

# EOS.IO 技术白皮书 第二版

译者： Harvey 老狼，谭智勇，宋承根@OracleChain，梓岑@YOYOW，荆凯

摘要：EOS.IO 软件引入了新的区块链架构，旨在实现去中心化应用的纵向和横向扩展。这是通过创建一个类似操作系统的架构来实现的，可以在上面构建应用。该软件提供了帐户，身份验证，数据库，异步通信以及跨越多个 CPU 内核或集群的程序调度。该技术的最终形式是一个区块链架构，在治理区块链的场景下，可以最终扩展，足以支持每秒数百万笔交易，消除用户费用，实现去中心化应用的轻松快速地部署和维护。

请注意：本文中提到的加密通证指的是使用 EOS.IO 软件所构建的区块链上的加密通证，并不是在 EOS 通证发行相关的 ETH 区块链上的 ERC-20 兼容通证。

Copyright © 2018 block.one

无需授权，出于非商业目的和教育用途（即，除了收费或商业目的），任何人都可以使用、复制或者分发本白皮书中的任意材料，需要注明原文出处和适用的版权声明。

DISCLAIMER: This EOS.IO Technical White Paper v2 is for information purposes only. block.one does not guarantee the accuracy of or the conclusions reached in this white paper, and this white paper is provided “as is”. block.one does not make and expressly disclaims all representations and warranties, express, implied, statutory or otherwise, whatsoever, including, but not limited to: (i) warranties of merchantability, fitness for a particular purpose, suitability, usage, title or noninfringement; (ii) that the contents of this white paper are free from error; and (iii) that such contents will not infringe third-party rights. block.one and its affiliates shall have no liability for damages of any kind arising out of the use, reference to, or reliance on this white paper or any of the content contained herein, even if advised of the possibility of such damages. In no event will block.one or its affiliates be liable to any person or entity for any damages, losses, liabilities, costs or expenses of any kind, whether direct or indirect, consequential, compensatory, incidental, actual, exemplary, punitive or special for the use of, reference to, or reliance on this white paper or any of the content contained herein, including, without limitation, any loss of business, revenues, profits, data, use, goodwill or other intangible losses.

- [Background 背景](#)
- [Requirements for Blockchain Applications 区块链应用程序的要求](#)
  - [Support Millions of Users 支持百万级用户](#)

- [Free Usage](#) 免费使用
- [Easy Upgrades and Bug Recovery](#) 轻松升级和故障修复
- [Low Latency](#) 低延迟
- [Sequential Performance](#) 串行性能
- [Parallel Performance](#) 并行性能
- [Consensus Algorithm](#) 共识算法 (BFT-DPOS)
  - [Transaction Confirmation](#) 交易确认
  - [Transaction as Proof of Stake](#) 交易作为权益证明 (TaPoS)
- [Accounts](#) 账户
  - [Actions & Handlers](#) Actions & 处理器
  - [Role Based Permission Management](#) 基于角色的权限管理
    - [Named Permission Levels](#) 命名权限级别
    - [Permission Mapping](#) 权限映射
    - [Evaluating Permissions](#) 权限评估
      - [Default Permission Groups](#) 默认权限组
      - [Parallel Evaluation of Permissions](#) 权限的并行评估
  - [Actions with Mandatory Delay](#) 强制延迟的动作
  - [Recovery from Stolen Keys](#) 丢失密钥的恢复
- [Deterministic Parallel Execution of Applications](#) 应用程序的确定性并行执行
  - [Minimizing Communication Latency](#) 通讯延迟优化
  - [Read-Only Action Handlers](#) 只读的动作处理器
  - [Atomic Transactions with Multiple Accounts](#) 多账户事务的原子化
  - [Partial Evaluation of Blockchain State](#) 区块链状态的部分评估
  - [Subjective Best Effort Scheduling](#) 排程的主观最优化
  - [Deferred Transactions](#) 延迟事务
  - [Context Free Actions](#) 上下文无关的动作
- [Token Model and Resource Usage](#) 通证模型和资源使用
  - [Objective and Subjective Measurements](#) 主观度量和客观度量
  - [Receiver Pays](#) 收款方付费
  - [Delegating Capacity](#) 代理容量
  - [Separating Transaction costs from Token Value](#) 将交易成本与通证价值区分
  - [State Storage Costs](#) 状态存储的成本
  - [Block Rewards](#) 出块奖励
  - [Worker Proposal System](#) 工作提案系统
- [Governance](#) 治理
  - [Freezing Accounts](#) 冻结账户
  - [Changing Account Code](#) 更改账户代码
  - [Constitution](#) 宪法
  - [Upgrading the Protocol & Constitution](#) 升级协议和宪法
    - [Emergency Changes](#) 紧急情形下的修改
- [Scripts & Virtual Machines](#) 脚本与虚拟机
  - [Schema Defined Actions](#) 由模式定义的动作
  - [Schema Defined Database](#) 由模式定义的数据库
  - [Generic Multi Index Database API](#) 通用的多重索引数据库 API

- [Separating Authentication from Application](#) 将验证与应用区分
- [Inter Blockchain Communication](#) 跨链通讯
  - [Merkle Proofs for Light Client Validation](#) 用于轻客户端验证的 [Merkle 证明 \(LCV\)](#)
  - [Latency of Interchain Communication](#) 跨链通讯延迟
  - [Proof of Completeness](#) 完成性证明
  - [Segregated Witness](#) 隔离见证
- [Conclusion](#) 结论

## Background 背景

区块链技术源于 2008 年推出的比特币，自那时以来，企业家和开发人员一直在努力使该技术通用化，以便在单个区块链平台上支持更广泛的应用。

虽然一些通用区块链平台还在努力实现第一个能正常运行的区块链应用，针对特定场景的区块链应用诸如 BitShares 去中心化交易所（2014）和 Steem 社交媒体平台（2016）已经成为日活跃用户上万的成功应用。这两个应用成功的把性能提高到每秒数千笔交易，延迟降低到 1.5 秒，降低交易费用，并实现了与当前中心化服务器的方案相似的用户体验。

由于现有的区块链平台使用费用高昂，计算性能有限，阻碍了区块链的广泛应用。

## Requirements for Blockchain Applications 区块链应用的要求

为了得到广泛使用，区块链上的应用程序要求区块链平台足够灵活，能够满足如下要求：

### Support Millions of Users 支持百万级别的用户

想要同 Ebay, Uber, AirBnB 和 Facebook 这些企业竞争，需要能够处理数千万日活跃用户的区块链技术。在某些情况下，除非用户数足够庞大，否则应用程序可能无法正常运作，因此，能够应对大量用户的平台至关重要。

