# Dante Network：Web3世界的“互联网协议”


## 摘要

Dante Network 是Web3多生态协作的中间件。在 Dante Network 中，我们定义并实现Web3互联互操作的协议栈，这将为Web3带来颠覆性的体验，就像“Internet协议”之于当前的互联网一样。基于 Dante Network 所实现的协议栈实例，未来，多链生态之间不仅可以实现通证的流通，还将实现信息的全面感知以及智能合约的无障碍互操作。


## 1. 概述

### 1.1 互联网协议栈（Internet protocol stack）

1969年，美国高级研究计划署(ARPA)建立了阿帕网(ARPANet)，实现了4台分别位于加利福尼亚州大学洛杉矶分校、加州大学圣巴巴拉分校、斯坦福大学、犹他大学的大型计算机之间的互联。这是互联网的前身，在随后的几十年中，互联网的协议栈和基础设施在不断地完善，比如大名鼎鼎的TCP/IP协议，就是在这个期间提出并逐步通用，现在，它已经成了互联网协议栈不可缺失的基石之一。

除了TCP/IP，还有很多基础的协议，比如ARP、DNS，以及我们都很熟悉的HTTP/HTTPS等。现代互联网，就是构建在这些协议栈，以及实际执行这些协议栈的路由交换网络之上的。这些协议栈，很少能够被用户真正直接感受到，但是，一旦离开它们，整个互联网以及构建于其上的事物都将崩塌，在线社交、网络游戏、视频网站、电商、直播，所有的一切。

## 1.2 区块链和Web3

由于去中心化、链上数据公开透明不可篡改等特性，以及开源共享、自治共建等协作方式，区块链为互联网的发展带来了新的可能性。也让关于Web3的愿景逐步从理念走向现实。当然，不可避免的，作为新事物，区块链也为Web3带来了一些困扰，在连接和通信方面。

在以太坊提出的愿景中曾经提到，想要构建一台世界计算机。如果抛开它相对低下的性能(TPS)和高昂的费用(GAS费)不说，那么，它确实在某种程度上构建了一台全新范式的计算机。分布式、去中心化，图灵完备，可以存储数据，可以部署和运行各类的DApp。即使是前面提到的性能和费用问题，也有很多开发者在尝试通过引入PoS（Proof of stake）共识机制、分片、Rollup等方式去解决。

然而，Web3的世界里并不是只有以太坊，正如世界上并不是只有一台计算机一样。Web3发展到今天，类似于以太坊这样的“世界计算机”还有不少，并且也都有相对稳定的技术架构和生态基础，比如Near、Avalanche、Flow、Filecoin、PlatON等。它们其中的一部分拥有更高的TPS和更低的GAS费，另外一部分拥有特别的能力，比如可扩展的存储能力或者隐私计算能力。总而言之，这些公链的存在让Web3的世界有了更多的可能。

正如在互联网发明之前，所有的电脑都是孤立的，或者有限范围内定制的局域网连接，现在Web3世界所面临的正是这样的情况，链与链之间并没有广泛有效的连接，相互之间的数据是隔离的。这种隔离是区块链的技术体制所天然带来的。在一个区块链网络中，每个节点只会去记录和验证自身网络中所发生的事件，这种方式保证了一个区块链网络能够在没有信任基础的情况下最终达成可靠的共识，不过与此同时，也会忽视其它发生在链外的事件。隔离由此产生。这种隔离制约了Web3多生态之间的可组合性以及互补性

，尽管目前看起来似乎没有带来太多的影响，不过我们相信，很多年以后，当我们回顾这一切的时候，就会发现，一个连接Web3世界的协议栈及其实例是多么的重要，就像互联网协议之于我们当前的世界一样。

### 1.3 Web3的中间件

我们希望将 Dante Network 构建为一个能够实现Web3多生态互联及协作的中间件。这个中间件将会由包含一系列多链互联互操作协议的协议栈，以及该协议栈的一个实例网络所组成。

对于开发者而言，这个中间件将帮助他们更方便地实现业务在多链之间的可组合性，使得他们可以将更多的注意力放到业务开发本身，而非数据通信这样基础且繁琐的问题上。对于用户而言，这个中间件将帮助他们在多链生态之间自主自由地流动，而不用过于在意他们当前的资产及身份是创建于哪一个生态的。

## 2. Dante Network

### 2.1 设计目标

当我们从互联网和通信的角度来更系统地看待跨链问题，对于中间件的要求就会更严格、更全面，而不是局限于某个特定的应用，比如资产跨链。

在 Dante Network 构建之初，我们就认识到，基础、共性、通用是这个中间件成功的几个关键特性，所以，我们拟定了 Dante Network 要达成了两个基本目标，并朝着这个方向持续前进。

- 全面互联：不光只是通证的跨链流通，还要实现多链信息的相互感知，甚至是跨链智能合约的相互调用，尽量保障多链环境和单链环境的一致。

- 灵活配置：在保障基础功能的情况下，可以对跨链操作的效率、安全、去中心化等方面的参数进行组合配置，以使其能够适应不同的应用场景需求。

## 2.2 协议栈

我们将Dante Newwork的协议栈称之为“世界树协议栈”，和互联网协议栈类似，它尝试定义一种基础、共性、通用的标准框架，遵循这个框架可以帮助web3世界各个公链生态实现互联互通。该协议栈分为三层：服务表述层、安全质量层、共识验证层。

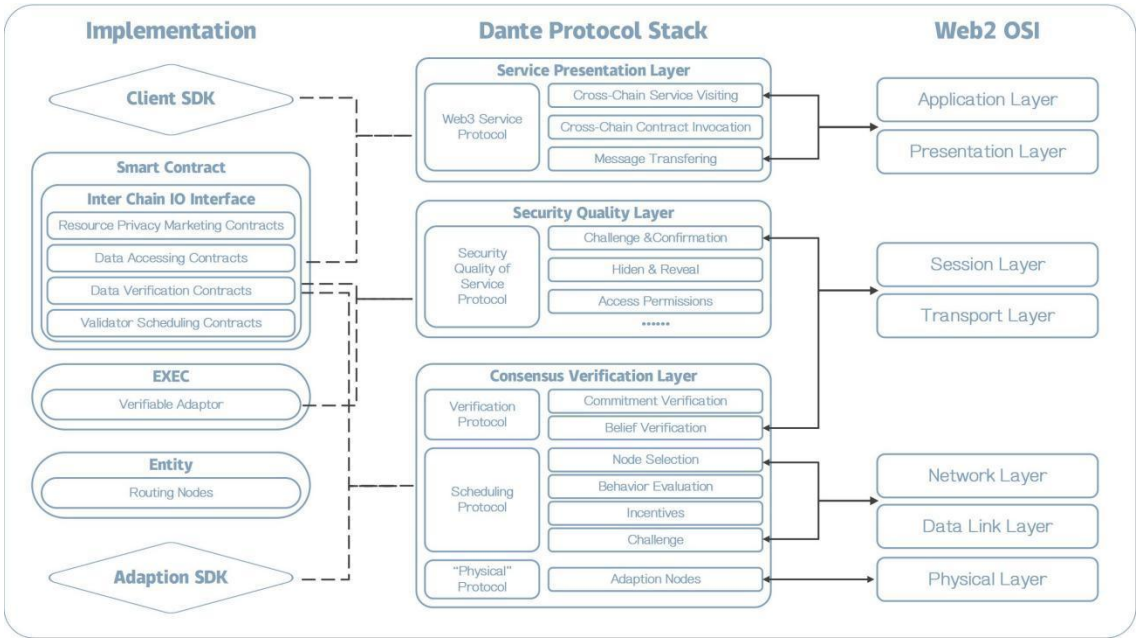


图1 Dante协议栈

### 2.2.1 服务表述层（Service presentation layer）

从信息科学的角度来看，多链之间的一切交互均可以抽象为基于数据的服务，无论是存储、计算，又或者资产跨链。我们将这些行为统一约定了一套标准服务协议，称之为Web3 service。

Web3 service具体定义了多链生态相互协作的消息表述方式（类似于互联网协议中的HTTP、FTP、SMTP等），智能合约跨生态相互访问的调用方式（类似于互联网协议中的RPC）和以及服务表述方式(类似于互联网协议中的RESTful webservice）。

