面向以太坊的活跃网络拓扑感知与分析研究

白销东, 刘代东, 魏松杰

南京理工大学 计算机科学与工程学院 南京 中国 210094

摘要 针对以太坊网络拓扑隐藏的特征以及当前拓扑感知方法精度较差的问题, 提出了一种基于通信协议和未来事务特征的以太坊活跃网络拓扑感知方法。首先, 通过改进的以太坊节点发现协议构建虚拟地址并主动对目标节点路由表进行映射, 感知目标节点的潜在活跃连接节点集合。其次, 向以太坊网络中部署感知节点, 通过收集目标节点转发的消息本体与消息哈希的数量, 并根据两者之间的比例关系, 对目标节点的活跃连接数量进行推测。最后, 根据以太坊内存池中未来事务的隔离特征, 本文提出了一种基于隔离验证的节点间活跃连接验证的改进方法。通过面向以太坊测试网络展开广泛的实验, 本文所提出的方法能够在理想的时间内感知以太坊的区块链覆盖层网络拓扑快照, 相较于现有方法所感知的网络拓扑具有更高的精确性。根据实验数据表明, 以太坊网络中活跃节点不足整个网络的 2%, 而且活跃节点出现聚集和重叠现象。通过对以太坊网络拓扑进行建模, 分析可得以太坊网络具有较小的平均最短路径, 信息传播在网络中表现出小世界效应, 并且节点的度值遵循幂律分布, 符合以太坊分布式网络的设计初衷。

关键词 以太坊网络; 拓扑感知; 网络分析; 点对点网络

中图法分类号 TP311 DOI 号 10.19363/J.cnki.cn10-1380/tn.2024.07.06

Research on Active Network Topology Perception and Analysis for Ethereum

BAI Xiaodong, LIU Daidong, WEI Songjie

School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China

Abstract This paper proposes a topology perception method for the Ethereum active network that is based on the Ethereum communication protocol and future transaction characteristics. The purpose of this method is to uncover the hidden features of the Ethereum network topology and improve the accuracy of the methods that are currently used to perceive the topology of networks. Firstly, the set of potentially active connected nodes of the target node are determined by creating virtual addresses and actively mapping them to the target node's routing table using an enhanced Ethereum node discovery protocol. Secondly, deploy sensing nodes to the Ethereum network, gather the number of message ontology and message hashes delivered by the target node, and speculate on the number of active connections of the target node based on their proportionate relationship. Finally, based on the isolation features of future transactions in the Ethereum memory pool, this article presents an enhanced mechanism for Isolated Witness-based active connection verification across nodes. Extensive experiments on the Ethereum test network have demonstrated that the approach presented in this research can perceive the network topology snapshot of Ethereum's blockchain overlay layer in an optimal amount of time and with greater accuracy than existing methods. precision. According to experimental data, less than 2% of the nodes in the Ethereum network are active, and the active nodes appear to cluster and overlap. Through modeling of Ethereum network topology, the analysis demonstrates that the Ethereum network has a short average shortest path, that information transmission in the network exhibits a small-world effect, and that the degree value of nodes follows a power law distribution, which is consistent with Ethereum's original design intent.

Key words ethereum network; topology perception; network analysis; peer-to-peer networ

1 引言

在区块链的概念被首次提出之后, 基于区块链技术的加密货币应用得到了迅速的发展, 同时引发了大量关于其可靠性和安全性的研究工作[1]。作为典型代表的比特币网络已经受到了彻底的安全审查, 而被视为第二代区块链代表的以太坊则采用了完全不同于比特币的结构化点对点(Peer-to-Peer, P2P)网

通讯作者: 魏松杰, 博士, 副教授, Email: swei@njust.edu.cn。

本课题得到江苏省研究生科研与实践创新项目(No. SJCX21_0115)资助。

收稿日期: 2022-10-24; 修改日期: 2023-02-20; 定稿日期: 2024-03-15

络协议。作为对等节点通信的基础, 以太坊的网络覆盖层(Overlay Network)在区块链参与者的通信中扮演着关键的角色。理解以太坊网络拓扑结构是研究以太坊系统安全性和可用性的关键。

