Polkadot（波卡链）：畅想一种异构的多链架构

草案 1 Gavin Wood 博士
以太坊&Parity 创始人 GAVIN@PARITY.IO

【原文链接：Paper】
【译者：岳利鹏 lipeng@chainx.org】

摘要：现有的区块链架构都存在诸多问题，不仅仅是从实用性角度所说的扩展性（extensibilty）和伸缩性（scalability）的问题。我们认为，问题源于把共识架构中两个很重要的部分：一致性（canonicality）和有效性（validity）绑定得太紧密了。这篇文章介绍了一种异构的多链架构，能从本质上把两者拆开。

为了分离这两者，且能保持最小化的绝对安全性（security）和传输性（transport）等基本功能，我们将介绍一种原生的支持内核可扩展（core extensibilty）的可行性方法。对于可伸缩性（scalability）的问题，我们通过对这两个问题分而治之的思路解决，通过非信任节点的激励机制，弱化他们的内生绑定关系。

本架构的异构本质，支持众多高度差异化的共识系统在非信任（trustless）、完全去中心化的联邦内交互操作，允许去信任（trust-free）地相互访问各区块链。

我们提出一种方式，支持向后兼容一个或多个现有的网络，比如以太坊等。我们相信这个系统能够提供一种有用的底层组件，能够实用性地支持全球商业级别的可伸缩性（scalability）和隐私性（privacy）。

1 前言

这篇论文的意图只是一个技术版本的概要，旨在用一些原则来描述将要开发的这个区块链示例，解释这个可能方向的合理性。它罗列了诸多区块链技术方面的具体改善措施，以及在此开发阶段所能够提供的尽可能多的细节。

它并不是要写成一个形式化证明的说明书。它并不完整，也不是最终版本。它并不是为了覆盖框架非核心的模块，例如 API、依赖、语言和用法等。这只是概念性实验，都很可能会修改提到的参数。为了响应社区的意见和评论，会新增、重定义、删除各组件。通过实验性的证据和原型，给出关于什么会有效、什么不会的信息，也很可能修正本论文中大部分内容。

这篇论文包含了一个关于协议和一些想法的核心描述，可能会被用来解决多个方面的问题。它将是能够用来在概念验证阶段开展一系列工作的核心描述。一个最终的“1.0 版本”会基于这个协议，再添加一些变得可证明而且决定包含到项目中来的想法。

1.1 历史

- 2016 年 10 月 09 日：0.1.0-proof1

- 2016 年 10 月 20 日：0.1.0-proof2
- 2016 年 11 月 01 日：0.1.0-proof3
- 2016 年 11 月 10 日：0.1.0

2 介绍

区块链已经承诺了它的伟大意义，能够应用于包括物联网（IOT）、金融、治理、身份管理、去中心化互联网和资产追踪等多个领域。然而抛开这些技术承诺和大话，我们还没有看到现有技术下，出现重大的关于现实世界的应用部署。我们相信这是因为现有技术的 5 个关键缺陷：

伸缩性（Scalability）：全球范围内花费了多少计算、带宽和存储的资源，来处理单个交易？峰值情况下能处理多少交易？

隔离性（Isolatability）：多参与方对于应用的差异化需求，能否在同一个框架下接近最优程度地满足？

开发性（Developability）：工具的工作效果有多好？APIs 是否已满足开发者的需求？教程资料是否可用？是否集成权力？

治理性（Governance）：网络是否保留了能够随着时间进化和适应的灵活性？制定决策能否高度地包容、合理和透明，来提供去中心化系统的高效领导力。

应用性（Applicability）：技术是否真的解决了用户的刚性需求？是否需要其他的中间件来嫁接真实的应用？

当前的工作，我们主要面向前两个问题：伸缩性和隔离性。也就是说，我们相信 Polkadot 架构可以在这两个方面，提供有意义的改进。

当前，例如 Parity 以太坊客户端这样的高性能区块链实现，已经可以在消费级高速硬件上每秒处理超过 3000 笔的交易。然而现实世界的区块链网络却限制在 30 笔交易每秒的情况下。这种限制主要是源于目前同步（synchronous）的共识机制，需要充分的计算缓冲时间来安全地处理，也就加重了其必须对于慢速硬件的支持。这归咎于其底层的共识架构：状态转换机，或者这种让所有参与方校对和执行交易的方式，在本质上将其逻辑绑定在了共识一致性（canonicalisation）的设计上，或者需要让所有参与方都同意所有的可能性、有效性和历史。

这种说法即适用于类似比特币和以太坊这样的工作量证明（POW）系统，也适用于 NXT 和比特股这样的权益证明（POS）系统，他们都本质上受制于同一个障碍，但这些共识算法却是个能让区块链成功的简单策略。然而，在一个协议里紧密捆绑这两个结构，我们也就捆绑了多个不同风险偏好、不同伸缩性需求、不同隐私需求的角色和应用。一种特征满足不了所有人的需求。因为这种场景，产生了很多次的广泛呼吁，但网络只能倾向于更保守，服务于少数人，最终导致在创新能力、性能和适应性方面的失败，非常戏剧化。

有一些系统例如公证通（Factom），整个地去除了状态转换机。然而大多数应用场景都需要依赖一个共享的状态机，来支持状态转换的功能。去除它只是隐藏了问题，却没有给出真正替代性的解决方案。

现在看起来清楚了，因此一个合理的方向是：像路由对于可伸缩去中心化计算平台那样，解耦共识组件和状态转换组件。而且不出意外的话，这也是 Polkadot 解决伸缩性问题的策略。

2.1 协议、实现、网络

和比特币、以太坊一样，Polkadot 希望一开始的时候只是个网络协议，并且是运行这一协议的主要公有网络（目前假设）。Polkadot 倾向于是个免费和开放的项目，协议在一个知识共享的许可证上制定，代码托管在 FLOSS 许可证下。这个项目以一种开放的状态开发，接收各方面有用的捐助。一个微意见提交系统（RFCs），但不像 Python 改进议程那样，会提供一种公众协作参与协议修改和升级的方式。

我们对 Polkadot 协议的初始实现，将称为 Parity Polkadot Platform，会包含协议的完整实现和 API 接口。像其他 Parity 的区块链实现一样，PPP 会设计成通用目的的区块链技术栈，并不限定于公有网络、私有网络或联盟网络。目前为止的开发已经被包括英国政府在内的几方资助。

但是，这篇论文还是在公有网络的场景下。我们在公有网络下预见的功能，是一个完整设想（比如私有或联盟网）的子集。另外在这个场景下，可以清晰地描述和讨论 Polkadot 的所有方面。这也是说读者需要知道，在非公有（有权限的）场景下，一些特定的机制（比如和其他公有网络的交互）并不直接和 Polkadot 相关。

2.2 前人工作

从状态转换中解耦底层的共识，已经私下讨论了两年，在以太坊的最早期的时候 Max Kaye 就提议过。

一个更复杂的可伸缩方案叫做 **Chain fibers**，这要回溯到 2014 年 6 月，在那年底也发表了。它创造了一个关于单个中继链（relay-chain）和多个同构链，可以透明地跨链执行的先例。退相干性（Decoherence）通过交易延迟（latency）来实现，这就使需要更长时间，来处理需要协调系统多个部分的交易。Polkadot 借鉴了很多它的架构以及随后跟很多人的讨论，虽然跟它的设计和规定也很不一样。

然而目前并没有运行在生产环境下的系统可以和 Polkadot 相比，有的也只是提出了些相关性功能，很少有本质层面的细节。这些提议可以归纳为：丢弃或减少状态机全局相关性的系统、试图通过同构分片提供全局相关性的单例状态机系统、目标仅是异构性（heterogeneity）的系统。

2.2.1 没有全局状态的系统

公证通（Factom）演示了个没有有效性的一致性系统，能够高效地记载数据。由于没有全局状态和其带来扩展性问题，它可以被看做是一个可伸缩的方案。然而前面也提到了，严格上来说它只解决了很少的问题。

Tangle 是个关于共识系统的概念性尝试。不把交易排序再打包到区块中，而是通过串联的共识得出一个全局的一致性状态改变排序，它在很大程度上抛弃了高度结构化的排序想法，而是推出一个有向无环图，后续的有依赖的交易通过明确的指向，来帮助前面的交易达成一致。对于任意的状态改变，这个依赖图就会很快地变得无法处理，然而对于更简单的 UTXO 模型，立即就变得合理了。因为系统总是松散地连贯，而且交易通常是相互独立的，大规模的全局并发变得非常自然。使用 UTXO 模型确实可以让 Tangle 定位成价值转移的货币系统，而并没有其他的更多通用和可扩展的功能。因为没有了全局依赖性，而和其他系统的交互又需要确定性地知道其状态，这种方法就变得不切实际了。

2.2.2 异构链系统

侧链是个支持比特币主链和附属链之间去信任交互的提案。但并没有任何和侧链进行富（rich）交互的具体规定：交互被限定在允许和侧链之间相互托管对方的资产，也就是行话所说的双向锚定（two-way peg）。最终也是为了做个框架，通过锚定比特币链和其他链，允许在比特币协议之外进行外部交易，为比特币添加附属的外围功能。从这方面讲，侧链系统更多着眼于可扩展性而不是可伸缩性。

根本上将，侧链确实没有关于有效性的条款，从一条链（比如比特币）的代币转到另一条链上，安全性只是寄希望于侧链能否激励矿工来一致性地验证交易。比特币网络的安全性无法简单地在其他链上起作用。进而一个确保比特币矿工联合挖矿（复制他们的一致性算力到侧链上），并且同时验证侧链交易的协议也被提出来了。

Cosmos 是个延续侧链思路提出来的多链系统，替换中本聪的 PoW 共识算法为 Jae Know 的 Tendermint 共识算法。本质上，它包含多个使用独立 Tendermint 实例的区块链（在空间 zone 中运行），和一个使用去信任通信的中心（hub）链。跨链通信仅限于转移数字资产（也就是代币），而不是任意信息，然而这种跨链通信是可以返回数据和路径的，比如给发送人通知转账的状态。

和侧链一样，空间链上验证人的经济激励问题也没有解决。一般的假设是每个空间链会各自持有通胀增发的支付代币。设计仍然还比较早期，现阶段的也缺乏在全局有效性上建立可伸缩一致性的经济手段细节。然而相比于那些需要强耦合的系统，为了空间链和中心链间的松耦合性，需要给空间链的参数添加更多灵活性。

2.2.3 Casper

目前关于 Casper 和 Polkadot 之间，还没有完整的讨论和比较，即使是公平和彻底（也不准确）地描述两者。Casper 是正在重塑 PoS 的共识算法，它研究如何让参与方在最终会确定的分叉上押注。本质上，需要考虑即使是长程攻击的情况下，也要保证应对网络分叉的健壮性，还需考虑基础以太坊模型上的可伸缩性。因此，在本质上 Casper 协议的目标比 Polkadot 和以往项目要复杂的多，也偏离了基础的区块链模型。它仍然还没有做出来，不知道将来如何运作，也不知道最终会开发出来的样子。

