# **第2章 密码学基础与共识机制**

区块链系统的核心创新之一，是在无需中心机构的前提下构建起可验证、可追溯、难以篡改的共识机制。而实现这一目标的前提，是一整套由密码学构造支撑的安全通信、数据完整性保障与身份验证体系。本章将从基础的密码学原理讲起，逐步深入到区块链系统中的交易验证结构与分布式共识算法[[1]](#footnote-0)。

## **2.1 非对称加密、哈希函数与Merkle树**

在进入具体的技术细节之前，我们需要先明确一个基本问题，也就是密码学为何以及如何在区块链中发挥作用？尽管许多加密方法早已在传统计算机安全中得到广泛应用，但在区块链环境下，它们承担的角色不仅仅是“加密”或“隐藏”，而更多地关乎“公开却安全”、“无需信任但可验证”的机制设计。

### 2.1.1 密码学在区块链中的角色定位

在传统认知中，密码学（cryptography）常常与“加密”或“保密”紧密联系，被视为信息安全的技术屏障。然而，在区块链系统中，密码学的作用远超保密功能，其核心地位在于构建了公开环境下的可信机制，支撑无需信任第三方的制度性保障。正如密码学家Bruce Schneier所言：“密码学不是保密术，而是制度保障（Cryptography is not about secrecy, it’s about control）。”这句话深刻地指出了密码学在分布式系统中的独特地位，它不再仅仅保护数据不被泄露，更关键的是用算法规则替代人为信任，在无主权、去中心的网络中维护秩序与信任。

在区块链系统中，密码学的核心功能可归纳为三大类，一是数据加密与机密性保护（Confidentiality），二是身份验证与签名确认（Authentication），三是不可篡改性与完整性保证（Integrity & Immutability）。

#### （1）数据加密是区块链中可选择的隐私保护机制

尽管比特币、以太坊等主流公链采用的是透明账本机制，交易数据默认公开，任何人都可以查询历史记录，但在某些应用场景中，数据保密性仍然具有重要价值。例如企业间贸易中的敏感交易记录，用户隐私数据（如KYC信息、身份认证文档），与链下世界连接的RWA（Real World Asset）资产信息。在这些场景中，对称加密与非对称加密仍然作为数据加密手段被用于保障内容不被泄露。

加密手段可分为对称加密和非对称加密，以及混合加密。对称加密（Symmetric Encryption）比如AES算法，用于数据存储和快速加解密场景；非对称加密（Asymmetric Encryption）比如RSA与ECC，用于加密通信或加密某些关键配置参数；混合加密机制结合非对称与对称加密优势，先用非对称加密安全交换临时对称密钥，再用对称加密高效处理实际数据，既解决对称密钥传输风险，又规避非对称加密效率低的问题，平衡安全与实用。

需要强调的是，区块链的“默认公开”并非否定加密的价值，而是在基础层设计中优先保障可验证性与透明性。实际使用中，开发者可以根据应用需求决定数据是否加密，甚至在应用层实现可选隐私保护（如zk-SNARK、Ring Signature等）。因此，数据加密在区块链中的角色并非基础性制度保障，而是应用层的可选隐私配置，服务于对保密有需求的特定场景。

#### （2）身份验证

身份验证是区块链系统的核心机制之一，但是区块链系统中的身份认证是基于非对称密码学实现的“去信任身份认证”。

在传统互联网中，身份的确认依赖中心化机构，例如网站验证用户身份依赖用户名密码 + 第三方认证（OAuth、验证码等）；金融系统中用户身份依赖KYC机构、银行账户体系；法律上“签字即承担责任”依赖公证机关、政府认证。

而在区块链系统中，这一切被一种根本性机制替代，这就是“私钥即身份，签名即授权”机制。具体而言，私钥（Private Key）是用户唯一拥有的，代表其控制权；公钥（Public Key）可公开分享，用于验证身份；而签名（Signature）则是对消息（如交易）进行加密，证明其确实由私钥持有者授权。

这种机制的制度意义在于四个方面。一是无需注册，用户无需申请账户或获得许可，只需生成一对密钥即可获得系统身份；二是不可伪造，除非获取了对方私钥，否则任何人都无法冒充签名；三是可验证性强，所有签名都可以通过公钥被快速验证，保障了系统运行效率；四是权限明晰，账户中通过私钥控制的资金或数据所有权，权限分配逻辑清晰。

在系统运行过程中，签名不仅用于转账交易，还广泛应用于智能合约的调用授权、DAO治理中的投票签名、多签账户（Multisig）中的联合控制，以及离链数据的验证（如链下预言机、零知识证明的验证者签名）等。

由此可见，非对称签名机制是区块链系统中身份验证与权限控制的制度支柱。这不仅是一种技术实现，更是一种新的信任范式，即由算法取代制度，由密钥取代机构，签名即获取信任。

#### （3）不可篡改性

区块链“不可篡改”的神话背后，其核心机制是哈希函数（Hash Function）与链式结构的组合。哈希函数具有三个特性，分别是单向性、抗碰撞性和雪崩效应。单向性（One-wayness）是指难以从哈希值反推出原始数据；抗碰撞性（Collision Resistance）是指难以找到两个不同的输入，哈希结果却相同；雪崩效应（Avalanche Effect）则是指输入微小变动，输出变化剧烈。

区块链正是利用这些特性实现了数据不可篡改。其核心做法，是每个区块都包含上一个区块的哈希值，形成“链”；区块内容一旦更改，其哈希值也随之变化，进而影响整个后续区块；共识机制确保大多数节点认可最长合法链，因此攻击者必须同时修改多个区块且重新计算工作量/权益量，成本极高。

此外，在区块内部，Merkle树结构将所有交易组织为树状结构，根节点（Merkle Root）写入区块头中，以此确保交易数据的完整性和可验证，且仅需提供少量哈希路径即可证明某一笔交易是否包含在区块中（即轻节点验证机制）。

哈希链机制带来的不只是技术上的数据完整性，更是制度上的信任架构。所有数据记录一经写入链中，理论上就不可被任意篡改或删除；并且任何用户都可独立验证某一数据是否存在于链上，无需依赖中心机构提供证明，审计、追踪、数据可得性变得标准化、自动化。这意味着，区块链将“谁控制账本”这一传统意义上中心机构的权力转化为公开可验证机制，从而实现“制度即代码”的转型。正如以太坊白皮书所提出的愿景，旨在通过区块链技术支持可编程的智能合约，使开发者能够在其上构建去中心化应用（DApps），实现 “无需信任中介即可执行代码” 的功能。

#### （4）密码学从工具到规则带来的制度性革命

区块链的本质是一种建立在密码学基础之上的去信任秩序重构机制，其关键转变不在于技术本身，而在于将密码学从“信息保护工具”升级为“制度设计规则”，如表2-1所示。

表2-1 密码学给区块链带来的制度设计规则

|  |  |  |
| --- | --- | --- |
| 功能层级 | 传统角色 | 区块链语境 |
| 数据加密 | 保密 | 可选隐私保护机制 |
| 身份认证 | 依赖中心化认证机构 | 私钥即身份，签名即授权 |
| 数据完整性 | 审计、日志系统 | 哈希链 + 共识构成不可篡改性 |
| 法律契约 | 需要第三方见证 | 签名与代码即法律执行逻辑 |

这正是“密码学不是保密术，而是制度保障”的真实体现。它不是仅仅保护数据不被泄露，而是建立“任何人都不能随意更改系统规则”的底层保障；它让身份、资产、权限在没有中央协调者的情况下自洽运行；它用数学结构与算法激励替代了传统组织中的制度安排、人为审批与信任博弈。

这种制度转变，是区块链成为“去中心化社会操作系统”的前提，也奠定了其在金融、治理、身份等领域的应用基础。

### **2.1.2 以RSA和ECC为代表的非对称加密**

要理解非对称加密在区块链中的应用，我们需要先掌握其基本工作原理与两种主流算法的异同。以下将从原理出发，依次介绍非对称加密的基本原理以及 RSA 与ECC的比较。

#### **（1）非对称加密的基本原理**

在区块链系统中，非对称加密（Asymmetric Cryptography）是构建账户体系、交易验证、身份认证等基础功能的核心工具。与传统的对称加密算法不同，非对称加密使用一对密钥，也就是公钥（Public Key）与私钥（Private Key），它们通过特定数学结构紧密关联。

非对称密码用于加解密运算时，加密（Encryption）是发送者使用接收者的公钥对信息进行运算，加密后的信息只有拥有相应私钥的人才能解密；解密（Decryption）是接收者使用自己的私钥对接收到的加密信息进行运算，从而恢复明文内容。

反之，非对称密钥结构也支持数字签名机制。签名（Signing）是发送者使用自己的私钥对信息进行运算；验证（Verification）是任何人都可以使用该发送者的公钥验证签名是否匹配，从而确认该信息确实是由该私钥持有者发布且未被篡改。

这种双向结构体现了非对称加密“加密通信”与“身份验证”这两种典型应用场景。在区块链系统中，尤其是比特币和以太坊，非对称加密并不用于加密交易数据，而是用于身份认证和交易签名验证。

#### **（2）椭圆曲线密码（ECC）简介**

椭圆曲线密码（Elliptic Curve Cryptography, ECC）是一种基于椭圆曲线数学结构的非对称加密方法，最早由Victor Miller和Neal Koblitz在1985年提出。相比早期的RSA或DSA算法，椭圆曲线密码在相同的安全等级下只需要更小的密钥长度，但能提供更高的计算效率。

ECC的基本数学对象是定义在有限域上的椭圆曲线，其标准形式为。其中，与是定义曲线形状的参数。为了满足密码学要求，这种曲线必须满足非奇异性条件。曲线上的每个点 都可以参与椭圆曲线点加法（点与点相加）和标量乘法（点与整数相乘）的运算。

ECC的核心在于椭圆曲线离散对数问题（ECDLP）的困难性。具体过程如下。

首先选择一条标准椭圆曲线和一个基础点；之后随机选择一个私钥，其中是的阶；最后计算公钥，即将点进行次加法运算。

已知和，要恢复私钥是目前计算上不可行的。这构成了ECC安全的基础。

#### **（3）为什么选择ECC？**

尽管RSA长期以来是加密通信的主流（如HTTPS协议使用RSA证书），但在资源受限、效率要求高的区块链环境中，ECC更具优势。比如比特币系统中每个节点都需要频繁地进行签名验证操作，且比特币系统对数据的传输和存储成本敏感，因此ECC在这些方面表现优异。表2-2是两种主流非对称加密算法的指标比较。

表2-2 RSA vs ECC 主要指标

|  |  |  |
| --- | --- | --- |
| 特征 | RSA | ECC |
| 安全性基础 | 整数因式分解难题 | 椭圆曲线离散对数难题 |
| 相当安全强度下的密钥长度 | 2048位 | 256位 |
| 公钥规模 | 大（2048位或更高） | 小（256位） |
| 私钥规模 | 大（2048位） | 小（256位） |
| 签名速度 | 慢 | 快 |
| 验证速度 | 快 | 中等 |
| 应用场景 | 传统互联网通信 | 区块链、物联网、移动设备等 |

此外，规模更小的密钥，带来了更短的交易数据，因而也带来了更低的Gas/手续费；更快的运算速度，还带来了更高的交易吞吐率；安全性等价条件下所需资源更少，也使得ECC更适合区块链节点运行环境。因此，现代大多数区块链系统，包括比特币、以太坊、Solana等，均采用ECC作为非对称加密基础。

#### **（4）比特币中的 ECC标准secp256k1 曲线**

比特币使用的椭圆曲线标准是secp256k1，由SEC（Standards for Efficient Cryptography）定义，曲线方程为。该曲线方程定义在素数域上，其中，基础点（Generator Point）G用于密钥生成，具有非常大的阶（接近）。

相较于NIST推荐的被认为可能存在后门参数的secp256r1，secp256k1参数选取透明且非随机，更受比特币社区青睐。其在实现上也避免了某些椭圆曲线中复杂的参数验证逻辑，使得secp256k1相对更高效，更适合在链上环境使用。

在实际应用中，比特币账户地址基于公钥生成，每笔交易均需要签名，也就是使用私钥对交易摘要（哈希值）签名，网络中其他节点通过公钥验证签名的有效性，确保交易未被篡改且由真实用户发起。

#### **（5）加密 vs 签名的功能差异与数学关系**

在区块链系统中，非对称加密可用于两个不同方向，也就是加密通信与数字签名。这两者虽然都基于相同的密钥对结构，但功能与实现目的不同，如表2-3所示。

表2-3 非对称加密算法的加密和签名功能差异

|  |  |  |
| --- | --- | --- |
| 特性 | 加密（Encryption） | 签名（Signature） |
| 用途 | 保密性（只有目标接收方可读取） | 身份验证与不可否认性 |
| 加密者/签名者使用密钥 | 接收方公钥 | 发送方私钥 |
| 解密者/验证者使用密钥 | 接收方私钥 | 发送方公钥 |
| 区块链中的主要用途 | 较少用于纯加密 | 用于交易签名与验证 |

比如在加密通信场景，Alice 想给 Bob 发送私密消息，就需要使用 Bob 的公钥加密内容，Bob 需要用自己的私钥解密。而在签名验证场景，Alice 发出一笔链上交易，需要使用自己的私钥对交易内容签名，任意人都可以使用 Alice 的公钥验证签名是否真实有效。

虽然加密与签名均依赖公私钥对，但在算法层面常使用不同的结构。比如ECIES（Elliptic Curve Integrated Encryption Scheme）[[2]](#footnote-1)是ECC中一种加密协议，在加密运算时就结合了对称密钥与非对称加密；而签名算法如 ECDSA（Elliptic Curve Digital Signature Algorithm），就是比特币使用的核心签名机制。

比特币网络并不使用ECC加密用户交易数据，而是依赖 ECC 签名机制来验证交易的真实性和完整性。用户私钥用于签名交易哈希值，网络通过公钥验证签名是否有效。

非对称加密机制是区块链安全体系中的核心技术支柱，其设计不仅保证了“链上身份”的唯一性与可验证性，也在交易签名、消息认证、资产控制等方面发挥着不可替代的作用。在多种非对称加密方案中，ECC因其优越的性能和安全性成为区块链的首选，特别是比特币系统采用了secp256k1曲线这一设计，使得密钥生成、签名验证等过程高度优化。

加密与签名作为非对称机制的两个方向，其差异体现了区块链系统在处理隐私与公开、认证与去信任的微妙平衡。而理解这些底层机制的数学基础与系统实现，是深入掌握Web 3技术栈的前提。

### **2.1.3 哈希函数的单向性、抗碰撞性和雪崩效应**

哈希函数不仅在密码学中占据基础地位，在区块链系统中也承担了多重关键任务。为进一步理解其作用，我们从其在区块链结构中的具体应用入手展开分析。

#### **（1）哈希函数在区块链中的基础角色**

哈希函数（Hash Function）是一类将任意长度的输入映射为固定长度输出的函数，在密码学、数据结构和分布式系统中广泛使用。对于区块链系统而言，哈希函数不仅是基础性工具，更构成了系统安全性与不可篡改性的核心支撑。从区块头的哈希计算，到交易的签名验证、地址生成，甚至在更高层次的Merkle Tree 构建、共识机制实现等方面，哈希函数都发挥着不可替代的作用。

区块链使用的是密码学哈希函数（cryptographic hash function），其区别于一般哈希函数之处主要在于，它不仅追求“快速映射”，更强调抗逆性、安全性与数学不可预测性。具体而言，一个合格的密码学哈希函数应具备三个基本性质。一是单向性（One-wayness），给定输入容易计算输出，但反过来从输出推导输入在计算上不可行；二是抗碰撞性（Collision Resistance），不可能找到两个不同的输入，使其哈希值相同；三是雪崩效应（Avalanche Effect），输入稍有改动，输出将发生剧烈变化，不可预测。这些特性保障了区块链中的数据完整性、链结构的不可篡改性，以及去中心化节点对数据的快速一致校验能力。

#### **（2）SHA-256 与 Keccak-256 两个典型哈希算法的核心性质**

在区块链系统中，不同的项目选择了不同的哈希算法作为底层支撑。

比特币的基础哈希算法是SHA-256。该算法被广泛部署于区块链系统中，尤其在比特币生态中，是区块头哈希计算、工作量证明（PoW）难度竞赛、Merkle Root 构建的基石。

SHA-256（Secure Hash Algorithm 256）是由美国国家安全局（NSA）设计的 SHA-2 系列中的一员，其输出长度为 256 位（即 32 字节）。SHA-256 的设计目标是抗碰撞性与高效性，虽然其设计较为古典，但其结构已在工业界和学术界长期验证为安全可靠。比特币的安全性，在很大程度上依赖于 SHA-256 的抗碰撞性。攻击者若想伪造一笔交易并重新构造区块链历史，需要构造与原始区块哈希相同的伪造哈希，这在目前计算资源限制下基本不可行。此外，比特币使用的是双重 SHA-256，即对输入先进行一次 SHA-256 哈希，对结果再哈希一次。这种设计增强了抗长度扩展攻击的能力，并增加了计算复杂度，有助于系统安全。

以太坊采用的是 Keccak-256 哈希算法，这是 SHA-3 标准的早期实现版本（标准化后称为 SHA3-256）。虽然二者在基本结构上类似，但参数略有差异。Keccak-256 属于海绵结构（sponge construction）哈希算法，相较 SHA-2 系列，其设计更为新颖，具备以下特征。一是灵活性更高，海绵结构使其更适用于零知识证明和更复杂的密码协议；二是抗攻击能力更强，Keccak 尚未在工业界被发现有有效攻击，其吸收—压缩结构理论上对多种攻击向量具备更高防御力；三是广泛兼容性，由于 Ethereum 的合约层需要频繁使用哈希计算（如地址映射、日志记录、合约调用等），Keccak-256 的性能与通用性得到了实际验证。

需要注意的是，虽然以太坊社区也支持 SHA-256，但在合约级计算中，Keccak-256 是主要接口（在 Solidity 中称为 keccak256() 函数）。

#### **（3）哈希链是不可篡改性的技术基础**

区块链之所以“不可篡改”，本质在于其数据结构的链式哈希设计。每个区块的头部（Block Header）都包含前一个区块的哈希值，这一结构构成了所谓的“哈希链”。

下面是区块N和区块链N+1的数据结构：

Block N:

- Previous Hash: Hash(Block N-1)

- Merkle Root: Hash(all tx in Block N)

- Nonce, Timestamp, etc.

