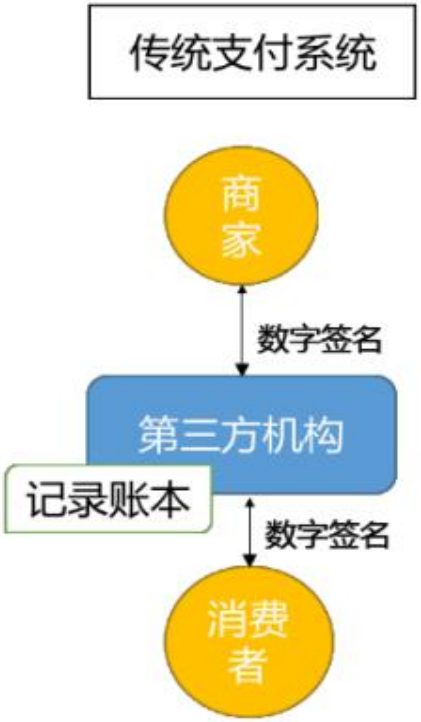
区块链发展史和区块链体系结构

区块链的起源

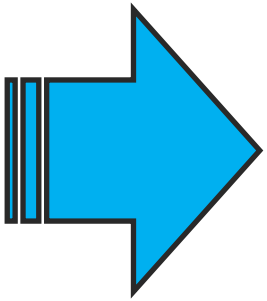
- 在比特币出现之前，一般采用建立可信第三方机构的方法，对交易进行记录。这种情景下，数字货币的流通对第三方机构的依赖程度很高，所以要求第三方具有很强的可靠性，同时存在第三方作恶的可能性。
- BitCoin网络中的任意两个用户可在无可信第三方参与的情况下进行P2P交易，并将每笔交易计入总帐中。

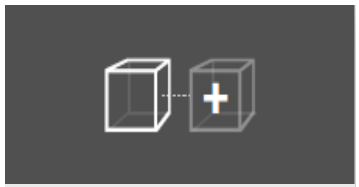


区块链的起源

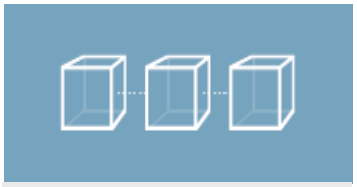
- 区块链技术是构建比特币区块链网络与交易信息加密传输的基础技术。它基于密码学原理而不基于信用，使得任何达成一致的双方直接支付，从而不需要第三方中介的参与。

通过区块链技术解决信任问题





交易被打包成区块，区块“链接”在一起形成区块链



选取最长链作为主链，全网共识，保证账本一致

在多数人参与者是诚实的情况下，系统是安全的

区块链的起源

- 假如区块链是一个实体账本，一个区块就相当于账本中的一页，区块中承载的信息，就是这一页上记载的交易内容。

区块链的特点

全网分布保存->防丢失

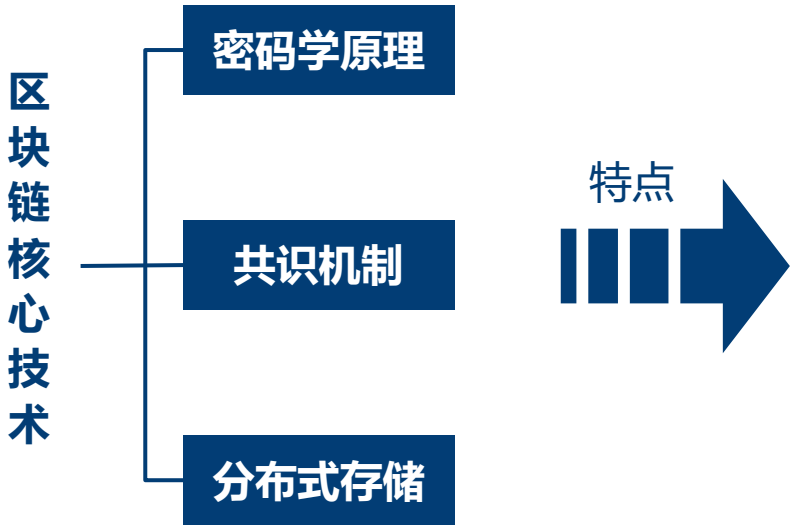
多方共识记账->防篡改

块的链式结构->易追溯

区块链的内涵

区块链的起源

- 区块链的技术特征



去中心化

分布式核算和储存，共享交易账本，共同维护

开放性

对所有节点开放，数据可随时下载提取

信息不可伪造和篡改

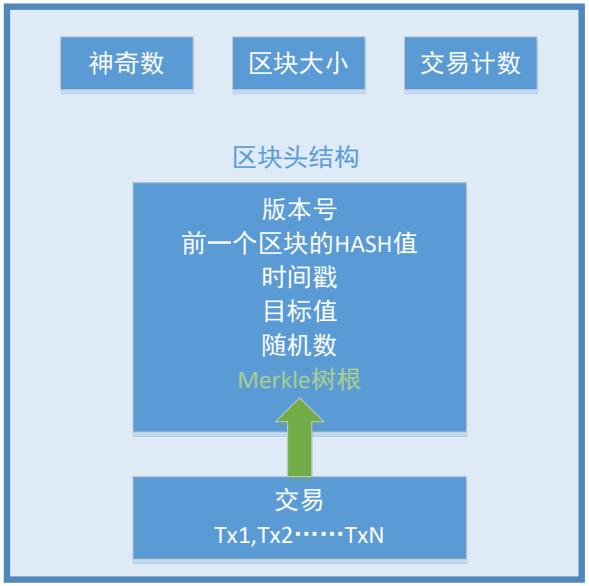
设置共识协议规范，在验证和可信条件下进行交易，信息永久地存储

自治性

交易条件和状态内嵌，不需要人为干预

区块链的起源

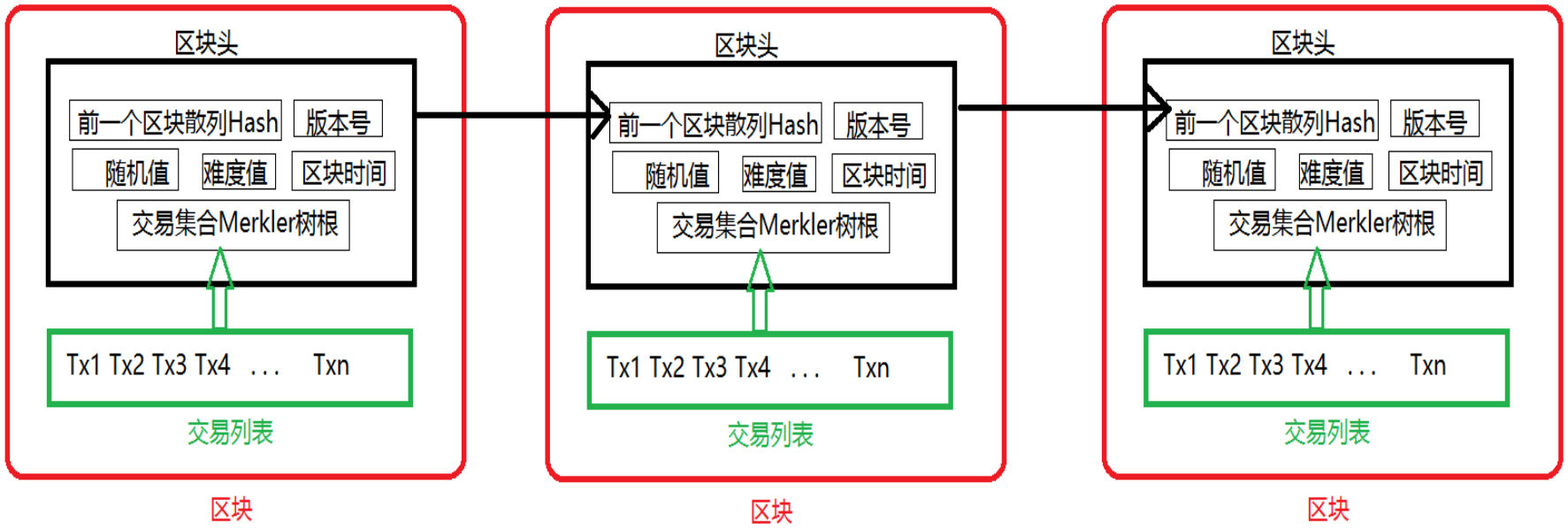
- 区块是一种记录交易的数据结构。每个区块由区块头和区块主体组成。区块头包含了除了交易相关信息以外的所有信息，区块主体负责记录前一段时间内的所有交易信息。