与比特币的非结构化对等网络有所不同, 以太坊的平台覆盖层是通过遵循 Kademlia 的对等发现协议和会话建立协议形成一个结构化的分布式哈希表(Distributed Hash Table, DHT)网络[2]。从以太坊 P2P 网络的多层视图来看, 网络中的每个节点维护着两层对等连接, 其中以太坊的区块链覆盖层则是特定于应用程序的网络结构, 节点维护一个活跃的邻居列表, 其功能主要包括对等节点选择和区块传播等方面, 在实现目标性能和正确运行发挥着重要作用。而在平台覆盖层上, 节点将所有对等邻居节点以非活跃连接节点的形式存储在分布式哈希表中, 而分布式哈希表中的活跃节点在未来有机会被提升到活跃邻居列表。在区块链覆盖层中, 每个节点自身维护一个活跃邻居列表, 该节点的消息传播依赖于活跃邻居列表中存储的节点[3]。因此, 维护区块链网络中信息流正常传播的是区块链覆盖层的活跃连接, 而非平台覆盖层的非活跃连接。

目前国内外已经涌现出许多针对以太坊网络层的研究, 其中大部分研究是通过以太坊的节点发现协议感知节点路由表中的非活跃连接, 并从平台覆盖层的角度对以太坊网络进行分析[4-6]。在真实环境中, 区块链覆盖层的活跃连接承载着网络中信息传播的确切流量, 维护着以太坊的正常运行, 所暴露出来的网络信息更多。由于以太坊的开发者采用拓扑混淆的策略使得网络拓扑结构隐藏在每一个节点中, 这为以太坊的网络拓扑感知带来了一定的困难, 而且以太坊的对等节点查找过程中的随机性与匿名性并没有改善以太坊的网络连接, 而是掩盖了拓扑的不平等。因此, 本文旨在感知以太坊的区块链覆盖层, 并对节点的活跃邻居列表进行推测。通过构建以太坊网络的活跃网络拓扑不仅能够更好的理解区块链的去中心化网络结构, 避免节点的集中化趋势, 而且能够更好的分析单个节点对网络的贡献, 促进事务和区块的传播, 从而降低双花攻击和自私挖掘发生的概率。由此可见, 以太坊的区块链覆盖层的活跃连接对整个以太坊的性能和可靠性具有十分重要的影响, 而且活跃拓扑结构的研究对优化网络管理具有重要意义。

本文的主要工作是针对现有以太坊的网络感知方法所暴露出的局限性, 实现了一种高效的以太坊网络拓扑感知方法, 并通过事务特征实现节点间活跃连接的推测, 最终对以太坊的活跃网络拓扑特征进行分析。本文的主要贡献如下。

(1)提出了一种面向以太坊的区块链网络覆盖层的活跃网络拓扑感知方法。本方法利用以太坊的协议特征和未来事务的处理策略, 实现了以太坊网络拓扑结构的感知和节点的活跃邻居列表的度量。

(2)通过实验测试和度量, 发现并验证典型以太坊网络的覆盖层网络结构的异质性, 节点之间的连接状况分布不均匀, 节点的度值遵循幂律分布, 且信息的传播表现出小世界效应。

2 背景与相关工作

2.1 以太坊网络

本小节对当前以太坊的网络协议规范进行简要介绍。以太坊采用结构化的组网方式, 通过重新设计节点发现、数据格式化等功能, 实现对传输内容的序列化和加密处理。以太坊的通信机制则是由 DevP2P 协议所实现, 该协议是构建以太坊点对点网络的协议集合, 旨在发现网络中的其他参与者, 并且保证参与者之间的可靠通信。如图 1 所示, 以太坊的网络采用三层协议, 以下对各层协议进行分析。