Block N+1:

- Previous Hash: Hash(Block N)

...

这种设计使得区块链在数据层具备如下性质。一是篡改不可局部化，对Block N中任一笔交易的任何更改，都会导致Block N中的Merkle Root值的改变，进而引起Block N的整体哈希值的改变。二是该影响的向后传播。Block N 哈希值一变，Block N+1 的“Previous Hash”字段也不再匹配，必须同步修改。三是改动代价呈指数增长。由于哈希函数的单向性和不可预测性，篡改历史需要从改动区块起重新计算每一个后继区块的哈希，同时满足所有的PoW（若采用PoW），这在现实中几乎不可行。这正是 Nakamoto 在比特币论文中所定义的“最长链安全性”的基础。只要多数算力诚实维护链的延续性，就没有单一节点或组织能篡改过去的账本。

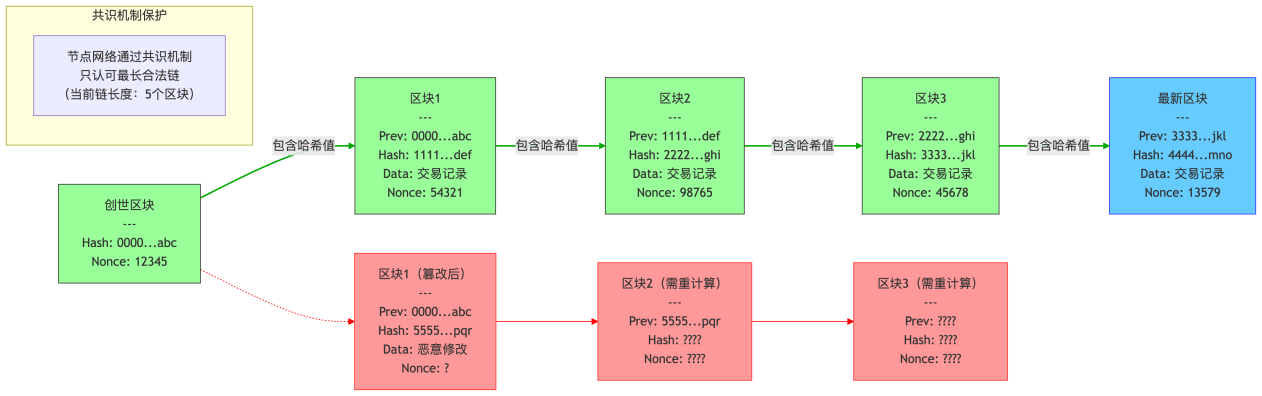


图2-1 区块链链式结构数据不可篡改原理

更具体的，在图2-1中，如果区块1的数据被篡改了，那么它的哈希值就会由原来的1111...def 变为 5555...pqr，由此导致后续所有区块的“Prev”引用失效（红色虚线部分）。为此，攻击者需重新计算被篡改区块及后续所有区块的工作量证明（Nonce值）。此例中篡改链只有3个区块，而主链已有5个区块，但网络节点始终认可最长合法链（绿色实线部分），因此攻击者需超过51%算力才有可能篡改成功，成本极高。

哈希链结构不仅维护账本数据的不可篡改性，也为共识机制提供了快速验证路径。例如，在比特币网络中，节点只需验证区块头中的哈希是否符合难度要求、是否连续，便可确认其是否有效，而无需下载整个区块体。PoS等权益证明类系统虽然共识逻辑不同，但哈希链结构依旧存在，该链结构仍然在维护链状态与区块顺序性，确保全网节点能快速达成同步。

#### **（4）应用示例解析**

理解了哈希函数的原理与结构后，我们可以通过一些实际应用场景，来具体观察它在区块链系统中的关键作用。首先，我们以比特币为例，区块头哈希的计算机制为此提供了一个清晰的切入点。

比特币中每一个新区块的产生，都需要经历“挖矿”过程，即寻找一个能够满足特定难度条件的哈希输出。这一过程实际上是对区块头中若干字段（包括上一区块哈希、Merkle Root、时间戳、Nonce等）进行2次 SHA-256 计算，目标是找到一个满足如下条件的随机数 Nonce，使得

SHA256(SHA256(Block Header)) < Target

由于 SHA-256 的结果完全不可预测，矿工需要不断变更 Nonce 进行尝试，这构成了工作量证明（Proof of Work）的核心机制。

区块链中账户地址（尤其是比特币与以太坊）并非随机生成，而是通过公钥哈希衍生得到的标识符。以太坊网络的地址生成机制如下：

Address = rightmost\_160bits(Keccak-256(Public Key))

该过程确保每个地址在全链唯一，且无法从地址反推回原始公钥（因Keccak的单向性），由此提升了用户隐私性。在比特币网络中，地址生成还需额外使用RIPEMD-160哈希与Base58Check编码，这些步骤同样增强了抗碰撞性与易读性。

Merkle 树是一种基于哈希函数的平衡二叉树结构，用于批量数据摘要压缩与快速验证。区块中每笔交易被哈希为叶子节点，之后每两个哈希值再次组合计算，最终构成顶层的“Merkle Root”。这使得轻节点只需下载部分交易与相应的“Merkle 路径”即可验证某交易是否存在于区块中；而对任意一笔交易的更改都会导致 Merkle Root 的变化，进而影响区块头哈希，增强了不可篡改性。Merkle 树也被广泛用于 Layer2、跨链验证、状态压缩等应用中，是哈希函数扩展性的经典体现。

从密码学角度看，哈希函数可被视作区块链世界的“数字指纹引擎”，其不可逆、抗碰撞与全局性结构构建能力，使得我们无需信任某个中心机构，便可通过数学结构维护数据真实性与系统安全性。无论是最底层的交易结构、区块链接，还是高层的数据压缩、地址映射、共识设计，哈希函数几乎无所不在。其雪崩效应保障了对任一比特变动的敏感性，其单向性保障了篡改的不可行性，其抗碰撞性保障了地址与身份的唯一性。

在构建和理解 Web 3 世界的去中心化信任机制时，哈希函数无疑是最值得掌握和深入理解的核心工具之一。

### **2.1.4 Merkle树结构与验证逻辑**

区块链系统的核心设计目标之一，是在保证数据不可篡改的同时，使得不同参与者可以以极低的计算和带宽成本验证海量数据是否属于某个状态。这一需求催生了多种数据结构的创新，其中最关键且广泛应用的数据结构之一，便是Merkle树（Merkle Tree）。Merkle树不仅是区块结构中交易数据索引与验证的基础，还为轻节点（SPV节点）提供了高效的验证手段。

#### **（1）Merkle树的基本结构与核心逻辑**

Merkle树又称哈希树（Hash Tree），最早由计算机科学家Ralph Merkle在1979年提出，其设计思想是在树状结构中，节点信息由其子节点的哈希值递归组合而成，最终形成一个唯一的根哈希（Merkle Root）作为整棵树的代表。其典型形式是一棵完全二叉树，叶子节点为数据块的哈希值，非叶子节点为其两个子节点的哈希值拼接后再哈希的结果。其形式化定义如下。

给定一组数据块，首先对每个数据块做哈希处理，得到叶节点；接下来，两两组合这些哈希值并哈希生成上一层节点，其中 || 表示拼接操作。重复此过程，直到只剩一个顶层节点，即根哈希（Merkle Root）。

举例而言，如果我们有四条交易数据，其Merkle树构建过程如下。

首先是叶节点，

其次是中间节点，

最后是根哈希，

这种递归式构建机制，带来了极强的数据完整性保证与高效的局部验证能力。

#### **（2）根哈希的验证逻辑与局部数据证明**

Merkle树最核心的功能之一，是通过根哈希进行全局一致性校验，同时支持对某个特定数据是否存在于数据集中进行快速证明，这就是所谓的“Merkle证明”（Merkle Proof）。

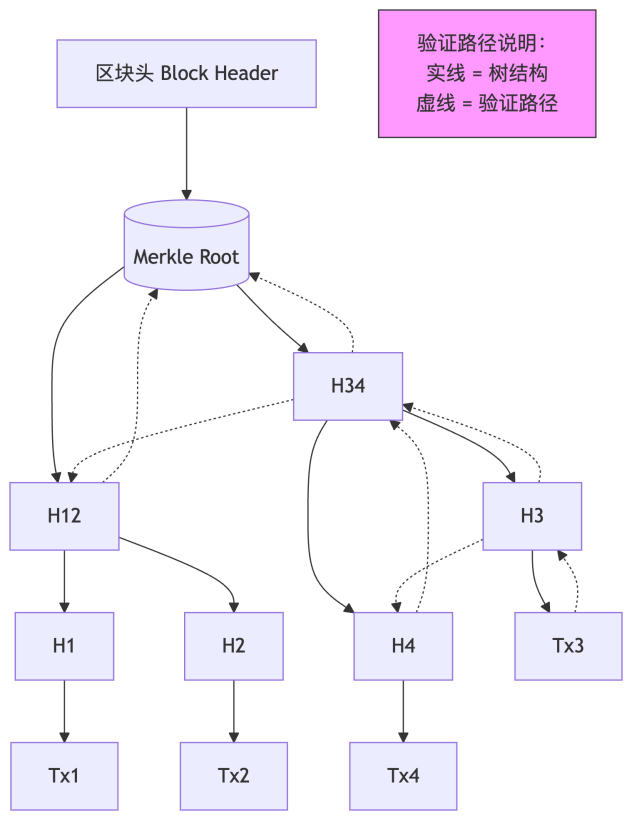


图2-2 Merkle树结构及其验证机制

图2-2是一个包含4个交易的Merkle树，树中有Tx1, Tx2, Tx3, Tx4等4个交易。为了验证Tx3是否包含在树中，需要提供以下信息：交易Tx3本身（用于计算H3）；兄弟节点哈希H4（因为H3和H4组合成H34）；上一层兄弟节点哈希H12（因为H34和H12组合成根）。验证时，首先用提供的Tx3计算其哈希H3' = Hash(Tx3) ，然后用提供的H4与H3'组合计算 计算H34' = Hash(H3' + H4)；再用提供的H12与H34'组合计算Root' = Hash(H12 + H34') 。将计算得到的Root'与区块头中的Merkle根比较，如果相等，则证明Tx3在树中；否则Tx3就不在这个Merkle树中[[3]](#footnote-2)。

这种证明机制的复杂度为，因为只需提供一条从叶节点到根节点路径上的对手节点哈希值，从而极大降低了验证数据量。若某区块包含 1024 条交易，完整下载并验证全部数据需要处理所有 1024 条交易，而借助Merkle树，验证某条交易是否存在仅需大约 10 个哈希值，因为 ，这极大提升了效率。

#### **（3）区块体内的交易索引与SPV客户端原理**

在比特币系统中，Merkle树被直接用于组织区块中的交易数据，其根哈希值被写入区块头（Block Header）中的 merkle\_root 字段。这一设计有两个显著作用，一是整体一致性校验，任意交易的更改都会影响merkle\_root，使得区块头的哈希值发生变化，进而影响整个区块链的哈希链结构；二是支持轻客户端（SPV，Simplified Payment Verification），轻客户端无需存储整个区块链，只需保存区块头，并通过Merkle证明验证交易是否存在于某区块中。比特币白皮书第8节提出了一种“简化支付验证”的客户端设计，即SPV节点仅需同步区块头（约80字节）而非完整区块体（可能超过1MB），这可以大幅降低存储和带宽成本。

验证交易的步骤如下。首先是获取目标交易的哈希值及其所在区块的Merkle证明路径；然后利用路径与交易哈希计算merkle\_root，比对该root是否等于区块头中记录的merkle\_root；进而验证区块头的工作量证明是否有效，且区块链中后继区块数（确认数）足够。这种验证方式虽然无法验证区块中是否存在双花等攻击（因为无法看到全部交易），但对于个人支付接收者而言，在区块足够确认后已具备较强的安全性。

Merkle树结构是SPV客户端可信度与效率之间的关键平衡点，这种设计使得比特币网络可以容纳不同资源等级的参与者。

#### **（4）Merkle树与其他数据结构的对比分析**

Merkle树并非唯一可用于数据完整性校验与索引的结构。为了深入理解其优势与局限，我们有必要将其与其他常见结构进行对比。表2-4就是Merkle树与其他常见结构的对比。

表2-4 Merkle树与其他常见结构的对比

|  |  |  |  |
| --- | --- | --- | --- |
| 数据结构 | 特点 | 优势 | 劣势 |
| 普通哈希（如 | 单个哈希表示整体数据 | 实现简单，占用空间小 | 无法局部验证或部分更新，缺乏可分解性 |
| Merkle树 | 树状结构，递归哈希组合 | 可进行局部验证、支持轻节点、更新局部数据效率高 | 构建与路径维护成本略高 |
| Patricia Trie（前缀树，Ethereum中使用） | 基于路径压缩的前缀结构，键-值映射 | 更适合状态存储，支持高效查找与修改 | 结构复杂，验证路径更长 |

由表2-4，我们可以看到，普通哈希适用于一次性全数据哈希的场景，如文件完整性校验，但不支持子数据验证；Merkle树在需要高效证明“某个数据属于整体”的场景中表现优越，尤其适合区块链中大规模交易索引与验证；而Patricia Trie 更适用于状态型系统（如以太坊的账户状态），其存储的是键值映射结构，支持更新、插入、删除操作，复杂度更高。

以比特币和以太坊为例，比特币使用Merkle树组织交易数据，以太坊则使用三棵Patricia Merkle Trie分别记录账户状态、存储状态和交易日志。这说明Merkle树和Patricia Trie在不同架构中具有互补性。