服务表述层类似于互联网OSI模型中的应用层和表述层。其中包含了对数据本体、数据描述、服务执行模式、数据校验及信息验证标准、安全服务质量配置信息等一系列服务相关信息进行编解码的标准协议。

### 2.2.2 安全质量层（Security quality layer ）

跨链行为的本质其实是通信范畴的问题，任何通信协议都必须考虑服务质量（Quality of Service，QoS）的问题，这在TCP/IP等互联网协议中都有具体的体现。

但是在当前大多数跨链服务中，都很少专门去考虑这方面相关的问题。所以，在 Dante Network 中，我们正式地提出了SQoS（安全服务质量）。该协议具体定义了web3多链生态的协作模式。从功能的角度，这类似于OSI模型中的会话层，以及传输层的部分能力。在不同的SQoS配置下，用户可以根据自身业务的特点，在安全性、效率和去中心化程度之间进行灵活选择，这是应对分布式系统不可能三角的一种非常灵活的方式。

### 2.2.3 共识验证层（Consensus verification layer）

区块链本身是一个分布式的、无需信任的系统，而多链场景中，这个现象被进一步的放大，对于任何跨链解决方案，这都是需要被系统性考虑的问题。

对此，我们专门规划了共识验证层来处理这个问题，在其中，我们将构建多节点协作共识协议栈。遵循该协议栈，多个路由节点可以以“无需信任”(trustless)的模式开展工作。该协议具体定义了各类跨链协作服务的验证方法，数据传递的路由方法，以及实际执行相关工作的路由节点的接入和运行方法。

共识验证层类似于OSI模型中传输层、网络层、数据链路层（决定访问网络介质的方式）和物理层。其协议栈主要包含以下几个方面的设计。

验证协议

包括承诺验证和信念验证。验证协议定义了以何种方式对工作节点提供的服务进行合法性判断。执行服务的具体节点将按照该协议的定义提交服务结果（在承诺验证中还需要提供完成服务的证明）。在链上智能合约中，对提交的服务结果、服务证明进行验证。

a) 承诺验证:定义执行承诺验证的相关模式。承诺验证是一种确定性的验证方式，它对底层“路由”没有任何要求，能够通过密码学算法保证服务执行过程和结果的合法性。这是一种执行效率相对较高，但适应范围有限的验证方式。
b) 信念验证：定义执行信念验证的相关模式。信念验证是一种不确定性的验证方式，每个服务需要由多个节点冗余的完成，无需信任其中的任何一个节点，服务结果取决于对多个服务副本的信念聚合。这是一种执行效率相对较低，但适应范围较广的验证方式。

路由协议

定义了执行某个具体服务时，路由节点的选择机制、评估机制、激励机制、挑战机制。类似于OSI模型中的网络层、数据链路层。

a) 路由选择机制：定义了执行某个特定的服务时路由选择的方式。例如，在执行承诺验证相关的服务时，由于承诺验证能够在密码学上保证结果可信，因此对底层路由没有太多安全性的约束，此时可采用执行效率高、路由安全性要求低的策略；在执行信念验证相关服务时，由于任何一方都是“无需可信”（trustless）的，所以需要多个路由节点同时完成任务。在选择中需要有一定的随机性，以保证合谋作恶；
b) 行为评估机制：定义了对节点服务执行行为的评估方式。执行任务成功（验证通过）的节点将得到正反馈；执行任务失败（验证未通过）的节点将得到负反馈。反馈信息将表现在节点的可信度上，可信度将影响节点被选中的可能性；
c) 激励/罚没机制：节点需要通过质押来执行任务。执行任务成功的节点将获得网络激励；执行任务失败的节点将受到罚没；
d) 挑战机制：挑战机制是一种维护系统安全的保障机制，该机制定义了向网络中的异常行为发起挑战的方式。任何节点都可以向它认为异常的行为发起挑战，例如跨链信息传递有误、信息的无中生有等恶意欺骗行为。

节点适配协议

定义了 Dante Network 中路由节点执行操作和提交证明的框架与规范，节点将依此来开展工作并提交相应的工作成果。类似于OSI模型中的物理层。

## 2.3 架构


图2 Dante Network 网络架构

### 2.2.1 路由节点（Routing Node，RN）

路由节点是实际存在的节点实体，将在物理层面实际执行数据的路由工作，将数据从源链送到目标链。路由节点是开放以及去中心化的，满足一定基础条件的用户均有机会参选成为路由节点，为多链互联提供服务。关于路由节点的具体要求和规则，将会在后续进行公布。

### 2.2.2 可验证适配器（Verifiable Adaptor，VA）

可验证适配器是链下的可执行程序，实际运行于路由节点之上，其将在软件层面执行数据的路由工作，包括对来自不同链的数据进行采集、聚合、分类，将经过分类的数据转换成目标链可读的信息格式，并依照当前设定的模式送入相应的目标链。

### 2.2.3 链间IO接口（Inter chain IO Interface，ICIOI）

链间IO接口是部署在各个链上的智能合约簇，也是每条链对外的IO接口，多链间的数据将通过该接口发送及接收。

此外，该智能合约簇也是跨链任务调度的大脑，可以根据具体的跨链业务需求，对本次跨链执行逻辑进行设定和指挥，包括匹配源链和目标链、指定数据的确认模式、选择和调度路由节点、对数据做最终的确认等一些列行为。

不同链因为虚拟机架构和支持开发语言的不同，智能合约的部署形式可能存在差别，不过业务逻辑都是类似的。

### 2.2.4 开发者和SDK

为了让用户方便地使用 Dante Network 提供的服务，更好地享受到多链互操作带来的无限可能性，我们将封装并提供两类SDK，即Client SDK和Adaption SDK，分别面向Dapp开发者和节点/社区开发者。

Client SDK：为DApp提供开发支撑，通过该SDK可在DApp内直接调用 Dante Network 的多链服务，使得Dapp可以实现多链间的信息同步以及智能合约调用。

Adaption SDK：为节点提供开发支撑，对于 Dante Network 暂不支持的链，只要符合协议标准的要求，开发者可基于该SDK进行二次开发，将该链接入 Dante Network 的连接范围。

## 2.4 视角

### 2.3.1 Web3视角

去中心化和自组织是Web3的特点，但是这并不意味着隔离与孤立。如果我们曾经深刻感受过全面互联为Web2带来的难以置信的成就，也就不难想象互联互操作对Web3发展的意义。

站在Web3的视角， Dante Network 定义了属于Web3原生的“互联网协议”。我们提出这套协议的初心，是探索和实践Web3的未来。协议尝试描述和定义Web3世界互联互操作的规范，关于协议的具体实现，可以有不同的方法实例，我们基于这种规范实现了一种实例，随着Web3的发展和深化，我们也会不断地迭代协议及实例，同时，我们也希望有更多优秀的开发者来参与优化，甚至发布自己的实例方案，世界会有自己的选择，我们只需要把自己的工作做到最好。

### 2.3.2 DApp视角

Dapp开发者在选择基于那条链去开发和部署其应用时，通常会从生态支持力度、社区繁荣程度、技术栈适配情况、基础设施建设状况等方面去考虑。

然而在实际中，对于具体的某个公链生态，这几个方面通常都不是同时具备的，这意味着开发者往往需要在其中做排序和取舍，某种程度上来说，选择了某一条链，也就是放弃了更多的可能性。从云原生的范畴来说，每个

公链本质上提供的就是资源和能力，而Dapp则是通过对这些资源和能力进行整合，最终支撑自己的业务成型。在多链互连互操作成为现实之前，每个Dapp通常只能使用其当前所部署的链的资源和能力， Dante Network 的意义，正是打破这种局限。

如果我们将Web3看作一片沃土，那么Dante就是一颗世界树，各个公链的能力和资源——比如ETH执行智能合约的算力、Filecoin的存储空间、PlatON的TEE资源和隐私计算的算力等——就是土壤中的养分，而其上众多的DApp则是结出的果实。世界树的重要作用就是从土壤中吸收养分，并转化为果实生长所需要的能量。