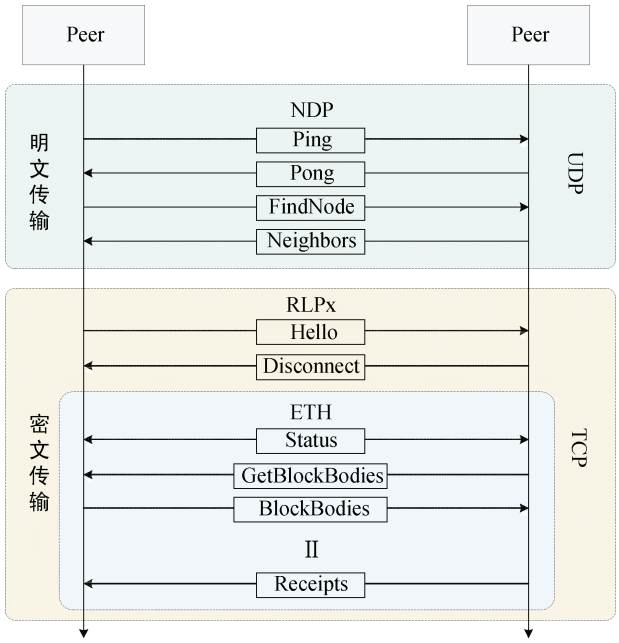


图 1 以太坊网络协议栈

Figure 1 Ethereum network protocol stack

2.1.1 节点发现协议

节点发现是一种用于在对等网络中寻找其他参与者的协议, 是 P2P 网络通信的前提, 不同于比特币, 以太坊中的节点发现协议受到 Kademlia DHT 的启发。Kademlia DHT 是一种分布式哈希表, 主要用于

高效低存储和检索内容。但是, 与大多数分布式哈希表不同的是, 以太坊的节点发现协议并没有以键值对的形式存储数据, 而是仅以值的形式存储和中继节点记录。因此, 以太坊的节点发现协议是基于 Kademlia DHT 改进实现的分布式哈希表, 唯一的目的是发现对等节点并组织节点的分布式索引。

以太坊中的每个节点都拥有经由椭圆曲线加密生成的 512 位公钥作为其全局标识符。在节点发现协议中节点之间的距离是基于全局标识符的逻辑距离, 该距离值可以由两个节点全局标识符的Keccak256哈希的异或值计算所得。假设两个节点的全局标识符分别为 $node_1$ 和 $node_2$, 则其节点之间的距离为:

$$dis(node_1, node_2)=\log_2(keccak256(node_1)XOR\ keccak256(node_2))$$

在节点发现协议中, 每个节点会保留自身附近的邻居节点信息, 并依赖逻辑距离实现节点的查找过程。在逻辑距离的定义中, 使用异或值表示节点间距离的原因是当节点计算到其自身的距离时异或值为 0, 且逻辑距离运算遵循对称性和三角不等式。

在节点发现协议 v4 版本中, 节点发现协议使用 UDP 数据包进行传输。首先, 节点通过 Ping 和 Pong 消息验证邻居节点是否可达。然后, 通过 FindNode 和 Neighbor 消息实现节点发现的请求和回复。以太坊的相邻节点存储机制也是节点发现协议的一部分, 在 Kademlia DHT 存储协议中, 每个节点维护一个路由表, 路由表中根据节点间逻辑距离划分为多个子列表, 每个后续子列表包含了比前一个子列表相距更远的节点。

2.1.2 RLPx 协议

为了增强以太坊节点通信的可靠性与匿名性, 以太坊设计了 RLPx 协议对通信过程进行加密。RLPx 本身没有特殊含义, 仅是跟随以太坊中内容的序列化方法 RLP 进行命名。在以太坊的可靠通信中, RLPx 协议主要负责节点间可靠会话的构建。

如图 1 所示, 当以太坊节点经过初始握手之后, 连接双方必须发送 Hello 和 Disconnect 消息构建可靠会话。其中, Hello 消息是会话双方发送的第一个数据包, 由双方各发送一次, 当接收到 Hello 消息时, 表明当前会话是有效的。Disconnect 消息则是用于通知对等端进行断开连接操作的信息。