然而 Casper 和 Polkadot 都代表了有趣的新一代协议，对于以太坊的争论，本质上也是他们的终极目标和实现路径上的差异。Casper 是以太坊基金会主导的一个项目，只是被设计用来作为 PoS 协议的替代，没有从本质上打造可伸缩区块链的意愿。关键还需要一次硬分叉来升级，而不能时可扩展的，因此所有的以太坊客户端和用户都需要升级，否则就得留在原来的前途不明朗的分叉上。因此，这类协议在去中心化系统上的部署会很困难，需要紧密的协调。

Polkadot 在几方面上不同；首先而且也是最重要的，Polkadot 将被设计成完全可扩展和可伸缩的区块链开发、部署和交互测试平台。他将被设计为面向未来的、可以吸收最新的可用区块链技术的平台，且不需要过于复杂的去中心化协调和硬分叉。我们已经预见到了几个应用场景，例如高度加密的联盟链和低区块时间的高频链等，它们不太可能在近期的以太坊上实现。它们最终和以太坊之间的耦合度也会很低，以太坊上也没有支持两者间非信任交互的想法。

简言之，尽管 Casper/以太坊 2.0 和 Polkadot 有一些相似点，我们相信从本质上它们最终的目标是不一样的，并非竞争，在可预见的将来，两个协议会大概率地并存。

3 概要

Polkadot 是一个可伸缩的异构多链系统。这意味着不像以往那些专注于不同程度潜在应用功能的单个区块链实现，Polkadot 本身被设计成不提供任何内在的功能应用。Polkadot 提供了中继链（relay-chain），在其上可以存在大量的可验证的、全局依赖的动态数据结构。我们称这些平行的结构化的区块链为平行链（parachains），尽管也不要求它们必须是一条链。

换句话说，Polkadot 会被设计成一个独立链的集合（例如包含以太坊、以太坊经典、域名币、比特币），除了两个非常重要的点：

- 合并的安全性
- 去信任的跨链交易性

这两点也是我们称 Polkadot 为可伸缩的原因。从原则上，一个问题在 Polkadot 上被彻底解决了：可以向外扩展，会有非常大数量的平行链。尽管每条平行链在各方面都通过不同的网络模式进行平行管理，但这个系统却有可伸缩的能力。

Polkadot 提供了一个尽量简单的架构，把大部分的复杂性都放在了中间件上。这是个刻意的决定，为了试图减少开发的风险，使必备的软件可以在短时间内开发出来，还能对安全性和健壮性持有信心。

3.1 Polkadot 的哲学

Polkadot 需要提供一个绝对坚实的基座，来在其之上建设下一代共识系统，覆盖从生产级别的成熟设计到初期想法的所有风险。通过对安全性、隔离性、通信能力提供强有力的保证，Polkadot 能够允许平行链从一系列特性中选择适合它们自己的。的确，我们预见了各种实验性的经过考虑的区块链特性。

我们看到，传统的高市值区块链（例如比特币和 Zcash）、低市值的概念性区块链和接近零手续费的测试网，是并存在一起的。我们看到，全加密的暗黑联盟链和高功能性的开放区块链（例如以太坊）也并存在一起，甚至还为之提供服务。我们看到，实验性的新虚拟机区块链，比如主观时间计费的 Wasm 区块链，在将难度计算问题从类似以太坊的区块链方式，修改成类似比特币的区块链方式。

为了管理区块链升级，Polkadot 将内生支持某种形式的治理结构，很可能基于现有的稳定政治体系，会有一个两院结构，类似于 Yellow Paper Council。底层权益代币持有者作为最高权力机构，会有全民投票控制权。为了反映用户的需求、开发人员的需求，我们期望建立一个合理的两院结构，采纳用户的意见（由绑定的验证人决定）、主要客户端开发者和生态系统玩家的意见。代币持有者会保留最高的合法权，可以形成一个最高法庭来参政、议政、替换或解散这个架构，还有那些我们不怀疑的最终需求。

借用一句马克吐温的谚语：“政府和尿布都得经常换，而且理由都一样”。

然而在大范围共识的机制下组织参政会很琐碎，更多关于替换和新增的质的改变，希望既不是通过非自动的弱法令（例如通过块高度和新协议的形式化证明文档的哈希）来达到一致性，也不是通过在核心共识算法中包含一个高效的高级语言，来改变他自身可能需要改变的各个方面。后者是一个最终目标，然而为了落实一个合理的开发路线图，更可能选择前者。

Polkadot 看重的主要原理和规则有：

最小：Polkadot 需要有尽可能少的功能性。

简单：只要他们可以推给中间件、放在平行链、或用下面要讲的一种优化手段，就不在基础协议里添加多余的复杂性。

通用：没必要在平行链中添加任何要求、约束或限制；Polkadot 需要成为共识系统开发的基石，要尽量通过给模型加入最具适应度的扩展和优化。

健壮：Polkadot 需要提供一个稳定的基础层。为了经济稳定性，需要采用分散的方法，来降低高额奖励这个攻击向量可能引发的问题。

4 Polkadot 的参与方

有四个基本的角色在维持 Polkadot 网络：收集人（collator）、钓鱼人（fisherman）、提名人（nominator）、验证人（validator）。在 Polkadot 的一个可能实现里，最后一个角色有可能会被拆分成两个：基础验证人和可用保证人（guarantor），将会在 6.5.3 节讨论。

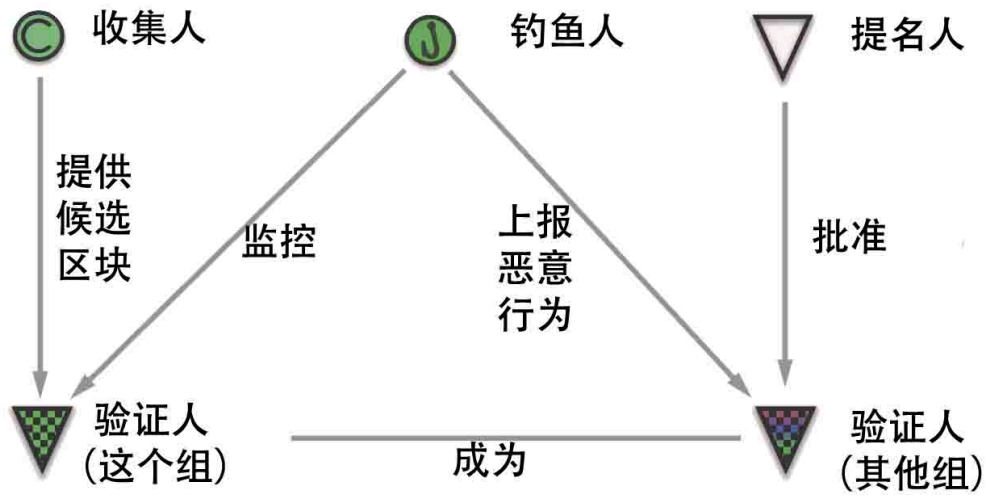

图 1. Polkadot 四个角色的交互

4.1 验证人

验证人有最高权限，帮助在 Polkadot 网络里打包新区块。验证人需要抵押足够多的押金，因为我们允许其他有资金的提名人推举一个或多个可以代表他们的验证人，所以验证人一部分的押金并不是他们自己所拥有的，而是属于提名人的。

一个验证人必须在高可用和高带宽的机器上运行一个中继链的客户端。每个区块上，节点都必须准备接收一个已提交的平行链上的新区块。这个过程涉及接受、验证、再发布候选区块。验证人的任命是确定性的，但实际上也很难预测。因为不能期望验证人拥有所有平行链的全同步数据，所以他们希望把这个提议平行链新区块的工作指派给第三方，也就是收集人。

不同的验证人小组一旦都确定性地批准了自己所属平行链的新块，他们就必须开始批准中继链自身的区块。这包括更新交易队列的状态（也就是从一条平行链的出口队列转移到另一条平行链的入队列）、处理已批准的中继链的交易集合、批准最终的区块、吸收平行链的最终改变。

在我们选择的共识算法下，会惩罚一个没有履行他们职责的验证人。最开始如果不是有意的错误，就只是会扣留他们的奖励，但如果是重复的错误会扣减他们的押金（通过烧

毁），例如双向签名（double-signing）或合谋提供一个非法区块等可证明的恶意行为，会导致他们丧失全部的押金（烧毁一小部分，大部分奖励给信息提供方和诚实的验证人）。

在某种程度上，验证人和目前 PoW 区块链的矿池相似。

4.2 提名人

提名人是一个拥有权益的群体，他们把安全性押金委托给验证人。他们没有更多的角色，除了通过有风险地投放资本来表示：他们信任某个特定的验证人（或群体）可以代表他们维护整个网络。按照他们的入金比例，他们也会受到和验证人总押金同样比例的奖励和扣减。

和下面的收集人一样，提名人和目前 PoW 网络的矿工相似。

4.3 收集人

交易收集人是帮助验证人制造有效的平行链区块的群体。他们会运行一个特定平行链的全节点，这也意味着他们有全部的必要信息，可以打包新块并执行交易，就跟目前 PoW 区块链的矿工一样。在正常情况下，他们会收集并执行交易，并创建一个”未密封”（unsealed）的区块，再加上一个零知识证明一起提交给一个或多个当前负责提议（proposing）该平行链区块的验证人。

关于收集人、提名人、验证人的精确关系可能还会修改。起初，我们希望收集人和验证人能够紧密合作，因为可能只有一些（甚至一个）交易量很小的平行链。最初的客户端实现会包含一个 RPC 接口，来支持一条平行链的收集人节点把可证明的有效平行链区块，无条件地提供给一个（中继链）验证人节点。由于维持所有的全同步平行链的成本越来越高，所以我们设计了附加的结构，有助于分离独立的、经济驱动的、和其他的参与者。

最终，我们希望看到收集人群体为了更多手续费，竞争性地去收集信息。在一段时间内，为了持续增长的份额收益奖励，这些收集人可能只服务于特定的验证人群体。或者自由职业（freelance）的收集人也可以简单地创建一个市场，提供有效的平行链区块，而不是获得立即支付的竞争性份额奖励。同样地，去中心化的提名人群体也会允许多个有抵押的参与者来协调和分担验证人的职责。这种能力保证了参与的开放度，有助于成为更加去中心化的系统。

4.4 钓鱼人

不像其他的两个参与方，钓鱼人并不直接和区块打包的过程相关。他们是独立的“赏金猎人“，激励他们的是一次性的大额奖励。

准确地说，由于钓鱼人的存在，我们才能减少恶意行为的发生，即使发生希望也只是因为私钥不小心泄露了，而不是故意的恶意企图。起这个名字的出发点是考虑到他们期望收益的频率和最终奖励的大小。

钓鱼人只要及时举报并证明至少一个有抵押的参与方存在非法行为，他们就能获得奖励。非法行为包括对两个有相同父块的不同区块进行签名，或在平行链上批准一个无效区块。为了预防由于私钥泄露给钓鱼人所导致的过渡奖励，钓鱼人上报关于单个验证人的非法消息签名的基础奖励是从最小开始的，这个奖励会随着其他钓鱼人上报更多的非法签名而逐渐增加。依据我们基本的安全性假设：至少三分之二的验证人是诚实的，渐近线将设置在 66%。