区块结构

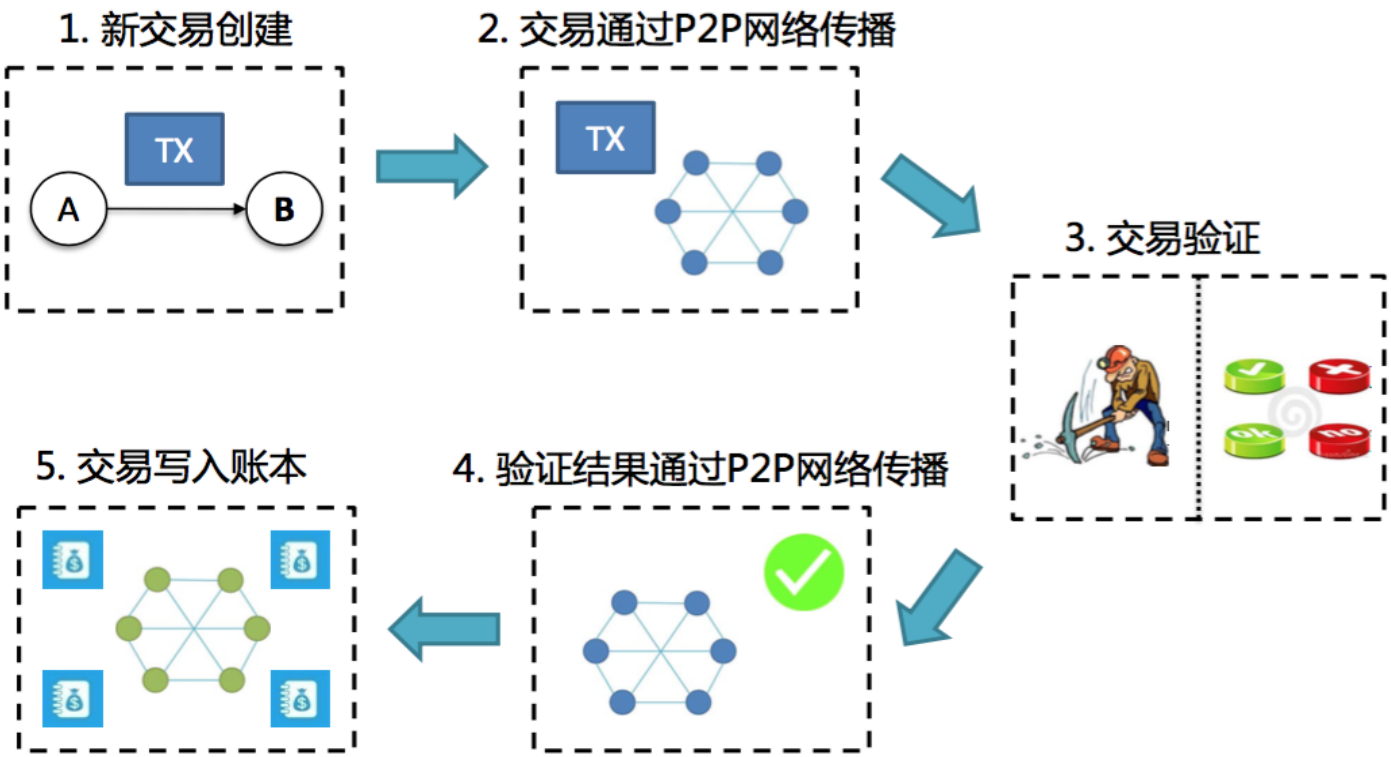
区块链的起源

- 每个区块通过包含上一区块hash值的方式，使得区块“链接”起来构成区块链。



区块链的起源

- 交易过程


1. 新交易创建
TX
A
B
2. 交易通过P2P网络传播
TX
3. 交易验证
4. 验证结果通过P2P网络传播
5. 交易写入账本

区块链的起源

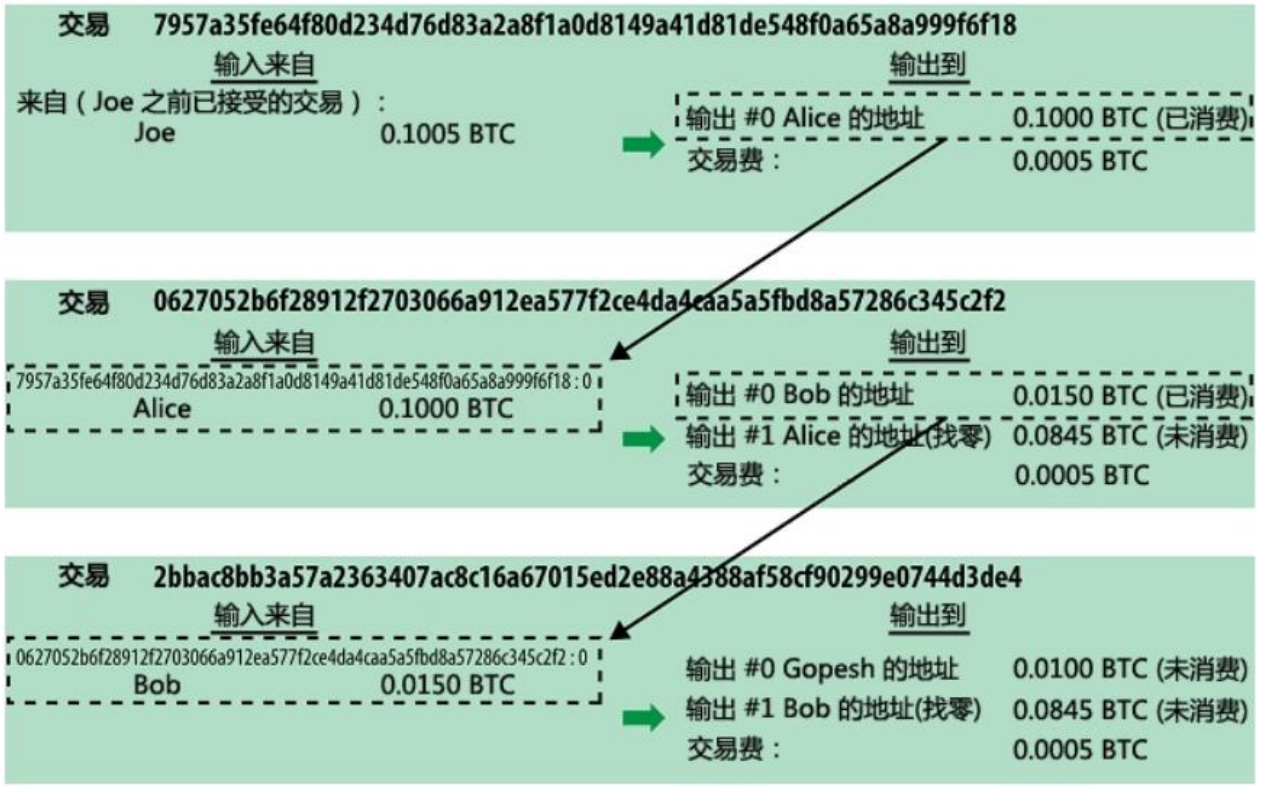
- 交易示例
  - Alice下载一个比特币客户端。
  - 客户端自动生成一个钱包，随机生成一个私钥和对应的比特币地址。
  - Alice可以分享这一个地址，这样其他人就可以通过这个地址发送比特币到Alice的钱包里。

区块链的起源

- 交易示例
  - 获取比特币，向朋友Joe现金购买。
  - 按下发送键后
  - 钱包创建一笔交易
  - Joe的私钥签名这笔交易
  - 公告比特币全网
  - 矿工通过“挖矿”，使交易包括在一个区块中
  - 将区块存储到全网公开账本（区块链）

区块链的起源

- 交易
  - 接下来Alice去Bob的咖啡店消费,在她支付Bob咖啡店的新交易中,使用了之前的交易作为输入,并以支付咖啡给Bob和找零给自己作为新的输出。
  - 这样交易在逻辑上就形成了一条链式结构,交易的输入对应之前交易的输出。



区块链的起源

- 挖矿（达成全网共识）
  - 验证交易
    - 每个全节点依据统一的标准对每个交易进行验证。验证交易通过后，将交易加入交易池中。
  - 构建区块
    - 矿工用交易池中的一组交易构建一个候选区块的主题，再构建区块头。
  - 校验并广播区块
    - 矿工成功构建一个区块后，广播该区块到邻近节点。其他节点接到区块后依据统一的标准对区块进行独立验证，验证通过后再广播该区块，并将区块组装到节点保存的区块链上。
  - 将区块组装进区块链
    - 因为区块链是去中心化的数据结构，所以不同节点间的状态会有不一致，但组装区块时，所有的节点都遵从选择“最长”的区块链，这样整个比特币网络最终会收敛到一致的状态。

区块链的起源

- POW 共识机制
  - POW（Proof Of Work），工作量证明。
  - 核心思想是通过计算能力竞争的方式来保证数据一致性从而达成共识。
  - 在比特币系统中，各节点(即矿工)基于各自的计算机算力的相互竞争来解决一个求解困难但验证容易的问题，最快解决该难题 的节点获得区块记账权，即该参与方创建了一个区块，所有其他参与方更新本地区块链。

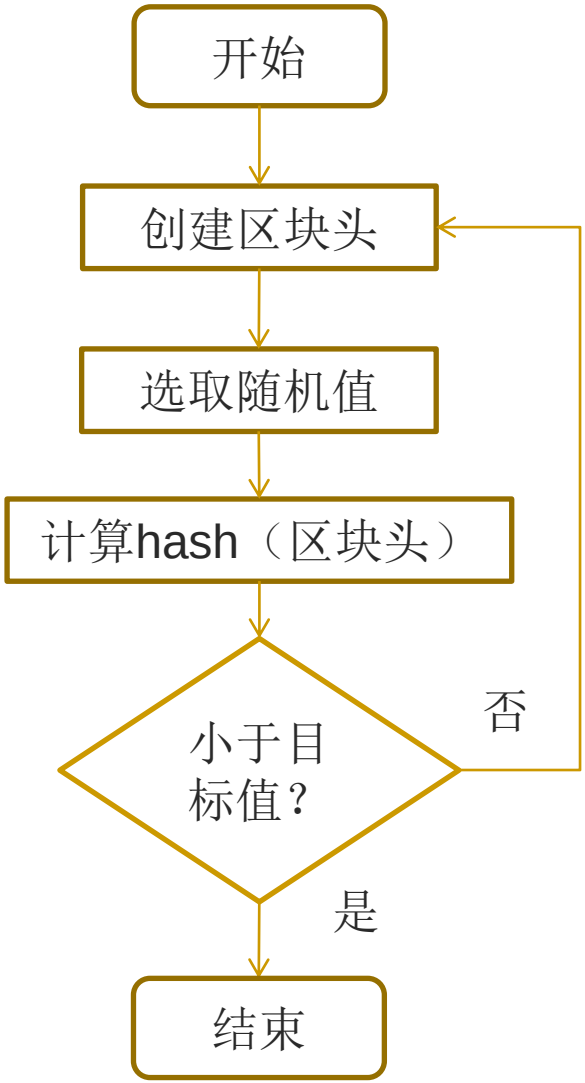
区块链的起源

- **POW过程**
  - 创建交易列表，通过Merkle Tree算法生成Merkle根Hash
  - Merkle 根Hash与其他相关字段组装成区块头，将区块头的80字节数据（Block Header）作为工作量证明的输入
  - 不停的变更区块头中的随机数即nonce的数值，并对每次变更后的的区块头做双重SHA256运算（即SHA256(SHA256(区块头))），将结果值与当前网络的目标值做对比，如果小于目标值，则解题成功，工作量证明完成。