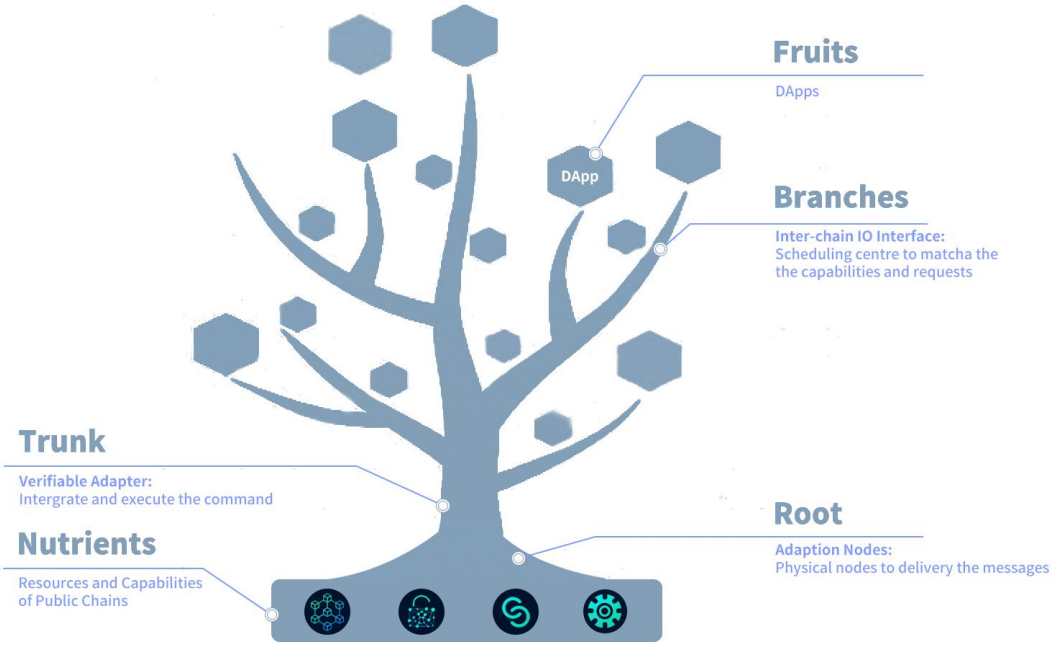


图3 Web3中的世界树

## 3. 关键实现

### 3.1 SQoS

SQoS定义了跨链操作将具备何种程度的安全服务质量。对于任何分布式系统，不可能三角都是无法逾越的问题，但是，无法逾越并不意味着无法应对。事实上，我们认为，在具体的应用中，不是所有场景的都同时对安全性、效率和去中心化存在要求。

所以，我们应该为用户提供根据其自身业务需求来进行偏好配置的能力。SQoS由一系列与安全相关的服务质量配置项组成。在一些情况下，用户可以选择非常高的安全性，但会在效率上有所牺牲，正如人们在web2中选择TCP来达到可靠传输的目的；而在另一些情况下，用户不需要非常高的安全性，他们可以选择效率偏好，正如人们在web2中选择UDP来进行“无连接”通信。

在对大量应用场景进行调研分析的前提下，我们为Dante protocol stack中有关SQoS的部分设计了以下可配置项。

a) 挑战确认：描述跨链服务是否需要确认等待，窗口大小为等待的时间（区块数量）。在确认窗口中，挑战节点可以发起挑战；

b) 隐藏揭示：描述该跨链服务是否在隐藏揭示模式下执行。在该模式下，相关跨链服务在实际执行以前，不对其明文内容进行揭示。这是为了避免跨链操作后执行的节点，“抄袭”已经执行完成节点的“答案”；

c) 验证阈值：描述执行信念验证时，信息内容的可信度阈值。如果验证执行后，最值得“信赖”（可信度最高）的信息内容，其可信度阈值超过了该阈值，则接受该消息。如果该阈值设置为100%，则必须所有信息副本内容一致，才算验证通过；

d) 优先级：一些因素将影响跨链服务的优先级，例如按支付的跨链订单费设置消息优先级；

e) 异常回滚：多链互操作将导致服务调用原子性的破坏，当发生错误时，错误信息无法直接返回到初始调用方，异常回滚是用于应对这种情况的；

f) 匿名交易：类似TLS/SSL中的加密传输，web3中的行为都是通过交易发生的，匿名交易将提供能够在一定程度上隐藏交易信息的能力；

g) 身份溯源：多链互操作将导致服务调用原子性的破坏，目标链无法直接获知源联调用者的身份信息，身份溯源将提供这一能力，类似TLS/SSL中的身份校验；

h) 收发隔离：服务及回执是否通过同一批路由节点来执行，收发隔离将增加路由节点合谋作恶的难度;

i) 交叉验证：这是在SQoS中专门设计的一种安全等级非常高的选项。虽然Dante protocol stack专门设计并实现了较为严谨的验证方式，但我们认为web3的世界是一个共建共享的世界，因此我们设计了可以利用其他类似的多链基础设施（如Axelar, LayZero, ChainLink, etc.），针对数据真实性，在消息层面进行交叉验证的方法。这种兼容并包的特性，也是Dante protocol stack可以被成为“多链共建协议栈”的原因。交叉验证在安全性上能够提供高于所有现有单一验证手段的能力，能够在对安全等级要求非常高的场景中（例如大金额的多链swap、DeFi、Token转账），有效避免在某一种验证手段遭到攻击而变得不可用，或者被恶意劫持时， Dante Network 依然是安全有效的。当然，没有免费的午餐，这是以牺牲效率以及支付更高的服务费来获得的。

## 3.2 数据路由

路由的本质是将信息从源地址搬运到目的地址，Dante Network 通过网络中的路由节点来完成这项任务。

由于Web3去中心化、无需许可、无需信任等特点，通常情况下，网络对可能参与路由节点都是开放的、弱约束的，所以，相较于互联网的信息传递，多链间的信息路由会有更多的设计机制，比如冗余，或者基于密码学的假设前提，来保证信息传递的可达性和真实性。

