，并将天气信息写入交易备注。智能合约查询特定地址的交易记录，就可以获取备注的天气信息了。

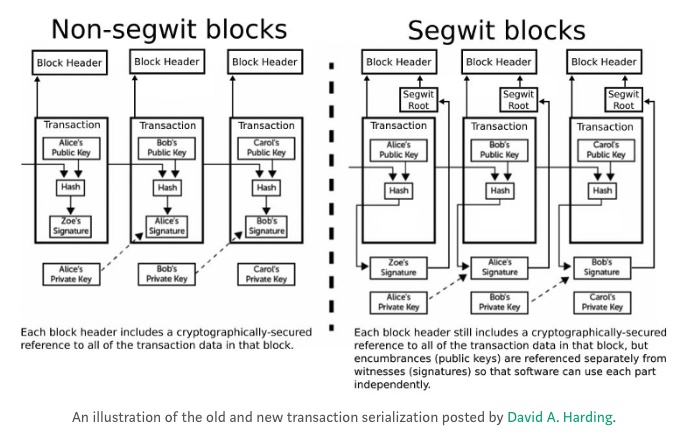
另外一项功能是预测市场，通过人们下注博弈，使数据更可信

参考文献：

<https://zhuanlan.zhihu.com/p/27981531>

# 隔离见证

比特币中每笔交易上会带有交易发送者的签名信息，隔离见证就是把这个信息从交易信息中隔离出来，如图：



但是这些签名信息仍然存在于区块之中。

隔离见证重新整理了每个交易内容的布局方式。把脚本签名从交易内容的结构里面拿出来，放到了最下面，并有一个指针Pointer指向它。

# 多重签名

# 加密算法

参考文献：

<https://blog.csdn.net/mfwing/article/details/51439708>

加密算法分为对称加密算法，DES，3DES，AES

非对称加密算法和散列算法，DSA，ECC，RSA属于非对称加密算法，公钥加密的信息可以用私钥解密，私钥加密的信息可以用公钥解密，

SHA256，MD5属于散列算法

量子计算机可以破解非对称加密算法，即公私钥的密码体系，短期内还无法破解散列算法

# 同态加密

同态加密是基于数学难题的计算复杂性理论的密码学技术。对经过同态加密的数据进行处理得到一个输出，将这一输出进行解密，其结果与用同一方法处理未加密的原始数据得到的输出结果是一样的。

# EKT

参考文献：

<https://github.com/EducationEKT/EKT/blob/master/docs/whitepaper.md>

# http://upyun-assets.ethfans.org/uploads/photo/image/a1c08251ec654a8c9392821362a17946.pngETH