钓鱼人某种程度上和目前区块链系统的全节点相似，他们所需要的资源相对较少，也没必要承诺稳定的在线时间和大的带宽。钓鱼人有如此大的不同，所以他们只需要提交很少的押金。这个押金用于预防浪费验证人计算时间和计算资源的女巫攻击。它是立即可以提现的，很可能不会比等值的几个美金更多，但如果监测到一个不当行为的验证人，可能会收获很大的奖励。

5 设计综述

本章试图给出一个系统的全局完整描述。对系统更加深入的解释会在接下来的一章中给出。

5.1 共识

在中继链上，Polkadot 通过一个现代的异步（asynchronous）拜占庭容错（BFT）算法达成对有效区块的相互共识。算法受简单的 Tendermint 和 HoneyBadgerBFT 启发。后者在有任意网络缺陷的架构下，只要满大部分验证人是诚实的，就能提供了一种高效的容错算法。

也许一个权限证明（PoA）模式的网络就足够了，然而 Polkadot 是个可以在全开放和公开的场景下部署的网络，不需要信任任何特殊的组织和当权者来维护它，因此我们需要一种管理验证人群体并且激励他们守法的方法。我们选择使用以 PoS 为基础的共识算法。

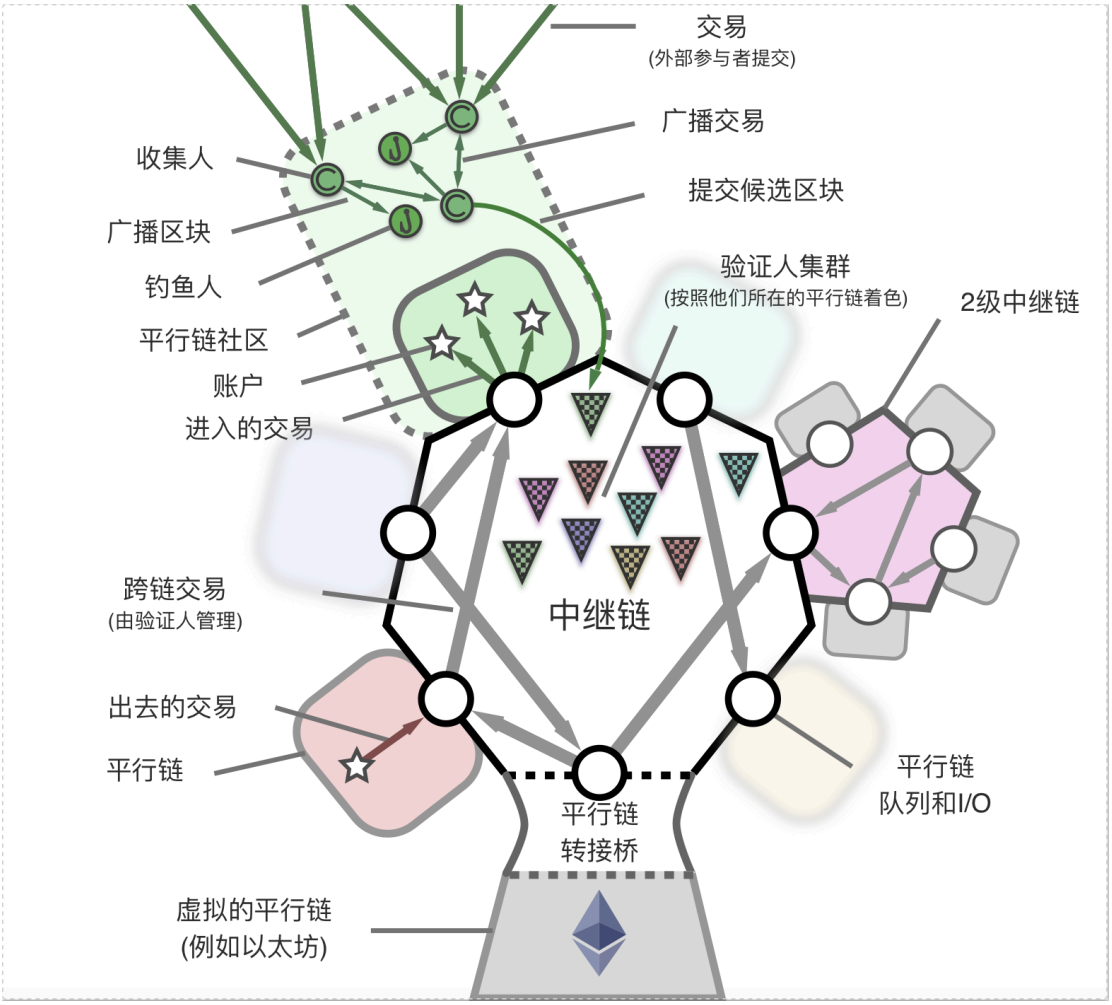

图 2：Polkadot 的概括性原理图。它展示了收集人收集并且广播用户的交易，也广播候选区块给钓鱼人和验证人。展示了用户提交一个交易，先转移到平行链外部，然后通过中继链再转移到另一条平行链，成为一个可以被那里的账户执行的交易。

5.2 权益证明

我们假设网络可以度量每个账户有多少权益（stake）。为了更轻松地和现有系统对比，我们把度量单位称为“代币”（tokens）。不幸的是由于它仅仅能作为对账户简单的价值度量，也没有任何个性化，因此多种原因使这个术语并不那么理想化。

通过一个被提名的权益证明（Nominated Proof-of-Stake NPos）结构，我们猜想验证人的选举不会很频繁（很可能是一个季度一次，最多一天一次）。通过按比例分配的增发出来的代币（很可能大约 10%，最多每年 100%）和收集到的交易手续费来进行激励。虽然货币增发一般都会造成通胀，但因为所有代币持有者都有公平参与的机会，所以代币持有者的资产不会随着时间而遭受损失，他们会很开心地参与到该共识机制中来。全网权益证明的开展所需的抵押必须达到一个特定的最小比例。会根据市场机制，达到有效的代币增发这个目标。

验证人严重依赖他们抵押进来的权益。现存验证人的押金会从他们离职的时候开始，要再保留更长时间（也许 3 个月左右）。这么长的押金冻结期是为了还能惩罚将来的不当行为，直到区块链周期性的检查点到来。不当行为会遭到例如减少奖励等的惩罚，如果是故意破坏网络的完整性，验证人将会损失部分或全部的权益，转移给其他验证人、信息提供者或全部权益持有者（通过烧毁）。例如一个验证人试图同时批准不同分叉上的两个分支（有时也被称为短程攻击），就会被后面的方法鉴别并遭到惩罚。

检查点锁定器（checkpoint latch）能规避长程“无权益抵押”（nothing-at-stake）攻击，防止比一般长度更长的高度危险的链重构（chain-reorganistation）发生。为了保证最新开始同步的客户端不会被误导进错误的链，网络会出现定期的“硬分叉”（最长也就是验证人的押金冻结期），把最近检查点区块的哈希值硬编码（hard-code）进客户端。将来通过逐步递减有限链的长度（finite chain length），或周期性地重置创世块（genesis-block），这种方法会运行得很好。

5.3 平行链和收集人

每条平行链将给中继链提供同样的安全性保证：平行链的区块头会被包含进中继链的区块中，还还跟着一些确认信息，用来保证不会发生链重构或双重花费（double-spending）。类似于比特币侧链和联合挖矿的安全性保证，Polkadot 也强力保证平行链状态交易的有效性。会依据密码学算法，把验证人随机地分成很多个组。一条平行链对应一组，甚至每个块的组也都可能不一样。这个设置意味着中继链至少也要和平行链的出块时间一样短。本文不讨论分组的特定决定方式，可能要么是围绕类似 RanDAO 的提交-披露（commit-reveal）框架，要么结合平行链前一个区块的密码学哈希值。

这样的验证人组需要提供平行链的候选块，还要保证它们是有效的（否则损失押金）。有效性围绕两个重要的点：第一，它是内生有效的，所有的状态转换被公正地执行，包括引用的外部数据也被公正执行（比如交易）。第二，参与方需要可以简便地访问候选块的任何外部数据，例如外部交易等，然后就可以下载这些数据并手工执行候选块。验证人可以提交没有包含任何外部交易数据的空块（null），如果他们这样做，就要承受奖励减少的风险。他们和收集人在平行链的一个 gossip 协议上工作，收集人把交易收集到块里，并且要提供一个非交互的零知识证明（noninteractive zero-knowledge），用来证明本子块的父块是有效的（为该工作收取任何手续费）。

防止垃圾（spam）数据的方法留给了平行链协议自身：中继链本质上不规定“计算资源计量”和“交易费”。本质上也不强制平行链规定相关协议（尽管权益持有者不太可能愿意接纳一个没有提供合理机制的平行链）。这里明确地说明了并不会都像以太坊的手续费规则，也可以类似比特币的区块链手续费模型，或其他任何还没有提出来的垃圾预防模型。

Polkadot 的中继链本身将很可能存在一个类似以太坊的账户和状态模型，可能是 EVM 的衍生版本。因为中继链节点将需要做大量的其他计算，将会通过提高手续费尽量减小交易吞吐量，我们的模型还会包含块大小的限制。

5.4 跨链通信

Polkadot 最关键的部分是跨链通信。因为在平行链间可以存在某种信息通道，我们才说 Polkadot 是可伸缩的多链系统。在 Polkadot 中，通信可以很简单：一条平行链中的执行交易的时候（依据那条链的逻辑），可以给第二条平行链或中继链转发一个交易。目前生产环境中的区块链外部交易，都只能是完全异步的，他们并没有给它的来源方返回任何信息的原生能力。