整个路由策略包含了对路由节点的选择、信息的校验以及对路由结果的反馈等一系列动作。

在 Dante Network 中，我们将引入节点信念评估模型、启发式随机选择策略，并结合质押、激励、罚没、挑战等机制来实现该路由策略。

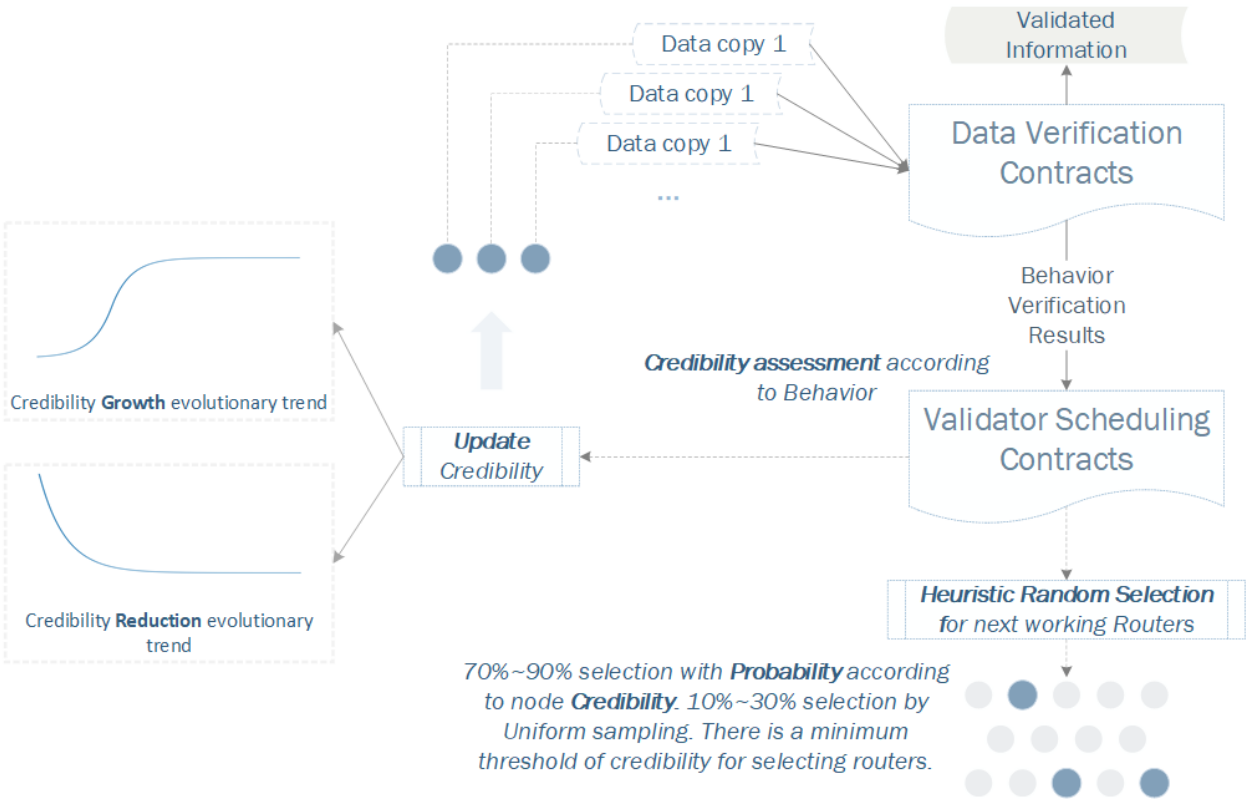


图4 信念验证

### 3.2.1 节点信念评估模型

节点信念评估模型是对路由节点行为是否合法的一个概率上的量化指标，表现为节点的可信度数值。适配节点在执行消息路由时，其效果会在验证环节得到反馈，该反馈将表现为其可信度的变化。设各个节点的可信度为$\{c_0, c_1, c_2, \ldots, c_m\}$，根据链上计算的约束，可信度可转换成相应的整数来执行，正如上文信念验证计算中消息验证所采用的方式，假设此处我们将可信度定义为[0, 10000]的数值（实际对应[0, 100%]）。

我们定义：(Let)

$$
\begin{aligned}
&min = 0 \\
&max = 10000 \\
&middle = min + (max - min)/2 \\
&range = max - min \\
&step_{success} = about \quad 100 \\
&step_{doEvil} = about \quad 200 \\
&step_{exception} = about \quad 100
\end{aligned}
$$

上述分别表示（从上到下）可信度的取值范围的下限、上限、中间值、跨度，合理行为的步进量、作恶行为的步进量、异常行为的步进量。

合理行为的步进量、作恶行为的步进量、异常行为的步进量反应了节点发生相应行为时对其可信度的影响，将在经过充分测试后，在主网上线时具体指定；

这些参数间将满足以下约束条件：

$$
\begin{aligned}
&step_* < range \\
&step_{doEvil} > step_{success} \\
&step_{doEvil} > step_{exception}
\end{aligned}
;
$$

新加入节点的可信度的值会设置在[3500， 5500]左右，具体值将在经过充分测试后，在主网上线时指定。

合理行为

当节点所“搬运”的消息内容（哈希值）与验证得出的消息内容一致时，系统将认为节点行为合理，节点将得到正反馈，其可信度将得到提升。

可信度提升的数学演进原理如下：

$$
c_i^{t+1} \leftarrow f_{success}(c_i^t)
$$

$$
in \quad which \quad f_{success} \quad satisfy \quad c_i^{t+1} > c_i^t
$$

$f_{success}$ 的具体范式将考虑链上计算执行的特点来具体指定，使得该演进趋势 $J_{growth}(c_i^0, f_{success}, t)$ 具备以下特征：

$$\frac{\partial J_{growth}}{\partial t} > 0$$

$$\begin{matrix} \frac{\partial^2 J_{growth}}{\partial t^2} > 0 & when & c_i^t < middle \\ \frac{\partial^2 J_{growth}}{\partial t^2} < 0 & when & c_i^t > middle \end{matrix} \quad , \quad c_i^t = f_{success}^{(t)}(c_i^0)$$

其中 $f_{success}^{(t)}(c_i^0) = f\left(f_{success}^{(t-1)}(c_i^0)\right)$ ， $f_{success}^{(0)}(c_i^0) = c_i^0$ ；

这将使得对于可信度很低的节点，它们可信度的提升速度刚开始会很慢，但会增长的越来越快。可信度接近中间值时增长速度将会最快。新加入节点的初始值接近中间值，这是一种对新节点的鼓励措施；可信度超过中间值后，增长速度将逐渐变慢；不断增长的可信度将最终无限趋近于上限。

合理行为可信度持续增长的演进趋势 $J_{growth}(c_i^0, f_{success}, t)$ 的仿真结果如下图（其中纵坐标为可信度，横坐标为t，代表演进节拍）：

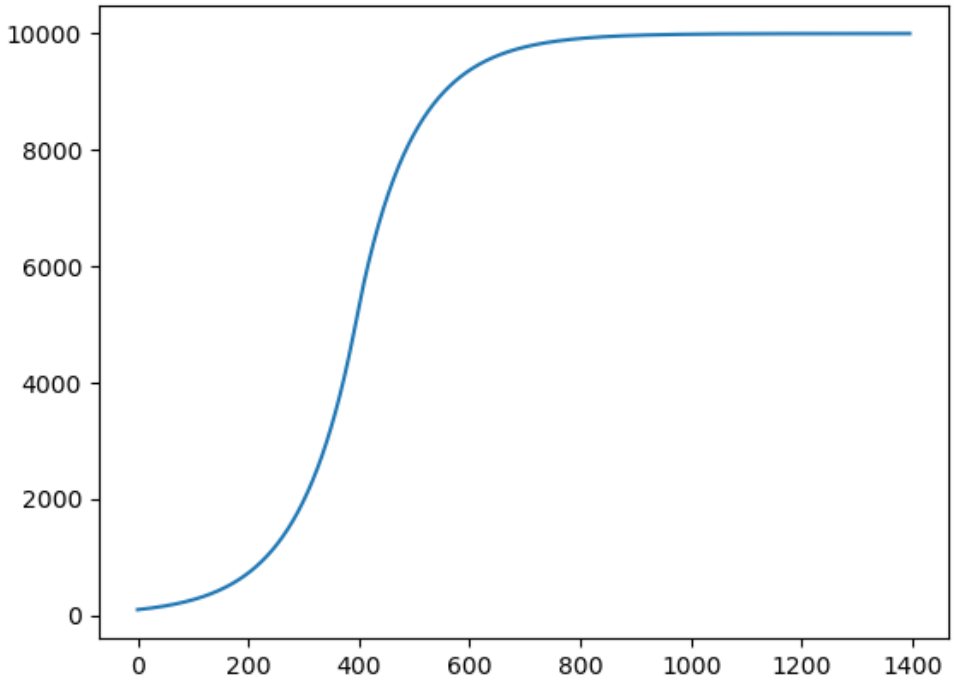


图5 信用增长趋势

这样的设计目的是希望给与新加入节点能够快速提升可信度的机会，而对于因为作恶而导致可信度很低的节点，需要付出更多的努力才能将可信度提升到正常水平，间接提高作恶的代价；

作恶行为

当节点“搬运”的消息内容（哈希值）与验证得出的消息内容不一致时，系统将认为节点行为作恶，节点将的到负反馈，其可信度将降低。

作恶行为可信度降低的数学演进原理如下：

$$c_i^{t+1} \leftarrow f_{doEvil}(c_i^t)$$

$$in \quad which \quad f_{doEvil} \quad satisfy \quad c_i^{t+1} < c_i^t$$

$f_{doEvil}$ 的具体范式将考虑链上计算执行的特点来具体指定，使得该演进趋势 $J_{reduction}(c_i^0, f_{doEvil}, t)$ 具备以下特征：

$$\frac{\partial J_{reduction}}{\partial t} < 0$$

$$\frac{\partial^2 J_{reduction}}{\partial t^2} > 0$$

不难看出，可信度越高，一旦作恶，则受到的惩罚将更加严重，这是因为在选择路由节点时，可信度越高，则被选中的概率越大。并且作恶行为可信度下降的速度，大于成功行为可信度上升的速度。