2.1.3 ETH 协议

以太坊的 ETH 协议是建立在 RLPx 协议之上, 是定义了传输内容具体格式的协议。为了实现顶层应用协议, 以太坊节点在完成对等节点的连接建立、握手通信的基础上, 需要与对等节点进行以太坊的网络状态信息交互, 从而实现整个区块链数据的同步与确认, 并为后续的交易信息和区块信息的传播建立基础。

2.2 网络拓扑感知的意义

作为对等节点通信的基础, 区块链的网络层因其在相关参与者之间的通信中扮演的关键角色而受到广泛的关注。Decker 等[4]研究了网络拓扑结构对信息传播效率的直接影响, 其中节点自身所拥有的连接数量是造成传播延迟的主要因素之一。除此之外, 非结构化 P2P 网络在拓扑结构上存在着拓扑失配的问题[8], 即逻辑链路与物理链路之间的不匹配, 从而导致了低下的数据传输效率。由此可见, 网络拓扑的结构特征直接影响了系统的安全性和传输效率。

对于具有匿名性的区块链网络而言, 开发者通过含蓄地隐藏网络拓扑结构的方式来实现维护区块链安全的目的[9]。具体来说, 区块链网络中的节点是公开的, 任何一个活动节点都可以获得网络中的其他节点信息, 但是节点之间的连接是隐藏的, 因为普遍认为隐藏节点之间的关系有助于保护区块链免受反匿名化和一些低级攻击手段的干扰。尽管隐藏区块链网络拓扑提高了攻击的难度, 但是通过侧信道攻击等技术来推断网络连接可以跨越这个限制。因此, 在针对网络级别的攻击下, 拓扑隐藏并不是一个有效的防御策略。

以太坊的对等节点发现规则定义了网络的拓扑结构。现有研究表明, 比特币网络结构并非是由随机选择节点所构成的随机图, 取而代之的是, 比特币网络拓扑被采矿机构严重重塑, 这表明现实世界中的比特币网络拓扑并非是预期的网络结构, 存在着较为明显的社区现象[10], 社区与社区之间表现着较低的聚集性。因此, 无法否认的是区块链对等节点查找过程中的匿名性和随机性并没有改善区块链网络的拓扑结构, 而是掩盖了拓扑的不平衡性。同样, 拓扑信息的缺失阻碍了对网络结构的分析, 而拓扑分析不仅可以改善网络的去中心化结构, 避免节点的集中化趋势, 而且能够促进事务和区块的传播, 进而降低节点进行双花攻击和自私挖掘的能力。由此可见, 区块链网络结构的分析对于提高区块链网络的效率和安全性至关重要。

2.3 相关工作

区块链作为一个开源的区块链平台, 允许任何个体随时加入或离开网络, 所以区块链网络的拓扑结构并非是一成不变的, 也给网络拓扑感知带来了挑战。在针对区块链网络中节点的研究中, Delgado-Segura 等[11]提出了一种被动的探测方法, 通过启动

一个超级节点对网络中所有可访问的节点建立连接并收集交换的信息, 以此来揭示了以太坊的网络规模、节点地理分布等特征。Essaid 等[12]基于上述研究, 通过启动超级节点并被动收集事务流量, 对比特币网络的对等节点的行为特征进行了描述。但是通过启动超级节点进行网络探测的方式, 网络开销较大, 执行过程具有随机性, 探索周期长。

以太坊在其对等节点发现协议中定义了 FindNode 方法, 该方法可以向邻居节点请求靠近目标节点的路由表信息。Kim 等[13]通过向以太坊中的所有节点发送 FindNode 消息, 并根据返回的数据直接推测以太坊的网络拓扑结构。但是, 该方法所探测的拓扑结构具有局限性, 无法区分节点之间的连接是否是活跃的。Gencer 等[14]对比特币和以太坊的网络拓扑的去中心化程度进行了度量。区块链网络的活跃连接是节点与节点之间的隐藏信息, 无法通过直接测量的方式获取, 只能通过合理的方式进行推测。Miller 等[15]以比特币为目标, 并通过比特币中 ADDR 消息的过期时间戳来推断节点间的活动连接。但是, 该方法随着比特币版本升级已经失效。尽管上述方法在结构上和模式上进行了各种优化, 但是仍旧存在这就测量精度较低和测量周期较长的局限性。