区块链的起源

- POW过程

区块头结构

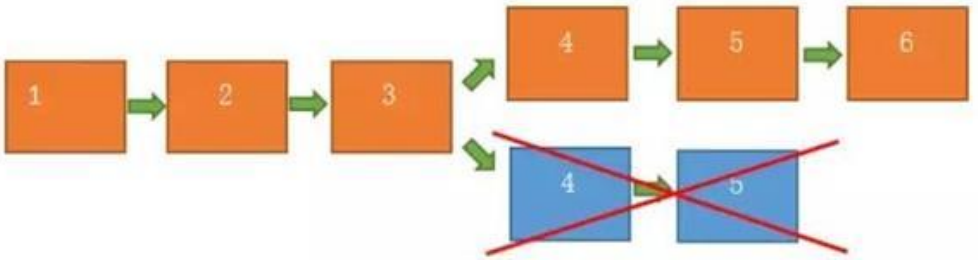


区块链的起源

- 挖矿的作用
  - 挖矿在构建区块时会创造新的比特币,类似中央银行印发新的纸币。比特币的总数是固定的,创建比特币的速度随时间下降。
  - 挖矿创建信任。挖矿确保只有在包含交易的区块上贡献了足够的计算量后,这些交易才被确认。区块越多,花费的计算量越大,数据越难篡改，意味着更多的信任。
  - 挖矿实现了在没有中心机构的情况下， 也能使整个比特币网络达成共识。

区块链的起源

- 区块分叉
  - 当有两个矿工几乎同时完成新区块的工作量证明解时，会各自进行广播，由于网络传播的时延，网络中各个节点收到两个解的次序有先后，因此会造成不同节点分别认可不同的区块，从而造成区块分叉；



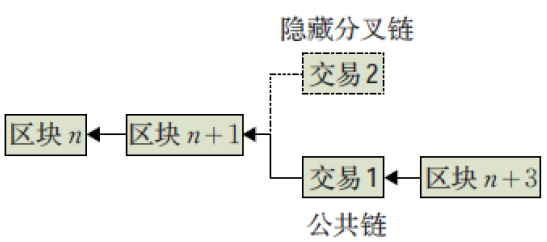
  - 所有节点都选择难度最大的链作为正确的区块链，放弃难度较小的其他链，比特币区块链系统最终会达成共识，消除分叉。

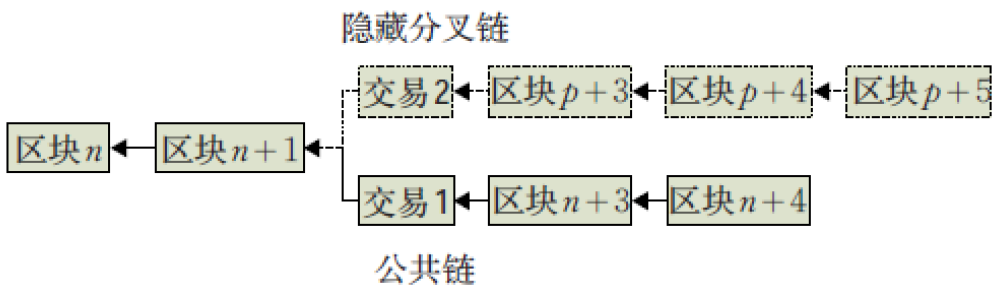
双花/双重支付攻击

- 双重花费攻击分为两种类型：
- 1. 攻击者在短时间交易场景中使用同一笔钱同时跟两个或多个商家进行交易，在交易没有被记录进合法区块链的情况下攻击者得到了货物或获得了服务，这时攻击者成功实施了双重花费攻击。
  - （1）.Race attack：一个人同时向网络中发送两笔交易，一笔交易发给自己（为了提高攻击成功率，他给这笔交易增加足够的小费），一笔交易发给商家。由于发送给自己的交易中含有较高的费，会被矿工打包成区块的概率比较高。
  - （2）.Finney attack：一个人挖到了一个区块（这个区块中包含一个交易：A向B转10BTC，其中A和B都是自己的地址），他先不广播这个区块，先找一个愿意接受未确认交易的商家向他购买一个物品，向商家发一笔交易：A向C转10BTC，付款后向网络中广播刚刚挖到的区块，由于区块中包含一个向自己付款的交易，所以他实现了一次双花

双花/双重支付攻击

- 双重花费攻击分为两种类型：
- 2. 攻击者自身拥有强大的算力，在第一笔交易完成并记录进区块链后，攻击者通过自身强大的算力，将同一笔钱产生的第二笔交易记录在自己维护的隐藏分叉链上，然后攻击者在自身维护的私有链上进行挖矿，直到维护的私有链的长度超过区块链中公共链的长度，由于区块链中的节点始终都将最长的链视为正确链。维护公共链的节点会选择在较长的隐藏分叉链上挖矿，包含第一笔交易的区块将被视为无效区块而抛弃





区块链的起源

- Pow共识算法可能面临的攻击
  - 51%算力攻击
  - 双花攻击
  - 自私挖矿攻击

- PoW共识算法并不是一个强一致性算法，而是概率性算法，经过数学推导，在落后6个区块的情况下，如果攻击者不能控制超过全网%51的算力，则攻击者几乎无法成功。

区块链的起源

- PoW共识算法的特点
  - 优点
    - 算法简单，容易实现
    - 节点间无需交换额外的信息即可达成共识
    - 破坏系统需要投入极大的成本，要有压倒大多数人的算力（51%攻击）。
  - 缺点
    - 浪费能源
    - 区块的确认时间难以缩短（10分钟左右），现在每秒交易量上限是7笔，因此性能比较低。

区块链的分类

- 公有链
  - 无官方组织及管理机构，无中心服务器，参与的节点按照系统规则自由接入网络、不受控制，节点间基于共识机制开展工作。
- 联盟链
  - 由若干机构联合发起，介于公有链和私有链之间，本质上是一个多中心化的区块链系统。
- 私有链
  - 建立在某个组织内部的许可链，其读、写和记账权限严格按照组织内部的运行规则设定。
  - 相比传统的中心化数据库，仍然具备可追溯、不可篡改、防止内部作恶的优点。

区块链技术

区块链发展阶段

2009年

区块链1.0-数字货币

加密数字货币

- 可编程货币，是与转账、汇款和数字化支付相关的密码学货币应用。
- 区块链技术还在探索的过程，并没有大型的金融区块链应用（非数字货币类）上线，尝试很多，普及尚早。

2020年

区块链2.0-智能合约

加密数字货币+智能合约

ethereum

HYPERLEDGER

- 可编程金融，是经济、市场和金融领域的区块链应用，例如股票、贷款、抵押、产品、智能财产和智能合约。
- 区块链不仅仅是技术，所以这一轮技术革命中区块链的影响要远大于其他技术，可能会有颠覆性的业务、技术或者企业出现。