作恶行为可信度持续降低的演进趋势 $J_{reduction}(c_i^0, f_{doEvil}, t)$ 的仿真结果如下图。

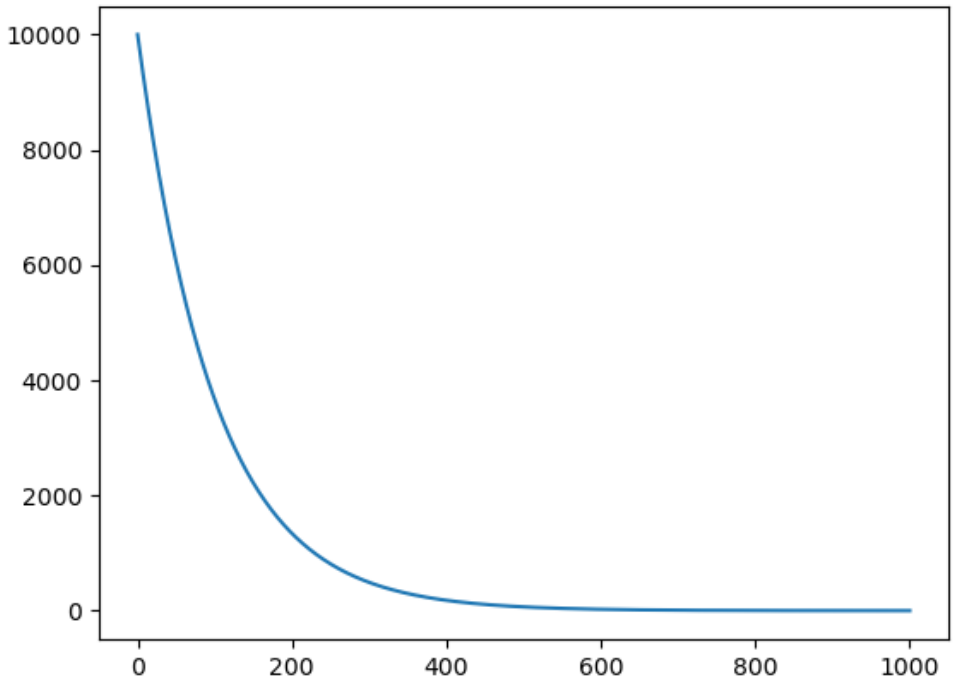


图6 信用降低趋势

这样的设计目的是提高节点作恶的代价，使可信度高的节点作恶在经济上是更加不划算的。一次作恶，不仅会损失质押金，同时还需要多次成功的合理操作，才能弥补其在可信度上的损失。

异常行为

当一次消息验证，无法得出一致消息时，系统判定该消息行为异常，本条消息的所有相关节点将得到负反馈，其可信度都会降低。降低的值受到其所属分组权重$q_i$的影响。

异常行为可信度降低的数学演进原理如下：

$$c_i^{t+1} \leftarrow f_{exception}(c_i^t, q_i)$$

$$in \quad which \quad f_{exception} \quad satisfy \quad c_i^{t+1} < c_i^t$$

$f_{exception}$的具体范式将考虑链上计算执行的特点来具体指定，使得该演进趋势$J_{exception}(c_i^0, f_{exception}, t, q_i)$具备以下特征：

$$\frac{\partial J_{exception}}{\partial t} < 0$$
$$\frac{\partial J_{exception}}{\partial q_i} < 0$$

$$\frac{\partial^2 J_{exception}}{\partial t^2} > 0$$

异常行为导致的可信度降低，其幅度将小于同等情况下的作恶行为，将取决于此次消息验证时该节点所传递消息所属的分组权重，分组权重越高，代表该节点执行的消息与更多、或者可信度更高的节点所执行的消息内容更加一致，那么该节点受到的处罚将越轻，可信度的降低值越小。

异常行为可信度持续降低的演进趋势 $J_{exception}(c_i^0, f_{exception}, t, q_i)$ 与作恶行为可信度降低的演进趋势 $J_{reduction}$ 的仿真结果相似，只是幅度更平缓。

这样设计的目的是对异常行为发生后相对“合法”的节点提供了一定保护措施。

### 3.2.2 启发式随机选择

由于节点的加入是完全开放式的，如果任由节点完全自由的执行操作，则可能会导致有的任务由于参与的节点太多而效率低下；有的任务由于参与的节点太少而达不到验证的要求。本项目采用了一种启发式的随机选择策略，来选择每个任务的执行节点。该策略是在链上智能合约中执行的，这样就可以利用公链所提供的共识机制，来保证选择策略执行的公平性。

每次选取服务节点的份额将分为两部分，可信度选择、随机选择。设选取服务节点的总数量为 $N_s$，可信度选择的份额 $N_c$ 占比系统上限为 $S^+$，下限为 $S^-$，则每次可信度选择的份额占比取值空间为 $[S^-, S^+]$。同时，设置每次节点可信度的最低阈值为 $P_c^{min}$，低于该阈值的节点将不再被选择。将可信度大于 $P_c^{trustworthy}$ 的节点称为值得信任的节点。选择算法流程如下：

在可信度选择中，节点被选中作为服务节点的概率受到其可信度的影响，设大于可信度最低阈值所有节点可信度为 $\{c_0, c_1, c_2, \ldots, c_n\}$，节点i可信度映射的选择概率为：

$$p_i = \frac{c_i}{\sum_{j=0}^{n} c_j}$$

所有值得信任的节点的总占比为：

$$P_{all}^{trustworthy} = \sum_{j=0}^{k} p_j \quad , where \quad p_j \geq P_c^{trustworthy}$$

则此时可信度选择份额为：

$$N_c = N_s \times max\{min\{P_{all}^{trustworthy}, S^+\}, S^-\}$$

此时随机选择份额为：

$$N_r = N_s - N_c$$

从所有可信度大于等于 $P_c^{min}$ 的节点中，按概率$\{p_0, p_1, p_2, \ldots, p_n\}$采样选取 $N_c$ 个节点作为服务节点；

再从剩下的可信度大于等于 $P_c^{min}$ 的节点中，按均匀采样选取$N_r$个节点作为服务节点。

### 3.2.3 激励/罚没

执行消息路由的节点需要质押一定的保证金，如果其执行结果被最终验证通过并采纳，则能够获得相应的激励，如果未通过验证，会按情形对质押的保证金进行罚没。

具体的激励罚没参数将在专门的文档中进行阐述。

### 3.2.4 挑战

对于SQoS中设定的安全性较高的操作（例如资产跨链），在跨链信息验证发生异常时，将设置任务确认窗口，在该时间窗口内，挑战者可以对异常行为发起挑战。此时 Dante Network 会随机选择多个节点对挑战进行确认(例如在跨链转账的场景中，查询源链上是否真实存在这笔交易），除非网络中所有节点都是恶意节点，否则是可以判断出被挑战对象的真伪的。如果结果是真，则挑战失败，罚没挑战者；如果结果是假，挑战成功，罚没之前执行任务

的相关路由节点；如果结果不一致，则撤销这条消息。这是为了避免一种极端情况，即网络中恶意节点占比过多，这种情况下，宁愿跨链失效，也不能让攻击者得逞。

挑战者发起挑战同样需质押一定数量的Token。此时合约将重新选择多个个节点来执行挑战确认验证，执行该验证并成功的节点会得到激励。

这种挑战机制实际是一种二级验证机制，所选择的二级验证节点也是采用启发式随机选择来完成的（但在参数的设置上上会更倾向于选择可信度更高的节点），除非网络中所有节点都是恶意节点，否则两次随机选择的适配节点，都恰好全部是恶意节点的概率是很小的。

我们考虑一种极端情况，即网络中所有的恶意节点，都是事先串通好的。此时设网络中适配节点恶意节点的占比是 $P_{evil}$，网络中适配节点的总量为 $N_a$，每次选择的执行任务的节点数量为 $N_{s-1}$。对于重要程度很高的场景，我们设置所有消息副本必须完全一致，才能通过验证。每次选择的执行二级确认验证的节点数量为 $N_{s-2}$，二级确认验证本身需要所有确认结果一致才能确认挑战结果。此时：任务验证所选择的所有节点都是恶意节点的概率 $P_{1-allEvil}$ 为

$$P_{1-allEvil} = \frac{C_{[N_a \times P_{evil}]}^{N_{s-1}}}{C_{N_a}^{N_{s-1}}}$$

二级确认验证选择的所有节点都是恶意节点的概率为 $P_{2-allEvil}$ 为：

$$P_{2-allEvil} = \frac{C_{[N_a \times P_{evil}]}^{N_{s-2}}}{C_{N_a}^{N_{s-2}}}$$

本次信念验证被恶意节点成功攻击导致验证错误的概率为 $P_{failed}$：

$$P_{failed} = P_{1-allEvil} \times P_{2-allEvil}$$

我们假设极端情况下，每次选择执行任务的节点数量$N_{s-1}$为21个，二级确认验证选择的节点数量$N_{s-2}$为31个，在不同的节点总量$N_n$下，恶意节点分别占比75%、80%、85%、90%、95%(区块链的大部分共识机制实际上在恶意节点超出51%时已经无法工作），恶意节点成功攻击的概率仿真结果分别如下（横坐标为$N_n$的取值，纵坐标为攻击成功的概率）。