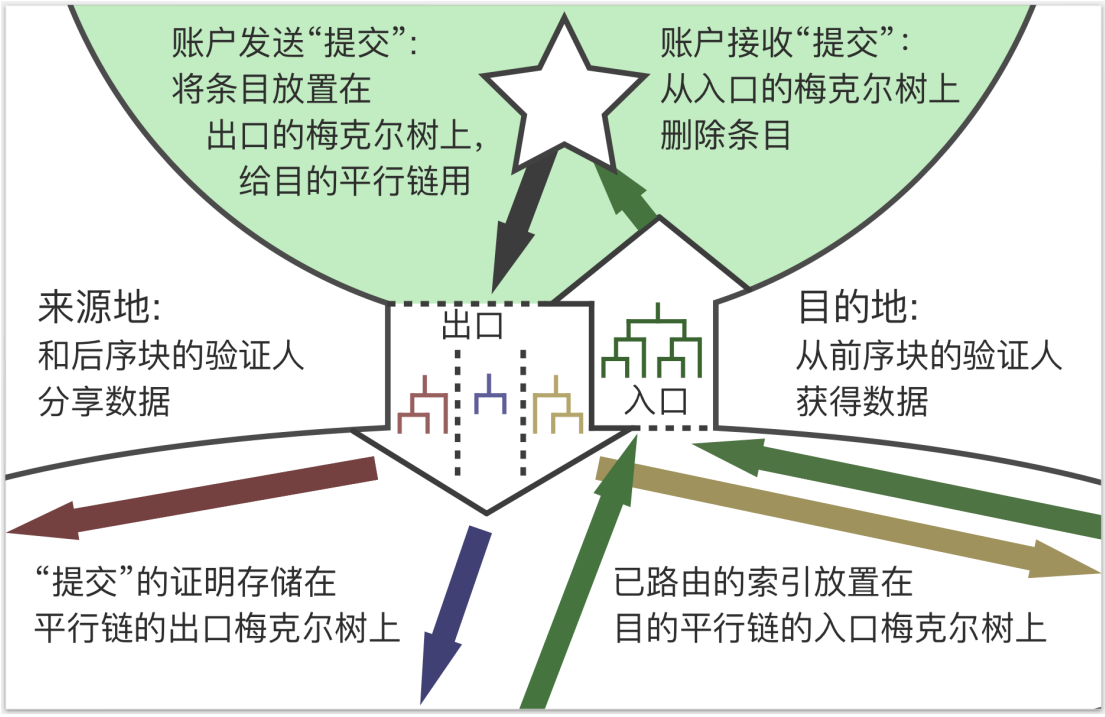

图 3：一个基本的原理图，展示了路由已提交的交易（“提交”）的主要逻辑

为了保证最小的实现复杂度、最小的风险和最小的平行链架构束缚，这些跨链交易和目前标准的外部交易没有区别。这些交易会有个来源方字段，用来辨别平行链的身份，还有个可以是任意长度的地址。跨链交易需支付的手续费，并不像目前的比特币或以太坊系统那样，而是必须通过来源平行链和目的平行链的谈判逻辑来管理。一个在以太坊的 Serenity 版本中提出的改进提案，会是一个简单管理这种跨链资源支付的方法，尽管我们假设其他人会提出更先进的方法。

跨链交易的问题可以用一个简单的队列机制解决，这个队列用梅克尔树（Merkle tree）来保证数据真实。中继链的任务是把交易从来源平行链的出口队列转移到目的平行链的入队列。已转发的交易会在中继链上被引用，而不是中继链自身的交易。为了预防一条平行链往另一条平行链发送垃圾交易，规定在在前一个块结束后，发送每一个交易时，

目标平行链的入队列不能太大。如果区块处理完后，入队列太大，那么目的平行链会被看做是饱和了，接下来的几个块里就不会再路由交易给它，直到入队列降到临界值以下。这些队列在中继链上管理，允许各平行链相互决定他们的饱和度大小。如果再往停滞的目标链发送交易，这样就可以同步地报告失败了（因为不存在返回路径，如果第二个交易也是同样的原因失败了，它可能也不会给来源调用者发送回复，这就需要用到一些其他的恢复方法）。

5.5 Polkadot 和以太坊

归功于以太坊的图灵完备特性，至少在简单的可论证的安全性边界内，我们期望 Polkadot 和以太坊有丰富的交互可能性。简而言之，我们预想到了，从 Polkadot 出来的交易，可以让验证人先签名，然后再喂给以太坊，在那里通过一个交易转发（transaction-forwarding）合约来解释和执行。反方向，我们也预想到了，从以太坊上的一个“外向合约”（break-out contract）中的特殊格式日志，可以快速地证明一个消息是否真的要被转发。

5.5.1 从 Polkadot 到以太坊

通过选择一个拜占庭容错算法，验证人经由授权投票产生的一系列权益持有者组成，我们能够获得一个安全的共识机制，用不经常更改的合适数量的验证人。在一个总共有 144 个验证人的系统内，4s 出块时间和 900 个块的最终性（允许举报、惩罚、修复类似双向投票的恶意行为），一个区块的有效性可以合理地考虑为用最少 97 个签名证明（144 的三分之二再加一）,然后跟着的是 60 分钟无风险注入的验证时间。

以太坊可以包含一个控制和维护 144 个签名的“内向合约”（break-in contract），由于椭圆曲线数字签名的验签操作只要花费 EVM 3000 gas 的计算量，而且因为我们只希望验证操作发生在大多数的验证人里（而不是全体），以太坊确认一个从 Polkadot 来的指令的基础花费不会超过 300,000 gas——仅仅是区块 550 万 gas 限制的 6%。增加验证人的数量（只有在处理数十个区块链的时候才必要）不可避免地会增加成本，然而很明显可以期望到随着以太坊技术的成熟和架构的改进，交易吞吐量会随着时间而增加。另一个事实是不是所有的验证人都会参与（例如只有最高押金的验证人才会做这个任务）这种结构的限制会比较合理。

假设这些验证人每天轮换（更保守的、更可能接收的是每周，甚至每月），网络给维持这个以太坊转接桥的成本大约是 540,000 gas 每天，或者按照当前的 gas 价格，45 美金一年。一个通过转接桥的基本转发交易会花费大约$0.11；当然另外的合约计算会耗费更多。通过缓存和捆绑多个交易，内向的交易花费可以简单地分担，减少每个交易的花费。如果一次转发需要凑够 20 个交易，那么转发一笔基本交易的花费会降低到大约 $0.01。

在这个模型中，Polkadot 的验证人节点除了签名消息之外只需要再做很少的事情。为了能够把交易路由到以太坊网络里，我们假设任何一个验证人需要属于以太坊网络，更可能的只需提供很少的奖励给第一个在网络上转发消息的人（奖励会支付给交易发起人）。

5.5.2 从以太坊到 Polkadot

使用一个叫做日志的概念，把交易从以太坊上转发到 Polkadot 上。当一个以太坊合约希望派生出一个交易给 Polkadot 上面的某一条平行链，它只需简单地调用一个特殊的“外向合约”就好。那个外向合约会索取任何必须的费用，然后生成一个日志打印指令，以便于通过梅克尔树和有块头哈希来证明它的存在。

在下面的两个情况中，可以非常简单地证明有效性。原则上，唯一的要求是每个 Polkadot 节点都要运行一个全同步的标准以太坊节点。然而这本身就是非常重的依赖。一个更轻量的方法是提供一个简单的证明，仅需要包含正确执行该交易所必须知晓的以太坊的那部分状态树，然后再检查日志的有效性。这种类似简单支付验证（SPV-like）的证明不需要提供大量的信息。更方便的是，验证人可能完全不需要自己运行节点，Polkadot 内的押金系统能支持第三方参与者来提交块头，因为其他第三方（也就是所说的钓鱼人）也可能提供一个他们块头是无效的证明（具体地说就是状态根和回执根是错误的），所以这些人也冒着损失他们押金的风险。

在一个类似以太坊这样的无最终确定性（non-finalising）的 PoW 网络上，不可能存在最终可证明的一致性。为了适应这个，程序需要依赖一定的块确认数量，或者直到那个依赖的交易已经在链内某一特定深度了。在以太坊上，这个深度从最脆弱的 1 个块（网络都还不完全知道）延伸至 1200 个块（从 Frontier 上线到以太可交易）。在 Homestead 的稳定版本上，大部分交易所选择了 120 个块这个数字，我们也可能会选择相近的参数。

所以我们可以想象 Polkadot 这边的以太坊接口有一些简单的功能：可以接受以太坊网络的新块头，并能验证它的 PoW，可以结合一个有足够深度的块头（还有 Polkadot 内转发的相应信息），来验证从以太坊那边的外向合约打印出来的特定日志的证明，还可以接收关于之前收到的但还没有确定的块头里包含无效的回执根的证明。

需要有一个转发激励机制，才能够真正地在 Polkadot 网络里得到以太坊块头的数据（还有任何关于有效性和一致性的 SPV 证明）。这可能设计成只是个简单的支付行为（由在以太坊那边收集的手续费资助），转给任何能够提供一个有效块头的人。为了能够应对分叉，验证人需要保留最近几千个块的信息，要么由协议原生支持，要么通过中继链上的合约。

5.5.3 Polkadot 和比特币

Polkadot 和比特币的交互是非常有挑战性的：从两边的网络角度考虑，一个所谓的“双向锚定”架构会非常有用。然而由于比特币的局限性，如何提供一种安全性的锚定个

是非常艰难的任务。可以使用类似以太坊的流程，从比特币转发一个交易到 Polkadot：由一个受 Polkadot 验证人控制的“外向地址”（break-out address）来托管转账过来的代币（和附属的数据）。可以通过结合一个确认期，来激励先知（oracles）提供 SPV 证明，先知们通过标识一个非一致性的区块，来证明一笔交易存在双花的可能。任何在外向地址里托管的比特币原则上也被相同的验证人群体控制。

问题是如何保证这些比特币，是被轮换的验证人集合所控制的。相比于以太坊那样可以根据在合约内任意组合签名规则的方法，比特币的局限性就更多了，大部分的比特币客户端只接受最多 3 方的多重签名。扩充至 36 个或者大家希望的最高至上千个的终极提议，在现有的比特币协议里还不可能实现。一个选择是修改比特币的协议来支持这个功能，然而硬分叉在比特币的世界里非常难以安排和讨论。另一个可能性是使用门限（threshold）签名的方法，用密码学的结构来构造一个被被多个私钥片段共同控制的公钥地址，要制造一个有效的签名需要这些人的大部分或全部人都参与。不幸的是，和比特币的 ECDSA 相比，门限签名计算起来非常耗资源，而且是多项式级别的复杂度（polynomial complexity）。

由于入金的安全根本性由有抵押的验证人决定，所以另一个选择是减少多重签名的私钥持有人数量至只有重度质押的验证人才能参与，这样门限签名就变得可行了（或者最糟糕的情况，也可能直接用比特币的原生多重签名）。由于要预防验证人的非法行为，这个方法会降低可托管的比特币总量。然而这是一个优雅的妥协，可以简单地设置能够安全地在两个网络里的转移的基金总额上限（验证人攻击失败可能会受到的押金损失，和攻击成功可能的会收到的比特币潜在收益对比）

因此，我们认为在现有的比特币框架下，开发出一个能够在两个网络间安全转移比特币的平行链是不现实的，尽管如此，比特币的持有者还可以在不确定的将来协调这些工作。

6 协议细节

本协议可以大致分为三个部分：共识机制、平行链接口、跨链交易路由系统。

6.1 中继链操作

中继链会类似以太坊，也是基于状态的，包含一个账户信息到状态存储的映射关系，其中信息主要包含余额和交易计数器（防止重放）。把账户系统放在这里的目标是：记录每个身份在系统里控制了多少权益。但还有一些值得注意的差异：

- 不能通过交易部署合约；这是为了让中继链尽量缺乏功能性，不支持公开部署合约。

- 没有资源计数器（gas）；因为公众能够调用的一些功能是固定的，gas 记录系统的原理就不适用了。因此在所有功能中，会使用一个更通用的手续费标准，这样就能更高效地执行那些动态代码，交易格式也会更简单。
- 会有一些包含特殊功能的默认合约，他们管理交易的自动执行和网络消息的输出。