2030年

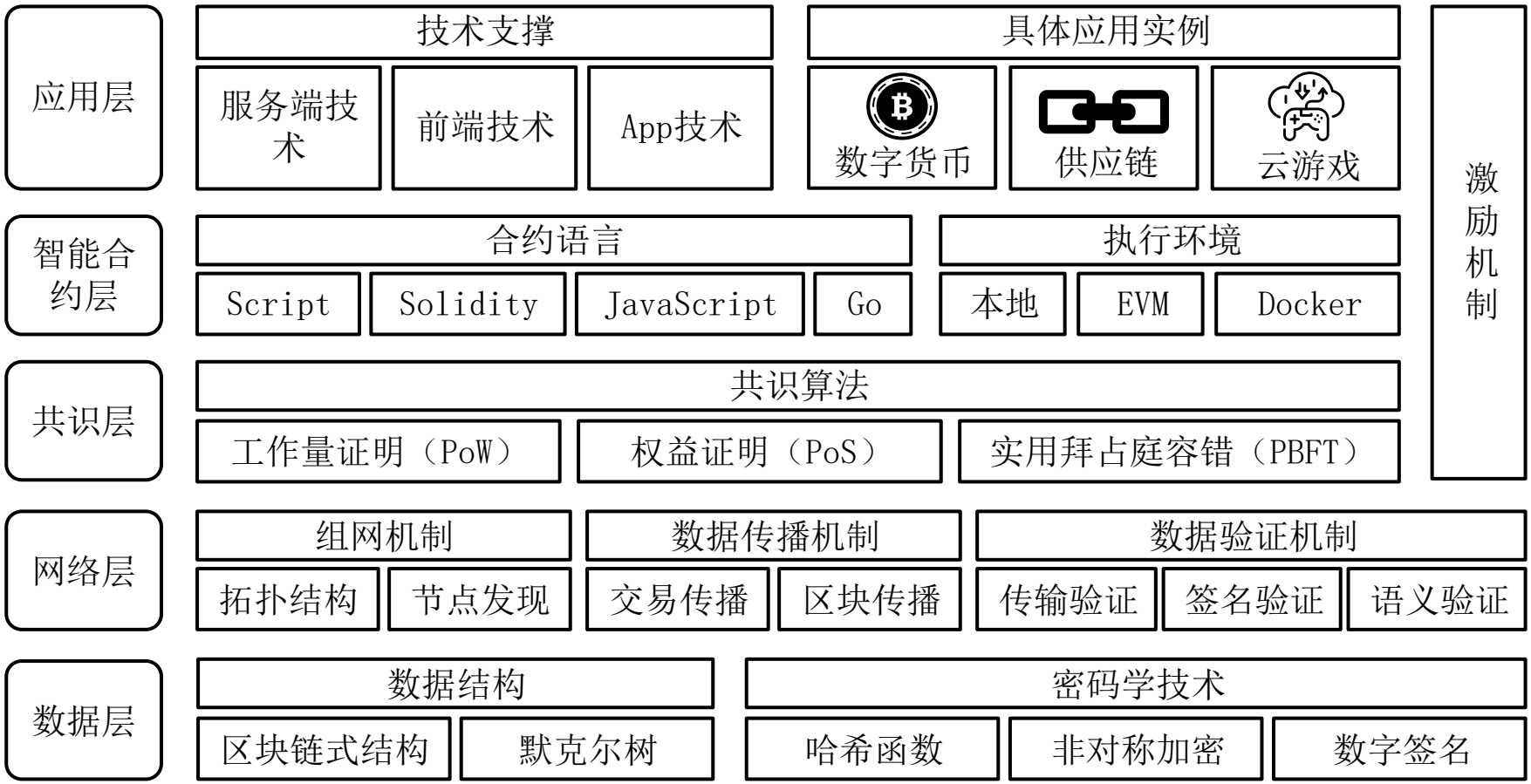
区块链3.0-去中心化互联网

成为成熟的数字经济基础设施

- 可编程社会，是超越货币、金融和市场的应用，特别是在政府、健康、科学、文化和艺术领域的应用。
- 应用生态决定最后的赢家。目前公链和私链（或联盟链）都有一些金融应用，但还不成气候，应该胜负未分，这里面大公司不一定有优势，开源力量不可小觑。

区块链技术体系结构

区块链技术体系结构



数据层

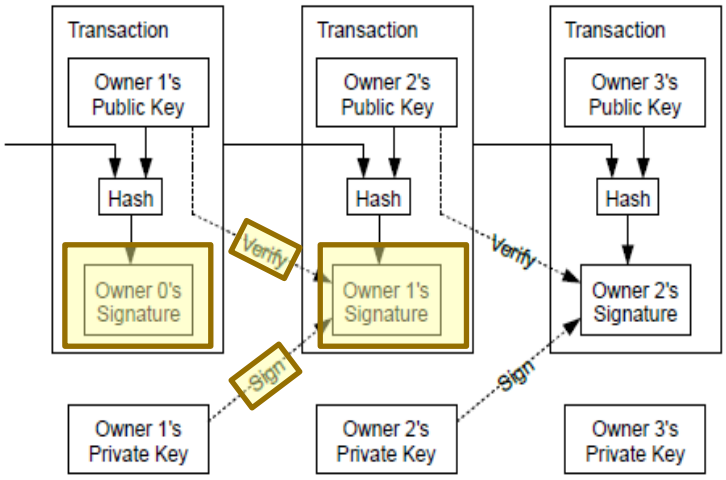
- 区块链数据层定义了各节点中数据的联系和组织方式，利用多种算法和联性和验证的高效性，从而使区块链具备实用的数据防篡改特性。
  - 信息模型
  - 关联验证结构
  - 加密机制

数据层

- 区块链承载了不同应用的数据，信息模型是指节点记录应用信息的逻辑结构。
  - UTXO（unspent transaction output）
  - 基于账户
  - 键值对模型

数据层

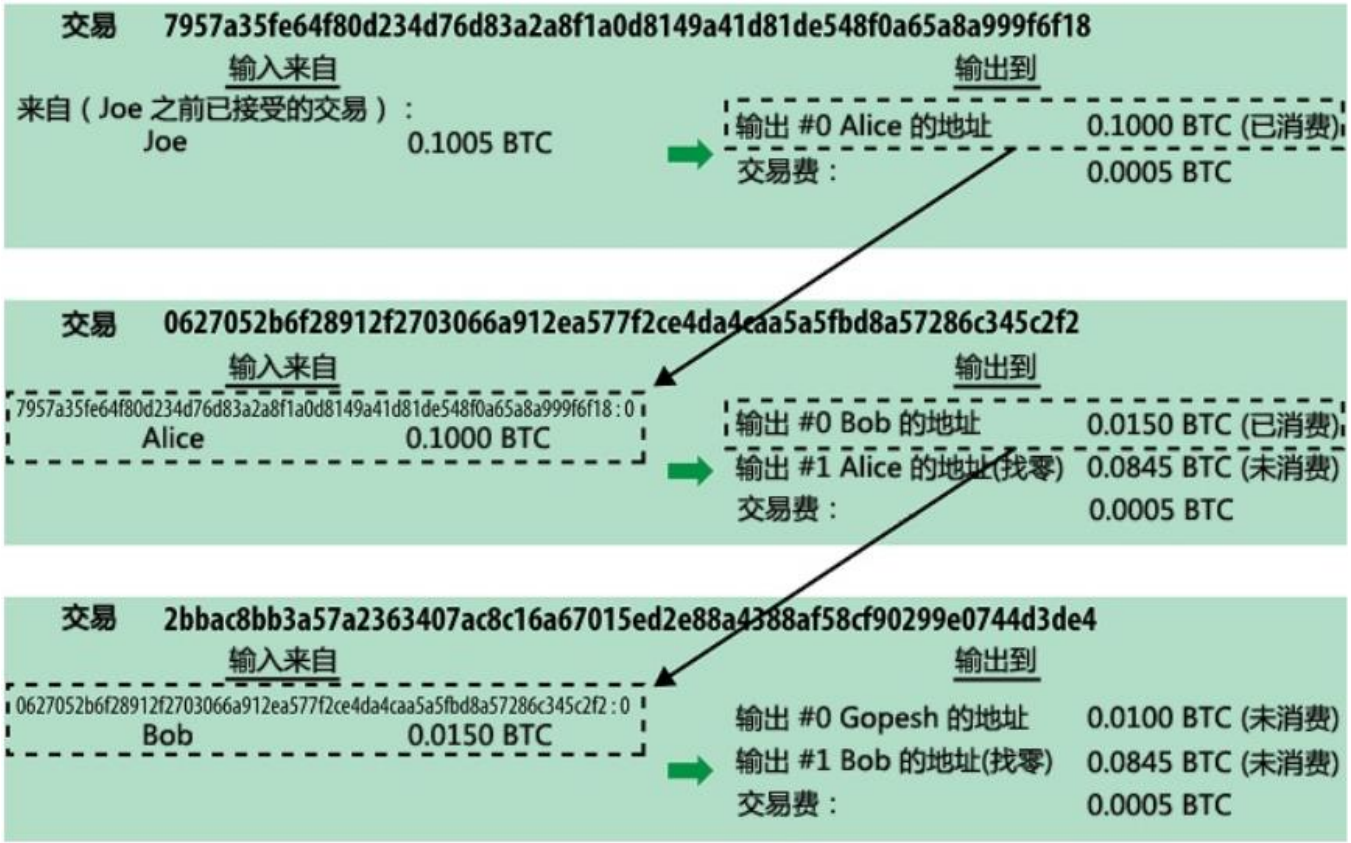
- UTXO
  - UTXO 是比特币交易中的核心概念，逐渐演变为区块链在金融领域应用的主要信息模型。
  - 每笔交易（Tx）由输入数据（Input）和输出数据（Output）组成；
  - 输出数据为交易金额（Num）和用户公钥地址（Adr）；
  - 输入数据为上一笔交易输出数据的指针（Pointer）。



txid : 001	
in	txid : 000
	vout : 0
	scriptSig : <sig><PubK(A)> from (<tx001> + <PriK(A)>)
out	value : 1
	n : 0
	scriptPubKey : OP_DUP OP_HASH160 <PubKHash(B)> OP_EQUALVERIFY OP_CHECKSIG

数据层

- UTXO实例


交易 7957a35fe64f80d234d76d83a2a8f1a0d8149a41d81de548f0a65a8a999f6f18
输入来自
来自（Joe 之前已接受的交易）：
Joe 0.1005 BTC
输出到
输出 #0 Alice 的地址 0.1000 BTC (已消费)
交易费： 0.0005 BTC
交易 0627052b6f28912f2703066a912ea577f2ce4da4caa5a5fbd8a57286c345c2f2
输入来自
7957a35fe64f80d234d76d83a2a8f1a0d8149a41d81de548f0a65a8a999f6f18 : 0
Alice 0.1000 BTC
输出到
输出 #0 Bob 的地址 0.0150 BTC (已消费)
输出 #1 Alice 的地址(找零) 0.0845 BTC (未消费)
交易费： 0.0005 BTC
交易 2bbac8bb3a57a2363407ac8c16a67015ed2e88a4388af58cf90299e0744d3de4
输入来自
0627052b6f28912f2703066a912ea577f2ce4da4caa5a5fbd8a57286c345c2f2 : 0
Bob 0.0150 BTC
输出到
输出 #0 Gopesh 的地址 0.0100 BTC (未消费)
输出 #1 Bob 的地址(找零) 0.0845 BTC (未消费)
交易费： 0.0005 BTC