## **Free Usage 使用免费**

应用开发人员需要具备灵活性，能够为用户提供免费服务；用户不必为了使用平台或从平台的服务中受益而付费。用户可以免费使用的区块链平台，自然会得到更多人的青睐。有了足够的用户规模，开发者和企业可以创建对应的盈利模式。

## **Easy Upgrades and Bug Recovery 轻松升级和故障修复**

基于区块链构建应用程序的企业，需要区块链平台具备灵活性，可以为其应用添加新特性来增强完善。区块链平台必须对软件 and 智能合约的升级提供支持。

所有的非小型的软件都可能会有缺陷，即使是用了最严格的形式验证也是如此。当 bug 不可避免出现时，区块链平台必须足够健壮，能够修复这些 bug。

## **Low Latency 延迟低**

及时的反馈是良好用户体验的基础。延迟时间如果超过了几秒钟，会大大影响用户体验，严重降低基于区块链的应用相对于现有的非区块链应用的竞争力。区块链平台应当支持低延迟的交易。

## **Sequential Performance 串行性能**

有些应用程序由于必须顺序执行命令，从而无法用并行算法进行实现。诸如交易所之类的应用经常需要足够的串行操作处理性能，以应对大量的交易。因此，区块链需要提供强大的串行性能支持。

## **Parallel Performance 并行性能**

大型应用程序需要在多个 CPU 和计算机之间分配工作负载。

## **Consensus Algorithm (BFT-DPOS)**

EOS. IO 软件采用了目前为止唯一能够符合上述性能要求的去中心化共识算法——委托权益证明 [Delegated Proof of Stake \(DPOS\)](#)。根据这一算法，在使用 EOS. IO 软件构建的区块链上持有通证的人，可以通过一个持续进行的投票系统来选择区块生产者。任何人都可以选择参加区块生产，只要能够说服通证持有人为其投票，就会有参与区块生产。

EOS. IO 软件可以让区块每 0.5 秒生成一个。任何时刻，只有一个生产者被授权产生区块。如果在计划的某个时间内没有成功出块，则跳过该块。如果有一个或更多的区块被跳过，则在区块链上会有 0.5s 或者更久的空白。

使用 EOS. IO 软件，区块的产生是以 126 个区块(每个出块者六个区块，乘以 21 个出块者)为一个周期。在每个出块周期开始时，会根据通证持有人所投票数选出 21 个区块生产者。被选中的区块生产者的顺序会根据 15 个及以上的区块生产者的同意，制定出块顺序的安排。

如果出块者错过了一个块，并且在最近 24 小时内没有产生任何块，则这个出块者将被剔除在考虑范围之外，直到他们通知区块链可以重新开始产生区块。这确保了网络的顺利运行，把被证明为不可靠的区块生产者排除在出块排程之外，通过这一方式使得错过区块的数量最小化。

在正常情况下，DPOS 块链不会经历任何分叉，因为区块生产者并非竞争关系，他们合作产生区块。如果有区块分叉，共识将自动切换到最长链。这一方式之所以有效，是因为区块链分叉上增加区块的速度，与具有相同共识的区块生产者的比例直接相关。换句话说，具有更多生产者的区块链长度将比具有较少生产者的区块链增长速度更快，因为，有更多生产者的区块链分叉上，丢块更少。

此外，没有块生产者可以同时两个区块链分叉上生产块。如果一个块生产者发现这么做了，就可能被投票出局。这类双重生产的密码学证据，也可能被用来自动移除作恶者。

在传统的 DPOS 算法上增加了拜占庭容错算法 (Byzantin Fault Tolerance) ，所有的出块者都要对所有区块签名，以此来确保在同一时间戳或者同一区块高度上，没有区块生产者能够同时在两个区块上签名。一个区块有了 15 个区块生产者的签名，该区块就被认为是不可逆的。任一拜占庭区块生产者如果想在同一时间戳或者同一区块高度的两个区块上签名，就不得不留下密码学证据。在这一模式下，一秒之内就可以达成不可逆的共识。

## Transaction Confirmation 交易确认

使用 DPOS 算法的区块链，一般出块者都是 100%参与的。一笔交易在广播后平均 0.25 秒，就可以认为具有 99.9%的确定性了。

EOS. IO 在 DPOS 之外，还增加了异步拜占庭容错 (aBFT, asynchronous Byzantine Fault Tolerance )，来实现更快的不可逆性。aBFT 算法使得在 1 秒内就可以对不可逆性得到 100%的确认。

## Transaction as Proof of Stake (TaPoS) -- 交易证明

EOS. IO 软件要求每个交易都要将最近区块的区块头哈希的一部分包括在其中。这一哈希有两个目的：

1. 防止在区块链分叉上的交易重放，这些交易并不包含参考区块；
2. 通知区块链网络，某一用户及其资产(stake)处于特定的分叉上

随着时间的推移所有用户都能直接对区块链进行确认，这使得伪造链难以作伪，因为伪造的链无法从合法的链上转移交易。

## Accounts 账户

EOS. IO 软件允许使用唯一的可读的名称来实现对帐户的引用，名称最长为 12 个字符。该名称由帐户的创建者选择。账户创建者必须留出 RAM 空间用于存储新的账户，直至新建的账户抵押了通证以获得自己的 RAM 空间。

在去中心化的情境下，应用程序开发人员将为创建帐户支付名义成本来注册新的用户。通常企业已经以广告和免费服务等形式，为所获取的每个用户花费了大量资金。相比之下，创建新的区块链帐户所需的资金成本是微不足道的。并且幸运的是，没有必要为已经由另一个应用程序注册的用户创建帐户。

## Actions & Handlers Actions 和处理器

Each account can send structured Actions to other accounts and may define

每个帐户可以将结构化的 Action 发送到其他帐户，并且可以定义 Action 被接受后的处理脚本。EOS. IO 软件为每个帐户提供其自己独有的数据库，只能由自己的 action 处理程序访问。Action 处理脚本还可以向其他帐户发送 Action。Action 和自动的 action 处理程序的组合正是 EOS. IO 定义智能合约的方式。

为了支持并行执行，每个账户都可以在其数据库中定义任意数量的范围(scope)。区块生产者会对事务进行排程，不会在访问 scope 的内存时出现冲突，因此事务可以进行并行执行。