中继链会有一个基于 EVM 的虚拟机，但为了最大程度地简化会做很多修改。它会有一些内置合约（类似于地址在 1-4 之间的那些以太坊合约）运行平台的特定功能，包括共识合约、验证人合约、平行链合约。

如果不用 EVM，很有可能会选择 Web-Assembly（Wasm）；这样的话，所有结构还是相似的，但是这些基于 Wasm 的内置合约使用的是通用功能的语言，而不再是 EVM 上面的那些带有很多限制的不成熟语言。

还可能借鉴目前以太坊上衍生出来的其他方面的概念，例如在 Serenity 版本中提出来一些改变，比如为了能在一个块里并行执行那些没有状态冲突的交易，将交易的回执格式简化等。

Polkadot 有可能会部署一个类似于 Serenity 的纯净（pure）区块链系统，它不包含链的任何基础协议。但我们觉得这会带来更多的复杂性和开发不确定性，所以不太值得在目前阶段就去实现这么一个更高效且简介的伟大协议。

为了管理共识机制，需要很多小片儿的功能：验证人集合、验证人机制、平行链等。这些都可以放在一个整体的协议中。然而为了实现模块化，我们会把这些描述成中继链的合约。这意味着他们都是中继链共识机制管理的对象（类似面向对象语言），但不一定是类似 EVM 的字节码，也不一定能通过账户系统寻址。

6.2 权益合约

这个合约管理着验证人集合：

- 哪些账户是验证人；
- 哪些在短期内可以变成验证人；
- 哪些账户为了提名验证人而质押了权益
- 每个人的属性，包括余额、可接受的押金比例、地址列表、会话（session）身份

它让账户在想成为验证人的时候可以来注册（需满足某些要求）、也可以提名某用户、在想退出验证人角色的时候还可以来退出。它自身还包含了一些用于验证和达成一致性的功能。

6.2.1 权益代币的流动性

通常我们希望能从网络中把尽可能多的权益代币都抵押进来，因为这关系到抵押权益的总市值和网络的安全性。这可以很简单地通过货币增发和收益分发来激励验证人。然而，这么做会出现一个问题：如果代币都被抵押在权益合约里，用于防止作恶，那么如何保证代币在一定程度上的基本流动性，进而支持价格发现呢？

一种方法是提供一个前向衍生合约来管理由抵押代币衍生出来的二级代币。但这在非信任的情况下很难实现。这些衍生代币无法等值交易，原因就和欧元区的不同政府发行的债券一样：抵押的代币有可能被扣减而价值降低。至于欧洲政府，他们还可能会违约。对于由验证人质押而来的代币，要考虑到验证人的恶意行为可能会遭到惩罚的情况。

基于我们的原则，我们选择了一种更简单的方案：不能把所有的代币都质押进来。这意味着一部分（可能 20%）代币会被强制保持可流通的状态。尽管从安全的角度上讲，这个方案不完美，但也没有从根本上影响网络的安全。相比于 100%的质押，也将只可能没收 80%的权益作为赔款。

我们还将会使用一个反向拍卖机制来公平地决定质押代币和流通代币的比例。有兴趣成为验证人的代币持有者可以给权益合约提交一个请求，说明他们希望支付的最小比例。每次会话（会话可能每个小时算一次）开始的时候，系统会根据每个意向验证人的押金和支出比例来填满验证人的插槽。一个可能的算法是从提交押金的验证人中，选择那些押金满足如下条件的人：押金不高于“总押金目标/插槽数量”且不低于“次低押金”的一半。如果不够填满这些插槽，那么我们会迅速降低这个“次低押金”来满足条件。

6.2.2 提名

用户可以把手中的权益代币非信任地交给一个已激活的验证人，让他们来履行验证人的职责。提名通过一个“批准-投票”系统来完成。每个准提名人可以给权益合约提交一个声明，指出他们信任的可以履行职责的一个或多个验证人的身份。

在每个会话期间，提名人的押金会散布给一个或多个代表他们的验证人。这些验证人的押金是等额分配的。提名人的押金用于验证人承担他们的责任，将能够获得利息或承受相应的扣减。

6.2.3 押金没收/烧毁

验证人的某些行为会导致惩罚性地没收他们的押金。如果押金降低到允许的最小值，会话就会提前结束，另一个会话开始。一个不完整的将导致惩罚的行为列表：

- 属于一条平行链的验证人小组，却不为该平行链的区块提供合法性验证；
- 签名了该平行链一个不合法的区块；
- 不去处理出口队列中被投票为已生效的消息；
- 不参与到共识流程中；
- 在中继链两个竞争性的分叉上同时签名。

有些行为会威胁到网络的完整性（例如签名不合法的平行链区块，或者签名多个分叉），为了驱逐这些验证人，会没收他们的押金。另外还有一些不那么严重的行为（例如不参与到共识流程中）或者那些无法清晰判别的行为（例如处于一个低效的小组），只会导致一小部分的押金被处罚。在后一种情况中，可以采用一个二级小组的搅拌功能来让恶意节点遭受到比正常节点更多的惩罚。

因为实时同步每条平行链的区块是个非常大的工作，所以在某些情况下（多叉签名和不合法签名），验证人无法很方便地检测到自身的不当行为。在这里有必要指出验证人之外的一些参与方也可以举报这些非法行为，并从中获得奖励，但他们和钓鱼人还不太一样。

因为有些情况非常严重，我们希望可以很简单地从没收的押金里支付奖金。我们通常倾向于使用烧毁代币的方法进行重分配，而不是采用批量转账的方法。烧币可以从整体上增加代币的价值，也就可以补偿整个网络而不仅是涉及到的特定几方。这主要是作为安全防范机制，只有非常恶劣的行为才会到会非常大金额的惩罚。

很重要的一点是奖金必须足够高才能让网络觉得验证工作是值得做的，当然也不能比成本高太多，否则会招致那些足够有钱的、精心策划的国际级别的犯罪黑客攻击那些不幸的验证人，迫使他们做出非法行为。

规定的奖金也不能比恶意验证人的押金高太多，否则会不正当地激励非法行为：验证人为了奖金自己举报自己。解决方法是要么直接限制成为一个验证人的最小押金量，要么间接教育提名人：如果验证人押金太少，他们可能没有足够的动机来遵守规则。

6.3 平行链的注册

这个模块用于记录系统中的每条平行链。它是个相对简单的类似数据库的结构，管理着每条链的静态信息和动态信息。

静态信息包括链的索引（一个整数）和验证协议的标识。协议标识用于区分不同的平行链，只有这样，验证人才能运行正确的验证算法 ，然后提交合法的候选块。一个最初的概念验证版本会关注于如何把一个新的验证算法放在客户端中，这样每增加一个新种类的区块链，就需要一次硬分叉。然而在保证严格和高效的情况下，还是有可能不用通过硬分叉就能让验证人知晓新验证算法。一个可能的实现方法就是用一种确定的、本地编译的、平台无关的语言来描述平行链的验证算法，例如 WebAssembly 等。为了验证这种方法的可行性，我们还要做更多的调查，毕竟如果能够避免硬分叉还是会有很大优势的。

动态信息涉及交易路由系统，比如必须对平行链的的入口队列进行全局共识（在下一节讨论）。

必须通过全民公投才能注册新的平行链。这本来可以直接内部管理，但通过一个外部的全民公投合约会更好，因为这个合约还可以用于更多其他场景的治理。关于平行链投票注册系统的具体参数（例如法定人数、多数派的比例）会用形式化证明做成一个不常更新

的“主宪法”系统，当然初始阶段也可能只是用传统的方法。具体的公式不在本文的讨论范围内，例如占 2/3 的多数派通过，并且全系统 1/3 的代币都参与了投票才算通过。

还有一些暂停和删除平行链的操作。我们希望永远不要暂停一条平行链，但这个设计是为了能应对平行链的一些紧急情况。最明显的情况是由于验证人运行了平行链的多种客户端实现，导致可能无法对某区块达成共识。我们也鼓励验证人使用多种客户端实现，以便能尽早检测到这类事情，防止押金被扣减。

因为暂停操作是个紧急措施，所以会采用验证人动态投票的方式，而不是通过全民公投。对于重启操作，可能直接通过验证人投票，也可能通过全民公投来完成。

删除操作平行链只能通过全民公投来进行，而且要提供一个宽松的平滑退出过渡期，能让它们成为一个独立的区块链或变成其他共识系统的一部分。这个期限可能是几个月，而且最好由平行链根据自身的需求来制定。

6.4 打包中继链区块

区块打包的过程本质上是共识的过程，也是把基本的数据变得有意义的过程。在一个 PoW 链里，打包有一个同义词叫挖矿。在本方案里，它涉及收集验证人对于区块有效性、可用性、一致性的签名，这些区块包括中继链区块和它所包含的全部平行链的区块。

底层的 BFT 共识算法也不是当前的工作范围。我们不描述它，而是使用一种原语描述一种由共识推动的状态机。最终我们希望能受到一些现有共识算法的启发：Tangaora（Raft 的 BFT 变体）、Tendermint 和 HoneyBadgerBFT。共识算法需要并发地对多条平行链达成共识。假设一旦共识达成，我们就可以不可辩驳地记录哪些人参与了其中。我们也可以在协议内把不正当行为的人缩小到一个小组中，里面仅包含哪些恶意参与者，这样就可以在惩罚时可以降低附带伤害。

以签名声明形式存在的这些证明、中继链的状态树根和交易树根一起存储在中继链的块头里。

对于中继链区块和平行链区块的打包过程是在同一个共识生成机制中，两类块共同组成了中继链的内容：平行链并不是由他们的小组隔离地进行“提交”之后再被收集的。这虽然导致中继链的流程更加复杂，但也让我们可以在一个阶段里就完成整个系统的共识，能够将延迟最小化，并且能支持更加复杂的数据可用性，这在路由流程中将会很有用。

可以用一个简单的表格（二维的）来建模每个参与方的共识机。每个参与方都有一系列以签名形式存在的来源于其他参与方的信息，描述着每条平行链的候选块和中继链的候选块。这些信息有两部分数据：

可用性：对于出口队列里这个块的已提交交易，验证人是否有足够的信息才能在下一个块正确地验证平行链的候选块？他们可以投 1（知道）或 0（不确定）。当他们投

了 1，他们就承诺在后续的投票中也要这么投票。后面的投票和这个不对应会导致惩罚。

有效性：平行链的区块是否有效，是否包含了引用的所有的外部数据（比如交易）？这和验证人对平行链的投票相关。他们可以投 1（有效）、-1（无效）或 0（不确定）。只要他们投了非 0，他们就承诺在后续的投票中也要这么投票。后面的投票和这个不对应会导致惩罚。

所有验证人都必须投票；在上面的规则限制下，还可以重新提交投票。共识流程可以像很多标准 BFT 共识算法那样来建模，每条平行链是并行的。除了有很小的概率把少数恶意参与者都被分配到了同一条平行链小组之外，共识算法在整体上还是能支撑网络，最坏的情况也不过只是出现一个或多条平行链因空块而死锁的情况（还有一个回合做出惩罚）