#### **（5）未来扩展与应用场景**

Merkle树的价值并不止于交易验证，其思想已广泛扩展至许多新型应用中。在状态通道与Rollup等二层扩展技术中，Rollup常使用Merkle树或Merkle根对链下交易集合进行哈希压缩，实现批量提交与验证；在IPFS等去中心化存储中，其内容寻址与版本管理机制中大量使用Merkle DAG（一种广义Merkle结构）；数字版权与文件分发系统通过Merkle Root快速验证某片段属于某个大文件，提高传输安全性。此外，随着Verkle Tree（向量承诺结构）等新型树结构的发展，Merkle树的角色在性能、安全性与扩展性方面也在不断演进。

Merkle树作为一种经典的递归哈希结构，为区块链提供了高效、可验证的数据组织方式，其核心优势在于支持对部分数据的快速验证，复杂度仅为对数级；在轻节点场景中实现有效安全性保障；通过根哈希维护了全局一致性和篡改检测能力。

比特币采用Merkle树构建交易索引并服务于SPV客户端，而以太坊在状态管理中则将其发展为更复杂的Patricia Merkle Trie。通过对比我们可以发现，Merkle树在分布式系统中提供了一种优雅的数据完整性证明机制，其思想已成为现代区块链系统不可或缺的基础。

## **2.2 公私钥签名验证与交易结构**

理解了哈希函数与Merkle树等密码学工具后，我们进一步探讨区块链中最关键的身份认证与操作授权机制，也就是公私钥体系及其在交易结构中的应用。

### 2.2.1 公钥、私钥与地址生成

在区块链系统中，用户账户的身份不依赖于传统的用户名密码或中心认证系统，而是建立在非对称加密技术基础之上。用户通过生成一对私钥与公钥，并进一步从公钥推导出地址，从而实现“自主控制资产、无需注册登录”的用户交互方式。

#### （1）私钥的生成与随机性

在比特币与以太坊等主流区块链系统中，私钥（Private Key）是用户控制其链上账户和资产的根本凭据。它是一个足够大的随机整数，通常取自一个特定的数域范围内。例如，在比特币系统中，私钥是一个介于  和 之间的数，其中是椭圆曲线 secp256k1 的阶，约为 ，是一个 256 位的整数。

私钥必须具备足够的熵值（entropy），也就是随机性强度。若私钥是可预测的或存在偏差，将有可能被穷举破解或提前预测。历史上就出现过“弱私钥”攻击，部分用户使用了低熵或重复的种子词导致资产被盗[[4]](#footnote-3)。因此，私钥生成需使用高质量的伪随机数生成器（CSPRNG），在硬件钱包或助记词生成工具中应谨慎设计用户交互熵、系统时间、环境噪声等种子来源。

#### （2）BIP-39 助记词与种子生成机制

为了解决用户无法记忆冗长随机私钥的问题，比特币社区提出了一套助记词规范，即 BIP-39（Bitcoin Improvement Proposal 39），它将二进制种子转换为一组易于记忆的人类可读单词序列。

助记词的生成步骤如图2-3所示。首先是熵生成（Entropy Generation），熵通常为 128–256 位的随机数，例如 128 位熵可映射为 12 个单词；然后是校验位添加（Checksum），对熵进行 SHA-256 哈希运算，并截取前若干位（熵长度 / 32）作为校验位，附加到熵末尾；再之后是分段映射（Word Indexing），将扩展后的位串按每 11 位进行分段，映射至 BIP-39 词表中的 2048 个单词之一。由此系统就会输出助记词序列，这也是用户获得的一组可抄写、背诵的英文单词，比如trophy fragile web episode flower ski squirrel surge moral ecology vivid foster。

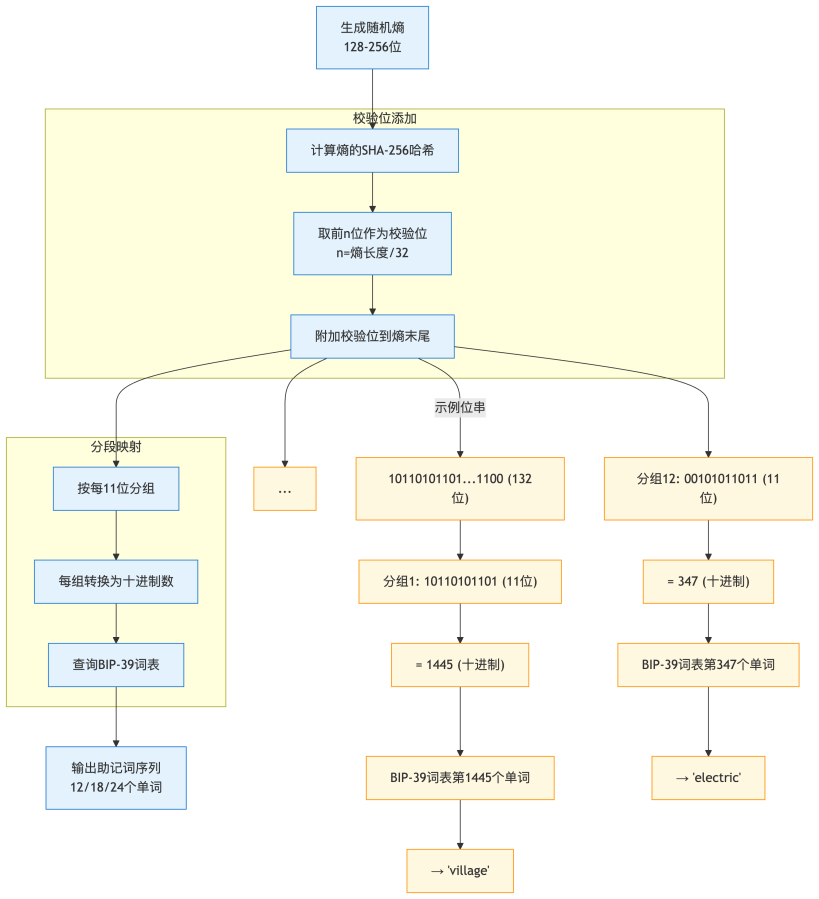


图2-3 助记词的生成步骤

助记词通过 PBKDF2-HMAC-SHA512 与用户设定的密码短语（passphrase，可选）共同生成种子。

seed = PBKDF2(HMAC-SHA512, mnemonic, "mnemonic" + passphrase, 2048 iterations)

这个种子用于生成主私钥（master private key）和派生路径，这也就是整个分层钱包（HD Wallet）[[5]](#footnote-4)的起点。

#### （3）公钥推导与椭圆曲线加密

私钥生成之后，下一步是利用椭圆曲线乘法（Elliptic Curve Multiplication）从私钥推导出对应的公钥。比特币采用 secp256k1 椭圆曲线，其基本操作如下：



其中，为私钥；G为曲线定义的基点； 为椭圆曲线上的标量乘法。

这是一种单向函数，因“离散对数问题”难解而不可逆，从公钥无法反推出私钥。

公钥有两种表示格式。未压缩格式（Uncompressed）为65字节，以 0x04 开头，后跟 X 与 Y 坐标各32字节；压缩格式（Compressed）为33字节，以 0x02 或 0x03 开头，表示 Y 坐标奇偶性，仅保留 X 坐标。压缩公钥更节省空间，是比特币主网中推荐的使用方式。

#### （4）地址编码与Base58Check格式

在比特币系统中，从公钥生成地址涉及多步哈希与编码，如图2-4所示。首先是对公钥做 SHA-256运算；之后对这一结果做 RIPEMD-160运算，得到 20 字节的公钥哈希；再之后是在上一步运算结果前面加上版本前缀（如 0x00 表示主网 P2PKH 地址）；然后对上述结果再做两次 SHA-256，并取前 4 字节作为校验码（checksum）；最后将结果拼接并使用 Base58Check 编码[[6]](#footnote-5)，得到最终地址。

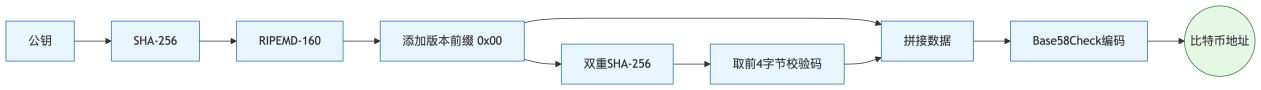


图2-4 比特币系统从公钥生成地址的过程

例如，下面的数字表示就是典型的比特币地址：

1BoatSLRHtKNngkdXEeobR76b53LETtpyT

比特币地址有多种类型，如表2-5所示。

表2-5 比特币系统的地址类型

|  |  |  |  |
| --- | --- | --- | --- |
| 地址类型 | 用途 | 版本前缀 | 特点 |
| P2PKH | 最基础的支付地址类型 （向公钥哈希付款） | 1字节0x00 | 最早的地址格式 |
| P2SH | 多签/复杂脚本地址 （向脚本哈希付款） | 1字节0x05 | 多签、复杂脚本用 |
| Bech32  （P2WPKH/P2WSH） | 原生隔离见证地址 （高效防错） | HRP bc1 | 隔离见证，编码更轻量 |

#### （5）多重签名与脚本化地址

除了基本的一人控制一把私钥外，区块链系统还支持多重签名地址（Multisig Address），允许多个密钥共同管理一个账户。其实现依赖于比特币脚本系统，尤其是 P2SH 与 P2WSH 类型。

例如，“2-of-3”多签地址要求 3 个密钥中任意 2 个签名才能支出资金，其脚本大致为：

2 <PubKey1> <PubKey2> <PubKey3> 3 OP\_CHECKMULTISIG

这个脚本可以通过哈希封装为一个地址，而不是直接暴露脚本内容。

P2SH（BIP-16）允许用户将上述复杂脚本封装为如下地址形式：

3J98t1WpEZ73CNmQviecrnyiWrnqRhWNLy

其逻辑为“付款至某脚本哈希”，支出时需要提供原始脚本与满足条件的签名。

P2WSH（Pay to Witness Script Hash）是在隔离见证（Segregated Witness）机制下的脚本地址版本，格式为 bc1...（Bech32 编码）。它将脚本数据从交易主数据结构中移除，减少了区块大小，提高了效率与安全性。

### 2.2.2 数字签名与验证流程

数字签名是现代区块链系统中保障交易安全性和身份验证的核心机制，其重要性等同于传统金融系统中的签字和盖章。在比特币和以太坊等去中心化网络中，数字签名不仅确保每一笔交易的真实性，而且可以防止中途被篡改，甚至可以构建复杂的智能合约授权逻辑。

#### （1）ECDSA 签名算法的简化逻辑

在区块链中，大多数网络采用椭圆曲线密码学ECC作为基础构建加密与签名机制，比如比特币网络出于高效的计算性能与较短的密钥长度下能够提供更强加密强度的原因，使用了secp256k1 椭圆曲线。ECDSA 是 ECC 的数字签名实现形式，其核心思想是利用私钥生成一个与消息内容强绑定的签名对，验证者则用公钥对其有效性进行验证。

以下是以比特币 secp256k1 曲线为例，展示的 ECDSA 签名流程简化数学逻辑说明。

##### ①签名阶段（由发送者进行）

首先是消息摘要生成，对需要签名的消息（如一笔交易）先进行哈希处理。这一步通常使用 SHA-256算法，，其中  是交易原文。然后生成随机数，选择一个在范围内的随机整数 （私密、临时）。再之后是计算点 ，其中为椭圆曲线上的生成元点，是曲线上的点，对应坐标。

取 。只取 的坐标，对  取模运算结果作为。

计算，其中 是私钥，是 的模反元素。最终签名为一个二元组。

##### ②验证阶段（由网络节点或接收者进行）

重新计算消息哈希，验证签名是否有效。

用公钥 计算以下内容：



如果，则签名有效。

这个过程的本质是借助椭圆曲线上的数乘与加法性质构建了一个不可逆的映射，即使攻击者拦截了和，也无法从中推导出私钥 ，因为这相当于求解椭圆曲线上的离散对数问题，是当前已知计算上不可解的数学问题。

以太坊目前也采用 secp256k1 与 ECDSA，不过未来可能过渡到更高效的 BLS 签名[[7]](#footnote-6)等新算法以支持聚合验证与零知识证明场景。

#### （2）签名机制的三个核心目标

ECDSA 所实现的数字签名机制本质上承担了真实性、完整性与不可否认性三个关键安全目标，这三个目标是所有密码学签名机制的基本设计准则，在去中心化系统中尤其关键。

签名能够证明消息确实是由拥有对应私钥的一方所发出。在区块链中，这意味着只有账户所有者可以“签名”授权其资金的转移。通过公钥可以验证签名是否由私钥生成，从而保证身份的真实性。

数字签名绑定于具体的消息哈希 ，即使内容变动一个比特，签名就不再有效。这一特性防止了消息在传输或存储过程中被篡改。由于签名绑定私钥，发送者在事后无法否认其交易行为。这对建立链上交易记录的法律与制度效力具有重要意义，尤其在 RWA、DAO 投票与司法取证的场景。

#### （3）签名在交易广播与链上确认中的作用

数字签名不仅是密码学操作，更是区块链运行流程中不可或缺的环节。一个交易从用户构造、签名到网络广播，再到最终上链确认，关联了多个关键节点。

以比特币为例，一笔标准交易的生命周期包括以下几个环节。首先是构造交易原文，包括输入（引用的 UTXO）、输出（接收地址与金额）、锁定脚本等；之后是对消息进行哈希处理，也就是对原文执行两轮 SHA-256 哈希生成 ；然后调用私钥，生成签名；再之后是封装签名数据，将签名与公钥一起嵌入 scriptSig。最后进入到本地验证和广播环节，通过钱包或节点将交易广播到网络中。

以太坊中的流程稍有差异，签名采用三元组，用于恢复公钥。

节点在接收到交易后，其验证逻辑包括校验签名是否符合格式要求以及是否在数值范围以内；校验签名对应的公钥是否能得出正确的地址；使用 ECDSA 验证签名是否对应交易哈希；检查发送方余额是否充足、Nonce 是否连续[[8]](#footnote-7)。签名有效后，交易才会被加入到内存池（mempool）中，等待矿工打包。签名失败的交易将被直接丢弃。一旦交易被矿工打包进区块，且连续多个区块对其进行了确认，则该交易被视为“不可逆”，其签名即成为链上行为的正式记录。

其安全性来源于工作量证明与签名认证的组合，篡改任何已确认交易都需重构区块链。Bitcoin Core、ethers.js 等审计工具可以对历史交易进行签名验证，确认交易合法性与责任方。

#### （4）多重签名与高级签名场景拓展

在复杂业务逻辑中，单签名机制已不足以满足安全性或灵活性的需求，例如企业冷钱包要求 2/3 多签；DAO 资金拨款需多方批准；闪电网络或 State Channel 等 Layer 2 协议等新场景也有对签名的更多要求。

比特币支持以下两种主流的多重签名方式，一种是P2SH（Pay to Script Hash），其具体步骤包括构造一个包含多个公钥和门限（如 2-of-3）的锁定脚本，对该脚本进行哈希，并生成一个地址。用户向此地址转账时，即完成资金锁定，花费时需要提供所有参与者签名与原始脚本。

另一种是P2WSH（Pay to Witness Script Hash）。P2WSH是 P2SH 的隔离见证版本，验证效率更高。具体来说，是将交易输入数据移出主交易体，节省空间，提升验证速度。这种方法更适合大规模多签或复杂脚本场景。

以太坊则通过智能合约灵活实现任意签名逻辑，包括多签钱包（如 Gnosis Safe）、社交恢复机制（account recovery）、时间锁、门限授权等扩展。随着账户抽象（ERC-4337）的发展，签名不再局限于椭圆曲线，还可支持 BLS 签名、零知识签名[[9]](#footnote-8)等形式，未来将更加灵活和可组合。

ECDSA 数字签名机制在区块链中的应用体现了密码学在去中心化系统中的制度基础功能。它通过数学上的不可逆性，保障了交易来源的真实性、防止了中途的篡改，并赋予签名行为法律上的不可否认性。同时，在实际系统运行中，签名机制还与交易结构、广播机制、节点验证流程紧密耦合，共同构成了一个“无需信任”的价值转移系统。

### **2.2.3 以比特币为例的交易数据结构**

在区块链系统中，交易（transaction）是链上状态变更的基本单位。在比特币网络中，交易的本质是“输入”和“输出”的连接与转移，体现了一种去账户化的资产管理方式。这种设计不仅降低了实现复杂度，更与比特币的“最小信任”思想相契合。