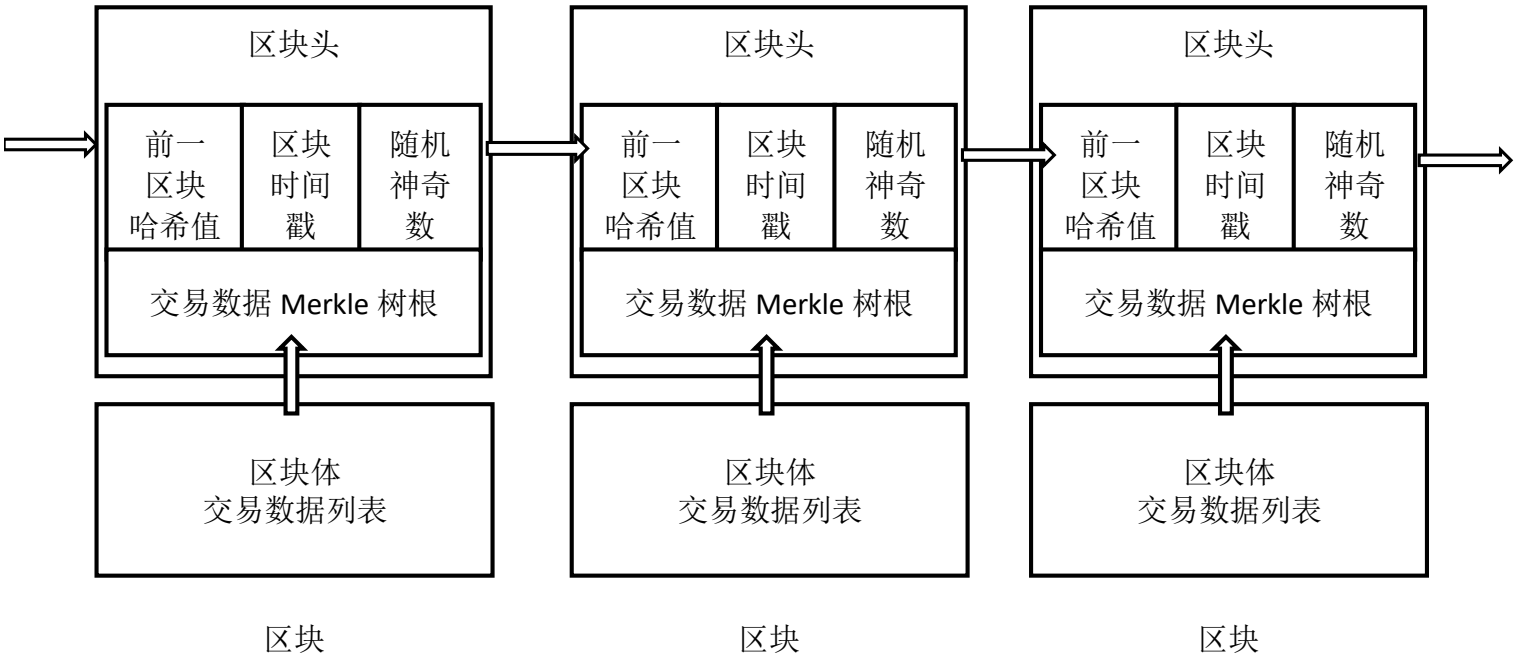
数据层

- 基于账户的信息模型以键值对的形式存储数据，维护着账户当前的有效余额，通过执行交易来不断更新账户数据。基于账户的信息模型与银行的储蓄账户类似，更直观和高效。

- 键值对模型可直接用于存储业务数据，表现为表单或集合形式。

数据层

- 关联验证结构
  - 区块链之所以具备防篡改特性，得益于链状数据结构的强关联性。该结构确定了数据之间的绑定关系，当某个数据被篡改时，该关系将会遭到破坏。由于伪造这种关系的代价是极高的，相反检验该关系的工作量很小，因此篡改成功率被降至极低。
  - 链状结构的基本数据单位是“区块（block）”



区块链中的密码技术

- 哈希函数
- Merkle树
- 非对称密码算法(公钥密码算法)
- 数字签名

区块链中的密码技术

哈希函数（Hash Function）也称杂凑函数或者散列函数，可以在有限且合理的时间内，将任意长度的二进制字符串映射为固定长度的二进制字符串，其输出值称为哈希值或者数字摘 要。一般而言，哈希函数的数学表达形式如下:

$$h=H(m)$$

其中m表示任意长度的输入，H表示哈希函数的具体实现，h则表示固定长度的输出哈希值， 常见的哈希函数包括MD5, SHA-1, SHA-2系列等，其中前两者已被理论破解。

例如：字符串abc的MD5哈希值为　　0bee89b07a248e27c83fc3d5951213c1
　　字符串abC的MD5哈希值为　　2217c53a2f88ebadd9b3c1a79cde2638

正向快速　**逆向困难**　**输入敏感**

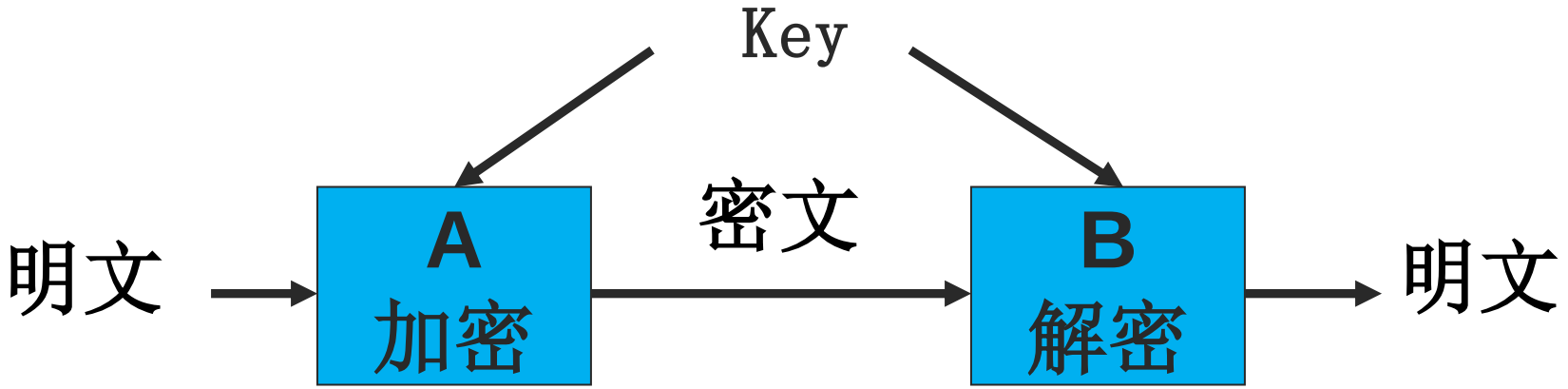
长度固定　**易压缩**　**抗碰撞**

区块链中的密码技术

- 在区块链中，哈希函数主要用于数据完整性、数据加密、共识计算的工作量证明、区块之间链接等。
- 区块链采用了双SHA256、RIPEMDl60等哈希函数，SHA256主要用于加密交易形成区块，RIPEMDl60则用于生成比特币的地址。
- 区块链系统中的双SHA256函数是将不同长度的消息经过两次SHA256计算处理，输出256位二进制字符串统一存储。

区块链中的密码技术

- 对称密码机制
  - 用于加密和解密的密钥是一样的或相互容易推出。
  - 密码算法应该达到高度扩散和高度混淆的作用。
  - 典型算法
    - DES(Data Encryption Standard—数据加密标准)
    - AES (Advanced Encryption Standard-高级加密标准)

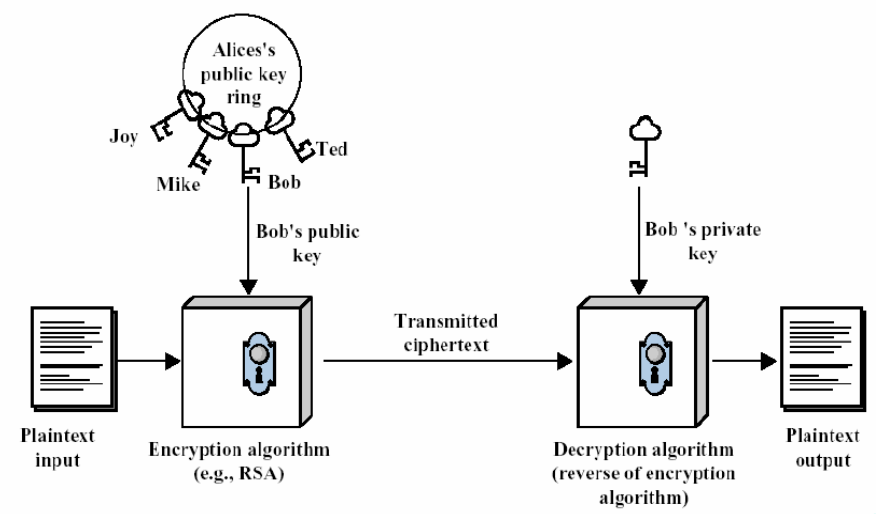


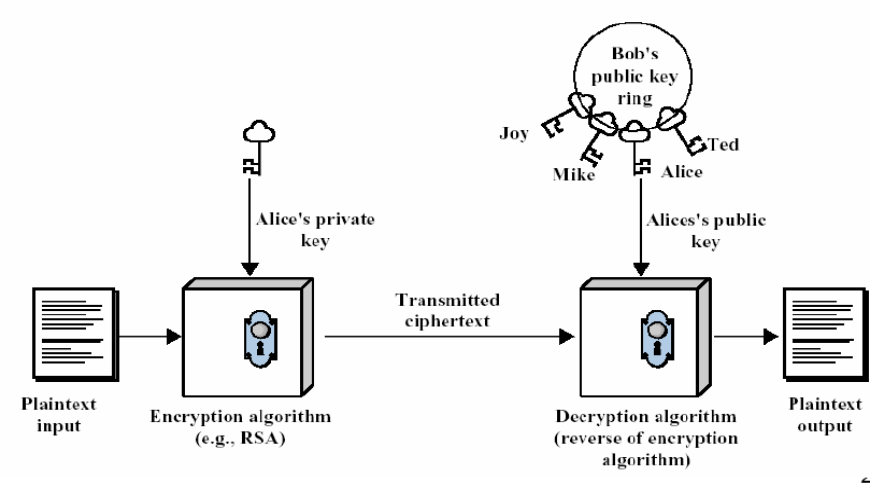
区块链中的密码技术

- 非对称密码/公钥密码机制
  - 非对称加密算法需要两个密钥：公开秘钥（简称公钥）和私有密钥（简称私钥）。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，反之一样。
  - 加密秘钥 ≠ 解密秘钥
  - 典型算法
    - RSA算法
    - Diffie-Hellman 密钥交换算法
    - ElGamal加密算法
    - 椭圆曲线密码算法