## **Role Based Permission Management 基于角色的权限管理**

权限管理主要涉及对某一特定 Action 是否得到正确授权进行确定。权限管理的最简单形式是检查事务是否具有所需的签名，但这意味着已经知晓了所需的签名。通常，授权是与个人或由个人组成的团队绑定在一起的，并且通常会进行划分。EOS. IO 软件提供了一个断言式的权限管理系统，可以让帐户就何人在何时能够做什么事上，进行细粒度和高级别的控制。

至关重要的是，身份认证和权限管理被标准化实现，并与应用程序的业务逻辑分离。这使得开发某种工具以通用方式管理权限成为可能，并为性能优化提供了巨大的空间。


每个帐户都可以通过其他帐户和私钥的任何加权组合来控制。这种机制创建了一个能够真实反映权限在现实中的组织情况的层次化权限结构，并使得多用户对帐户的控制比以往任何时候都更容易。多用户控制是提升安全性的最重要因素，如果能正确地使用，可以极大地消除黑客盗窃的风险。

EOS. IO 软件允许帐户可以定义何种密钥和/或账户 3 的组合，可以向另一账户发送特定类型的 Action。例如，可以为用户的社交媒体帐号提供一个密钥，并提供另一密钥用于访问交易所。甚至用户可以给予其他帐户许可让其代表自己的帐户行事，而无需向其他帐户分配密钥。


### **Named Permission Levels 命名权限级别**

## Authorities

### Signing

 STM7xh66F5ZHyfN9u4rNZmTioBteZhvWdqDwaR2kR55t  
BXeCjTr2z

### Owner

 STM7iTj8quuiqX7aUHZWrYXfAqqTDbpL8FwxoCUzEeKE  
745mvBY41


### Active

 STM5DiwrKfp4ngW7fWcDej1Kd3efogYSGxsMKfkz3xi4  
AsNGYWU2D

### Posting


 [streemian](#) 1 33.3%

 [esteemapp](#) 1 33.3%

 STM89kBiwpi5R8CBrgAFWd5p5uayTvvS7B  
6Zb4oBenWx8ChjeLMTf 1 33.3%

Threshold 1 33.3%

### Memo

 STM7S4xvQgdvuQ8SFAD8vzFcLskoUdLqzEPbudcXuyok  
u2irwzt6v

使用 EOS.IO 软件，帐户可以定义命名权限级别，每个权限级别可以从更高级别的命名权限派生。每个命名权限级别定义一项授权；这一授权是密钥和(/或)其他账户的命名权限级别所组成的多签名检查的阈值。例如，帐户的“朋友”权限级别可以设置为帐户中的某项 Action 能被该账户的任何朋友的帐户平等地控制。

另一个例子是 Steem 区块链，其具有三个硬编码命名的权限级别：owner, active 和 posting。Posting 权限只能执行诸如投票和发布等社交行为，而 active 权限除了更改所有者之外，可以做其他任何事情。owner 权限用作冷存储，它能够做



所有事情。EOS. IO 对这一概念进行了通用化，允许每个帐户持有者定义自己的权限层次结构以及动作的分组。

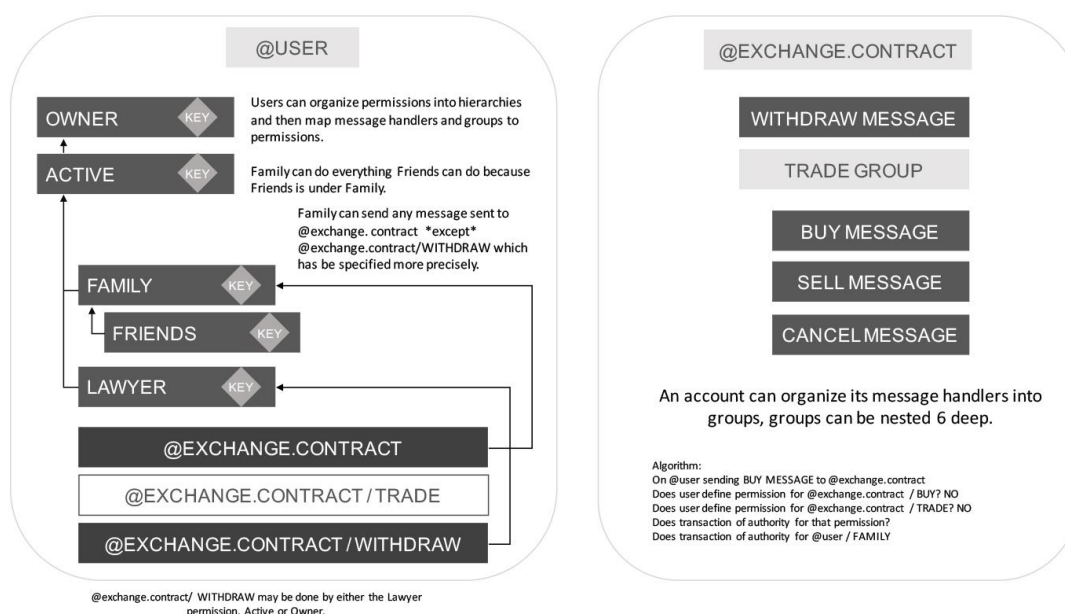
## Permission Mapping 权限映射

EOS. IO 软件允许每个帐户定义在合约/动作或任何其他帐户的合约，以及账户自己的命名权限级别之间进行映射。例如，账户持有人可以将账户持有人的社交媒体应用程序映射到帐户持有者的“朋友”权限组。通过此映射，帐户的任何朋友都可以和帐户持有者一样，在帐户的社交媒体上发布内容。即使他们会作为帐户持有者来发帖，他们仍然使用自己的密钥来为 Action 签名。这意味着总是可以辨别出来哪些朋友以何种方式使用了其帐户。

## Evaluating Permissions 权限评估

当从@Alice 向@bob 发送类型为“Action”的动作(Action)时，EOS. IO 软件将首先检查@alice 是否为@bob. groupa. subgroup. Action 定义了权限映射。如果没有找到任何结果，那么将会检查@bob. groupa. subgroup，然后是@bob. groupa，最后是@bob 的映射。如果此时仍然没有找到匹配的结果，那么假定的映射将是命名权限组 @alice. active。

一旦识别出权限映射，则启动多签名阈值校验过程对签名授权进行校验，把该授权与命名权限相关联。如果失败，那么它会遍历父权限，最终遍历到其所有者的权限， @alice. owner。



## Default Permission Groups 默认权限组

EOS. IO 的技术还允许所有帐户都有一个可以做所有事情的“owner”权限组，和一个除了更改所有者组之外可以执行所有操作的“active”权限组。所有其他权限组均派生自“active”权限组。