#### **（1）交易的输入与输出结构**

比特币交易由输入（Input）和输出（Output）两个基本组成部分构成，每一笔交易的本质是使用先前某笔交易产生的“未花费输出”作为输入，从而产生新的输出。其结构可以抽象为如下形式：

Transaction {

version: int32,

inputs: list[TxIn],

outputs: list[TxOut],

locktime: uint32

}

每个输入代表“要花掉的先前输出”，其结构如下：

TxIn {

previous\_output: OutPoint, // 指向某一UTXO

scriptSig: Script, // 解锁脚本

sequence: uint32 // 时间锁辅助字段

}

previous\_output是一个引用指针，由前一交易的 txid（交易哈希）和该交易中输出的索引 vout 组成；scriptSig是一个用于解锁前一交易输出的脚本（unlocking script），通常包含签名和公钥；sequence用于实现 nLockTime 相关的时间锁逻辑，默认为 0xFFFFFFFF 表示立即可用。

每个输出定义了“钱要发给谁”，其结构如下：

TxOut {

value: int64, // 金额（单位：satoshi）

scriptPubKey: Script // 锁定脚本

}

value为该输出分配的金额（satoshi，1 BTC = sat）；scriptPubKey是一个锁定脚本（locking script），规定了花费该输出所需满足的条件。每笔交易可以有多个输入和多个输出，这允许比特币交易具备极大的灵活性，例如从多个来源聚集资金，或将一笔钱分发至多个地址。

#### **UTXO模型与交易引用方式**

在理解了交易在区块链中的基本组成以后，我们需要进一步探讨底层的数据模型，也就是系统如何组织与追踪“谁拥有多少币”以及“哪些币可以被花费”。这是区块链系统设计的核心问题之一，不同的数据模型不仅决定了交易的表达方式，也直接影响了系统的可扩展性、安全性与可编程能力。

目前主流区块链系统采用了两种不同的交易模型，一种是以比特币为代表的UTXO（Unspent Transaction Output）模型，另一种是以太坊采用的“账户模型”（Account-Based Model）。在本节中，我们将首先聚焦于UTXO模型，理解其数据结构、交易引用机制及其在比特币中的具体实现方式。

比特币不使用账户余额的结构，而是采用UTXO模型（未花费交易输出模型），其核心思想是系统中的“钱”不是保存在账户里，而是作为“输出”存在的，一旦某个输出被花费（即作为后续交易的输入），它就不再是UTXO。

每个交易的“输入”必须引用一个已存在且未花费的输出。如下所示：

Tx A → Output[1] (50 satoshis) → 被 Tx B 作为输入引用 → Tx B 的输入

这带来了如下优势。一是并行性，多个交易只要不花费相同的UTXO，就可以同时验证，易于并行处理；二是可追溯性强，资产来源完全可验证，形成清晰交易链条；三是隐私性弱，交易路径全部公开，每个地址行为易于分析（除非使用CoinJoin[[10]](#footnote-9)等隐私增强手段）。

每个输入显式引用一个 <txid, vout> 元组，表明要花的是哪个交易的哪个输出。网络节点会检查该输出是否仍为UTXO；当前输入的签名是否满足该输出脚本要求。由于每个UTXO只能被引用一次，这种设计天然避免了双重支付（double spending）问题。交易一旦被打包入区块，对应的UTXO即被移出未花费集合，后续再试图花费将验证失败。

#### **（3）比特币脚本系统与签名验证逻辑**

比特币使用了一种简化的脚本语言（Script）来描述资产的所有权与控制权逻辑。该脚本是非图灵完备的、基于栈的执行模型，目的是实现可验证但不可执行任意程序的限制，从而确保网络的可控性与安全性。

在验证一笔交易时，Bitcoin Core 节点将把输入和输出中的两个脚本拼接起来：

scriptSig (输入脚本) + scriptPubKey（输出脚本）

这两个脚本会顺序执行，整个脚本被当作一个操作栈处理，最终结果是栈顶为 True 才表示验证通过。举例说明：

scriptSig: <signature> <public\_key>

scriptPubKey: OP\_DUP OP\_HASH160 <pubKeyHash> OP\_EQUALVERIFY OP\_CHECKSIG

其执行过程是先将<signature> 和 <public\_key> 压入栈；OP\_DUP 复制 <public\_key>；OP\_HASH160 对复制的公钥做哈希处理；再将 <pubKeyHash> 压入栈。OP\_EQUALVERIFY 比较两个哈希是否一致，若否则失败。OP\_CHECKSIG 验证该公钥对签名是否有效。

OP\_CHECKSIG 是比特币交易中实现签名验证的关键操作码，其基本流程为，从栈中取出签名和公钥；构造要验证的“消息哈希”，包含交易内容的一个标准化版本（即sigHash，去除当前输入的scriptSig，防止攻击）。使用公钥对消息哈希进行验证，是否与提供的签名匹配。若验证通过，返回 True，否则失败。OP\_CHECKSIG 实现了交易解锁过程的核心信任机制，使得只有私钥所有者才能解锁锁定在某个 scriptPubKey 中的输出。

这种脚本机制提供了比账户模型更灵活的表达能力。开发者可以构造多种复杂交易逻辑，如多重签名（OP\_CHECKMULTISIG）；时间锁（OP\_CHECKLOCKTIMEVERIFY）；哈希锁（HTLC，用于闪电网络）等。这为后续的可编程金融提供了基础框架，虽然功能受限，但稳定、安全、足够表达常见场景。

#### **（4）交易结构的扩展性与未来方向**

虽然比特币交易结构非常简洁，但其背后蕴含的设计哲学具备高度工程美学。UTXO模型提供了“原子性 + 可组合性”，输入/输出结构带来灵活的资金分配方式，脚本系统提供足够的表达能力，又避免了图灵完备所带来的计算不确定性。

随着比特币生态的发展，交易结构也经历了诸如隔离见证（SegWit）、Taproot等升级。SegWit将签名信息分离到 witness 字段中，避免了交易哈希重用问题，提升了效率；Taproot通过默克尔化的脚本路径简化复杂交易脚本，同时提升隐私性。这些改进逐步增强了比特币的可编程性和隐私性，也为未来与链下协议（如闪电网络、RGB协议[[11]](#footnote-10)）的联动提供了基础。

比特币的交易数据结构体现了极简但强表达力的设计原则。通过输入引用、UTXO模型和脚本验证系统的协同工作，比特币构建了一套无需中心账户记录、无需中介确认、却能实现防篡改、防双花、强可验证的交易体系。这不仅是比特币系统安全性的基础，也对后续所有区块链系统的交易设计产生了深远影响。

## **2.3 共识机制的分类与基本模型**

在一个去中心化的系统中，既没有中央服务器来裁定状态的真伪，也无法依赖传统意义上的信任关系来组织协作，因此，系统中所有节点如何在网络延迟、信息不对称，甚至部分节点可能作恶的前提下，仍然就“哪一个状态是被全体认可的”达成一致，成为系统能否正常运转的关键。这种在多方参与者间达成一致的过程，正是“共识机制”（consensus mechanism）的核心目标。

从比特币引入的“最长链规则”，到以太坊的权益证明（PoS）、拜占庭容错算法（BFT）系列，以及后续如Avalanche、DAG等创新方案，去中心化系统中涌现出多种共识机制，它们分别在安全性、效率、去中心化程度与容错能力之间作出了不同权衡，构成了Web 3技术栈中的基础支撑层。

### 2.3.1 “共识”在去中心化系统中的含义

“共识”是去中心化系统最关键的机制之一，它回答了这样一个根本问题，即在没有中央权威的前提下，我们如何就某个状态的真实性达成一致。在区块链这样的系统中，这个状态可能是某笔交易的执行结果、某个账户的余额，或是整个账本的最新状态。理解共识的本质、设计挑战及其与传统分布式一致性的差异，是深入掌握区块链原理的关键。

#### （1）从“数据库一致性”到“区块链共识”的范式转变

在传统的分布式系统中，尤其是分布式数据库领域，“一致性”指的是多个副本的数据保持同步，且对用户而言表现得如同一个单一的、可靠的数据库。这种一致性通常建立在一组可信节点之上，依赖强同步机制与故障检测能力，其经典模型包括Paxos、Raft等协议。

区块链的“共识”虽然在词义上接近“分布式一致性”（Distributed Consistency），但两者的语境、假设与目标却截然不同，如表2-6所示。

表2-6 分布式数据库一致性和区块链系统共识的区别

|  |  |  |
| --- | --- | --- |
| 对比维度 | 分布式数据库一致性 | 区块链系统中的共识 |
| 系统角色 | 通常为可信服务器集群 | 不可信节点之间的对等网络 |
| 故障模型 | 假设最多为崩溃故障（Crash Fault） | 假设存在拜占庭故障（Byzantine Fault） |
| 节点激励 | 通常无显性激励，属于内部部署 | 节点行为与经济激励机制紧密绑定 |
| 网络假设 | 近似同步网络，故障可控 | 异步或部分同步网络，延迟不可预知 |
| 核心目标 | 数据一致性与事务隔离 | 状态确定性与抗审查能力 |
| 代表协议 | Paxos、Raft | Nakamoto共识、PBFT、Tendermint、HotStuff 等 |

由表2-6可见，这种范式转变的核心，在于区块链所面对的是一个“不可信的世界”。在这个世界中，任何节点都可能撒谎、延迟、作恶，甚至串通攻击系统。因此，区块链中的“共识”不仅要解决一致性问题，更要在开放环境中保障安全性与抗审查性，这就对协议设计提出了更高的要求。

#### （2）从非拜占庭环境到拜占庭容错环境的故障模型升级

要理解区块链共识机制的复杂性，必须从“故障模型”（Fault Model）的视角出发。分布式系统的理论基础告诉我们，系统的设计必须在特定的故障假设下运行，不同的故障模型决定了协议的设计难度与可能性边界。

非拜占庭环境Crash Fault是分布式数据库最常见的假设条件，这里的“故障”通常是节点挂掉、失联、重启等“诚实但受损”的行为。在这一模型下，Paxos 等协议通过多数派写入（quorum write）、成员轮换与选主机制（leader election）、成员的心跳检测与日志复制等机制保障一致性。由于节点不会恶意说谎或传递错误信息，因此系统主要关注如何容忍有限数量的宕机节点，并在多数节点保持在线时继续运行。

相比之下，区块链系统运行在一个更为严酷的环境中，即“拜占庭容错环境”Byzantine Fault。该模型最早由 Leslie Lamport 在1982年的论文中提出，其假设包括节点可能完全被攻击控制，发送任意错误信息；节点可能串通作弊；网络延迟不可控，信息可能重复、乱序、丢失。在这种环境下，传统的容错机制完全失效。我们不能假设某个主节点是“值得信赖”的，也不能依赖一致的系统时间或可靠的网络连接。

拜占庭环境要求的“共识”机制，必须满足安全性和活性两个条件。安全性（Safety）是指如果某个状态被系统达成共识，它就永远不会被撤销或修改；活性（Liveness）是指在没有网络永久分裂的情况下，系统最终能够继续推进状态更新。这两个要求是区块链协议设计中的基本“生命线”，但正如我们将在下节分析的，它们存在天然的张力。

#### （3）安全性与活性的基本张力

“安全性”（Safety）与“活性”（Liveness）是分布式共识协议的两个核心指标，它们描述了系统运行过程中的两种能力。一种能力是安全性，即共识达成的内容不会被破坏，一旦网络中某个状态被确认（如一个区块被链上多数节点认可），其他节点就不能再“退回”或“分叉”到另一个状态上。安全性确保系统不会出现“双重支付”等逻辑矛盾。第二种能力是活性，也就是系统不会永远停滞，即使有节点掉线、网络延迟，也应保证新的交易最终可以被打包、确认，链条不断前进。

然而，FLP 不可能定理（Fischer, Lynch, Paterson, 1985）指出，在一个异步网络环境中，不可能构造出同时满足安全性与活性的分布式一致性协议，除非放弃其中之一。这就是区块链设计中最根本的“权衡困境”。

以比特币为例，其采用的“最长链规则”与PoW机制构成了Nakamoto共识的核心。它并不试图解决FLP定理带来的困境，而是采取了一种概率性安全与最终一致性（eventual consistency）的折中方案。这种共识的安全性保障表现为，只要多数算力是诚实的，系统就不会发生长分叉；活性容忍表现为，只要存在至少一个诚实矿工，系统就能继续推进区块延续；而权衡表现在系统的活性强依赖于网络延迟和算力分布，不具备强同步保证，通常通过“等待6个区块确认”的方式间接确保安全。换言之，比特币牺牲了短期内的“即时确定性”，换取了在长期内的“不可篡改性”。

另一类常见的共识机制是基于BFT模型的协议，如 PBFT、Tendermint、HotStuff。这类协议采用投票轮次、多数确认等机制，在一定同步假设下提供更强的一致性保障。在这类共识机制下，安全性表现在最多容忍 f 个拜占庭节点的前提下，系统能保证状态不可篡改；活性通常要求部分同步网络，即网络延迟有上限；在性能权衡方面，这种共识机制更适用于低节点数量、高吞吐率场景，如企业链或PoS链。然而，这些协议通常存在较高的通信开销，无法像PoW系统那样轻松扩展到成千上万个节点的开放网络。

#### （4）从共识机制看区块链系统的设计哲学

区块链共识机制并非单一组件，而是整套系统安全性与信任模型的核心体现。理解共识的含义，有助于我们回答系统的安全性边界在哪里；对抗攻击者时，我们依赖的最小假设是什么，是算力占比、节点数量，还是代币质押？用户为何能信任这个系统给出的结果“不可更改”？

可以说，每一个去中心化系统的设计者，都是在“安全 vs 活性”、“性能 vs 容错能力”之间进行抉择。这些选择直接决定了系统的可用性、可扩展性与抗审查能力。“共识”在区块链中的含义远非传统“一致性”的简单延伸，它是信任架构的核心机制，是制度与技术结合的产物，是博弈、网络、密码学与容错理论的交汇点。

### **2.3.2 PoW机制详解**

在区块链系统中，PoW（Proof of Work）机制是一种通过“计算资源竞争”来实现共识与安全的机制。其核心思想是，参与节点必须完成某种形式的“工作”以获得出块权，这种“工作”应具有对计算资源消耗显著、验证过程快速容易、难以伪造或作弊等特性。PoW 是比特币等第一代公链系统中最重要的共识基础，也是推动区块链迈入去中心化可信计算的新范式的起点。

#### **（1）算力竞争与概率选主决定谁有资格出块**

PoW 系统中，区块的生产者（即“矿工”）需要通过不断尝试计算哈希值来满足某种特定条件，从而“赢得”当前区块的记账权。以比特币为例，这一过程要求矿工找到一个“Nonce”（随机数），使得该区块头的哈希值小于当前网络设定的目标值（Difficulty Target）：

SHA256(Block Header + Nonce)<Target

由于哈希函数（如比特币中的 SHA-256）具备不可预测性和抗碰撞性，该过程本质上是一次大规模的猜数游戏。每一次尝试都是独立的，不能从之前的尝试中学习，因此只能靠暴力枚举。

在这种机制下，“选主权”并不由节点身份、权益或信誉决定，而是由谁先找到合法解来决定。由于寻找有效解所需尝试的次数是随机分布的，因此每个矿工成功出块的概率，取决于其单位时间内可以进行尝试的次数，即其算力（Hash Rate）在全网总算力中所占比例。

我们可以将 PoW 的选主模型理解为一个概率选主系统，该系统具有如下特征。一是概率公平性，算力越高，胜出的概率越高；二是抗女巫攻击[[12]](#footnote-11)，节点无法通过创建多个身份提高成功率；三是不可预测性，系统以及矿工都无法提前知道哪一个矿工会成功。换句话说，PoW 机制将“信任”建立在大量不可预测计算的基础上，这使得网络参与者无需预先信任也能达成一致。