区块链中的密码技术

- 公钥加密机制



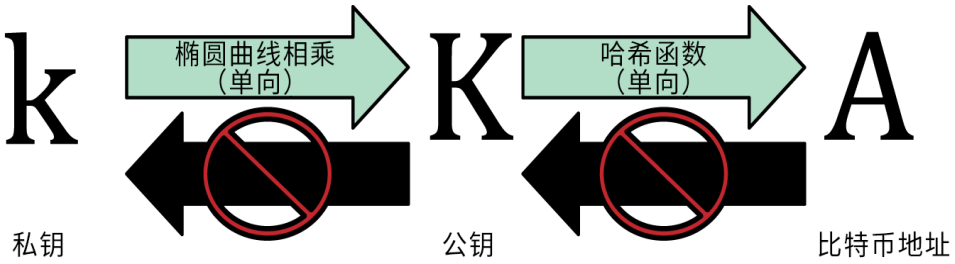


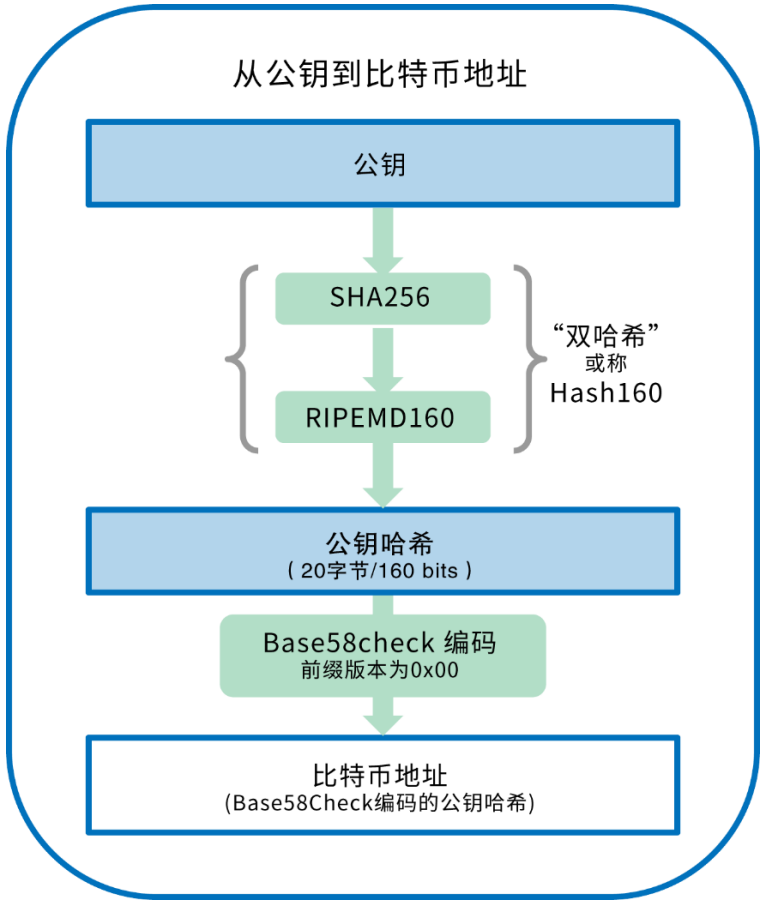
区块链中的密码技术

- 公钥密码算法
  - 由于区块链完全无可信中心，不存在PKI等可信第三方。在区块链上，私钥由用户自己选取，并产生相应的公钥，该公钥对应记录在区块的地址上。
  - 用户使用公钥对消息进行加密，只有对应的私钥才能解密。同时，私钥可用于对自己的交易信息进行数字签名，而别的用户可利用对应公钥对消息的签名进行验证。

区块链中的密码技术

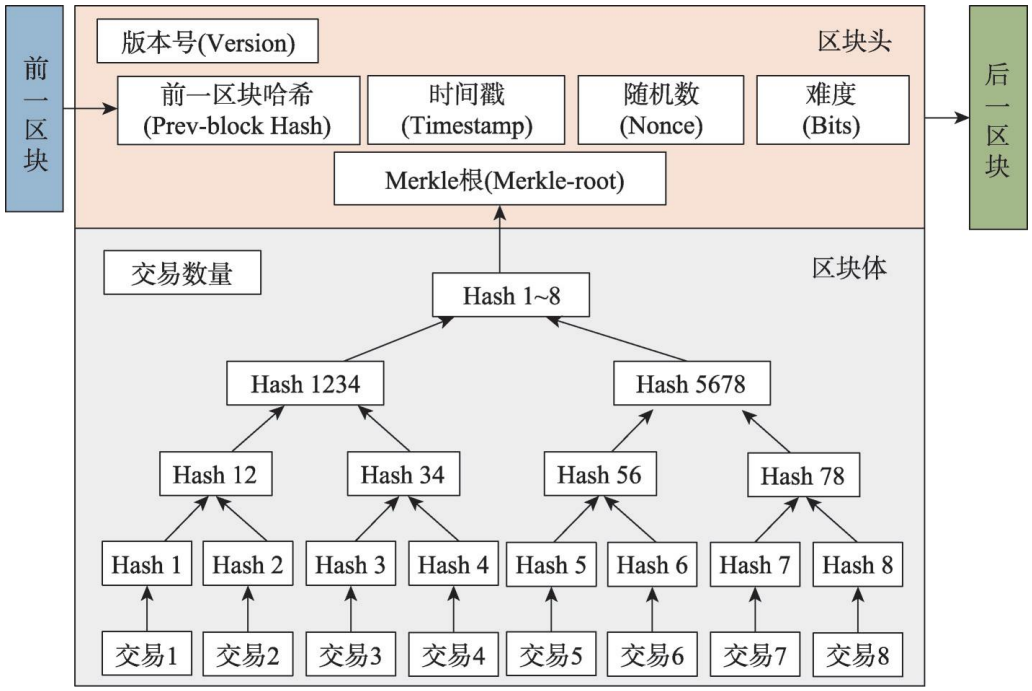
- 私钥、公钥和比特币地址之间的关系





区块链中的密码技术

- Merkle树
  - 在区块链的数据区块中数据结构主要为二叉Merkle树，每个交易记录对应于一个哈希值，并对应于Merkle树的叶子节点，两个叶子节点再次两两配对哈希计算，通过递归的方式直到最后一个哈希值作为Merkle树根存入区块体。

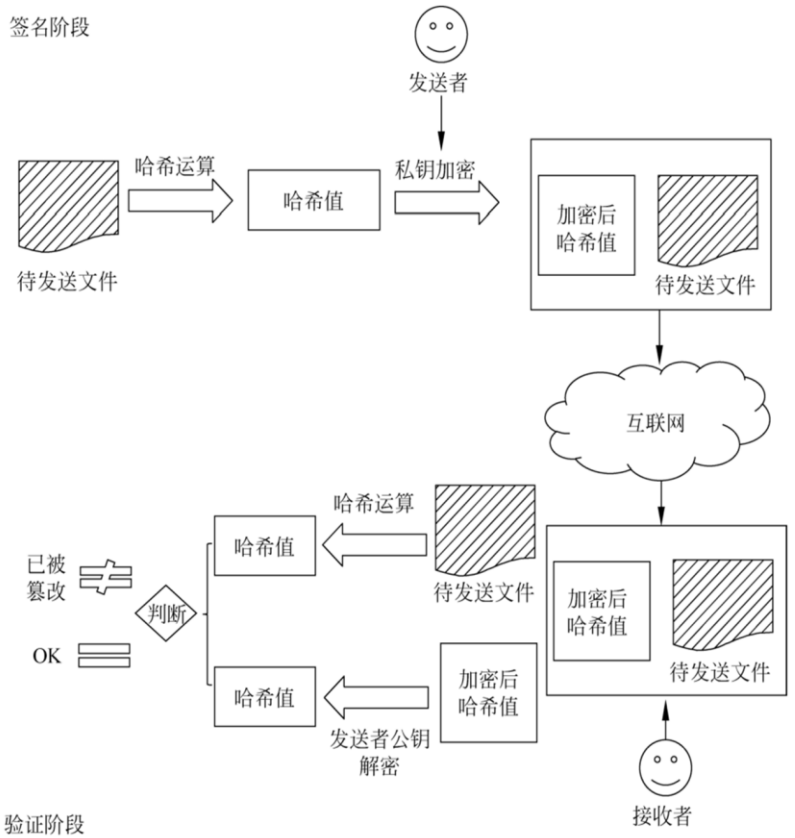


区块链中的密码技术

- Merkle树
  - 基于Merkle树，中本聪在比特币中提出简化支付验证（Simplified Payment Verification,SPV）。
  - 用户不用保存区块链的全部节点数据，仅需保存所有的区块头就可以进行支付验证，此时的节点称为轻节点。
  - 当用户要验证某笔交易，只需要向其他节点请求从交易所在叶节点到Merkle树根路径上的所有节点信息即可。

区块链中的密码技术

- 数字签名
  - 签名者对消息进行处理，生成别人无法伪造的一段数字串，这段数字串同时也是对消息的签名者发送消息真实性的一个有效证明。
  - 利用数字签名技术，能够确保消息传输的完整性、发送者的身份认证，防止交易中的抵赖发生。