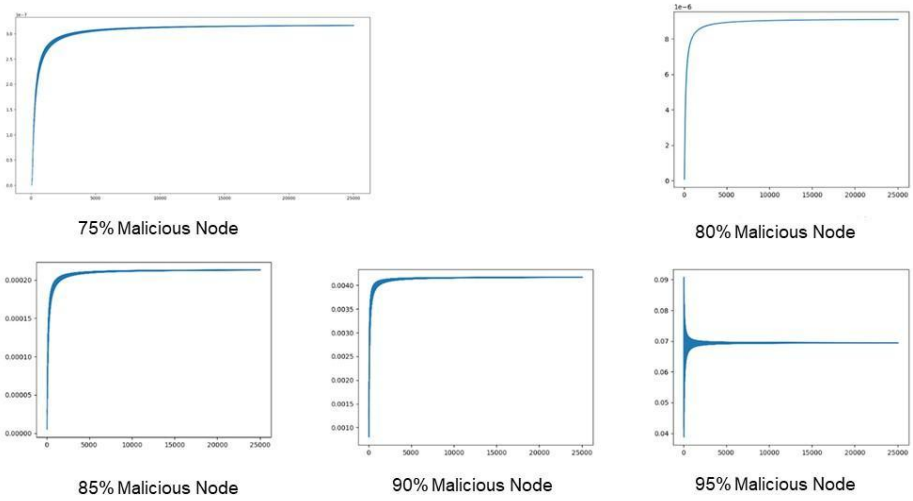


图7 作恶节点占比不同的系统表现趋势

可以看出，恶意节点占比80%以下时，攻击成功的概率几乎可以忽略不计；占比85%时依然只有最高0.02%（万分之二）；占比90%时为最高0.4%（千分之四)；占比95%时将不可忽略，最高可达9%；当然，我们是在非常极端的假设下进行的仿真，也就是假设所有恶意节点都事先串通好了，但这种情况在实际情况中真正发生的工程代价是非常大的（需要为协同合谋付出非常大的工程代价)。实际上，一个去中心化的网络，如果存在51%以上的恶意节点，那么实际上该网络中的所有参与者（包括恶意节点）都不太可能在经济上获利了；

另外，即使在 Dante Network 初期，假设我们设置最少需要21个节点才会主网上线，此时设置$N_{s-1}$初始为3个，逐渐递增，$N_{s-2}$与$N_{s-1}$相同，这是一种非常宽松的设置，当存在60%的恶意节点时（这已经远大于常规区块链共识能够容忍的恶意节点占比），作恶者攻击成功的概率仿真结果如下（横坐标为$N_{s-1}$和$N_{s-2}$的数量）；

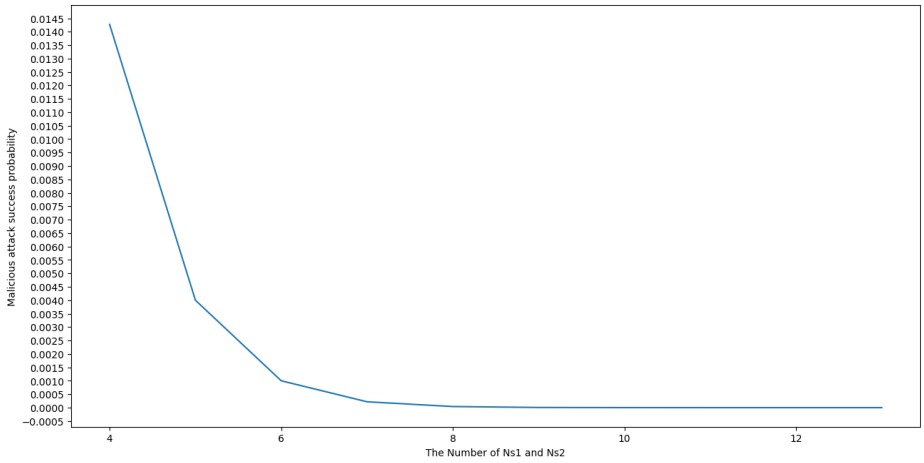


图8 节点数量对于攻击概率的影响趋势

可以看到，攻击成功的概率，在 Dante Network 初期（仅21个节点），即使采用完全开放式的节点加入模式，我们在非常宽松的安全设置之下（此时的$N_n$为固定21个，且$N_{s-2}$与$N_{s-1}$相同），只要选择的初次执行节点以及二级确认验证节点数量超过8个，即使存在60%的恶意节点，它们合谋攻击成功的概率不到0.02%。

我们对初期网络（适配节点总量为21个）中作恶节点占比70%、80%以及90%也进行了仿真。在恶意节点占比70%的情况下，只要执行节点和二级确认验证节点数量不小于8个，作恶成功的概率不超过0.1%（千分之一）；在占比80%的情况下，执行节点和二级确认验证节点数量为8个时，作恶成功概率为1.5%，10个节点时为0.4%（千分之四）；在占比90%占比的恶意节点，此时需要

超过16个执行节点和二级确认验证节点，才能将攻击成功的概率变得很小，约为0.3%。

### 3.3 信息验证

在互联网中，对数据的可信性验证并不是必须的，一般是应用层根据自身的业务特点来决定是否对进行验证工作。通常的场景中，一般只会对数据包进行校验，以保证数据传输的完整性，在某些场景中，甚至对数据的完整性都不会进行判断，比如UDP。

然而由于区块链网络自身以及相互之间都是无信任的状态，所以数据的验证在跨链的场景中，是一个必须的考虑的点。

从抽象的角度来讲，我们认为可以根据数据的驱动模态，将验证方式分为两种不同的模式。

一种是承诺验证，这种模式下数据请求方会预先提交数据需求，该数据需求包含一个具体的数据承诺，数据响应者在交付数据时，需要同时提交数据证明。该数据证明可以根据数据承诺在密码学上进行验证，若验证为真，则可以认为响应者提供的数据在密码学上是值得信任的。在本项目中承诺验证支持多样化的表现形，如数据本身对应的哈希值，块存储数据的默克尔根，VDE(可验证延迟编码）处理，同态隐藏形式下的承诺，可信执行环境中的隐藏承诺等。同时，在本项目中，针对这些承诺提交证明进行验证时，将考虑生成ZK Proof来缩短证明长度。承诺验证的实现有一定的技术难度，但是其执行过程只需要一个节点就能独立完成，因此执行效率相对较高。

另一种是信念验证，这种模式下要么不存在直接的数据请求方，要么数据请求方无法预先给出密码学上的数据承诺。我们在传递普通消息，或者一些复杂的rollup消息（例如执行某些复杂的链下计算的结果，将复杂计算算术

化并生成ZK Proof，其执行过程的工程代价可能非常大）时，将会遇到这种情况。由于单个节点所执行的链下过程是不可信的，因此在这种模式下我们需要每次同时选择多个节点来共同完成操作。本项目采用了基于概率信念的模式，即每条消息的验证，都是对所有相关执行节点所提供副本的内容进行对比、聚合后，所得出的结果。信念验证必须考虑恶意节点的存在，此时将得到多个内容不一致的结果，以及各个结果相关的可信度，可信度代表了我们对于结果的信念分布。

### 3.3.1 承诺验证

这是一种在密码学上可验证真伪的证明方式。每个证明都是针对某个特定承诺的证明，即对证明的验证，是计算该证明是否满足某个预先提交的承诺条件。这种证明方式更加高效，可以基于ZK Proof，以及可信计算环境等硬件设施来实现。