#### **（2）难度调整机制成为网络自适应的重要保障**

PoW 系统的稳定运行要求出块速度保持相对恒定的节奏。例如，比特币设定平均每 10 分钟产出一个区块，这一节奏不仅决定了交易确认的速度，还与系统的代币发行节奏（即通胀模型）密切相关。然而，网络算力并非固定。随着更多矿工加入、硬件性能提升或退出网络，整体算力会发生波动。如果不加以干预，这种波动将直接影响出块时间，进而破坏系统的运行秩序。

为解决这个问题，PoW 系统引入了难度调整机制（Difficulty Adjustment），通过动态调整目标哈希值的大小（Target）来控制出块速度。以比特币为例，每 2016 个区块为一个“调整周期”，理论时间为 2 周（2016 × 10 分钟）。如果实际所用时间 < 2 周，说明算力增长，系统将提高难度；如果实际所用时间 > 2 周，说明算力下降，系统需要降低难度。单次调整的倍率被限制在 4 倍以内，避免剧烈震荡。

目标哈希值与难度值（Difficulty）的关系可以表示为 Target=MaxTarget/Difficulty。其中，MaxTarget是系统设定的最大容忍哈希值，对应最小难度。难度越高，Target越小，寻找合法哈希所需的计算次数也就越多。这种自适应调整机制保证了系统能够在算力剧烈变化的情况下维持出块速率的稳定，从而维持共识的节奏和交易处理能力。

#### **（3）能耗问题与“最终性”的概率滞后性分析**

PoW 的突出优势是其具有无需预信任前提下的强一致性保证，但与此同时PoW机制也带来了能源消耗与共识“最终性”不确定两个核心问题。

PoW 的安全性来源于算力资源的消耗。为维持网络安全，全体矿工需要持续进行大量无效计算，这在本质上是以能源消耗换取抗攻击能力。剑桥大学估算，2022 年比特币网络年均能耗在 91–150 TWh，相当于一个中等国家的年用电量[[13]](#footnote-12)。这一能耗常被诟病为“浪费”或“不环保”，也引发了各国政府与环境组织的审视与监管。反方则认为，这种能耗不是无谓浪费，而是一种以物理资源为代价换取虚拟世界共识安全的成本，如同黄金开采之于黄金本位。合并后的以太坊等部分项目在技术路线上已通过转向 PoS 来规避 PoW 的高能耗问题，而比特币社区则更倾向于将 PoW 的物理消耗视为去信任系统稳定性的必要代价。

PoW 的另一个特点是其“概率最终性”而非“确定最终性”。当一个新区块被挖出后，它并不是立刻就被所有节点认可为“最终状态”。由于网络传播存在延迟，可能有多个矿工几乎同时挖出新区块，造成临时分叉。因此，比特币采用最长链原则（Longest Chain Rule），即以累积工作量最大的链为主。当发生分叉时，主链不一定是第一时间出现的链，而是随后获得更多后继区块支持的那一条分支。

在这种机制下，交易的“最终确认”是一个概率过程。随着后续新区块的叠加，某个区块被替代的概率迅速降低。以比特币为例，社区通常认为，1 个区块被确认的分支，有较高概率最终确认，但存在一定风险；有6 个区块被确认的分支，几乎不可能被回滚，称为“准最终性”。最终性滞后是 PoW 与类 BFT 共识系统（如 Tendermint、HotStuff）的根本区别之一。后者通过节点投票达成即时共识，而 PoW 倾向于依赖“算力历史”的累积来达成后验确认。

#### **（4）双重支付攻击与PoW的安全边界**

双重支付（Double Spending）指攻击者尝试使用同一笔加密资产进行多次消费，通过控制链的分叉与重组以欺骗收款方。双重支付是 PoW 系统中最典型的攻击场景，也是评估其安全性的关键指标。

攻击者广播一笔交易A，将资金支付给商家并获取商品；同时，攻击者私下构造另一笔交易B，将同样的资金转回自己。攻击者不广播B，而是在包含B交易的链上继续挖矿，构造一条不包含交易A的链。若其私链最终长度超过主链，网络会根据最长链原则切换到私链，交易A被“回滚”，商家遭受损失。

上述攻击能否成功，取决于攻击者算力比例q与商家等待的确认数z。中本聪在白皮书中用泊松分布建模了这一概率，核心结论是攻击者算力越小，成功概率越低；商家等待的确认数越多，安全性越高。例如，若攻击者拥有 30% 的算力，商家等待 6 个确认，攻击成功概率约为 0.0005。

因此，PoW 的安全边界建立在攻击者算力不能超过网络总算力的一定比例之上。一旦掌握超过 50% 的算力（即“51% 攻击”），攻击者就具备稳定改写交易历史的能力。这类攻击虽然难度极高、成本巨大，但在一些小币种（如 ETC、BTG）上已多次发生[[14]](#footnote-13)。

PoW 机制以“工作量换取信任”作为设计核心，其逻辑可以概括为以下五个方面。一是激励驱动的算力竞争确保了随机公平选出区块链记账者；二是难度动态调整机制实现了网络节奏稳定；三是能源消耗成本构筑了攻击门槛与共识安全；四是概率性最终性机制提供了后验一致性保证；五是抗双花攻击能力建立了交易不可逆的边界。

PoW 是第一个在大规模开放网络中成功实现无需许可共识的机制，它以极具物理性的设计为数字世界建立了“信任最小化”的范式。然而，PoW 并非没有代价，它的能耗问题、确认滞后性与中心化矿池问题，也引发了日益激烈的讨论与替代机制（如 PoS）的探索。

### **2.3.3 PoS与变体机制**

随着区块链相关技术的发展与能源消耗问题的日益突出，工作量证明机制（Proof of Work, PoW）逐渐暴露出其高能耗、资源集中等弊端。为了寻找更高效、低能耗且更易扩展的替代方案，权益证明机制（Proof of Stake, PoS）应运而生。PoS机制的核心思想，是用“持有代币的数量与时间”来决定节点参与共识与记账的权力，从而取代PoW中通过算力竞争所获得的区块生成资格。

#### **（1）PoS以币为“权”，以质押为“信”的基本原理**

在PoS机制中，系统中的每个参与者都可以将其所持有的加密资产“质押”（Stake），作为其参与共识的保证金。质押越多的节点，被选中提议新区块的概率越高。这种机制以“经济权益”作为系统安全的基础，显著减少了对能源密集型计算过程的依赖。不同于PoW中矿工消耗电力资源以获得新区块的打包权，PoS强调“资本性诚实”，即如果质押者作恶，将面临质押资产的削减（Slashing）或锁定，从而形成激励相容机制。

PoS的主要优势包括节能环保，无需进行大量哈希计算，能耗极低；准入门槛更低，参与者不再需要专业矿机，仅需持有一定数量代币即可参与；经济安全模型清晰，攻击成本等于或高于系统中总质押资产价值的一定比例；易于实现链上治理和自我升级，Stake 本身可作为治理权的载体，与升级机制绑定。

采用PoS机制的典型区块链系统包括以太坊2.0、Cardano、Tezos、Polkadot等。

#### **（2）代理人选举制与投票民主化的DPoS**

DPoS（Delegated Proof of Stake）是PoS机制的代表性变体之一，由BitShares创始人Daniel Larimer首次提出。DPoS引入“代表投票制”概念，将记账权交由一组经过选举产生的“代理节点”（或称为见证人、验证人）承担。所有持币用户根据持仓数量拥有一定投票权；用户可将投票权委托给可信节点（代理人），形成代理共识；被选出的代理节点轮流负责区块生成，形成快速高效的共识流程。

DPoS的优势是明显的。首先是更高的出块速度与吞吐能力，由于节点数量受控（通常为21至100之间），可实现秒级共识；其次是可治理性强，选票机制可动态淘汰不活跃或恶意节点；最后还有运行成本极低，节能、低延迟，适合高频交易场景。然而，其缺点也不容忽视。首先是权力集中趋势，票权通常集中于大户，形成“投票寡头”；其次是利益捆绑问题，代理人可能出现“票权贿选”或串通行为。另外在理论上更易受到协作攻击，一旦主要代理节点合谋，将对链的安全造成威胁。

DPoS机制被广泛应用于EOS、TRON、Lisk、Steem等项目，强调性能优先、治理灵活，但往往牺牲了一部分去中心化属性。

#### **（3）概率性 vs 决定性的最终性处理方式**

在分布式共识中，“最终性（Finality）”指的是一个交易被系统不可逆确认的程度。不同共识机制在最终性实现上存在显著差异。

PoW与Nakamoto PoS机制属于概率最终性（Probabilistic Finality）。交易被打包后并不立即确认，而是随着后续区块的增加，篡改难度随之增大，从而实现事实上的不可逆。例如，比特币社区习惯性地等待6个确认区块以视为“最终确认”。

现代PoS系统（如Casper FFG、Tendermint）多采用确定性最终性（Deterministic Finality）机制。一旦交易被共识多数投票确认，即立即不可撤销。这使得用户无需长时间等待多个确认，提高了用户体验与系统效率。以太坊2.0中的Casper FFG机制便是这种方式的典型代表。它在每一轮共识中引入了“检查点”（Checkpoint）和“超级多数投票”（2/3质押权）机制，确保系统在完成足够投票后即可“最终确认”。

最终性设计对系统安全、交易延迟与用户信任具有直接影响。确定性最终性虽然技术门槛更高，但更适合高价值、低容错的金融场景。

#### **（4）安全性争议与挑战**

尽管PoS机制在节能性与治理灵活性方面表现优越，但也引发了诸多安全性争议，特别是针对以下两类攻击。

一是Nothing at Stake问题。在PoW中，链的分叉存在显著代价，因为挖错链意味着资源浪费。而在PoS中，由于验证者仅需验证签名而无算力成本，因此可以在多条分叉链上同时“投票”，以提高获利概率。这种“零代价双投票”的行为将削弱系统最终性保障，并可能加剧分叉，甚至引发双花攻击（Double Spending）。

解决思路包括引入Slashing机制，如以太坊2.0，若验证者在两个互斥区块上签名，将被惩罚性削减质押资产；使用最终性检查点，强制共识只在“确认过的”链上推进，拒绝未经认可的分支；实施时间同步与投票限制机制，确保验证者在某一时间段仅能参与一个链分支。

二是长程攻击（Long-Range Attack）。长程攻击是PoS特有的攻击方式。若攻击者控制了某一历史时期的大量私钥，即使当前质押者已经变更，其也可以伪造一条从远古区块开始的“伪历史链”，从而欺骗新节点或轻客户端。例如攻击者购买（或恢复）早期验证者的密钥，通过离线重构链历史以覆盖当前主链。这种攻击对无法访问最新区块头的轻节点构成严重威胁。

缓解方式包括弱主观性假设（Weak Subjectivity），要求新节点必须从可信渠道（例如检查点签名）引导同步，而不能盲目信任旧链；采用定期快照机制，系统定期确认一组状态快照，并将其视为“不可篡改”的信任起点；验证人集切换与锁定机制，限制验证者更替速度，增强抵御历史攻击的能力。

#### （5）PoS安全、性能与治理的生态平衡演化趋势

PoS机制并非一成不变，其随着项目需求、攻击经验与设计思潮的演进而持续演化。目前主流PoS系统的演化和探讨主要集中在以下几个方向上，如表2-7所示。值得注意的是，PoS的治理维度也在迅速扩展。例如，部分系统将质押者视为治理代币持有者，拥有对升级、参数调整等的决策权。这种机制虽然增强了去中心化治理的可能性，却也引发了“质押即权力”的政治集中性问题。

表2-7 当前PoS 的演化方向

|  |  |  |
| --- | --- | --- |
| 演化方向 | 说明 | 代表项目 |
| 混合型共识机制 | 结合PoW、PoS优点，增强安全性 | Decred、NEAR |
| Layer 0与验证层分离 | 提高模块化与跨链验证能力 | Polkadot、Cosmos |
| 动态质押权调整 | 引入质押权动态调节，抗寡头 | Cardano |
| 参与者角色分离 | 验证者、委托者、治理者分工明确 | Tezos、Avalanche |

PoS机制代表着区块链共识机制设计的一个重要方向，其以“持币即权益”的理念，构建起一种低能耗、高效率的共识模型。PoS与其变体（如DPoS）在性能、治理机制、经济模型等方面展现出与PoW截然不同的设计哲学。但 Nothing at Stake、长程攻击等独特风险也提醒我们，PoS系统的安全性依赖于参与者行为约束与协议规则的设计严谨性。未来，随着跨链互操作、Layer 2解决方案与ZK证明等技术的融合，PoS可能会进一步发展出多层次、分角色的共识参与机制，使其更好地服务于全球性、高可信需求的去中心化系统。

### **2.3.4 PBFT与拜占庭容错算法**

拜占庭容错（Byzantine Fault Tolerance, BFT）问题，是分布式系统研究中最为关键和困难的议题之一。它要求系统在部分节点作恶、传递虚假信息，甚至彼此协同攻击的情形下，依然能够达成一致结果。为了解决该问题，研究者提出了多种算法，其中最经典且应用广泛的就是实用拜占庭容错算法（Practical Byzantine Fault Tolerance, PBFT）。该算法不仅在学术研究中奠定了基础，也被多个区块链底层协议采纳，用于构建高吞吐、低延迟、可预期的共识系统。

#### **（1）消息投票与阈值达成共识**

PBFT最核心的设计理念是通过多轮“消息投票”机制实现对客户端请求的共识确认，其目标是在最多 f 个节点可能拜占庭失效的前提下，通过n≥3f+1 个节点组成的集群达成一致，确保系统具备“拜占庭容错能力”。

PBFT共识通常由以下三轮消息投票组成，如图2-5所示。在预准备阶段（Pre-prepare），客户端发送请求至主节点（Primary）；主节点将请求打包成预准备消息（Pre-prepare），广播给所有副本节点（Replica）。在准备阶段（Prepare），每个副本节点收到主节点的预准备消息后，验证请求合法性（如签名、顺序等）；验证通过的副本节点向其他所有节点广播准备消息（Prepare）。在提交阶段（Commit），节点收到至少2f个不同副本的准备消息后，进入提交阶段；然后再次向所有节点广播提交消息（Commit）。当某个节点收集到2f+1条提交消息时，视为该请求“共识达成”，可以安全执行。