网络层

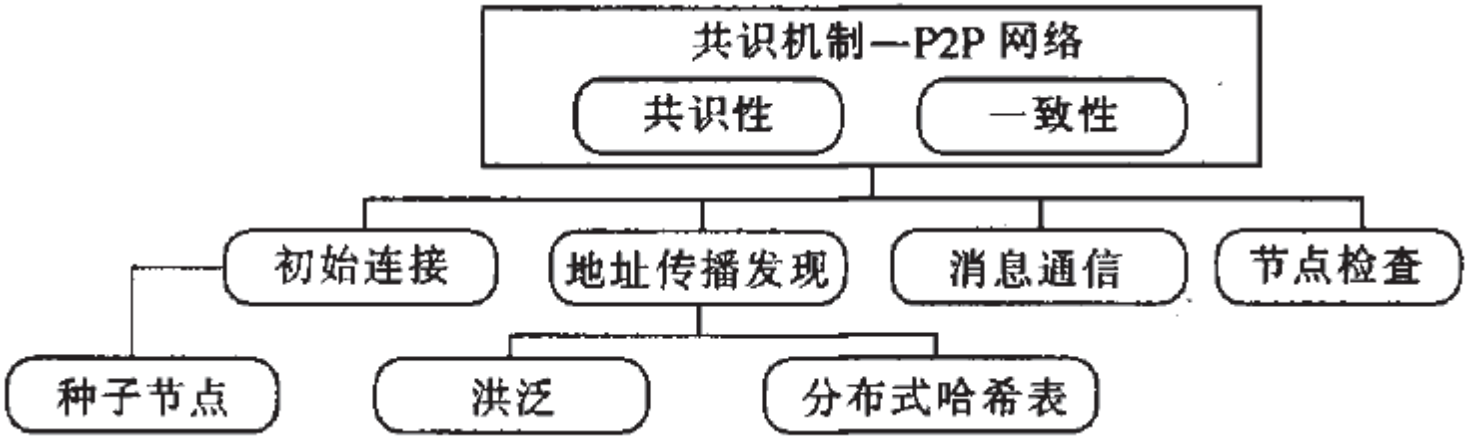
- 区块链网络层
  - 区块链网络层采用不受任何权威节点控制或层次模型约束的完全去中心化的P2P（对等网）组网方式。
  - 可以分为三个层次
    - 组网结构描述节点间的路由和拓扑关系
    - 通信机制用于实现节点间的信息交互
    - 安全机制涵盖对端安全和传输安全。

网络层

- 传统的因特网应用采用客户-服务器模式：
  - 所有内容与服务在服务器上，客户向服务器请求内容或服务，客户自己的资源不共享；
  - 这种集中式结构面临服务器负载过重、拒绝服务攻击、网络带宽限制等难以解决的问题。
- 对等网络：
  - 每个节点都有一些资源（处理能力、存储空间、网络带宽、内容等）可以提供给其它节点；
  - 节点之间直接共享资源，不需要服务器的参与；
  - 所有节点地位相等，具备客户和服务器双重特性；
  - 可缓解集中式结构的问题，充分利用终端的丰富资源。

网络层

- 组网结构——P2P网络在区块链中的架构

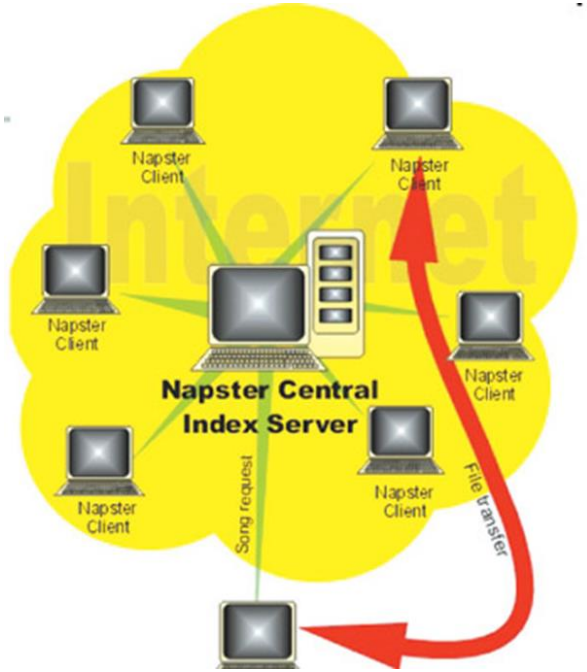


网络层

- P2P网络拓扑结构
  - 中心化拓扑
  - 全分布式非结构化拓扑/无结构对等网络
  - 全分布式结构化拓扑/结构化对等网络
  - 混合结构拓扑

网络层

- 中心化拓扑对等网络
  - 也称集中目录式结构或非纯粹的P2P结构。
  - 优点：
    - 维护简单，资源发现效率高
  - 缺点：
    - 单点故障
    - 扩展性差
  - 典型案例
    - Napster
    - QQ



网络层

- 无结构对等网络
  - 是指网络中不存在特殊中继节点、节点路由表的生成无确定规律、网络拓扑呈现随机图状的一类对等网络。
  - 该类网络结构松散，设计简洁，具有良好的容错性和匿名性，但由于采用洪泛机制作为信息传播方式，其可扩展性较差。
  - 典型案例：比特币网络（公有链）

网络层

- 比特币的网络层实现
  - 采用非结构化方式组网，路由表呈现随机性。
  - 节点间则采用多点传播方式，早起基于Gossip协议，现在使用Diffusion协议。
  - 新节点入网时，首先向硬编码DNS 节点（种子节点）请求初始节点列表；然后向初始节点随机请求它们路由表中的节点信息，以此生成自己的路由表；最后节点通过控制协议与这些节点建立连接，并根据信息交互的频率更新路由表中节点时间戳，从而保证路由表中的节点都是活动的。

网络层

- 比特币的DNS种子是写在源代码里的，这些节点在初始启动时提供最初接入网络的入口节点。新节点通过这些稳定节点作为中介连接其他节点，并且可以持续获取区块链网络节点地址列表，所以这些节点也称之为种子节点。

```
seed.bitcoin.sipa.be
dnsseed.bluematt.me
dnsseed.bitcoin.dashjr.org
seed.bitcoinstats.com
seed.bitcoin.jonasschnelli.ch
seed.btc.petertodd.org
```

网络层

- 结构化对等网络
  - 是指网络中不存在特殊中继节点、节点间根据特定算法生成路由表、网络拓扑具有严格规律的一类对等网络。
  - 该类网络实现复杂但可扩展性良好，通过结构化寻址可以精确定位节点从而实现多样化功能。
  - 常见的结构化网络以DHT（distributed hash table）网络为主。
  - 典型案例：以太坊区块链网络（公有链）

网络层

- 以太坊的网络层实现
  - 将节点公钥作为标识，采用Kademlia 算法计算节点的异或距离，从而实现结构化组网。
  - 新节点加入时首先向硬编码引导节点（bootstrap node）发送入网请求；然后引导节点根据Kademlia算法计算与新节点逻辑距离最近的节点列表并返回；最后新节点向列表中节点发出握手请求，与这些节点建立连接并保持。

网络层

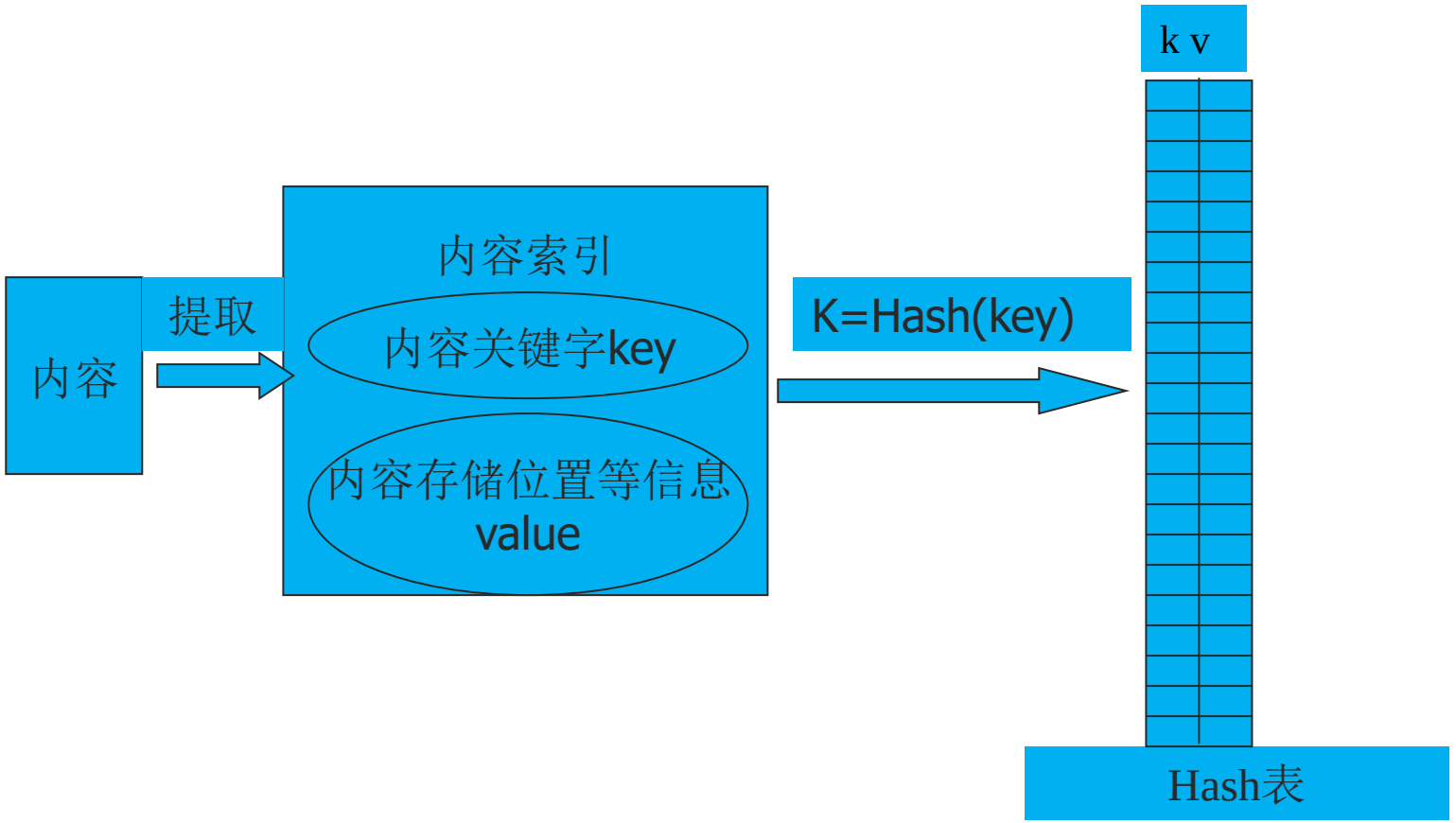
- 分布式哈希表DHT网络
  - 在结构化对等网络中，内容一般使用内容索引来表示,内容索引包括key和value两部分,其中key是内容的关键字,value是存放内容的实际位置,因此内容索引也表示为<key, value>对。
  - 将内容索引抽象为<K, V>对
    - K是内容关键字的Hash摘要：K = Hash(key)
    - V是存放内容的实际位置，例如节点IP地址等