判断一个独立区块是否有效的基本规则（允许全部的验证人作为一个整体达成共识，然后这些平行链区块就成为中继链上具有一致性的数据引用）：

- 需要有至少三分之二的验证人投票“是”，并且没人投“否”。
- 需要超过三分之一的验证人对出口队列消息的可用性与否投票“是”。

对于有效性而言，如果至少有一个“是”且至少有一个“否”投票，一个特殊的条件就开启了，整个验证人就必须投票决定是否有恶意参与者，或者是否产生了意外的分叉。除了有效和无效之外，还支持投第三种票，等效于同时投了“是”和“否”，表示这个节点有相互冲突的意见。这可能是因为节点所有者运行的多种客户端实现而产生了分歧，也预示着平行链协议可能存在不清楚的地方。

当所有验证人的票都被记录过后，发现赢的那个意见少于一定数量的票（详细参数最多可能是一半，也许更少），那就可以假设平行链发生了意外的硬分叉，这条平行链的共识就会被自动暂停。否则，我们假设就是有恶意行为发生，并惩罚那些给输的那个意见投了“是”票的验证人。

结论是只有足够的签名票数才能达成一致性，然后中继链的区块就打包完成了，开始打包下一个区块。

6.5 中继链区块打包的改进

打包区块的方法确保着系统的正常运行，因为每条平行链的关键信息都要由超过三分之一的验证人来保证可用性，所以它并不能很好地伸缩。这意味着随着更多平行链的增加，每个验证人的工作也会增加。

在开放的共识网络中，如何保证数据的可用性还是个有待解决的问题，然而还是有一些方法可以缓解验证人节点的性能瓶颈。一个简单的方案是：其实验证人们只是负责验证数据的可用性，那他们就没必要自己真正地存储、通信和复制数据。第二个方案是数据隔

离，这个方案很可能和收集人如何组织数据相关，网络可以对收集人有一定的利息或收入激励，让他们保证提供给验证人的数据是可用的。

然而，这个方案也许可以带来一点伸缩性，但仍没有解决根本问题。因为添加更多平行链通常需要增加验证人，网络资源的消耗（主要是带宽）以链总数的平方的速度增长，长期来看这是不可持续的。

最终，我们可能会思考对于保证共识网络安全的根本限制，网络对带宽的需求增长速度是验证人数乘以消息总进入数。我们不能信任那些将数据分开在不同节点存储的共识网络，因为这会将数据和运算分离。

6.5.1 延迟性介绍

简化这个规则的方法是先了解即时性的概念。33%+1 的验证人最终（eventually）需要对数据的有效性进行投票，而不是立刻（immediately）投票，我们可以更好地利用数据指数级传播的特性，来帮助应对数据通信的峰值。一个合理的等式（尽管未证明）：

$$\text{延迟} = \text{验证人数} * \text{区块链数} \quad (1)$$

在目前的模型下，系统的规模只有随着链的个数而伸缩，才能保证数据的分布式运算；因为每个链至少需要一个验证人，对于可用性投票的复杂度，我们把它降到了只和验证人个数呈线性关系。现在验证人数可以和链个数同样增长了，我们终结了：

$$\text{延迟} = \text{数量}^2 \quad (2)$$

这意味着随着系统增长，网络内带宽和延迟性的增长是可知的，但达到最终确定性所需的区块数目仍然是以平方增长。这个问题将会继续困扰我们，也可能迫使我们打造一个“非平层”（non-flat）的架构，也就是会有很多按层级结构排列的 Polkadot 链，通过一个树形的结构来路由消息。

6.5.2 公众参与

微意见（micro-complaints）系统是一种可以促进公众参与的方式。可以有一些类似于钓鱼人的外部参与方来监管验证人。他们的任务是找到提供了非可用数据的验证人。他们可以给其他的验证人提交一个微意见。这个方案需要用 PoW 或押金机制来防止女巫攻击，否则它会让整个系统失效。

6.5.3 可用性保证人

最终的一个方案是从验证人里提名出第二个小组作为可用性保证人（Availability Guarantors）。他们也需要和普通验证人那样交押金，而且有可能来源于同一个组（会在一个长周期里选择他们，至少也是一个会话）。和普通验证人不同的是，他们不需要在各条平行链间切换，而只需要形成一个单一的小组，监管所有重要跨链数据的可用性。

这个方案还有个优势是能缓解验证人数和链个数之间的等式关系。链个数可以最终增长（与原始链的验证人小组一起），然而各参与方仍可以保持次线性增长或常量增长，尤其是那些参与数据可用性验证的人。

6.5.4 收集人设置

系统需要保证的一个重要方面是：合理地选择那些制造平行链区块的收集人。如果一条平行链由某个收集人控制了，那么外部数据是否可用就会变得不那么明显，这个人就可以比较简单地发动攻击。

为了尽可能地广泛分配收集人，我们可以用伪随机的方法来人工衡量平行链区块的权重。在第一个示例中，我们希望验证人倾向于选择权重更大的候选块，这是共识机制的一个重要部分。我们也必须激励验证人找到最大权重的候选块，验证人可以把他们的奖励按比例分配给这些候选块。

在共识系统里，为了确保收集人的区块被选中的机会是平等的，我们用一个连接所有收集人的随机数生成器来决定每个候选块的权重。例如用收集人的地址和一些密码学安全的伪随机数做异或（XOR）运算来决定最优的块（获胜票）。这给了每个收集人（更准确地说是每个收集人地址）随机公平地打败别人的机会。

验证人通过女巫攻击来生成一个最接近于获胜票的地址，为了阻止这种情况，我们会给收集人的地址加上一些惰性。一个很简单的方法是需要他们的地址有基本的余额，另一个更优雅的方式是综合考虑地址的余额来计算获胜的概率。这里还没有完成建模，我们很可能会让很少余额的人也可以成为收集人。

6.5.5 区块超重

如果一个验证人集合被攻击了，他们可能会生成一个虽然有效但要花费大量时间来执行的区块。这个问题来源于一些特定的难解数据题，比如大质数因式分解难题等，验证人小组可能需要非常长的时间才能解出答案，如果有人知道一些捷径，他们的候选块就有巨大的获胜优势。如果一个收集人知道那个信息，而其他人都在忙着计算老的块，那么他就有很大的优势让他的候选块获胜。我们称这种叫超重（overweight）块。

为了防止验证人提交这些大幅超出普通区块的超重块，我们需要添加一些警告：因为执行一个区块要花费的时间是相对的（根据它超重的程度），所以最终可能的投票结果会有三种：第一种是这个区块绝对没有超重，超过 2/3 的验证人声明他们可以在一定时间内算完（例如出块时间的 50%）；另一种是这个区块绝对超重了，超过 2/3 的验证人声明他们无法在限定的时间内执行完这个区块；再一种就是意见分歧基本持平，这种情况下我们会做一些惩罚。

为了保证验证人能预测他们提交的区块是否超重，他们可能需要公布自己在每个块上的执行表现。经过一段时间后，他们就可以通过和其他节点的比较来评估自己处理器的性能。

6.5.6 收集人保险

还有一个问题留给了验证人：为了检查收集人区块的有效性，他们不能像 PoW 网络那样，而是必须自己计算里面的交易。恶意收集人可以填充非法或超重的区块给验证人，通过让他们受害（浪费他们的资源）来获取大量的潜在机会成本。

为了预防这个，我们为验证人提供了一个简单的策略。第一：发给验证人的平行链候选块必须要用有钱的中继链账户签名，如果不这么做，验证人会立即丢弃这个块。第二：会用组合算法（或乘法）对这些候选块进行排序，因素包括高于一定限额的账户余额、收集人过去成功提交的区块数（除去那些有惩罚的）、和获胜票的接近程度。这里的限额应该等于提交非法块的惩罚金。

为了警示收集人不要发送非法或超重的交易给验证人，任何验证人都可以在下一个区块中打包一个交易，指出那个非法的区块，并将那个收集人部分或全部的余额都转给那个受害的验证人。这种交易的优先级高于其他交易，使得收集人不能在惩罚之前转走他的余额。惩罚金额可能是动态决定的，也很可能是验证人区块奖励的一部分。为了阻止验证人任意没收收集人的钱，收集人可以对验证人的决定进行上诉，成立一个由验证人随机组成的陪审团，并交一些押金。如果陪审团发现验证人是合理的，那这笔押金就给陪审团了。如果是不合理的，押金退回给该收集人，而验证人要受到惩罚（因为验证人是核心角色，惩罚会比较重）。

6.6 跨链交易路由

跨链交易路由是中继链和其验证人的核心功能。这里管理着主要的逻辑：一个提交的交易（简言之为“提交”）是如何从一个来源（source）平行链的出口被强制地路由到另一个目标（destination）平行链里，而且无需任何信任人。

我们很小心地选择了上面的词语；在来源平行链里，我们无需一个明确约束这个提交的交易。我们模型里的唯一约束是：平行链必须尽力按照全部的出口能力打包，这些提交就是他们区块执行的结果。

我们用一个先进先出（FIFO）的队列组织这些提交。作为路由基准（routing base）的队列个数可能在 16 个左右。这个数字代表着我们可以直接支持的平行链性能，而不用采用多相（multi-phase）路由。Polkadot 一开始会支持这种直接路由，然而我们也可能会采用一种多相路由操作（超路由 hyper-routing）作为将来系统伸缩的方式。

我们假设所有参与方都知道下两个区块 n，n+1 的验证人分组情况。概括而言，路由系统有如下阶段：

- 收集人 s：合约成员中的验证人 V[n][S]。
- 收集人 s：FOR EACH 小组 s：确保合约里有至少一个验证人 V[n][S]。
- 收集人 s：FOR EACH 小组 s：假设出口[n-1][s][S]是可用的（上个区块里所有对 S 提交的数据）
- 收集人 s：为 S 构造候选块 b：（b.header, b.ext, b.proof, b.receipt, b.egress）。
- 收集人 s：发送证明信息 proof[S] = (b.header, b.ext, b.proof, b.receipt, b.egress)。
- 收集人 s：确保外部交易数据 b.ext 已经对于其他收集人和验证人可用了。
- 收集人 s：FOR EACH 小组 s：发送出口信息 egress[n][S][s] = (b.header, b.ext, b.receipt, b.egress)给下个区块的接收方小组的验证人 V[n+1][s]。
- 验证人 v：预连接下一个区块的同一个组的成员：让 N = Chain[n+1][V]；连接所有的验证人使 Chain[n+1][v] = N。
- 验证人 v：收集这个块所有的入口数据：FOR EACH 小组 s：检索出口 egress[n-1][s][Chain[n][V]]，从其他验证人 v 获得使 Chain[n][v] = Chain[n][V]。可能是通过随机性地选择其他验证人的证明数据。
- 验证人 v：为下个块接收候选块的出口数据：FOR EACH 小组 s，接收 egress[n][s][N]。对区块出口的有效性投票；在意向验证人间重新发布使 Chain[n+1][v] = Chain[n+1][V]。
- 验证人 v：等待共识。

egress[n][from][to]代表：在区块 n 里，从来源 from 平行链到目标 to 平行链的当前出口队列信息。收集人 s 是属于平行链 S 的。验证人 V[n][s]是平行链 s 在区块 n 时的验证人小组。相反地，Chain[n][s]是验证人 v 在区块 n 所属的平行链。block.egress[to]是从平行链区块 block 发送给目标平行链 to 的出口队列。