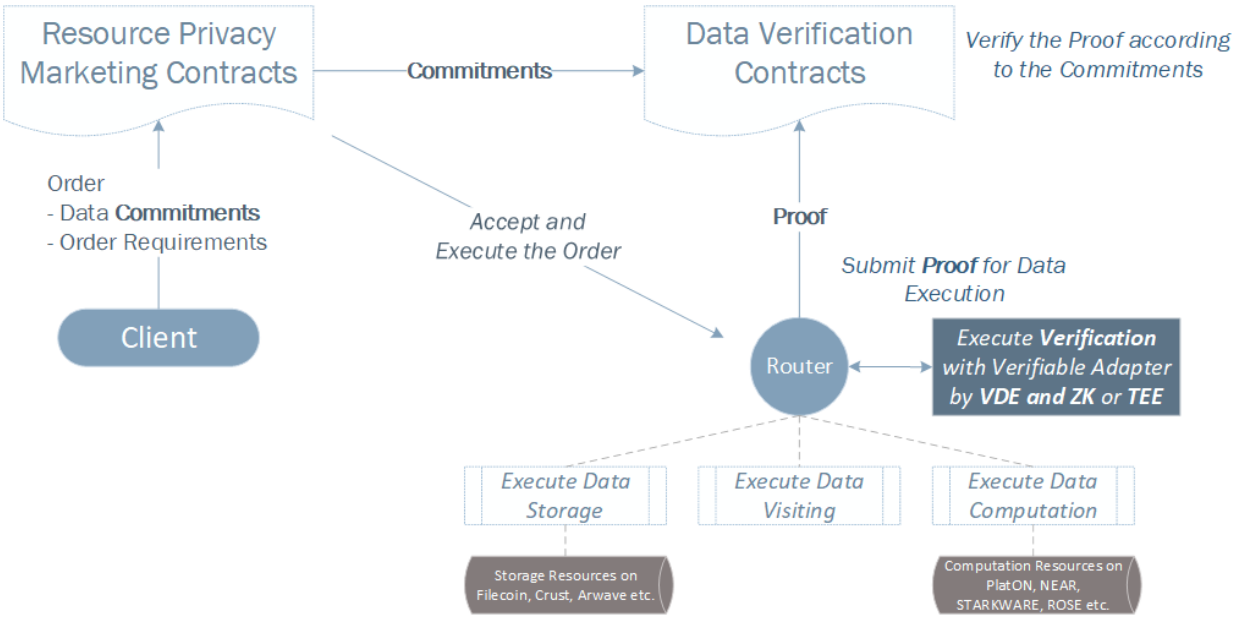


图9 承诺验证

承诺验证的原理如下：

请求者提出数据请求，该请求可以是数据存储的请求、数据访问的请求、数据计算的请求，并根据数据的具体情况，采用承诺函数$f_{commitment}(x_{data})$创建数据承诺并提交

$$D_{commitment} = f_{commitment}(x_{data})$$

其中$x_{data}$为生成承诺的相关输入，例如当承诺为默克尔根时，$x_{data}$为数据块向量；

$f_{commitment}(x_{data})$具有以下特征：

➔ 隐藏性：$f_{commitment}(x_{data})$的反函数不可得，即已知$D_{commitment}$，无法计算出$x_{data}$，例如哈希函数；

➔ 绑定性：$D_{commitment}$可唯一的代表$x_{data}$，即不同的$x_{data}$将得出不同的$D_{commitment}$。

响应者执行相关数据操作后，提交结果以及相关的$Proof_x$，通过$f_{verification}(D_{commitment}, Proof_x)$进行验证：

$$\begin{cases} f_{verification}(D_{commitment}, Proof_x) = True & \text{if} \quad passed \\ f_{verification}(D_{commitment}, Proof_x) = False & \text{if} \quad failed \end{cases}$$

### 3.3.2 信念验证

信念验证是一种基于冗余的验证方式。简单的讲，接收者对信息的判断，取决于它基于所接收到多个信息副本而计算出来的信念值。每个证明是针对信息本身的证明，每条信息的验证需要接收多个节点的信息副本。信息接收者无法信任每一个单个信息副本，需要对多个来源的信息副本内容进行聚合判断才能完成验证。信息副本间的冲突可能导致接收者无法得到确定的信息。

在信念验证中，每一条上链的消息都会通过多个节点进行操作，每个节点将搬运的消息副本递送至目标链上，根据每个消息副本的内容hash值，以

及搬运每个消息副本所对应的节点的信念度，在ICIOI中中对该条消息进行验证。

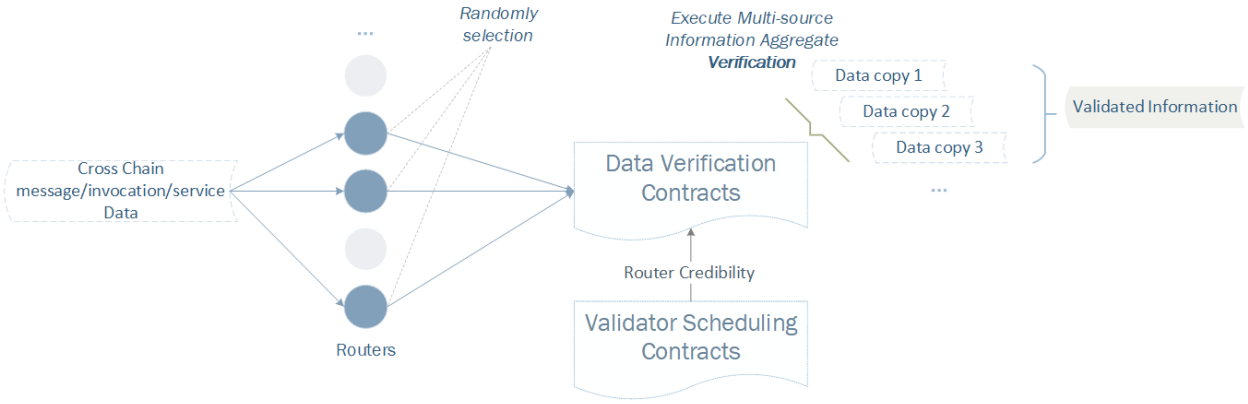


图10 信念验证

承诺验证的原理如下：

消息副本聚合，根据hash值对消息副本进行聚合，将hash值相同的消息副本放置在一起作为一个分组；

如果存在多个分组，则按每组消息副本的数量由多到少对分组进行编号，起始编号为0；

消息副本的搬运者为路由节点，每个路由节点拥有一个标识其可信度的数值，该数值的取值范围为[0, 100%]，在链上合约中处理时将转换为合约所接受的整数值，假设此处我们将数值转换为[0, 10000]

设一共有K个分组，每个分组包含$k_i$条消息副本，$i \in [0, K)$，分组i中的消息副本分别来自节点$\{n_{i_0}, n_{i_1}, n_{i_2}, \ldots, n_{i_{k_i-1}}\}$，每个节点的可信度分别为$\{c_{i_0}, c_{i_1}, c_{i_2}, \ldots, c_{i_{k_i-1}}\}$，对分组i中的消息可信度度量值$v_i$进行计算：

$$v_i = \sum_{j=0}^{k_i-1} c_j$$

计算所有分组的消息可信度度量值总和V为：

$$V = \sum_{i=0}^{K-1} v_i$$

分组i的消息可信度权重 $q_i$ 为：

$$q_i = \frac{10000 \times v_i}{V} = \frac{10000 \times v_i}{\sum_{j=0}^{K-1} v_j} = \frac{10000 \times v_i}{\sum_{j=0}^{K-1} \sum_{l=0}^{k_i-1} c_l}$$

设当前设置的消息验证权重阈值为T，如果存在消息可信度权重 $q \geq T$，其中可信度权重最高的分组作为当前消息的采纳分组，该分组对应的消息内容即为所采纳的对象；如果所有分组消息可信度都小于消息验证权重阈值T，则无法得出一致性结果，当前消息验证无法通过，此时将影响这条消息相关的所有适配节点的可信度，具体方式在“数据路由”一章中有详细说明；与验证采纳消息内容相同的消息副本所对应的适配节点，其可信度将得到提升；同时，与验证采纳消息内容不一致的消息副本所对应的其他适配节点，其可信度将会降低。具体方式在“数据路由”一章中有详细说明。

# 4. 应用场景

## 4.1 DeFi信用借贷

信用贷是一种在传统金融领域常见的方式。借贷者只需要向出借机构出示他合格的信用记录，而不需要进行额外的抵押，就可以借出符合其信用额度的贷款。这种信用记录，来自于一个人的身份信息、行为记录、以及自有资产情况。

在现实生活中，这样的信息往往来自于不同的地方，你的身份信息可能来自政府或者某大型机构的授权，你的行为记录可能来自于你信用卡的征信报告，而你的资产信息可能来自于你的房屋产权证，又或者你的股票持仓情况。你不需要把这些东西都交出去，但是你需要证明你拥有这一切。比如你的房产，否则那就是抵押贷款了，你只是想告诉大家，你是个守信用的人，你也具备还款的能力。即使是在现实世界中，向不同的机构来获取这些授权和证明也不是一件轻松的事情，但是不管怎么样，它至少可以做到。