## Parallel Evaluation of Permissions 并行权限评估

权限评估是个“只读”的过程，通过事务对权限进行的修改，在一个区块结束时才会生效。首先，这意味着所有事务的所有密钥和权限评估可以并行执行。其次，这种机制意味着可以快速验证权限，而不需要启动可能会回滚的代价高昂的应用程序逻辑。最后，这意味着当接收到挂起的事务时，可以执行事务权限验证；而在取消挂起、事务生效时，无需重新执行验证。

考虑到所有因素，权限验证占据了交易验证中很大比例的计算量。使之成为只读和并行的过程，可以显著提高性能。

在对区块链重放，以便从 Action 的日志中重新生成确定性的状态时，并不需要再次评估权限。事务被包含在一个已知的状态良好的区块中这一事实，使其可以跳过评估的步骤。这极大地减少了在对日益增长的区块链重放时的计算工作量。

## Actions with Mandatory Delay 强制延迟的动作(Actions)

时间是安全的关键组成部分。在大多数情况下，在私钥被使用前不可能知道其是否已经被盗用。基于时间的安全机制在人们使用一些特殊的应用程序时更为关键，这些应用程序需要在连网的电脑上保存密钥，便于人们日常使用。EOS. IO 软件允许应用程序的开发者指定某些 Action 在包含在区块后、实际应用之前，必须等待的最短时间段。在此期间，这些动作可以被取消。

当这类 Action 被广播时，用户可以通过电子邮件或短信收到相应通知。如果他们未曾授权，那么他们可以登录其帐户来还原帐户数据并撤回 Action。

所需的延迟取决于操作的重要程度。支付一杯咖啡可以在几秒钟内确认，不需延迟，而买房子可能需要 72 小时清算周期。将整个帐户转移到新的控制者手上可能最多需要 30 天。具体延迟的选择取决于应用程序开发者和用户。

## Recovery from Stolen Keys 密钥被盗后的恢复

EOS. IO 软件为用户提供了一种在密钥被盗时恢复其帐户控制权的方法。帐户所有者可以使用在过去 30 天内活动的所有者(owner)权限的密钥，以及账户所有者

所指派的帐户恢复合作伙伴的许可，来重置其帐户上的所有者密钥。不经过帐户所有者的配合，帐户恢复合作伙伴无法重置其帐户的控制权。

对于攻击帐户的黑客而言，由于其已经“控制”该帐户，因此尝试执行恢复过程没有任何收获。此外，如果他们的确进行恢复的过程，那么恢复合作伙伴可能需要身份认证和多因素认证（如电话和电子邮件）。这或者会暴露黑客的身份，或者黑客在恢复过程中毫无所得。

这个过程与简单的多重签名机制有极大的不同。通过多重签名的交易，有一个对象会执行并参与每一笔交易。然而，通过恢复过程，恢复过程的合作伙伴仅参与了恢复的过程，并没有权力参与日常的交易。这极大降低了相关参与者的成本和法律责任。

## **Deterministic      Parallel      Execution      of** **Applications    应用程序的确定性并行执行**

区块链共识取决于确定性（可重现）行为。这意味着所有并行执行都能在不使用互斥体或其他锁的原语的情况下正常运行。没有锁，必须有方法来确保可能需要进行并行操作的事务，不会产生非确定性的结果。

2018 年 6 月份发布的 EOS. IO 软件版本会是单线程运行的，但是包含了未来需要进行多线程和并行执行所用到的数据结构。

基于 EOS. IO 软件构建的区块链，一旦并行执行的功能开启，区块生产者的工作是将 Action 传递到单独的碎片 (shard) 中，以便它们可以并行地评估。每个帐户的状态只取决于传递给它的消息。区块生产者的输出会产生排程表，并且将被确定性地执行，但是生成排程的过程不必是确定性的。这意味着区块生产者可以利用并行算法来对事务进行排程。

并行执行的部分还意味着当脚本生成新的 Action 时，它不会立即发送，而是在下一个周期中发送它。无法立即发送的原因是因为接收方可能会在另一个碎片 (shard) 中主动修改自己的状态。

## **Minimizing Communication Latency    通信延迟优化**

延迟时间是一个帐户将动作 (Action) 发送到另一个帐户并收到响应所需的时间。EOS. IO 软件的目标是使两个帐户能够在单个区块内来回交换 Action，而不必在每个 Action 之间等待 0.5 秒。为了实现这一点，EOS. IO 软件将每个区块分为周期 (cycle)。每个周期分为多个碎片 (shard)，每个碎片 (shard) 包含一组事务列表。每个事务包含一组要传递的动作 (Action)。该结构可以被可视化为树，其中各层依据其特性被顺序处理或者并行处理。

Block 区块

Region 区

Cycles (sequential) 周期(顺序)

Shards (parallel) 碎片(并行)

Transactions (sequential) 事务(顺序)

Actions (sequential) 动作(顺序)

Receiver and Notified Accounts (parallel) 接收者和通知的账户(并行)

在一个周期中生成的事务可以在任何后续的周期或区块中传送。区块生产者不断地向一个区块中添加 cycle，直至达到了最大的时钟时间，或者没有新生成的需要传送的事务为止。

可以使用区块的静态分析来验证在给定周期(cycle)内是否存在两个碎片(shards)包含了修改同一个帐户的事务。只要这种静态分析机制一直起作用，就可以通过并行运行所有线程来处理区块。

## **Read-Only Action Handlers 只读 Action 处理器**

部分账户可能会处理一些只需要决定通过与否的动作(Action)，而不会改变自己内在状态。这种情形下，只需要在一个特殊的周期内的一个或多个碎片(shards)中只有某一特定账号的只读 Action 处理器被包含在其中，这些处理器就能并行进行。

## **Atomic Transactions with Multiple Accounts 多账户的原子事务**

有时，我们希望确保 Action 可以被多个帐户以原子化地方式传递和接收。在这种情况下，两个 Action 会被放置在同一笔事务中，两个帐户将被指派相同的碎片(shard)，Actions 会按顺序执行。

## Partial Evaluation of Blockchain State 对区块链状态的部分评估

大规模区块链技术组件应该是模块化的。不需要每个人都运行所有东西，特别是如果他们只需要使用应用的一小部分时，更是这样。

出于将交易状态显示给用户的目的，交易所应用程序的开发者将维护一个完整的节点。这款交易应用不需要其他社交媒体应用的相关状态。EOS. IO 系统允许任何完整节点可以选择性地运行任意的应用子集。如果你的应用程序不需要依赖其他程序的状态，那么传递给其他应用的 Action 将被安全地忽略。