收集人因为希望能够采纳他们出的块，所以收集（交易）手续费作为激励，并保证下一个区块的目标小组成员都能知晓当前块的出口队列。验证人的激励是达成中继链区块的共识，所以他们并不关心最终采纳哪个收集人的区块。一个验证人原则上可以勾结一个收集人，合谋减少采纳其他收集人的概率，然而因为平行链的验证人是随机分配的，所以这也很难得逞，而且还可能会遭到手续费减免，最终影响共识流程。

6.6.1 外部数据可用性

如果要在一个去中心化的系统里完成分布式的全部流程，一个长年的遗留问题是：如何确保一条平行链的外部数据都是可用的。这个问题的核心原因是：不可能生成一个关于可用性与否的非交互式证明。在一个拜占庭容错的系统内，我们需要依赖外部数据才能验证任意交易的有效性。假设我们能容忍的最多的拜占庭节点数为 n，我们一共至少需要 n+1 个节点才能证明数据的可用性。

Polkadot 是个希望可以伸缩的系统，这带来了一个问题：如果必须由一个固定比例的验证人来证明数据的有效性，并且假设他们真会存储这些数据来用于判断，那么我们如何避免随着系统的增长而带来的对带宽/存储空间等需求的增长。一个可能的答案是成立一个验证人小组（就是保证人），他们的数目随着 Polkadot 整体的增长而线性增长。这在 6.5.3 里提到了。

我们还有第二个技巧。收集人有内在的激励去确保所有数据的可用性，否则他们就不能再生产后续区块了，也就不能再获得手续费了。收集人也可以形成一个小组，成员复杂多样（因为平行链验证人成员的随机性），很难进入。允许最近的收集人（可能是最近几千个块）对某条平行链区块的外部数据发起挑战，来获取一点验证人的奖励。

验证人必须联系这些有明显进攻行为的小组，这些小组会举证、获取并返回数据给收集人，或者直接通过证明数据的非可用性来升级事态（作为原告方直接拒绝提供数据记录，不当行为的验证人会直接断开连接），并联系更多的验证人一起去测试。在后一种情况中，收集人的押金会被退回。

一旦超过法定个数的验证人都证明交易的非可用性，验证人小组就可以解散了，非法行为的收集人小组会被惩罚，区块被回退。

6.6.2 路由“提交”

每条平行链的头部都包含一个出口树根（egress-trie-root）。这个树根包含了一个路由信息的格子列表，每个格子里都有一个串行（concatenated）结构的出口提交。可以在平行链的验证人之间提供梅克尔树证明，这样就能证明某条平行链的区块对应着另一条平行链的出口队列。

在开始处理平行链区块之前，每条平行链指定区块的出口队列会被并入我们区块的入口队列。假设密码学安全的伪随机数（CSPR）能用来保证公平地对平行链区块进行配对。收集人计算新队列，并根据平行链的逻辑抽干出口队列。

入口队列的内容会被明确地写入平行链区块。这么做有两个目的：第一，平行链可以独立地进行非信任同步，而不用依赖其他链。第二，如果整个入口队列无法在一个块内处理完，那么这种方法可以简化数据逻辑；验证人和收集人可以继续处理下面的区块而不用再做数据引用了。

如果平行链的入口队列超过了区块处理的阈值，那么在中继链上就会被标记为已满，在队列清空之前不会再接收新的消息。使用梅克尔树来证明收集人在平行链区块里的操作是可信的。

6.6.3 弊端

这个架构的小瑕疵是可能发生后置炸弹攻击（post-bomb attach）。所有的平行链给另一个平行链发送最大数量的提交，这会瞬间塞满目标链的入口队列，不造成任何伤害地进行了 Dos 攻击。

正常情况下，假设有对于 N 条平行链和一系列正常同步的非恶意的收集人和验证人，那么总共需要 N x M 个验证人，每条平行链 L 个收集人，每个块可能的数据路径（data path）有：

验证人：M-1+L+L：M-1 代表平行链集合里的其他验证人，第一个 L 代表每个收集人提供了一个平行链候选块，第二个 L 代表下一个块的全部收集人需要放入出口队列的前块数据。（后一种情况可能会更糟，因为收集人之间会分享这些数据）。

收集人：M+kN：M 代表和每个平行链区块相关的验证人的连接数，kN 代表着下一个区块播种（seeding）到每个平行链验证人小组的出口队列的负载（很可能是一些很受喜爱的收集人）。

因此，每个节点数据路径的可能性随系统的复杂度的增长而线性增长。这也是合理的，当系统伸缩到上百上千个平行链的时候，通信的延迟也会变大，进而降低复杂度的增长速度。在这种情况下，会用一个多级的路由算法来减少峰值期的数据路径，但需引入缓存和交易延迟。超方路由（Hyper-cube Routing）是一种可以建立在上面描述的基础路由方法上的一种新机制。对节点来说，他们的连接数从需跟平行链和节点小组数一起增长，变成了只跟平行链个数的对数增长。这样就可能需要经过多个平行链的队列才能最终传送“提交”。

路由本身是简单和确定性的。我们从限制入口/出口队列的格子数开始；平行链的总数目是 routing-base（b），这个数字会随着平行链的改变而修正，增长为 routing-exponent（e）。在这个模型下，我们的消息总量以 $O(b^e)$增长，而数据路径保持为常量，延迟（或传递需要的块数）以 O(e)增长。

我们的路由模型是一个 e 维的超方体，每个立方体的面有 b 种可能位置。对于每个块，我们围绕一个轴来路由消息。为了保证最坏情况下的 e 个块的传递延时，我们用 round-robin fashion 来轮换每个轴。

作为平行链处理的一部分，只要给定当前的块高度（路由维度），入口队列里外部范围的消息就会立即路由给合适的出口队列的格子。这个过程需要在传送路由上发送更多数据，然而这会是个问题，也许可以通过一些替代性的数据负载发送方式解决，比如只包含一个引用，而不是在提交树（post-trie）里包含全负载。

一个拥有 4 条平行链的超方路由系统示例，b = 2、e = 2：

阶段 0，对于每个消息 M：

- sub_0：如果 $M_{dest} \in \{2, 3\}$，那么 sendTo（2），否则保留
- sub_1：如果 $M_{dest} \in \{2, 3\}$，那么 sendTo（3），否则保留

- sub_2：如果 $M_{dest} \in \{0，1\}$，那么 sendTo（0），否则保留
- sub_3：如果 $M_{dest} \in \{0，1\}$，那么 sendTo（1），否则保留

阶段 1，对于每个消息 M：

- sub_0：如果 $M_{dest} \in \{1，3\}$，那么 sendTo（1），否则保留
- sub_1：如果 $M_{dest} \in \{0，2\}$，那么 sendTo（0），否则保留
- sub_2：如果 $M_{dest} \in \{1，3\}$，那么 sendTo（3），否则保留
- sub_3：如果 $M_{dest} \in \{0，2\}$，那么 sendTo（2），否则保留

这里的两个维度很容易看做是目标索引的前两位（bits）。第二个块处理低序的位。一旦全部发生（任意顺序），提交就会被路由。最大化随机性（Serendipity）。一个对基本提议的修改是把验证人数固定为 c^2-c 个，每个小组 c-1 个验证人。摒弃原来每个区块时都在平行链间松散地分配验证人的方案，而改成对于每个平行链小组，在下一个区块时，会分配每个验证人到唯一的不同平行链小组。这导致了两个区块之间的不可变性，对于任意配对的平行链，都会有两个验证人调换他们的职责。然而我们不能用这个来确保绝对的可用性（单个验证人可能时常掉线，即使是非恶意的），但可以优化这个方案。

这个方案也会有后遗症。平行链需要重组验证人集合。进而验证人的数量会被绑定在平行链数量的平方级别，从很少开始最终快速增长，在 50 条平行链时就会变得无法承受。这些都不是什么本质问题，对于第一个问题，本来也需要频繁重组验证人集合，无论验证人集合的数量多少。当集合数很少的时候，多个验证人可能被分配到同一条平行链，那么对于全部平行链影响的因素是常量的。对于在很多平行链时的需要很多验证人的问题，可以用在 6.6.3 里讨论的这个多阶段的超方路由机制来缓解。

6.7 平行链的验证

验证人抵押了大量的保证金，他们的主要目标就是校验平行链区块是否有效，包括但不限于：状态转换、囊括外部交易、执行等待在入口队列的提交、执行出口队列的最终状态。这个过程本身是比较简单的。验证人一旦完成了前一个区块的打包，他们就可以自由地为后面的几轮共识准备平行链的候选块。

验证人一开始通过平行链收集人（下面介绍）或他的某个副验证人找到一个平行链区块。平行链候选块的数据包含区块头、前块头、外部数据输入（对于以太坊和比特币，这些数据被称为交易，然而他们也可能是任意结构、任意目的）、出口队列数据、状态转换有效性的内部证明数据（对于以太坊，这可能是用来执行每个交易的很多状态/存储树节点）。实验性的证据显示对于目前的以太坊区块，这个数据集最多有几百 K 字节（KiB）。

如果校验没有完成，验证人会尝试从前一个块的转换中获取相关信息，从前一个块的验证人开始，之后到所有签名了这个数据的验证人。

一旦一个验证人接收到了这么一个候选块，他们就在本地验证它。验证过程包含在平行链这个大类的验证人模块里，这个需要共识的软件模块必须写在所有的 Polkadot 实现里（原则上可以在多个实现里共享一个 C ABI 的库，但这会降低安全性，因为他们只是单一实现的引用）。

这个过程会提取前块头，然后用刚达成共识的中继链区块中记录的哈希值来检验。一旦父块头的有效性得到了验证，就会调用平行链类中特定的验证函数。这是个会接收很多数据项（大概就是目前给出的几种）的函数，返回值是对于区块是否有效的简单判断。

大多数这种验证函数都将首先检查头部的数据项，这些数据都可以直接从父块衍生出来（例如父块哈希、高度）。之后为了处理交易或提交，他们会尽力填充内部数据结构。对于以太坊这样的区块链，需要执行全部的交易才能往梅克尔树填充这么大量的数据。其他类型的区块链可能有其他的处理措施。

一旦完成验证，入口提交和外部交易（或代表的其他）都会根据链的规则而被固定。（一个可能的默认方式是需要所有入口提交都在服务外部交易之前处理，然而这应该由平行链的逻辑决定）。通过这个规定，一系列的出口提交都会被创建，而且确实符合收集人的候选块要求。最终会一起检查合理填充的块头和候选块头。