很遗憾的是，在加密的世界中，由于区块链与区块链之间的天然隔离性，以及去中心化的特性，这一切反而变得更加的困难。我们知道，在DeFi领域，借贷是一项重要的应用，但是目前较为成熟的借贷方式都是抵押贷款，需要借贷用户进行超额抵押以借出一部分流动性资金。抵押贷款当然是一种很好的方式，能够在一定程度上解决资金流动性的问题。但是，信用贷才是实现普惠金融的更重要的方式。我们在现实生活中已经毋庸置疑的感受到了这一切。所以，我们觉得世界树可以帮助DeFi达成这一切。

接下来我们以一个案例来说明。如图所示：

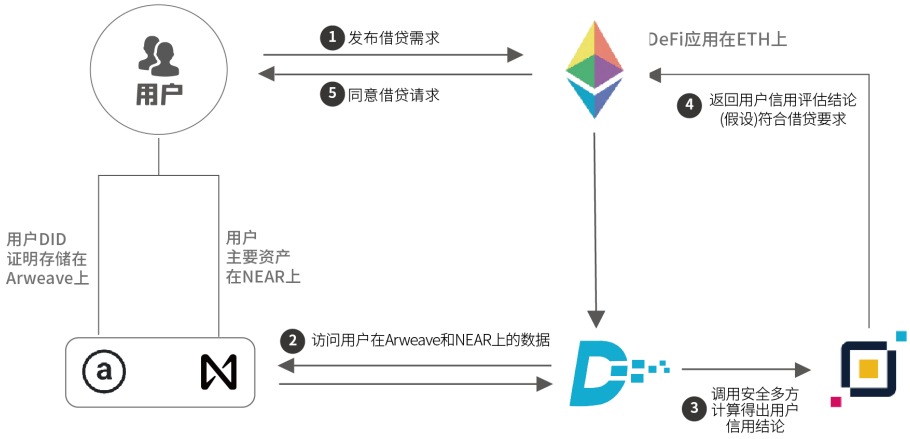


图11 信用贷Dante应用场景

一个部署在以太坊上的DeFi应用，它的主要功能就是审核用户的信息，生成一个信用评估，并对应相应的贷款额度。为了更充分地评估用户的信用，并控制贷款的风险，它需要用户至少向其提供两个有效的信息，一个是来自DID应用的身份信息，一个是资产信息。

我们假设该身份信息是永久存储在Arweave网络上，而用户的主要资产是放在Near或者Avlanche等其它公链上的。Dante可以帮助验证并将这些信息提交到以太坊网络。但是，我们认为这样可能会存在一些隐患，因为当前的DID通常都会将链上的身份信息和现实中的身份信息进行绑定，以保证身份的唯一性。用户未发生违约事件的情况下，贸然将用户的信息以及其关联的资产信息公示到链上，并不是一件友好的行为。

考虑到DeFi应用其实真正需要的并不是用户的完整信息，它需要的只是关于这些信息得出的信用评估结论，因此，我们可以借用隐私计算来完成这个过程，而避免直接暴露用户隐私。

一个值得欣慰的现象是，已经有很多公链在尝试解决隐私计算方面的问题，比如Oasis，或者PlatON，我们不需要再去重复造轮子。

Dante Network 可以帮助将这些信息导入这些隐私计算公链，调用他们的安全多方计算能力完成信用评估算法的执行，最终得出信用评估结论。

而最终DeFi应用可以根据这个信息向用户发放或者拒绝发放贷款。

### 4.2 去中心化加密

在实际场景中，数据所有者拥有数据，但是不一定具备执行加密的能力。很多终端设备，比如大量的手机、物联网传感器等，往往是数据持续产生的源头，但是考虑到它们的轻量型，指望它们完成抗攻击的复杂加密，或者同态加密等，是一件难以完成的任务。因此可以通过安全多方计算等方式，以去中心化的方式，在数据不暴露的情况下，在有执行加密计算能力的节点完成对数据的加密。

有的公链天然具备这样安全多方计算的资源和能力，其上应用可以很好地调用这种能力，用户也能够很便利地享受到其带来的好处。但是，并不是公链通用的，事实上，由于专业方向的限制，我们现在所熟知的大部分公链都不具备这样的资源和能力。在一个封闭的世界里，这是一种可以接受的常态，不过，在更加开放的Web3世界里，我们认为需要打破这一切。

Dante Network 可以帮助将这些资源和能力导入更多的Web3公链，使得用户或者DApp可以在任意地方发起关于隐私计算的请求，并得到响应。

如图所示，这是一项关于原始数据去中心化加密之后进行存储的案例。

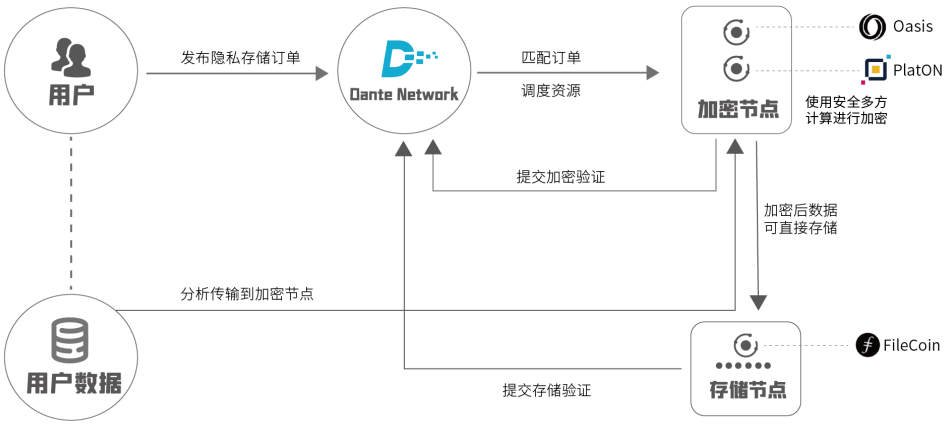


图12 去中心化加密Dante应用场景

在整个过程中，将完成用户原始数据、隐私计算公链，如Oasis或者PlatON，以及存储公链，如Filecoin的交互，最终使得原始数据隐私不暴露的情况下得到了妥善的存储。

## 4.3 NFT公开展示

事实上，相比前两项而言，这个应该场景并不是什么现象级的存在。但是它又的确来自于现实需求，并且已经处于实践中。

考虑到NFT在当下的普及程度，，当前发布一套NFT相关的链上工作已经很成熟了，同样的，NFT的公开展示并不是什么太困难的事情，比如著名的以太坊上的NFT平台OpenSea就可以很好地胜任这个事情。得益于以太坊的知名度和生态的成熟度，相关类似的配套设施非常普及，以至于我们常常认为这是理所当然的。多时候，我们会忽略了，很多新兴公链在生态早期基础设施是非常匮乏的，类似于OpenSea这样的NFT平台只是其中一项而已。当有NFT项目的创作者在某个新兴公链发布了一套NFT，如果该公链相关的NFT

展示方面的基础设施不够完善，同时缺乏一定用户基础NFT展示平台，那实际上这套NFT的真正价值将很难展现出来。

并不是所有的用户都有能力去查阅智能合约的源代码来找到这个NFT到底是什么的。如果这个NFT是艺术作品，那么，用户希望可以直接开始欣赏这幅画作，而不是告诉他们，你去看代码的链接。当然，NFT的发行方可以选择去其他NFT基础设施相对成熟的公链发布，但如果该NFT与这条新兴公链有着非常密切的关系(例如记载了该公链发展的重要里程碑，或者是凝聚生态内的原生价值的创造），那在该新兴公链发布就成了最合适的选择。

其具体使用流程如下:

- 用户在新兴公链（源链）上部署特定的NFT合约，例如记录项目发展历程中重要里程碑的数字收藏品；
- 用户在ETH上部署相关的标准化NFT映射合约，所有在源链上的相关操作，都通过 Dante Network 跨链合约调用来完成同步；
- ETH上部署的映射合约，通过 Dante Network 跨链合约调用中的权限管理机制，进行操作权限验证。
- ETH上部署的标准化NFT合约，在OpenSea中可以通过搜索直接展示出来。

NFT的展示只是其中一个很小的案例，从更广泛的角度来说，我们希望Dante能够为现在和未来更多的新兴公链提供类似的便利，进而帮助它们度过生态基础设施尚不完善的初创期。