## Subjective Best Effort Scheduling 自主最优任务安排

EOS. IO 系统不能强制阻止区块生产者向其他帐户发送的任何 Action。每个区块的生成者对处理一笔事务的计算复杂度和时间都有自己的主观度量，无论这一事务是由用户生成的还是由智能合约自动生成的。

在使用 EOS. IO 软件来构建而启动的一条区块链中，在网络层上所有交易的带宽成本，都会基于所执行的 wasm 命令的数量来计费。但是，使用该软件的每个单独的区块生产者会使用它们自己的算法和测量方式来计算资源的使用情况。当一个区块生产者发现一笔事务或帐户已经消耗了大量的计算能力时，他们会在生成自己的块时拒绝该交易；但是，如果其他区块生产者认为该事务有效，他们仍然会处理。

一般来说，只要有一个区块生产者认为一笔事务是有效的，并且所消耗的资源是在限制范围内的，那么其他所有的区块生产者也会接受，但可能要花费多达 1 分钟才能使该笔事务传播到这一区块生产者处。

在某些情况下，区块生产者可以创建一个区块，其中包括在可接受范围之外的事务。在这种情况下，下一个区块生产者可能会选择拒绝这个区块，而这一僵局将会被第三个区块生产者终结。这与一个大区块导致网络传播延迟所引发的情况没有什么不同。社区会注意到这种滥用权力的模式，并最终撤销对该恶意区块生产者的投票。

对计算成本的主观评估，可以让区块链不必去精确地度量运行某些任务所需要的时长。有了这个设计，就不需要精确地对指令计数，这将极大地增加优化的可能，而不会打破共识。

## Deferred Transactions 延迟交易

EOS.IO 软件支持延迟事务，事务可以安排在未来执行。这可以让计算转移到不同的碎片(shard)和/或创建长期运行的持续的流程，可以为持续的事务排队。

## Context Free Actions 上下文无关的 Actions

上下文无关的动作(Action)涉及到这样的计算：只依赖事务的数据，而不依赖区块链的状态。例如，签名验证的计算，只需要事务数据和签名来确定签署该事务的公钥。这是区块链上必须执行的最昂贵的计算之一，但是因为这一计算是上下文无关的，所以可以并行执行。

上下文无关的操作就像其他的用户 Action 一样，只是它们无法访问区块链状态来执行验证。这不仅能使 EOS.IO 可以并行处理所有上下文无关的动作 (Context Free Actions)，例如签名验证，更重要的是，这为通用的签名验证提供了支持。

在支持上下文无关动作(Context Free Actions)的情况下，可伸缩性技术如 Sharding、Raiden、Plasma、State Channels 等，变得更加可并行化，实用性更强。这一发展可以实现高效的跨区块链通信和潜在的无限可扩展性。

## Token Model and Resource Usage 通证模型和资源使用

请注意：在本篇白皮书中，所指的加密通证是使用 EOS.IO 软件所构建区块链中的加密通证。并不是在 EOS 通证发行过程中所用的基于以太坊的 ERC-20 兼容通证。

所有的区块链都是资源受限的，需要系统防止其滥用。在使用 EOS.IO 软件的区块链中，应用程序会消耗三大类资源：

1. Bandwidth and Log Storage (Disk);
2. Computation and Computational Backlog (CPU); and
3. State Storage (RAM).
4. 带宽和日志存储 (磁盘);
5. 计算和计算积压 (CPU);
6. 状态存储 (RAM).

瞬时使用和长期使用的这两类组件都会消耗带宽和计算。区块链系统将维护所有 Action 的日志，这些日志将会被所有的完整节点下载和存储。通过日志，可以重构所有应用程序的状态。

计算债务( computational debt)是指通过对 Action 的日志重新生成状态的计算消耗。如果计算债务的增长太大，就有必要对区块链的当前状态进行快照，并抛弃区块链的历史状态。如果计算债务增长过快，区块链将会耗用 6 个月的时间来重放 1 年的交易。因此，对计算债务进行谨慎管理是至关重要的。

区块链存储的状态指那些可从应用程序逻辑访问的信息。它包括诸如订单和帐户余额等信息。如果应用程序不读取该状态，则不应该存储它。例如，博客文章的内容和注释不被应用程序逻辑读取，因此它们不应该存储在区块链的状态中。与此同时，博文或评论是否存在这一状态信息、投票数和其他属性将作为区块链状态的一部分被存储起来。

区块生成者可以公布它们可用的带宽、计算资源和状态的容量。EOS. IO 系统根据账户在期限为三天的抵押合约中所抵押的通证数量，允许每个帐户可以消耗一定比例的可用容量。例如，假设一个基于 EOS. IO 系统的区块链应用启动，如果一个帐户持有该区块链提供的总通证的 1%，那么这个帐户就有可能利用该区块链 1% 的状态存储容量。

使用 EOS. IO 系统的区块链上，带宽和计算能力的分配是基于部分储备机制的，因为它们是短暂的(未使用的容量不能存储下来为将来使用)。EOS. IO 系统将使用类似于 Steem 的算法来限制带宽使用速率。

## **Objective and Subjective Measurements 客观和主观度量**

正如前面所讨论的，可度量计算的使用对性能和优化有很大的影响；因此，所有资源的使用限制最终都是主观的，并且根据各自的算法和估计来执行。通常由区块生产者创建自定义的插件来实现。

除此之外，有些无足轻重的东西，可以进行客观衡量。所传递动作(Actions)的数量和存储在内部数据库中的数据大小，这些都很容易进行客观测量。EOS. IO 软件允许区块生产者在这类客观的度量上应用相同的算法，但是也可以在主观度量上选择更严格的主观算法。

## **Receiver Pays 接收方支付**

历来是由企业为办公空间、计算电力以及运营业务所需的其他费用买单。客户从企业购买特定产品，而这些产品的销售收入将用于支付企业的运营成本。同样，没有任何网站要求访问者为维护服务器而支付小额费用。因此，去中心化应用程序不应该强迫它的客户为使用区块链而向区块链支付直接费用。

使用 EOS. IO 软件的区块链不要求用户直接向区块链支付使用费用，因此不限制或阻止企业制定其产品的货币化策略。

虽然接收方可以付费，EOS. IO 软件允许发送方为带宽，计算和存储付费。这样的话，应用开发者可以选择最适合他们应用程序的方式。在很多情况下，对于不想要自己实现定量配给系统(rationing system)的开发者而言，发送方付费这一方式显著降低了复杂性。应用开发者可以将带宽和计算能力代理给他们的用户，然后采用“发送方付费”的模式来使用应用。从终端用户的角度来看仍然是免费的，但是从区块链的视角看，这是属于发送方付费的模式。