验证人完成了对候选块的校验后，就对块头哈希进行投票，并发送必要的验证信息给小组里的其他副验证人。

6.7.1 平行链收集人

平行链收集人不需要交押金，他们完成的是类似目前区块链网络里矿工的任务。他们属于特定的平行链。为了开展工作，他们必须要有完全同步的中继链和平行链。

完全同步的精确含义取决于平行链的种类，尽管它们都包含平行链入口队列的当前状态。在以太坊这个例子中，它还至少要有最近一些块的梅克尔树数据库，但也可能包含非常多的其他数据结构，例如证明账户存在的 Bloom 过滤器、遗传（familial）信息、日志输出、和对应高度区块的分叉回退表单。

为了保持两条区块链的同步，他们必须维护一个交易池来“钓取”（fish）交易，并接收公网上正确验证的交易。有了链和交易池，收集人就可以为每个块的被选验证人（由于同步了中继链所以知道他们身份）打包新的候选块，再附属一些必要信息（例如从节点网络来的有效性证明等），然后提交给验证人。

他们收集所有交易的手续费作为回报。这里有很多经济激励手段。在一个激烈竞争的市场中，如果收集人有富余的话，还可以跟平行链验证人分享手续费，以激励他们打包特定收集人的区块。同样地，一些收集人可能提高所需支付的手续费，使区块对于验证人更有吸引力。在这种情况下，正常的市场机制会使那些更高手续费的交易跳过队列，并能更快地打包到链里。

6.8 网络设计

以太坊和比特币等传统区块链中的网络设计需求一般比较简单。所有的交易和区块都未受引导地用 gossip 广播。同步模块中牵涉到的东西会更多一点，以太坊就可以根据不通类别做出不同的响应，但现实中这更多是节点的策略，而不是协议本身的内容。

以太坊基于 devp2p 协议改进了目前的网络协议，支持在单一节点连接中进行多个子协议的多路复用，因此同时支持多个 p2p 协议，但以太坊的协议仍然相对比较初级，而且它还没有完成例如支持 QoS 等重要功能。当初创造一个无所不在的“web3”协议的愿望基本上失败了，只剩下从以太坊众筹出来的几个项目。

Polkadot 的需求更加根本。相比于一个完整的统一网络，Polkadot 有很多种参与方，每方都有不同的需求，参与方需要有很多不同的网络信道来交换数据。从本质上讲，这意味着需要一个能支持更加层级化的网络结构的协议。另外为了促进更多新类型的区块链来扩展网络，也需要有一个新的层级结构。

对于网络协议更深层面的探讨不在本论文范围内，我们需要更多的需求分析。我们可以把网络参与者分为两类（中继链、平行链），每个都有三小类。每条平行链的参与方之间相互通信，而不和其他链通信：

- 中继链参与方
- 验证人：P，为每条平行链分割成多个子集 P[s]
- 可用性保证人：A（在基础协议里由验证人代替）
- 中继链客户端：M（每条平行链的成员）
- 平行链参与方：
- 平行链收集人：C[0]，C[1]，…
- 平行链钓鱼人：F[0]，F[1]，…
- 平行链客户端：S[0]，S[1]，…
- 平行链轻客户端：L[0]，L[1]，…

通常我们认为网络成员和他们的设置间会发生如下几种通信：

- P | A <-> P | A：为了达成共识，验证人/保证人必须连接。
- P[s] <-> C[s] | P[s]：每个作为平行链成员的验证人会和其他成员连接来发现区块并分享区块，例如收集人。
- A <-> P[s] | C | A：每个可用性保证人将需要从验证人那里收集签过名的共识相关的跨链数据；收集人可以广播给保证人来优化对他们区块的共识。一旦完成，数据会广播给其他保证人来促进共识。
- P[s] <-> A | P[s’]：平行链验证人将需要从前一个验证人或可用性保证人集合收集额外的输入数据。
- P[s] <-> A：当需举报时，钓鱼人公告给任何参与方。
- M <-> M | P | A：中继链客户端输出数据给验证人和保证人。

- S[s] <-> S[s] | P[s] | A：平行链客户端输出数据给验证人和保证人。
- L[s] <-> L[s] | S[s]：平行链轻客户端从全客户端获取数据。

如果为了保证高效的传输，那种每个节点无差异的平层网络（类似以太坊 devp2p）就不再适应了。协议里很可能扩展引入一个合理的节点选择和发现机制，还可能计划一些前瞻性的算法，保证节点的顺序在适当时候是“偶然”连接的。

各类不同参与方节点的具体策略会不一样：对于一个能伸缩的多链系统，收集人要么需要持续地重新连接被选的验证人，要么连接一个验证人小组来保证他们永不断线，即使大多数时间他们对于自己是无用的。收集人也会保持和可用性保证人集合的一个或多个稳定连接，来确保需要共识数据的快速传播。

可用性保证人将保持相互连接，还要保持与验证人（为了共识和需共识的平行链数据）、一些收集人（为了平行链数据）、一些钓鱼人和一些全节点（为了缺失的信息）的稳定连接。验证人倾向于寻找其他验证人，特别是那些在同一个小组里的，还有那些可以提供平行链区块的收集人。

钓鱼人和一般中继链或平行链客户端会倾向于和验证人或保证人保持一个连接，但和他们相似的很多节点却不这么做。平行链轻客户端除了连接其他轻客户端外，也会连接一个平行链全客户端。

6.8.1 节点轮换的问题

在基础协议的预案里，每个块的验证人小组随机变换，验证人被随机分配去验证某条平行链的交易。如何在不相关的节点间传递数据会是一个问题，这就必须依赖一个全分布式并且连接良好的节点网络，才能保证所需的跳跃距离（最坏的延迟）只按照网络规模（一个类似的 Kademlia 的协议会有帮助）的 log 级别增长，要么就必须延长区块时间，来支持必要的连接谈判，建立能够满足该节点当前通信需求的节点集合连接。

这些都不是好的方案：强迫变成更长的出块时间会让网络无法支持一些特定的程序或区块链。即使是一个完美公平的网络连接也会导致带宽浪费，因为要推送大量数据给不相关的节点，所以会影响到网络的伸缩功能。

然而这些方向都会促进问题的解决，一个可以降低延迟的优化方案是降低平行链验证人集合的易变性，在一段区块后才重新分配（比如 15 个区块，如果是 4s 的区块时间，那么只需要每分钟才重新连接），或者一段时间内只轮换一个验证人（例如如果有某条平行链分配了 15 个验证人，那么平均情况是一分钟内才全部轮换）。通过提高平行链局部的可预测性，来降低节点轮换的次数，并仍然保证连接的优势，我们就可以保证节点间连接的随机性。

6.8.2 通往高效网络协议的路径

最高效和合理的开发方向是专注于改造一个现有协议而不是自己从头开发一个。我们将会探讨的几个点对点协议包括：以太坊的 devp2p、IPFS 的 libp2p、GNU 的 GNUnet。关于这些协议的全面介绍、以及其中关于如何打造一个能支持特定结构的模块化节点网络、动态节点转换、可扩展子协议等内容，本文也不做过多介绍，但这会是实现 Polkadot 的重要一步。

7 协议的可实践性

7.1 跨链交易支付

我们去除了以太坊那样的计算资源统计的 gas 机制，这虽然带来了更多的自由和简便性，但也引出了重要的问题：没有了 gas，一条平行链如何防止其他平行链逼迫他们做运算？然而我们可以依赖“交易提交”的入口队列缓存来阻止一条链给另一条链塞满交易数据，我们还没有提供其他相同效果的防垃圾机制。

这是个更高层面的待解决问题。由于链可以在“交易提交”中附属任何数据，我们需要保证在开始之前就支付计算费用。类似于以太坊 Serenity 版本中的一个提案，我们可以想象平行链有一个“内向”（break-in）合约来给验证人提供保证，付费换取特定数量的计算资源预分配。这些资源可能用类似 gas 的机制来度量，但也可能用完全不一样的概念模型，比如主观计算时间模型或类似比特币的一般计费模型。

无论“内向”合约定义什么价值模型，我们都不能简单假设链下调用器对他们是可用的，所以这个方案的用处也不大。然而我们可以想象在来源链里的第二个 “外向”（break-out）合约。这两个合约形成一个桥梁，相互认识并且提供等值交换（相互之间的权益代币可以用来支付结算）。调用其他链意味着要用这个桥梁做代理，可以通过谈判来协商如何支付目标平行链的计算资源消耗。

7.2 添加链

添加一个平行链是相对比较便宜的操作，但并不免费。平行链越多意味着每个平行链的验证人就越少，更多的验证人又意味着每人的押金也会变少。通过钓鱼人缓解了强迫一条平行链的问题。由于共识机制本身的问题，增长中的验证人集合本质上导致了更高的延迟。将来每条平行链都可能给不幸的验证人造成很大的验证算法负担。

因此，验证人和/或其他权益持有者将会对添加一条新链来定价。这个链的市场要么是如下两种：

- 对网络没有净贡献的链（通过锁币或烧毁权益代币的方式，比如联盟链、Doge 链、特定应用链）；
- 通过提供一些别的地方没有的功能，能给网络带来更多价值的链（例如隐私性、内部伸缩性、内置服务）

我们最终会激励社区的权益持有者来添加子链——通过经济手段或根据给中继链添加功能的意愿程度。

可预期的是刚添加进来的新链会有一个短暂的移除期，这就允许新链可以先试验，无需任何妥协和长期价值风险。

8 结论

我们提出了一个异构多链协议的可能方向，它是可伸缩的且能够向后兼容目前已存在的区块链网络。在这个协议下，各参与方为了自身利益共同创造了一个完整系统，它可以用非常自由的方式来扩展，而且没有目前那些普通区块链对用户的固有成本。我们给出了这个架构的大体轮廓，包括需要的参与方角色、他们的经济激励模型和他们需要做的操作。我们已经弄清楚了一个基本的设计，并讨论了它的优势和限制；我们未来的方向就是消除这些限制，向完全可伸缩的区块链方案迈进。

8.1 遗漏的材料和开放问题

网络设计一般都是从协议实现中分离出来的。还没有完全讨论网络如何从这种实验性的条件中恢复。网络需要一个确定性的非零时间，他们从中继链分叉中恢复过来应该也不是个大问题，然而需要小心地集成入共识协议中。

本文也没有具体探讨押金的没收和对应的奖励规则。目前我们假设提供的是赢者全拿（winner-takes-all）的奖励原则：但可能给钓鱼人最好的激励。一个短周期的提交-披露流程支持很多钓鱼人来索取赏金，进而达到一个更公平的奖励分配制度，但这个举报恶意行为的流程也将会引入更大的延迟。

8.2 鸣谢

感谢那些帮助发表这篇框架性文章的校对者。特别提下 Peter Czaban、Ken Kappler、Robert Habermeier、 Vitalik Buterin、 Reto Trinkler 和 Jack Petersson。感谢那些贡献想法的人，特别提下 Marek Kotewicz 和 Aeron Buchanan。感谢其他所有提供过帮助的人。所有的错误仍都是我的。

本文其中一部分的工作，包括对共识算法的初始调查，是由英国政府的 Innovate UK 项目资助。