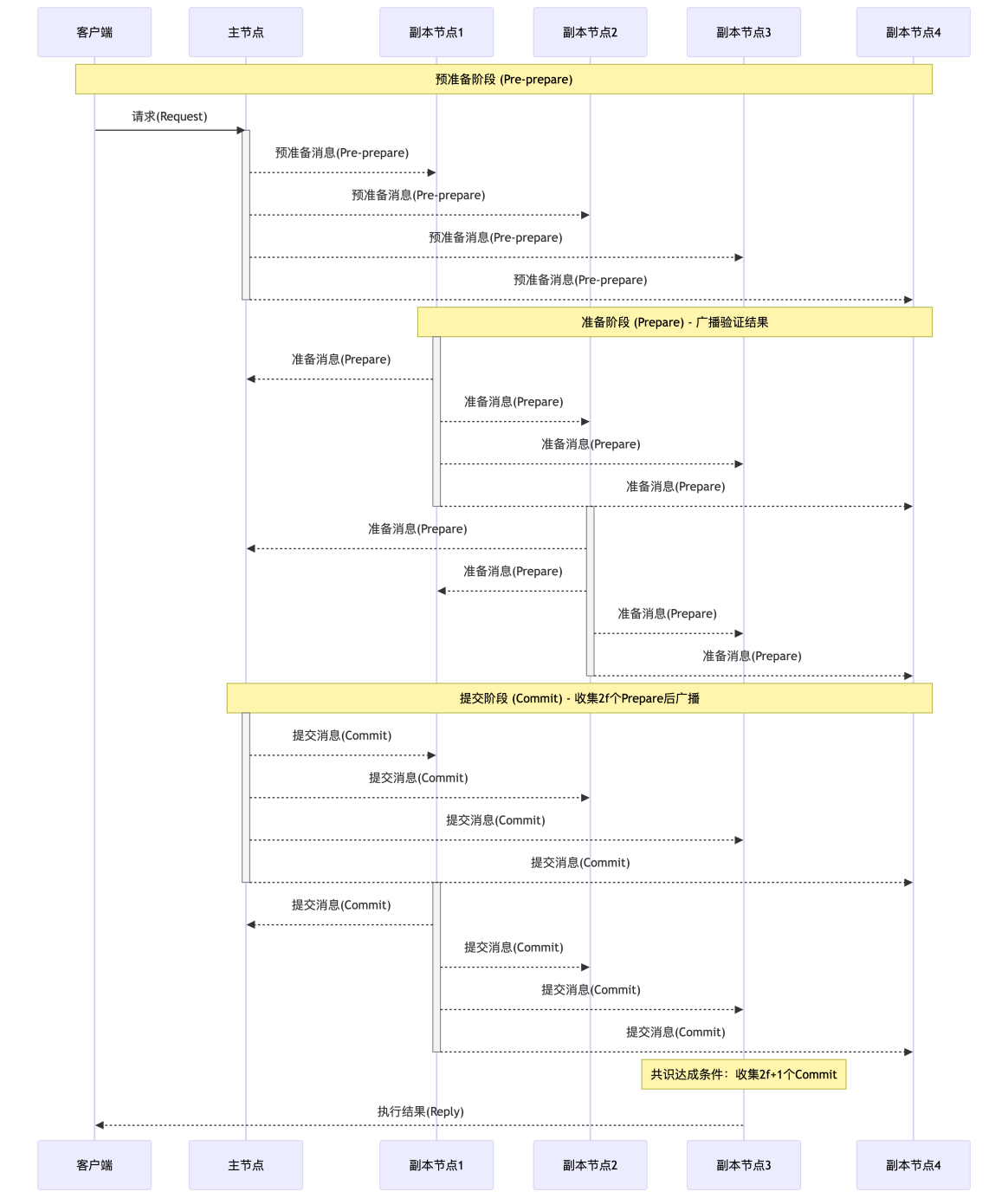


图2-5 PBFT共识达成过程

上述流程确保了只有当大多数节点都认为某个请求是合法且可执行的，系统才会采纳其状态变更。这个设计在理论上确保了安全性和一致性，能够抵御最多f个恶意节点。

PBFT的容错阈值是n≥3f+1，这是为了满足以下两个条件。一是安全性（Safety），也就是不管有多少客户端同时发起请求，不会有两个合法节点对同一个共识轮次产生不同的结果；二是活性（Liveness），即在满足部分网络同步与多数节点正常工作的情况下，客户端请求最终会被处理。该阈值设计保证了即使有f个节点作恶，只要剩余的2f+1个节点中，至少有f+1个诚实节点，就可以完成准备和提交流程。

#### **（2）通信复杂度与节点信任假设**

尽管PBFT在安全性方面表现优异，但其主要缺陷也非常突出，也就是通信复杂度。

PBFT的三阶段消息广播要求所有节点互相通信。在每一轮中，每个节点都需要向其它所有节点发送信息，这导致整个系统的通信复杂度达到O(n^2)。当节点数较小时（例如 4 到 20 个），这种通信成本尚可接受，但随着节点数增加，网络负担呈指数级增长。

具体来说，假设有n个节点，预准备阶段需要处理的信息条数为1 → n-1（由主节点发出）；准备阶段为 n-1 → n\*(n-1)；提交阶段为n\*(n-1)。总计三轮约为O(n^2)，这在大规模系统中容易成为性能瓶颈。

PBFT并非一个“完全开放”的网络共识算法。它假设节点是已知且可认证的，属于“许可型网络”（permissioned network）；节点需要彼此知晓身份，具备公钥注册机制；网络拓扑是稳定的，节点加入与退出必须经过控制；默认假设网络具有同步性或部分同步性，即消息最终可以在一定时间内到达。这种假设虽然不适用于比特币、以太坊等完全去中心化的“开放型网络”，但却非常适合企业级应用、联盟链或特定公共服务系统。

#### **（3）应用案例分析**

PBFT由于其高确定性（finality）与快速确认的特性，在区块链与分布式系统领域得到诸多实际应用。

Tendermint 是 Cosmos 网络的核心组件，其共识机制源于PBFT理念，但进行了关键优化。核心特性包括每个区块都有一个“提议者（Proposer）”，由投票权重决定；共识轮次被明确定义，如果某一轮失败（如提议者失联），会自动进入下一轮；使用验证者集合（validator set）投票，每一轮需要达到 2/3 的投票权确认；默认允许部分同步网络，且支持异步重试机制。

在10-100节点规模内，Tendermint可达到数千TPS，并具备区块级“即时最终性”（即一旦区块生成，不会被回滚）。Tendermint在Cosmos Hub及各区块链应用链（AppChain）上得到了应用。与Cosmos SDK的结合，也使其成为构建特定功能区块链的基础架构。Tendermint证明了PBFT理念在现实区块链系统中的可行性，尤其适用于注重性能与治理可控性的应用场景。

Hyperledger Fabric 是 Linux 基金会支持的企业级区块链平台，广泛应用于供应链、金融清算、身份管理等领域，其共识架构包含多种模块，PBFT为其早期默认方案之一。该架构明确划分了 Orderer（排序节点）、Peer（执行节点）与 Client（发起者）；提交过程可配置共识插件，其中包括PBFT类模块，默认采用“先排序、后执行”模型，这与“先执行、再排序”的公共链形成对比。

其适配场景具有以下特点。一是节点身份由组织控制，符合许可链设计；二是通信链路可以优化，具备高吞吐需求；三是注重事务隐私与可审计性，适合监管环境。Fabric中的PBFT模块强调稳定性与安全性，较少关注“极端开放性”，是一种“工程化”的拜占庭容错应用。

#### **（4）高吞吐 vs 去中心化的权衡点**

PBFT及其相关算法展现出了典型的在性能与去中心化之间的共识机制权衡。

PBFT的优点在于即时共识确定性，不像Nakamoto式共识需等待多轮确认，其低延迟、高并发适合交易频繁的金融或供应链场景。此外还有强安全性，在理论保证下最多1/3节点拜占庭仍可容错。但其缺点也同样明显。首先是无法扩展至大规模网络，O(n^2)通信复杂度难以承载上千个节点；节点加入的门槛高，需先注册、公钥认证、配置网络结构；存在主节点瓶颈，主节点失效或作恶需重选，这会带来延迟与攻击风险。

与比特币等系统相比，PBFT系统在去中心化程度上存在一定让步。节点是“可控的”，系统设计者需预先决定参与者集合；治理机制偏“链外”，节点投票权与管理策略多由治理层外部设定；缺乏抗审查机制，一旦部分节点或通信被审查，可导致共识停滞。因此，PBFT更适合部署在准去中心化或企业内部联盟链中，而不适合完全开放、匿名、无需许可的环境。

PBFT作为一种经典的拜占庭容错共识算法，在理论上提供了强大的安全性保障，尤其在小规模节点网络中展现出高吞吐、低延迟、强一致性的优势。其通过三轮消息投票实现阈值共识，有效防御了部分节点作恶带来的不一致风险。然而，O(n^2)的通信复杂度和许可型网络假设，限制了其在公链等大规模分布式环境中的应用。

在实际工程实践中，Tendermint与Hyperledger Fabric等项目通过改进PBFT或采纳其思想，构建了适用于特定场景的高性能共识引擎。这些应用表明，尽管PBFT不一定适合所有Web 3系统，但在需要快速确定和安全保障的金融、治理、供应链等领域，它依然是一种值得信赖的技术基石。

随着模块化区块链架构的逐步演化，PBFT与其变体仍将在更广泛的Layer 1或Layer 2设计中发挥作用，尤其是在异步环境下的BFT优化、去中心化验证者集的演化机制等方面，仍有诸多值得探索的方向。

## **2.4 FLP不可能定理与拜占庭容错问题**

在分布式计算领域，FLP不可能定理和拜占庭将军问题是理解系统设计约束的基础。它们揭示了在一定假设条件下，不可能同时满足安全性、活性和完全确定性的理论极限。只有充分理解这些问题，我们才能深入掌握区块链共识协议设计的权衡逻辑与现实挑战。

### 2.4.1 分布式系统的根本性难题

分布式系统的核心目标之一，是在多个节点之间达成一致性的决策，无论是数据库的状态同步，还是区块链中的交易确认。这一需求贯穿整个去中心化网络。然而，分布式系统并非万能，现实世界中的通信延迟、节点故障、网络分区等因素，使得分布式共识面临理论上的根本限制。

1985 年，计算机科学家 Michael J. Fischer、Nancy A. Lynch 与 Michael S. Paterson 发表了一篇影响深远的论文《Impossibility of Distributed Consensus with One Faulty Process》，提出了著名的 FLP 不可能性定理（FLP Impossibility）。该定理指出，在异步分布式系统中，即使只有一个节点可能发生故障，整个系统也无法同时保证一致性（Consistency）与活性（Liveness）。这一结论成为分布式系统设计中的“物理极限”，奠定了此后区块链、分布式数据库等技术架构的理论边界。

#### （1）FLP不可能性定理详解

FLP定理的核心结论可以表述为，在一个异步网络（即消息传递存在无限延迟的可能性）中，如果存在至少一个可能出现故障的节点（即存在容错需求），则任何确定性算法都无法保证系统同时满足一致性和活性。

一致性（Consistency）是指所有正确节点对某一状态的最终认知是一致的；而活性（Liveness）是指所有正确节点最终都会做出决策。换句话说，在异步系统中，面对节点可能崩溃的环境，要么牺牲一致性（出现状态分叉），要么牺牲活性（系统可能永远无法达成共识）。

该网络模型的假设有两个方面。一是异步，即网络消息的传输延迟不可预知，可能无限长；二是消息不会丢失，但可能无限迟到。因此，该网络的故障类型就是崩溃故障（Crash Fault），节点可能停止工作，但不会作恶或发送错误信息。

该算法类型是确定性算法（Deterministic Algorithm），即给定输入与状态，输出完全确定。

FLP定理的证明建立在如下推理框架上。系统状态可以划分为决定状态（Decided）与未决定状态（Undecided）。在异步系统中，消息可能无限延迟，算法无法区分消息是否尚未送达，还是消息已永远丢失。因此，存在某些初始状态与消息延迟顺序，使得系统可以被永远保持在未决定状态，从而无法终结。简而言之，异步延迟的不确定性，导致系统无法安全推进共识决策，除非放弃一致性或者允许系统永远无法完成决策。

FLP定理并不是告诉我们分布式共识“不可能”，而是告诉我们必须在一致性与活性之间做出权衡。这成为后续所有分布式共识协议设计的核心前提。而在拜占庭容错场景下，问题变得更为复杂，因为节点不仅可能宕机，还可能作恶，因此 FLP 定理也成为拜占庭共识算法设计的理论基石。

#### （2）区块链系统对 FLP 的应对策略

FLP 定理并非无法突破的“技术绝境”，而是提醒我们，设计分布式系统时，必须在网络模型、共识规则与故障假设之间找到合理的折中。区块链系统正是在此基础上，通过引入概率性与同步假设，构建了可以实际运行的去中心化网络。

部分共识算法选择通过引入部分同步（Partial Synchrony）或同步假设（Synchrony Assumption），突破 FLP 定理的限制。这类算法通常适用于联盟链、许可链等对节点身份有部分控制的场景。以 PBFT（Practical Byzantine Fault Tolerance） 为代表的拜占庭容错算法体系，即基于这种策略。

PBFT的核心设计是假设网络在一定时间内消息必达，即存在已知的最大消息延迟 Δ（同步窗口），然后通过三阶段投票（预准备、准备、提交）机制，确保所有节点最终对交易顺序达成一致。PBFT 能容忍最多f个拜占庭节点，只要总节点数满足n≥3f+1。这实际上是通过同步假设规避了 FLP的限制。在 PBFT 中，系统默认网络延迟虽然可能不可预知，但在大多数时间内处于同步状态。超过最大延迟时间的消息可以认为已经丢失。

因此，PBFT 通过引入“同步时段”的假设，避免了完全异步系统的消息不可达问题，从而绕开了 FLP 定理的限制，允许系统在确定性条件下保证一致性与活性。PBFT的优势是具有确定性终局（finality），交易一旦确认即不可逆；局限是通信复杂度高，O(n^2)消息量限制了其扩展性，因此适合小规模节点集群。

以比特币、以太坊等为代表的公有链体系，采用了概率性终局设计思路，主动接受 FLP 定理的存在，通过引入随机化机制与概率性确认，构建可容错的去中心化共识。工作量证明（PoW）的核心在于通过消耗大量计算资源（Hash 碰撞计算），选出当前轮次的“记账权”节点。由于 PoW 挖矿过程的随机性，系统中每个节点都可能成为区块生产者。网络默认存在临时分叉，节点通过最长链原则逐步收敛至同一链上。

在异步网络中，可能会出现区块同步延迟甚至短暂分叉，但通过累计算力作为“投票权”，系统最终会倾向于接受最长链。PoW 接受系统在短期内可能存在不一致，但概率上随着更多区块产生，链的重组概率趋近于零。这种设计放弃了同步假设与确定性活性，转而追求最终一致性（Eventual Consistency）与概率性终局（Probabilistic Finality）。

权益证明（PoS）延续了 PoW 的设计哲学，但将节点“投票权”与其质押的代币数量绑定。区块生产者按质押权重随机选取，避免了高能耗问题。PoS 的概率性共识逻辑同样允许短时间内的链分叉，但通过引入“罚没机制”（Slashing）来降低节点作恶的动机。系统通过链上经济激励与博弈设计，促使多数节点理性遵循协议规则。类似于 PoW，PoS 系统通常采用一定数量区块确认后视为终局。某些 PoS 协议（如以太坊 Casper FFG）引入了半同步投票机制，在引入概率性的同时，逐步靠近确定性终局。

同步假设与概率性路径，分别代表了联盟链与公有链的主流设计思路，都是对 FLP 不可能性定理的有效工程应对，表2-8是两种主流设计思路的对比。

表2-8 同步假设与概率性路径的对比

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 设计路径 | 核心假设 | 终局类型 | 优势 | 局限 |
| PBFT | 网络部分同步 | 确定性  终局 | 快速、稳定、一旦确认不可逆 | 扩展性差，通信复杂度高 |
| PoW / PoS | 异步网络 | 概率性  终局 | 高容错、无需信任节点身份、强开放性 | 可能存在短暂分叉，最终确认需要等待 |

#### （3）区块链设计中的一致性与活性权衡

在联盟链（如 Hyperledger Fabric、Tendermint）中，节点数量有限且彼此可验证身份，通常更倾向于牺牲部分开放性，追求强一致性与快速确认。交易一旦确认即不可逆，同时交易确认延迟低，但容错范围有限，最多可容忍 1/3 拜占庭节点。此类系统适合企业级、监管友好场景。

在比特币、以太坊等开放型网络中，系统必须面对节点数量大、网络高度异步、节点身份不可信的问题。这类网络短期内可能存在链分叉，最终确认需要多轮区块生成，但容错能力强，允许任意节点加入，开放性高。此类设计更契合无需许可、去中心化的金融应用场景。

近年来，许多区块链协议开始尝试将 BFT 一致性与 PoS 概率性设计相结合，如Cosmos（Tendermint）就由BFT 确定性共识，并且结合了质押选举；Ethereum 2.0（Casper FFG），基于 PoS 的概率性共识，叠加了投票机制保障最终性。这种融合设计试图在活性与一致性之间取得更优平衡，提升系统性能，降低用户等待时间，同时兼顾去中心化属性。

FLP 不可能性定理为分布式系统设计划定了理论边界，提醒我们在设计去中心化网络时，必须在一致性、活性与容错性之间做出权衡。区块链系统通过两条主要路径应对 FLP 挑战，一类通过引入同步假设（如 PBFT）保证确定性终局，适用于节点数量有限、身份可验证的联盟链；另一类通过引入概率性（如 PoW、PoS）实现最终一致性，接受短期分叉，适用于开放型、无需许可的公有链。

FLP 定理并未阻止分布式共识的发展，反而激发了更加丰富的系统设计思路。理解这一理论背景，有助于深入把握区块链架构的根本逻辑与安全边界，也是后续研究共识机制演化、链间互操作、跨链通信等问题的重要基础。