网络层

- 分布式哈希表DHT网络
  - 所有的<K, V>对组成一张大的Hash表，因此该表存储了所有内容的信息；
  - 每个节点都随机生成一个标识(ID)，把Hash表分割成许多小块，按特定规则(即K和节点ID之间的映射关系)分布到网络中去，节点按这个规则在应用层上形成一个结构化的重叠网络；
  - 给定查询内容的K值，可以根据K和节点ID之间的映射关系在重叠网络上找到相应的V值，从而获得存储文件的节点IP地址。

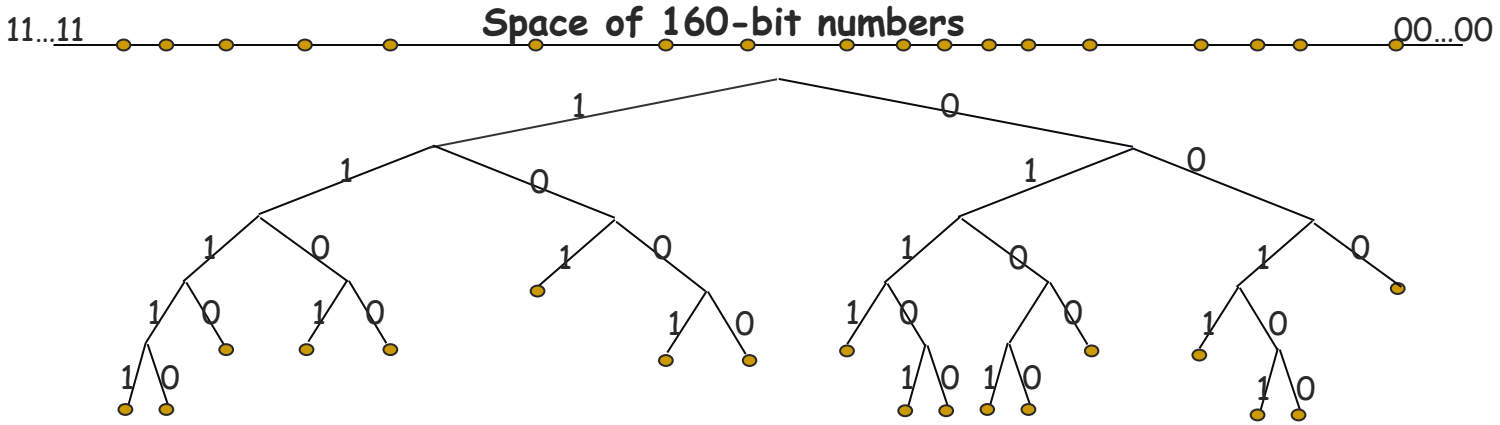
网络层

- 分布式哈希表DHT网络



网络层

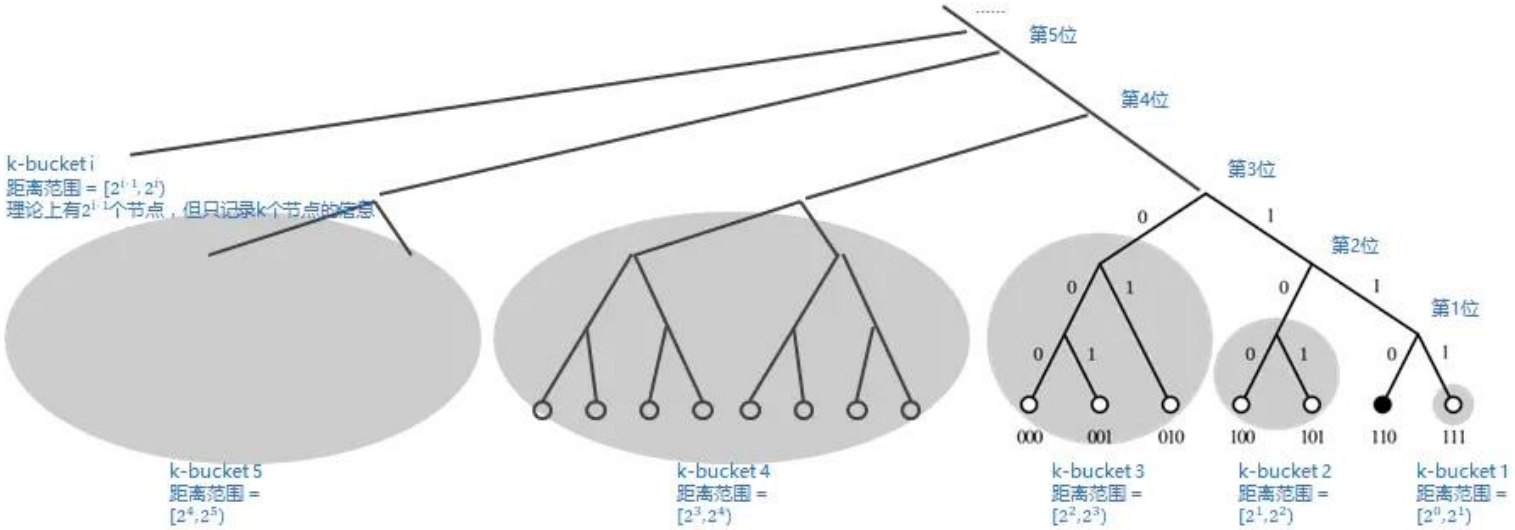
- 以太坊KAD算法
  - 每个节点都有一个160bit 的ID 值，<key,value>对的数据就存放在ID 值最接近key 值的节点上。
  - 判断两个节点x,y 的距离远近是基于数学上的异或运算 d(x,y) = x ⊕ y
    - d(010101b, 010110b) = 000011b = 3



网络层

- 以太坊KAD算法
  - 对每一个0≦i≦160，每个节点都保存有一些和自己距离范围在区间[2^i,2^{i+1}]内的一些节点信息。
  - 每一个这样的列表都称之为一个K 桶，并且每个K桶内部信息存放位置是根据上次看到的时间顺序排列，时间最长的放在头部，最新的放在尾部。每个桶都有不超过k 个的数据项（例如k=20）。
  - K 桶存放的是有相同ID 前缀的节点信息，每个K桶都覆盖了ID 空间的一部分，全部K 桶的信息加起来就覆盖了整个160bit的ID 空间，而且没有重叠。

I	距离	邻居
0	$[2^0,2^1)$	(IP address,UDP port,Node ID)$_{0\text{-}1}$ ..... (IP address,UDP port,Node ID)$_{0\text{-}k}$
1	$[2^1,2^2)$	(IP address,UDP port,Node ID)$_{1\text{-}1}$ ..... (IP address,UDP port,Node ID)$_{1\text{-}k}$
2	$[2^2,2^3)$	(IP address,UDP port,Node ID)$_{2\text{-}1}$ ..... (IP address,UDP port,Node ID)$_{2\text{-}k}$
.....		
i	$[2^i,2^{i+1})$	(IP address,UDP port,Node ID)$_{i\text{-}1}$ ..... (IP address,UDP port,Node ID)$_{i\text{-}k}$
.....		
160	$[2^{160},2^{161})$	(IP address,UDP port,Node ID)$_{160\text{-}1}$ ..... (IP address,UDP port,Node ID)$_{160\text{-}k}$



网络层

- 以太坊KAD算法
  - 假如节点x要查找ID值为t 的节点：
    - 计算到t 的距离： d(x,t) = x⊕t
    - 从 x 的第[log d]个K 桶中取出α 个节点的信息，同时进行查找节点操作。如果这个K 桶中的信息少于α 个，则从附近多个桶中选择距离最接近d 的总共α个节点。
    - 对接受到查询操作的每个节点，如果发现自己就是t，则回答自己是最接近t 的；否则测量自己和t 的距离，并从自己对应的K 桶中选择α 个节点的信息给x。
    - X 对新接受到的每个节点都再次执行查找节点操作，此过程不断重复执行，直到每一个分支都有节点响应自己是最接近t 的。

网络层

- 混合式对等网络
  - 是指节点通过分布式中继节点（超级节点）实现全网消息路由的一类对等网络。
  - 每个中继节点维护部分网络节点地址、文件索引等工作，共同实现数据中继的功能。
  - 是一种层次式结构：
    - 超级节点之间构成一个高速转发层
    - 超级节点和所负责的普通节点构成若干层次。
  - 典型案例：超级账本区块链网络（联盟链）

网络层

- 超级账本Fabric的网络层实现
  - 以组织为单位构建节点集群，采用混合式对等网络组网；
  - 每个组织中包括普通节点和锚节点（anchor peer），普通节点完成组织内的消息路由，锚节点负责跨组织的节点发现与消息路由。
  - Fabric 网络传播层基于Gossip 实现，需要使用配置文件初始化网络，网络生成后各节点将定期广播存活信息，其余节点根据该信息更新路由表以保持连接。

网络层

- Gossip协议的基本思想
  - 关于所有节点的表，称为全局视图（total view）。每个节点维护一个部分视图（partial view），含有c个邻接点的列表。
    - 表项：=<IP，age>
  - 节点之间定期交换表项。由主动线程（可主动发起通信）和被动线程完成。
  - 节点的加入
    - 与任意一个已知的节点进行视图交换
  - 节点的删除
    - 可自行离开，无需通知其他节点。
    - 当其他节点发现某节点P不再响应时，将其从表中删除。