## **Delegating Capacity 授权能力**

在一条使用 EOS. IO 系统开发的区块链上，通证的持有人可能不需要立即消耗可用带宽的全部或部分资源，他们可以选择将未消耗的带宽委托或租赁给他人；在这一区块链上运行 EOS. IO 软件的区块生产者将识别这一授权并分配相应的带宽。

## **Separating Transaction costs from Token Value 将交易成本与通证价值区分开**

EOS. IO 软件的主要优点之一是，应用程序可用的带宽数完全独立于通证的价格之外。如果应用程序所有者持有相应数量的通证，那么应用程序可以在固定的状态下，使用固定的带宽资源持续运行。开发人员和用户不会受到通证的市场价格波动的影响，因此不会依赖于喂价。换句话说，使用 EOS. IO 程序运行的区块链，可以让区块生产者能够自然地增加每单位通证可用的带宽、计算资源和存储资源，这与通证的价值无关。

使用 EOS. IO 软件的区块链，区块生产者每次产生区块，都会得到一定的通证奖励。通证的值将影响一个区块生产者能够有钱购买的带宽、存储和计算量；这个模型自然会利用通证价值的上涨来提高网络性能。

## **State Storage Costs 状态存储成本**

带宽和计算可以代理给他人，但是应用程序状态的存储需要开发者持有通证，直至状态删除为止。如果程序的状态永不删除，那么实际上这部分通证就退出了流通。



## Block Rewards 出块奖励

在使用 EOS.IO 软件构建的区块链上，每生成一个区块，出块者都会得到一些新的通证作为奖励。在这一状况下，新创建的通证数量是由所有区块生产者公布的期望报酬的中位数而决定的。可以配置 EOS.IO 软件，限制区块生成者所得奖励上限，使得通证供应的年总增长率不超过 5%。

## Worker Proposal System 工作提案系统

在基于 EOS.IO 软件的区块链上，通证持有人除了选举区块生产者，还可以选出一些旨在造福社区的工作提案。获胜的提案能够得到通证奖励，所配置的每年通证的膨胀率减去已支付给区块生成者的部分，就是这部分奖励的最大值。这些提案将按照所得到的选票比例来获得通证的分配，上限是他们进行工作所要求的通证数量。所选出的提案可以由通证持有人新选出的提案所替代。

实现工作提案的系统合约，可能不会在 2018 年 6 月份首网启动时就绪，但是资金机制会上线。区块生产者开始获得奖励时，就会开始为提案系统累积基金。由于工作提案系统会以 WASM 实现，所以日后可以无需分叉就能够将其加上。

## Governance 治理

治理是社区之中成员的如下过程：

1. Reach consensus on subjective matters of collective action that cannot be captured entirely by software algorithms;
2. Carry out the decisions they reach; and
3. Alter the governance rules themselves via Constitutional amendments.
4. 有些事实无法通过软件代码来收集，而人们通过搜集这些事实，就主观问题达成共识；
5. 执行他们达成的决策；
6. 通过宪法修正案，来变更治理规则。

基于 EOS.IO 软件的区块链实现的治理过程，有效地引导了区块生产者的现有作用。以前的区块链缺乏定义好的治理过程，依赖于临时、非正式和经常存在争议的治理过程，从而导致不可预知的结果。

基于 EOS.IO 软件的区块链认为，权力来自于通证持有人，他们将权力代理给区块生产者。区块生产者被赋予了有限和经审查的授权，可以冻结账户，更新有缺陷的应用，并提出对基础协议进行硬分叉的变更。

EOS. IO 软件内嵌了区块生产者的选举机制。在对区块链进行任何更改之前，这些区块生产者必须批准它。如果区块生产者拒绝按通证持有人的期望做出变更，那么可以投票将其替换。如果未经通证持有者的允许区块生产者就擅作更改，那么所有其他非出块的全节点验证者(交易所等)将拒绝该更改。

## **Freezing Accounts 冻结账户**

有时，智能合约会发生异常或不可预知的状况，无法按预期执行；有时，应用程序或帐户可能会利用漏洞，使其消耗不合理的巨量资源。当此类问题不可避免地发生时，区块生产者应当有权力纠正。

所有区块链上的区块生产者都有权选择将哪些事务包含在区块之中，这给予了他们冻结账户的能力。使用 EOS. IO 软件的区块链将这一权力正式化了，对一个账户冻结的决策会提交给 21 个节点进行投票，如果得到 15 个节点及以上的通过，则会冻结账户。如果出块者滥用权力，他们可以被投票换掉，而冻结的账号也会解冻。

## **Changing Account Code 更改账户代码**

当其他一切都失败了，而“无法停止的应用程序”以一种不可预知的方式运行时，使用 EOS. IO 软件的区块链，允许区块生产者在不需硬分叉整个区块链的情况下就能替换掉帐户的代码。与冻结帐户的过程类似，替换账户的代码需要得到被选中出块节点中 17 / 21 个节点的投票同意。

## **Constitution 宪法**

EOS. IO 软件使得区块链可以在签名用户之间建立 P2P 的服务协议或约束性合约，即所谓“宪法”。宪法内容定义了无法通过代码来强制履行的用户义务；通过确立司法权和适用法律以及其他公认的规则，促进争议的解决。每一笔在网络中广播的事务，其签名信息中必须包含宪法的哈希值，因此明确的将签名者绑定至合约。

宪法还定义了源代码协议的人类可读的意图(intent)。在错误发生时，这一意图用于分辨 bug 和特性的区别，帮助社区确定什么样的修复措施才是合理的。

## **Upgrading the Protocol & Constitution 协议和宪法的升级**

EOS. IO 软件定义了如下过程，借助于此，可以对由源代码和宪法所定义的协议，进行变更：

1. Block producers propose a change to the constitution and obtains 15/21 approval.
2. Block producers maintain 15/21 approval of the new **constitution** for 30 consecutive days.
3. All users are required to indicate acceptance of the new constitution as a condition of future transactions being processed.
4. Block producers adopt changes to the source code to reflect the change in the constitution and propose it to the blockchain using the hash of the new constitution.
5. Block producers maintain 15/21 approval of the new **code** for 30 consecutive days.
6. Changes to the code take effect 7 days later, giving all non-producing full nodes 1 week to upgrade after ratification of the source code.
7. All nodes that do not upgrade to the new code shut down automatically.
8. 区块生产者提出变更宪法的动议，获得出块者中 15/21 的投票通过。
9. 区块生产者对新 **宪法**的赞成态度，需要维持持续 30 天。
10. 所有的用户都需要表示接受新的宪法，作为未来的交易能够处理的条件。
11. 区块生产者采纳对源代码的变更以反应宪法的变化，并用新宪法的哈希值将变更提交到区块链上。
12. 区块生产者对新**代码**的赞成态度，需要维持持续 30 天。
13. 代码的修改 7 天之后生效，源代码通过后，给非出块的全节点一周的时间进行更新。
14. 所有未升级代码的全节点，会自动关闭。