### 2.4.2 拜占庭将军问题与容错模型

在分布式系统设计中，如何在存在部分故障甚至恶意节点的情况下，依然保证全网一致的决策，是一个根本性的挑战。传统的容错问题可以应对节点失效（crash fault），即部分节点可能因为硬件损坏、网络异常等原因停止响应，但这类系统假设节点行为诚实，不会主动发送错误信息。然而，实际环境中，节点可能会出现任意形式的故障，甚至可能被攻击或恶意操控，导致它们不仅会停止响应，还可能向其他节点传播虚假信息。这种更复杂的系统威胁，被称为拜占庭故障（Byzantine Fault），而如何在存在拜占庭节点的情况下达成系统共识，便形成了所谓的拜占庭容错问题（Byzantine Fault Tolerance，BFT）。

#### （1）拜占庭将军问题的提出

拜占庭将军问题最早由 Leslie Lamport 等人在 1982 年提出，是分布式系统容错领域的经典模型，其核心逻辑可重新表述为：一群分布在不同位置的将军需要通过信使传递消息，共同决定是否进攻某一目标。其中，部分将军可能是“叛徒”（对应分布式系统中的恶意节点），会发送虚假消息或隐瞒信息，试图干扰共识；其余“忠诚的将军”（对应正常节点）则需通过协议设计，在存在叛徒的情况下仍能达成两个关键目标，一是共识性，即所有忠诚的将军最终必须就同一作战计划（如“进攻”或“撤退”）达成一致，避免因指令混乱导致失败； 二是正确性，若发出指令的主帅是忠诚的（即指令本身真实有效），则所有忠诚的将军必须最终采纳该指令，确保共识结果符合真实意图。

这一问题的核心挑战，在于如何设计一套消息传递与决策规则，让忠诚节点在面对恶意节点的欺骗、干扰时，既能保持行动一致，又能在源头指令可信时做出正确选择，为分布式系统的容错共识机制提供了思想原型。

在没有任何限制的条件下，Lamport 等人证明了在异步网络中，若存在恶意节点，且节点间只依赖消息传递，系统无法保证一定达成共识。这一结论即为后来著名的 FLP 不可能定理。不过，在同步或部分同步网络中，通过设计合理的拜占庭容错协议，可以在容错上限之内达成共识。

#### （2）拜占庭容错系统的基本定义

拜占庭容错（Byzantine Fault Tolerance, BFT）系统的定义，是在最多有 f 个节点出现拜占庭故障（即节点可以任意行为，包括停止响应、发送虚假信息等）的情况下，系统仍然能够达成一致共识。这一点与传统的故障容忍（Crash Fault Tolerance, CFT）系统不同，后者仅能容忍节点失效，不能处理恶意节点。

拜占庭容错系统要求一致性和可用性。一致性（Consistency）是指所有诚实节点最终的状态必须保持一致；而可用性（Liveness）是只要满足一定的同步假设，系统就必须能够持续对外提供服务，继续处理新的事务或区块。

经典拜占庭容错模型（如 PBFT 算法）假设系统中共有n个节点，其中最多有f个节点是拜占庭节点。在此前提下，系统能够正常工作的前提是f≤(n-1)/3。换句话说，系统必须保证诚实节点数量至少达到总节点数的三分之二以上，即n≥3f+1。

这一约束背后的直觉，是如果拜占庭节点超过三分之一，恶意节点可能单方面操控部分节点，制造网络分裂（fork），或通过信息不对称诱导不同诚实节点做出相互矛盾的决策，导致系统无法保持一致。当拜占庭节点不超过三分之一时，恶意节点无法在全网范围内伪造一致的假信息，系统可以通过多轮投票或消息确认，排除错误路径，从而收敛于正确共识。这一容错上限被称为拜占庭容错理论的基础限制，广泛适用于 PBFT、Tendermint 等算法。

拜占庭容错协议通常要求节点之间具备可靠的点对点消息传递机制，并能验证消息的真实性（通常通过数字签名）。同时，大多数 BFT 协议基于同步或部分同步网络假设，即存在可界定的消息传播延迟，或系统可以在足够长时间内达成时钟同步。在实际应用中，消息传播模型、网络延迟与节点响应速度等因素，都会对 BFT 系统的安全性与性能产生重要影响。

#### （3）比特币的“诚实多数假设”与攻击模型

比特币作为第一个大规模应用的去中心化系统，尽管并未直接采用传统拜占庭容错协议（如 PBFT），但其安全性同样建立在“拜占庭容错”的思想之上。不同的是，比特币引入了工作量证明（Proof of Work, PoW）机制，通过消耗算力的竞争，构建了一种概率性的拜占庭容错系统。

比特币放弃了经典 BFT 中节点记名投票的设计，转而依赖于算力分布的经济假设。其核心安全前提被称为“诚实多数假设”，即在任何时刻，网络中多数算力（超过 50%）掌握在诚实节点手中。这里的“多数”指的是算力比例，而非节点数量。这一假设隐含的逻辑是，如果诚实节点控制了大部分算力，那么攻击者无法持续构造比诚实链更长的分支，无法逆转交易历史。如果攻击者掌握了超过 50% 的算力，便可能发起“51% 攻击”，重写区块链历史，甚至实现双花（Double Spending）。

这种安全模型有别于传统拜占庭容错的三分之一容错上限，但其背后仍然遵循拜占庭问题的基本逻辑，即在系统中存在潜在恶意节点的情况下，需要保证全局一致性与不可篡改性。

对比特币安全模型的攻击，大致可分为以下几种类型。

一是51% 攻击。51% 攻击是比特币最著名的攻击场景，即攻击者掌握全网超过一半的算力，就可以连续挖出更长的区块链，回滚已经确认的交易，并将已消费的资产重新消费（双花攻击）。这种攻击在理论上是成立的，但在实际中存在极高的成本。攻击者需要投入巨大算力，短时间内要超过诚实矿工的总和。同时，攻击可能导致比特币信任基础崩塌，币价暴跌，攻击者获利空间有限。

二是自私挖矿（Selfish Mining）。自私挖矿是一种更为隐蔽的攻击策略，攻击者通过暂时隐瞒所挖出的新区块，等待在掌握多个区块之后一次性广播，以此打断诚实矿工的链条，增加自身挖矿优势。研究表明，即使攻击者算力低于 50%，在一定条件下自私挖矿也可能产生优势，但区块传播优化[[15]](#footnote-14)、矿池去中心化[[16]](#footnote-15)等比特币网络设计降低了此类攻击的实际威胁。

三是重组攻击（Chain Reorganization）。在跨链交易或低确认数支付场景中，攻击者可以通过构造短期链分叉，回滚小额交易。对此，比特币生态通常通过提高交易确认数、使用支付通道（如闪电网络）等方式降低攻击风险。

四是女巫攻击（Sybil Attack）。比特币系统天然抗女巫攻击。由于节点的影响力取决于其算力，而非节点数量，攻击者无法通过创建大量虚假节点干扰系统共识。这是比特币 PoW 设计的一个核心优势。

尽管比特币的 PoW 共识机制与传统 BFT 算法在技术实现上不同，但二者的安全目标高度一致，也就是防止恶意节点通过控制网络或消息欺骗破坏系统一致性。PoW 的创新，在于将节点投票权与资源消耗（算力）挂钩，避免了节点由于身份伪造带来的威胁。通过随机竞争的概率模型，允许节点自由加入或退出，提供极高的去中心化程度。然而，PoW 牺牲了部分性能（如确认速度慢、能源消耗高），并不能完全规避概率性分叉，这些问题在后续 PoS、PBFT 等共识算法的发展中被逐步优化。

#### （4）容错模型的比较与演化

不同模型的容错上限及共识方式见表2-9。从表2-9中可以看出，拜占庭容错问题与各类共识算法设计紧密相关。比特币通过 PoW 机制，实现了一种无需节点身份验证的拜占庭容错方案；以太坊等后续系统则尝试通过 PoS、BFT 等算法，提高了共识效率，降低了能源消耗。

表2-9 不同模型的容错上限及共识形式对比

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 模型 | 容错上限 | 共识形式 | 网络要求 | 性能 | 代表系统 |
| CFT | f ≤ (n - 1)/2 | 多数节点投票 | 异步或同步 | 高 | Raft、Paxos |
| BFT | f ≤ (n - 1)/3 | 多轮消息确认 | 通常假设部分同步 | 中 | PBFT、Tendermint |
| PoW | 算力诚实多数 | 工作量竞争 | 异步容错 | 低 | 比特币、早期以太坊 |
| PoS | 权益诚实多数 | 代币权益投票 | 异步或部分同步 | 中高 | 以太坊 2.0、Cosmos |

未来，随着跨链、Rollup、模块化区块链的发展，拜占庭容错问题仍将继续作为去中心化系统设计的基础理论，被应用于更加复杂和高性能的共识架构之中。

### 2.4.3 现代BFT算法演化

在经典拜占庭容错（Byzantine Fault Tolerance，BFT）算法的基础上，随着区块链应用对高吞吐、低延迟和去中心化程度的更高要求，现代BFT算法经历了显著的技术演化。特别是在Tendermint、HotStuff以及基于有向无环图（Directed Acyclic Graph，DAG）的异步BFT协议（如Avalanche、Aleph）中，开发者尝试在安全性、性能与可扩展性之间寻找新的平衡。

#### （1）Tendermint BFT低延迟、高最终性的新范式

Tendermint 是由 Jae Kwon 于 2014 年提出并实现的拜占庭容错共识引擎，其最重要的应用是作为 Cosmos 生态的核心共识层。Tendermint 在传统 PBFT（Practical Byzantine Fault Tolerance）的基础上进行了重要改进，致力于在异步网络假设下，提供更简单的协议逻辑，以显著降低通信复杂度，并实现更高的吞吐量与更快的区块最终性。

Tendermint 的最大优势，在于其低延迟和高最终性。一旦交易被写入区块并获得超过 2/3 验证者签名即不可逆，避免了如 Nakamoto 共识中存在的链重组问题。Tendermint 通过支持灵活的验证者集管理，使其既可服务于开放的去中心化公链，也可应用于许可制环境。

Tendermint 采用轮换出块（Proposer-based Round Robin）机制，具体流程包括三个阶段，见图2-6所示。在Propose 阶段，当前轮次的提议者提出新区块；在Pre-vote 阶段，所有验证者对提议的区块进行投票。在 Pre-vote 阶段达到 2/3 多数后，验证者进入 Pre-commit 阶段并再次投票。当某个区块在 Pre-commit 阶段获得超过 2/3 的投票时，该区块被认为已达成共识，可被立即提交。

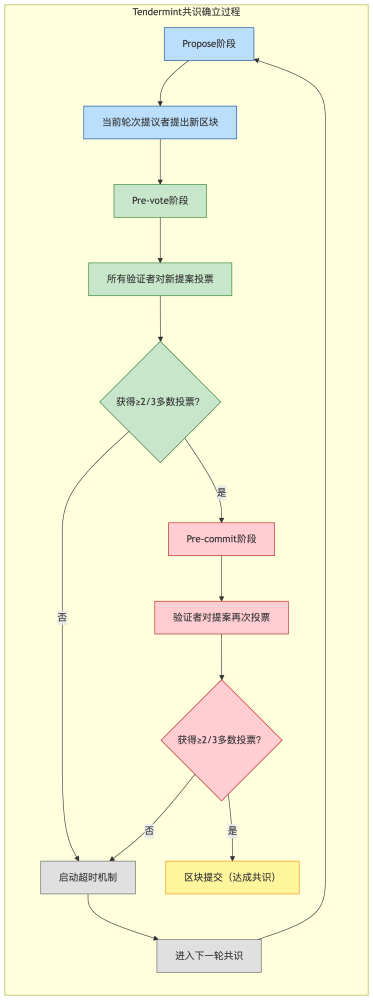


图2-6 Tendermint 共识达成过程

Tendermint 通过超时与轮换机制确保协议不会因为某个提议者离线而卡死。同时，Tendermint 假设网络存在短暂同步性（即 eventual synchrony），在大多数情况下可快速达成共识。

在安全性与局限性方面，Tendermint 能容忍最多 1/3 的拜占庭节点，只要超过 2/3 的验证者保持诚实，系统即可持续安全运行。虽然 Tendermint 在中小规模链上可实现数千 TPS，但在节点数量极大时，仍面临通信复杂度随节点数增加而上升的问题。

Cosmos Hub 通过Tendermint 构建了一个跨链生态（Cosmos SDK + IBC 协议），验证了 Tendermint 的工程可行性和生态活力。Tendermint 的设计代表了第一代现代 BFT 系统向公链环境的成功迁移，显著提升了去中心化共识的可用性。

#### （2）HotStuff 协议的简化线性 BFT 与模块化设计

HotStuff 协议由 VMware Research 于 2018 年提出，其核心设计目标是极大简化 BFT 协议的实现复杂度，同时提升理论性能上限。HotStuff 首次在 Facebook 的 Libra（后更名为 Diem，现为 Aptos）项目中获得应用。

HotStuff 相较 PBFT 的重要突破包括以下几个方面。一是线性通信复杂度，每轮共识仅需 O(n) 消息，远优于 PBFT 的 O(n^2) 复杂度；二是模块化设计，HotStuff 协议结构清晰，易于实现与扩展，降低了工程实现风险；三是流水线式共识，支持连续多轮共识决策并行处理，显著降低了延迟。

HotStuff 共识过程包括三个阶段，如图2-7所示。Prepare 阶段是验证者投票支持当前提案；Pre-commit 阶段是验证者确认上一阶段的提案；Commit 阶段是验证者最终提交提案。与 Tendermint 相似，HotStuff 也采用轮换提议者（Leader Rotation）机制，但在 HotStuff 中，状态传递通过“锁定节点”与“安全规则”实现，保证系统在网络异步情况下依旧保持安全性。更重要的是，HotStuff 支持链式区块连接（类似区块链结构），通过链头锁定（locked QC）有效避免了 PBFT 中频繁的视图切换和复杂的回滚逻辑。

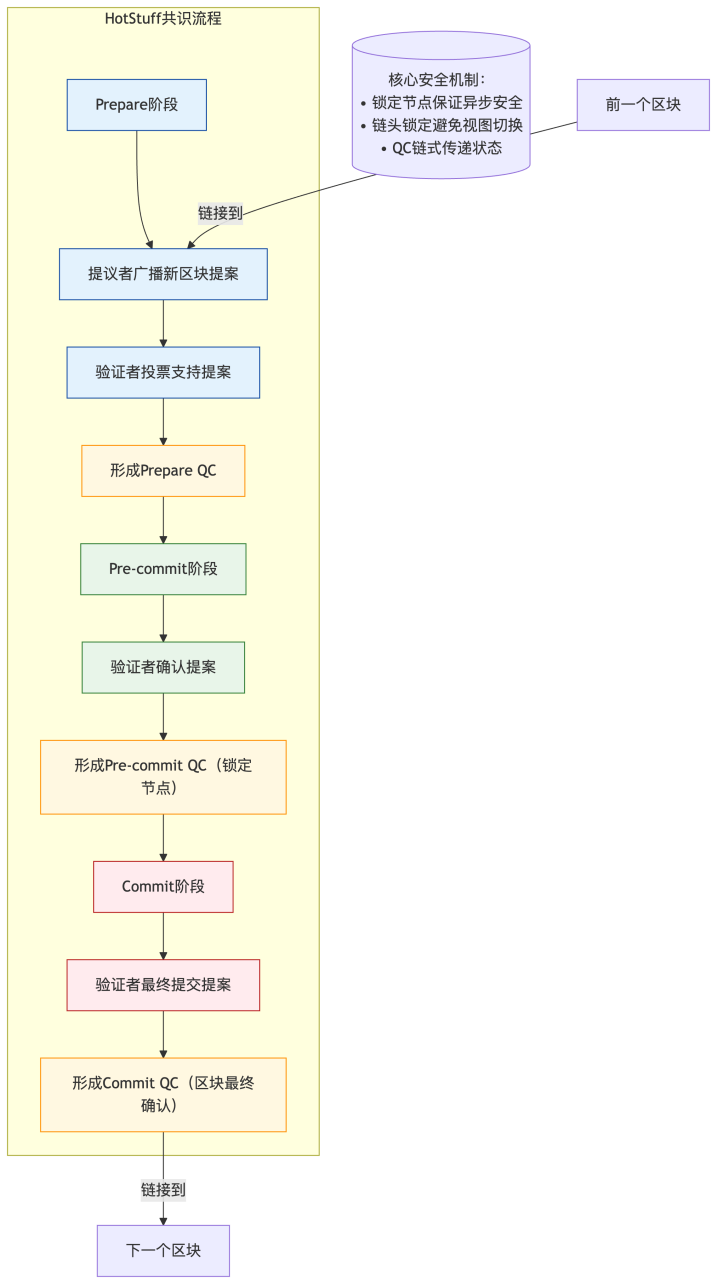


图2-7 HotStuff 共识达成过程

Facebook Libra 项目（后更名 Aptos）将 HotStuff 作为底层共识协议，针对高吞吐支付场景进行了优化。此后，DiemBFT、AptosBFT 等 HotStuff 变体陆续推出。

HotStuff 协议具有以下典型优势。一是高并发能力，支持流水线式并行，提高了交易处理速度；二是去中心化适配，适用于节点参与权限可动态调整的开放式系统；三是良好的安全性，容错能力与 PBFT 相同，容忍 f < n/3 的拜占庭节点。不过，HotStuff 也存在一定局限。一是延迟敏感，协议轮次依赖网络延迟，全球分布式系统的实际延迟可能影响 HotStuff 的优势发挥；二是集中度问题，在实际应用中，若节点分布不均，可能导致提议者轮换失衡。

总体来看，HotStuff 在 BFT 设计中引入了工程简约性的新范式，成为当前许多高性能链（如 Aptos、Sui）的基础。

#### （3）DAG协议与异步BFT趋势

随着近年来对高并发、异步容错与更强去中心化的追求，基于有向无环图DAG的异步 BFT 共识成为重要的技术趋势。DAG协议通过非线性区块结构与概率性最终性，尝试突破传统链式共识的性能瓶颈。

Avalanche 是由 Team Rocket 于 2018 年提出的新型共识协议，后发展成为 Avalanche 网络的核心机制。Avalanche 的关键设计有以下几个方面。一是随机子采样（Repeated Random Sampling），每个节点在局部采样少量邻居节点的意见，通过多轮采样快速逼近全网共识；二是DAG 结构不采用严格的链式区块，而是构建具有依赖关系的有向无环图（DAG），允许多个交易并行确认；三是概率最终性，每笔交易在经过足够采样轮数后，以统计意义上的高概率被全网接受，虽然不是绝对不可逆，但在工程上可认为不可篡改。

Avalanche 的优点包括极高吞吐量与低延迟，可达数千至数万 TPS，确认时间在秒级甚至亚秒级；强异步适应性，无需全网同步，可在高度异步网络下运行；去中心化潜力，支持大规模节点参与而不会引发性能显著下降。缺点包括概率性最终性，无法提供像 Tendermint 那样的即时绝对最终性；参数敏感性，协议安全性依赖采样参数、网络同步假设等，设计需谨慎。

Avalanche 的设计代表了 BFT 共识从同步链式结构向异步 DAG 结构的重要转变。

Aleph 是基于 DAG 构造的另一种异步 BFT 协议，特别强调在无全网同步假设下实现全序共识。Aleph 的主要特点包括无需同步假设，完全容忍任意信息传递延迟；快速达成全序，通过递归确认关系与快速消息传递，节点可在局部快速判定全局交易顺序；高安全性与高并发，容忍 f < n/3 的拜占庭节点，支持高并发交易处理。

Aleph 的核心思想是构建“最早可见的消息路径”，通过 DAG 中的“层级结构”来快速排序节点提交的交易。与 Avalanche 相比，Aleph 更关注理论安全性与去同步化环境的严格证明，但其实际应用案例尚处于相对早期阶段。

DAG协议作为异步 BFT 共识的重要方向，未来可能成为支持 Web 3 应用高并发、高可扩展性的主流选择。然而，DAG协议仍存在若干工程与理论挑战。一是交易顺序的全局一致性与公平性难以保证；二是概率性最终性对部分高价值场景（如金融结算）存在不确定性风险；三是DAG数据结构的可视化、维护与垃圾回收较为复杂。尽管如此，DAG共识在高频交易、物联网、社交等场景展现出强大潜力。

现代 BFT 共识算法的演化呈现出从 PBFT → Tendermint → HotStuff → DAG 异步协议的技术路线。Tendermint 实现了同步假设下的低延迟与高最终性，适合 Cosmos 等多链生态；HotStuff 则通过线性通信复杂度与流水线优化，成为 Diem（Aptos）等高性能公链的核心；DAG 异步 BFT 协议，如 Avalanche 与 Aleph，进一步突破传统 BFT 的结构约束，走向高并发、强异步与极致扩展性的未来。

从当前趋势看，线性 BFT（如 HotStuff）与 DAG 异步 BFT 很可能将成为未来十年高性能区块链系统的基础。对于开发者而言，理解这些共识协议的设计原理与适用场景，将有助于根据需求选择更合适的技术路径，也为进一步的学术与工程创新打下坚实基础。

## **本章小结**

本章从密码学基础与共识机制两大核心支柱出发，系统解析了支撑去中心化系统安全性、可验证性与运行秩序的技术原理。去中心化并非“去安全”，也非“去规则”，而是通过密码学手段与博弈机制，建立的一种无需信任第三方即可运行的制度框架。

首先，我们详细介绍了非对称加密、哈希函数与Merkle树在区块链中的应用。非对称加密为身份认证与交易签名提供了数学保障，哈希函数通过单向性、抗碰撞性与雪崩效应，成为数据不可篡改与区块链结构的基石。Merkle树作为链上数据验证的重要工具，使得即便节点不保存完整账本，也能高效验证区块内任意交易的有效性。通过这些密码学构造，区块链建立了一套无需中心背书的信任路径。

其次，本章深入解析了公私钥体系、数字签名与交易结构的运行细节。通过私钥生成、地址推导与数字签名流程，区块链实现了用户资产的自主控制与可验证交易授权。以比特币为例，UTXO模型中的输入、输出与脚本签名机制，确保了交易的不可伪造性与资产流转的完整记录。

共识机制作为去中心化系统的核心制度层，是本章的重点之一。我们对PoW、PoS、PBFT等主流共识算法进行了对比分析，揭示不同机制在安全性、可扩展性、能耗效率和节点信任假设上的权衡逻辑。PoW通过算力竞争与概率性保证了开放环境下的安全性，但存在高能耗与确认延迟问题。PoS与DPoS则以资本参与代替算力竞争，显著降低能耗，但面临Nothing at Stake与长程攻击的理论争议。PBFT类拜占庭容错机制则强调同步通信与快速确认，适用于高吞吐、较小节点规模的场景。

此外，本章通过FLP不可能定理与拜占庭将军问题，剖析了去中心化系统在理论上必须面对的一致性与活性的根本矛盾。区块链系统通过引入同步假设（如PBFT）或概率性（如PoW、PoS），在不同场景下做出了设计取舍。现代BFT算法如Tendermint、HotStuff，以及异步BFT与DAG协议的兴起，进一步丰富了共识设计的多样性。

通过本章的学习，读者应当理解密钥体系、哈希函数与Merkle树是区块链安全架构的基础；签名验证与交易结构确保了资产的自主管理与交易不可篡改；共识机制决定了去中心化系统的安全边界、性能瓶颈与攻击模型原理；去中心化的核心并非消除所有风险，而是通过密码学与博弈设计，实现最小信任下的安全秩序。

下一章将以比特币系统为具体案例，深入解析其交易结构、区块设计、激励逻辑与脚本语言，帮助读者将密码学原理与链上系统设计深度结合，建立对真实区块链运作机制的系统认知。

1. 由本书作者参与完成的《区块链中的密码技术》（第一版和第二版），已由浙江大学出版社出版发行。 [↑](#footnote-ref-0)
2. ECIES（Elliptic Curve Integrated Encryption Scheme）是一种基于椭圆曲线密码学的集成加密方案，它将非对称加密、对称加密和消息认证码（MAC）结合，提供高效的端到端加密和数据完整性保护。ECIES的核心在于巧妙融合了三种密码学组件，椭圆曲线用于密钥交换，KDF用于密钥派生，对称加密算法用于数据加密。这种组合既继承了ECC的短密钥优势，又通过对称加密解决了大数据量处理的效率问题。 [↑](#footnote-ref-1)
3. 这里需要注意的是，在验证路径中我们还需要知道每个节点在组合中的位置（左或右），因为哈希组合的顺序是重要的（Hash(A+B) != Hash(B+A)）。所以通常还需要提供每个节点是左节点还是右节点的信息。在图中我们用箭头方向了表示，但在实际中需要位置信息。 [↑](#footnote-ref-2)
4. 加密做市商Wintermute因使用Profanity生成以太坊地址，导致私钥泄露，遭受了1.625亿美元的资产损失。Profanity生成私钥时随机性不足，攻击者可利用这一漏洞暴力破解私钥，进而转移资金。

   分析机构Independent Security Evaluators报告显示，一名黑客通过查找错误代码和错误随机数生成器，成功猜测出安全性较弱的私钥，窃取了近4.5万个ETH，在以太坊价格较高时，这些资产价值超5000万美元。 [↑](#footnote-ref-3)
5. 分层钱包（HD Wallet，Hierarchical Deterministic Wallet）是一种通过单一种子（Seed）生成一系列私钥和地址的钱包结构，遵循BIP-32、BIP-39等协议标准。其核心特点是“分层派生”，从主私钥出发，可按预设路径（如按账户、币种、用途分类）逐层生成子私钥、孙私钥等，形成树状结构。所有密钥均由初始种子推导而来，因此只需备份种子即可恢复整个钱包的所有资产，无需单独备份每个私钥，大幅提升了管理便利性和安全性。这种设计广泛应用于比特币、以太坊等区块链生态，解决了传统钱包中密钥分散管理的痛点，同时支持多账户隔离和权限控制。 [↑](#footnote-ref-4)
6. Base58Check 编码采用 58 个字符并且排除了 0、O、l、I 等易混淆字符，增强人类可读性与书写安全。 [↑](#footnote-ref-5)
7. BLS签名是由Dan Boneh、Ben Lynn和Hovav Shacham在2001年提出的一种基于椭圆曲线配对的数字签名方案。基于椭圆曲线密码学，利用双线性配对来实现签名生成和验证。对消息使用私钥计算，验证则通过双线性配对检查等式是否成立。具有短签名特性，签名长度通常仅为单个椭圆曲线群元素。基于Gap Diffie - Hellman问题的计算困难性，安全性较高；对于同一消息和密钥对，签名具有确定性。BLS聚合签名是其重要扩展，允许将多个由不同私钥对不同消息生成的独立签名聚合成一个单一签名，并用对应的公钥集合一次性验证，能极大减少存储和计算开销。在区块链、分布式系统和加密协议中广泛应用，如以太坊2.0的信标链、Symbiotic这类再质押协议的验证机制等。 [↑](#footnote-ref-6)
8. 在账户模型区块链（如以太坊）中，Nonce 是账户专属的递增计数器，初始为 0，每发一笔交易就 + 1 。“连续” 指新交易的 Nonce 需与账户已执行交易的最大 Nonce+1 一致（如已发 0、1、2 号交易，新交易 Nonce 得是 3 ），用于防双花、定交易顺序，保证账户状态稳定，Nonce 不连续的交易（如跳号、重复）会被节点拒绝。 [↑](#footnote-ref-7)
9. 零知识签名（Zero - Knowledge Signature）是基于零知识证明概念衍生出的密码学技术。签名者可在不泄露任何关于消息本身以及私钥等关键信息的前提下，向验证者证明自己对消息进行了合法签名。验证者仅能确认签名的有效性，却无法从签名过程及结果中获取任何除“签名有效”之外的其他信息。在区块链、隐私保护通信等领域应用广泛，例如区块链中，可在保护交易双方隐私（如交易金额、资产来源等信息）的同时，确保交易签名的真实性和合法性，实现隐私性与安全性的平衡。 [↑](#footnote-ref-8)
10. CoinJoin是一种增强数字货币交易隐私的技术。其核心原理是将多个用户的交易输入和输出合并到一个交易中，通过混淆资金流向来割裂交易输入地址和输出地址之间的联系，使外界难以追踪每笔资金的真正来源和去向，从而提升交易的匿名性。该技术由比特币隐私专家Gregor Maňuch于2013年提出，它并非是一种新的协议，而是利用比特币现有功能来增强用户隐私。目前，不少钱包提供商和服务已集成了CoinJoin功能，方便用户在无需专业技术知识的情况下进行混币交易。 [↑](#footnote-ref-9)
11. RGB协议是一种用于比特币等区块链的隐私扩展协议。它能在不改变比特币底层协议的基础上，实现对数字资产的隐私发行、转移和管理。通过隐藏交易细节（如资产类型、数量、流向等），增强交易隐私性；还支持复杂智能合约逻辑，像创建条件支付、匿名资产发行等，可用于构建隐私金融应用、匿名数字资产交易平台等。 [↑](#footnote-ref-10)
12. Sybil Attack一词源于 1974 年小说《Sybil》中多重人格的女主角。在 P2P 网络中，指单一实体伪造大量身份以获取不正当影响力。该概念由微软研究院 John R. Douceur 于 2002 年正式提出。 [↑](#footnote-ref-11)
13. Neumueller, A. (2023). Bitcoin electricity consumption: an improved assessment. Cambridge Digital Assets Programme. Cambridge Judge Business School. 据上述报告，2022 年比特币网络的年度电力消耗估算从原本的 105.3 TWh 下调至 95.5 TWh。其他机构（如 Crypto.com）引用剑桥数据显示，比特币年耗电量在 91–150 TWh 之间。 [↑](#footnote-ref-12)
14. 2018 年 5 月，BTG（Bitcoin Gold）首次遭 51% 攻击，攻击者双花约 18 万枚 BTG，市值约 1,800 万美元；2020 年 1 月再次遇袭，造成约 7 万美元损失。2019 年 1 月 5–7 日，ETC（Ethereum Classic）网络连续遭遇两次 51% 攻击，攻击者通过重组区块链完成双花，共损失约 219,500 ETC，时值约 110 万美元；随后 2020 年 8 月又发生三起类似事件，累计损失超 500 万美元。2020 年 5 月，BCH（Bitcoin Cash）也曾短暂出现算力集中现象，虽未造成大规模双花，但引发社区对安全性的担忧。这些案例均发生在算力相对分散、市值较低的小币种网络，印证了“算力市场越小，越易成为 51% 攻击目标”的结论。 [↑](#footnote-ref-13)
15. 区块传播优化是指通过技术改进加速新生成区块在比特币网络中的扩散效率。比特币网络中，矿工挖出新区块后需将其广播给其他节点以确认区块有效性。若传播速度过慢，可能导致部分节点因未及时接收而继续在旧链条上挖矿（即“区块分叉”），为自私挖矿等攻击提供可乘之机。常见优化手段包括：采用更高效的网络协议（如压缩区块数据、优先传输关键信息）、优化节点连接拓扑（如让节点与更多高带宽节点建立连接）等。这些措施能减少区块在网络中的传播延迟，降低攻击者利用信息差实施攻击的可能性。 [↑](#footnote-ref-14)
16. 矿池去中心化是指降低比特币挖矿算力集中于少数矿池的程度。矿池是矿工联合起来共同挖矿、按贡献分配收益的组织，若少数矿池掌控过大比例的全网算力，可能增强其实施自私挖矿等攻击的动机和能力（算力越集中，攻击者越容易通过协调行动隐瞒区块、操纵链条）。矿池去中心化的核心是通过技术设计或市场机制，让算力分布更分散（如鼓励小型矿池发展、限制单个矿池的最大算力占比、采用更公平的收益分配模式等）。这一过程能减少单一实体对网络的影响力，降低大规模协同攻击的可行性，从而增强比特币网络对自私挖矿等攻击的抵抗能力。 [↑](#footnote-ref-15)