根据 EOS. IO 软件的默认配置，更新区块链增加新特性的过程往往耗时 2~3 个月，而只需要进行非关键问题的修复却不需要修改宪法的更新，需要 1 到 2 个月时间。

## Emergency Changes 紧急情况下的变更

如果需要进行软件变更，以便修复正在损害用户利益的有害漏洞或安全漏洞，区块生产者可以加速这一变更过程。通常而言加速新特性更新过程或修复无害的 bug，都是违反宪法的行为。

## Scripts & Virtual Machines 脚本 & 虚拟机

EOS. IO 软件首先是一个平台，协调已认证信息(称为 Action，动作)在账户间的传递。脚本语言和虚拟机的实现细节将独立于 EOS. IO 技术。任何开发语言或虚拟机，只要具有确定性，经过了恰当的沙盒化并具有足够的性能，都可以与 EOS. IO 软件的 API 集成。

## Schema Defined Actions 模式(schema)定义的动作(Action)

在账户之间发送的所有动作都是通过模式(schema)来定义的,这是区块链共识状态的一部分。这一模式(schema)使得 Action 可以在二进制和 JSON 格式之间无缝转换。

## Schema Defined Database 模式定义的数据库

数据库状态也是由类似的模式所定义的。这确保了所有应用程序所存储的所有的数据,都能够以人类可读的 JSON 格式解析,又利用了二进制文件的高效性进行操作和存储。

## Generic Multi Index Database API 通用的多重索引数据库API

开发智能合约需要定义好的数据库模式,用于追踪,存储和寻找数据。开发者通常需要按照多个字段对相同的数据排序或索引,并维持所有索引的一致性。

## Separating Authentication from Application 验证与应用程序相分离

为了尽可能优化并行运算,并把从程序日志中重新生成应用程序状态时相关的计算债务( computational debt )降至最低, EOS. IO 软件将验证逻辑分为三个部分:

1. Validating that an Action is internally consistent;
2. Validating that all preconditions are valid; and
3. Modifying the application state.
4. 验证动作( Action) 的内在一致性;
5. 验证所有前置条件的有效性; 以及
6. 修改应用程序的状态

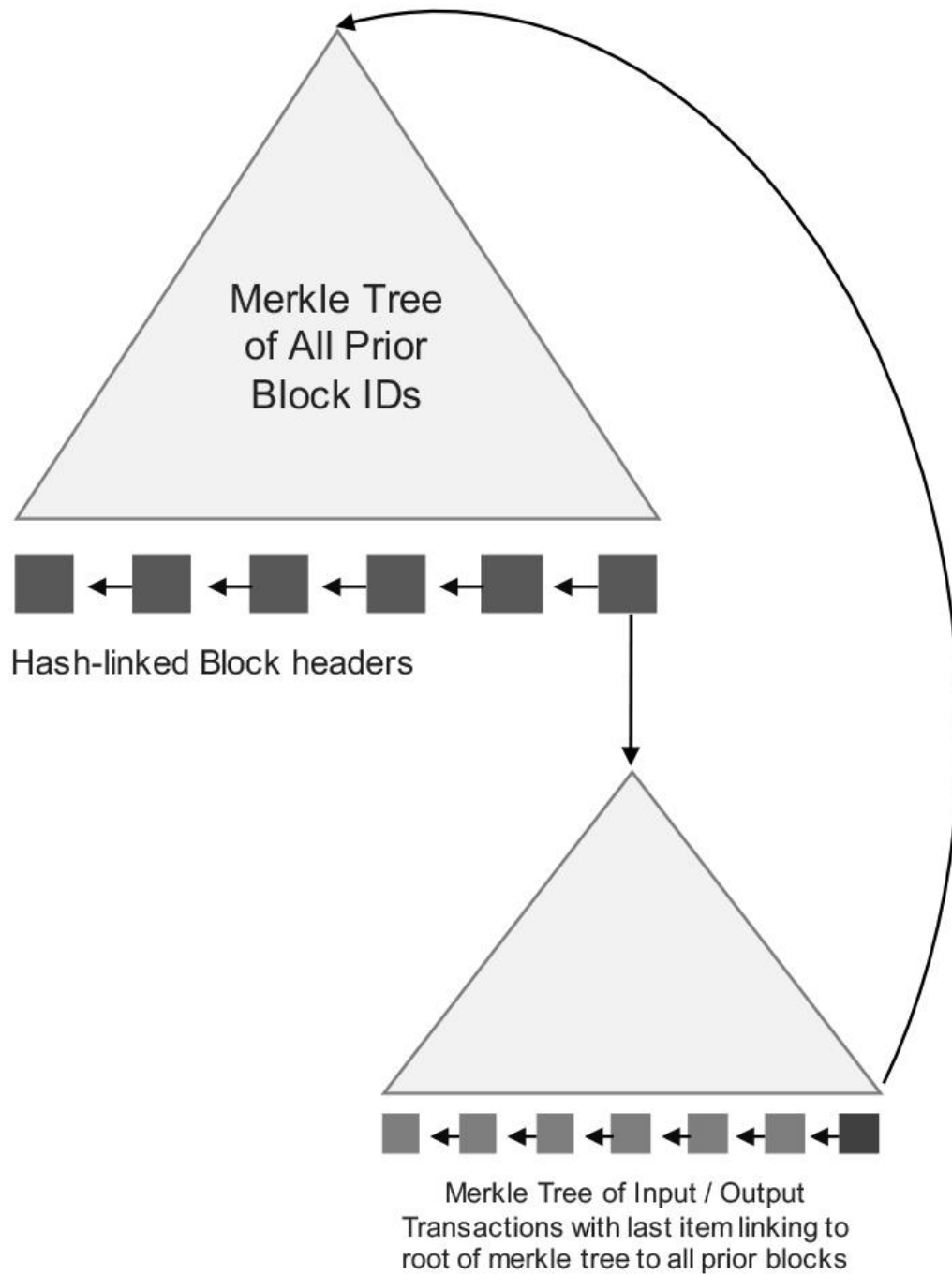
对 Action 内部一致性的验证是只读的,不需要访问区块链状态,这意味着它执行时,可以将并行运算最大化。对前置条件(如所需的账户余额)的验证也是只读的,因此也可以受益于并行运算。只有对应用程序状态的修改才需要写的权限,必须对每个程序顺序处理。

验证(Authentication)是一个只读过程,用于检验一个动作(Action)是否可以生效。应用(Application)是实际起作用的过程。这两种计算都需要实时进行,不过,一旦一笔事务包含在了区块链之中,就不需要再进行验证操作了。

## Inter Blockchain Communication 区块链跨链通讯

EOS. IO 软件旨在促进区块链间的跨链交互,这通过简化 Action 存在证明(proof of Action existence)和 Action 顺序证明(proof of Action sequence)的生成过程来实现。这些证明与围绕 Action 传递而设计的应用架构结合起来,将跨链通讯以及验证证明的细节对应用的发者隐藏,展现给开发者的是高层次的抽象。

## Merkle Proofs for Light Client Validation (LCV) - 用于轻客户端验证的 Merkle 证明(LCV)



如果客户端不需要处理所有的事务(transaction)，那么，与其他区块链交互将变得非常容易。毕竟，一个交易所只会关注交易所的出账和入账信息，而不会关心其它。更理想的情况是，对于交易所自身所维持的链来说，如果可以将轻量级的默克尔存款证明应用其中，那么就不必完全依赖自己的区块生产者。至少，某个区块链的生产者在同步另一条区块链时，会希望尽可能减小开销。

LCV 的目标是能产生相对轻量级的交易存在证明，其他人只需要追踪一个相对轻量级的数据集，就可以对此进行验证。既然如此，目标就是证明一笔特定的事务被一个特定的区块所包含，并且 某一条特定的区块链的验证历史中，已经包含了该区块了。

比特币的轻量级验证方式是，假设所有节点都可以读取区块头数据的完整记录，区块头数据每年增长 4MB。假设每秒产生 10 笔交易，一个有效的证明需要 512 bytes，这对于一个出块时间为 10 分钟的区块链来说是可行的。但对于一个出块时间为 0.5 秒的区块链来说，这就远远谈不上“轻量”了。

EOS. IO 软件的轻量级证明中，在某笔交易被包含到区块链之后，只需要对任意一个不可逆的区块头进行验证即可。使用下图中的哈希链表架构，可能只需要不到 1024 个字节大小的证明，就可以验证任意一笔交易的存在。

给定区块链上任意一个区块的区块 id，以及一个不可逆的可信区块的区块头。可以证明某个区块是包含在区块链上的。这一证明的算法复杂度是  $(\log_2(N))$ ，其中  $N$  是区块链上的区块数量。给定 SHA256 加密算法的类型，只用 864 个字节，你就能够证明在一条包含了 1 亿个区块的链上，一个任意的区块是否存在。

生成区块的时候如果使用合适的哈希链表来产生这些证明，只会带来很小的增量开销，这意味着没有理由不以这种方式去生成区块。

对其他链上的证明进行验证时，在时间、空间和带宽方面都有很大的优化空间。跟踪所有区块头数据(420 MB/年)可以让证明的尺寸最小。只跟踪最近的区块头，可以在长期存储文件的最小化和证明尺寸的最小化之间，得到均衡。或者，一个区块链可以采用惰性评估的方式，只记录过去证明的中间哈希值。新的证明只需要包含指向已知的 sparse tree（稀疏树）结构的链接。实际使用的方式，需要根据在 merkle 证明中所引用的交易位于外链上所占的比例来决定。

在一条区块链上，可以将另外一条区块链的所有区块历史都包含其中，不需要再进行跨链的证明。在跨链关联密度达到一定程度之后，这会是更高效的做法。出于性能考虑，理想情况是尽可能降低将跨链证明的频率。

## Latency of Interchain Communication 跨链通讯延迟

与外部区块链通讯时，区块生产者必须等到一笔事务(transaction)经过外部区块链的确认，达到了 100%的不可篡改的确定性之后，才能够接受该笔事务，认可为有效的输入。在一条基于 EOS. IO 软件的区块链上，借助于 DPOS 0.5 秒的

出块速度，并使用了额外的拜占庭容错的不可篡改性，这一过程大约耗时为 0.5 秒。如果某个链上的区块生产者不等到交易不可篡改，就像一个交易所接受了一笔存款而后这笔操作又撤销了的情况一样，这会影响这条链共识的有效性。EOS. IO 软件使用了 DPOS 和 aBFT(异步拜占庭容错)算法，提供快速的不可篡改性。

## Proof of Completeness 完成性证明

使用外部区块链的 merkle 证明，知道所有处理过的事务都是有效的，和知道没有事务被跳过或忽略，这两者之间有明显差别。虽然无法证明最近的所有事务都是已知的，但是，要证明在事务的历史中不存在遗漏，还是可能的。EOS. IO 软件为传递给每个账户的每个 Action 都指定了一个序列号，使得这一点成为可能。用户可以用这些序列号来证明，与某个账户相关的所有的 Action 都得到了处理，并且是按顺序处理的。

## Segregated Witness 隔离见证

隔离见证(SegWit) 是指，在不可逆地将一笔事务包含到区块链中之后，事务的签名不再具有相关性了。一旦事务具有不可篡改性了，签名数据可以被剪掉，其他所有人都能够达成当前的状态。由于在大多数的事务之中签名数据占据了很大一部分，SegWit 能够明显地降低磁盘使用量，减少同步时间。

同样的概念可以用在跨链通讯的 merkle 证明中。一旦一笔证明被接受，并不可逆的记录到区块链之后，想达成正确的区块链状态，用作证明的 sha256 哈希的 2kb 文件就不再需要了。同样使用隔离见证，给跨链通讯所带来的节省程度，是给普通签名带来节省程度的 32 倍。

隔离见证的另外一个例子是 Steem 的文章。在这一模式下，(在区块链中)一篇帖子只包含博客内容的 sha256 哈希函数值，博客正文会被放在隔离见证数据中。区块生产者只需要验证内容是存在的，并且具有给定的哈希值，而不需要在链上存储博文的内容，就能够从区块链的日志中恢复到当前的状态。这样做可以在不必永久保存博文内容的情况下，就能够证明所说的内容曾经存在过。

## Conclusion 结论

EOS. IO 软件是基于已为实践所证实的概念和最优实践的经验来设计的，代表着区块链技术的根本性进步。它是全球性可扩展的区块链社会宏伟蓝图的一部分，在其中，可以轻松部署和管理去中心化的应用程序。