3 网络感知方案设计

本文的感知方案流程如图 2 所示, 该方案主要包括如下三个过程: 节点路由表感知、节点活跃连接数量推测和活跃连接验证。节点路由表感知过程为感知节点向目标节点发起请求, 获取到目标节点的路由表信息的阶段。节点活跃连接数量推测过程为感知节点通过接收目标节点所转发的消息数量, 并根据消息本体及其消息哈希的比例对节点的活跃连接数量进行推测的阶段。活跃连接验证过程为根据推测所得活跃连接数量, 验证路由表中的对等节点是否为当前的活跃连接。通过上述感知过程即可得到节点在以太坊中的活跃连接, 如下对感知过程的各个步骤进行分步详细描述。

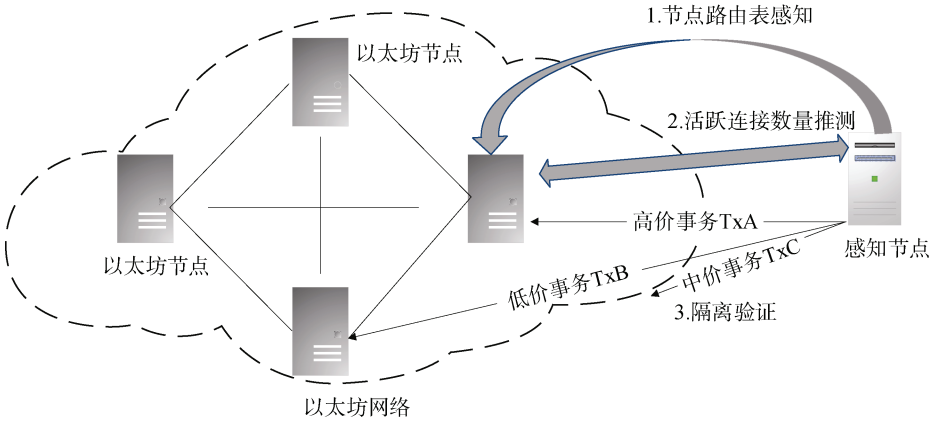


图 2 网络感知流程示意图

Figure 2 Network perception procedure diagram

3.1 节点路由表感知

不同于比特币、门罗币的对等发现协议, 以太坊采用 Kademlia DHT 协议来发现网络中的对等节点并维护对等节点信息。因此, 度量以太坊中节点的对等节点数量及其分布并非易事, 一般来说, 节点会随着时间的推移而离开或加入网络, 但是离开网络的节点并不会直接从其他节点的路由表中删除。所以, 节点的活跃对等节点连接数量要远小于以太坊节点路由表中存储的对等节点数量。但是, 无法否认的一个事实是节点的活跃连接节点必然来自于其路由表中存储的节点, 因此, 节点路由表中存储的对等节点被称为潜在活跃连接节点。

以太坊的节点路由表是不能够直接进行节点存储, 邻居节点被存放在以太坊节点路由表的 bucket 中, 当新的节点需要加入到 bucket 时, 通过计算新节点和目标节点的异或距离实现 bucket 的选择。与 Kademlia 协议不同的是, 在以太坊的具体实现过程中, 每个节点所保留的 bucket 数量是有限的。自 Geth1.8.1 之后, 每个客户端的 bucket 数量被限制为 17 个, 且每个 bucket 最多存储 16 条对等节点。与 Kademlia 协议相似的是, 以太坊的对等节点存储机制仍旧以前缀匹配的二叉树作为存储结构, 并为逻辑距离为 *dis* 的节点保留一个 bucket。由于采用对数的逻辑距离度量方式, 节点的每个 bucket 所能代表的节点 ID 范围也有所不同, 其中 bucket 的编号越大, 则代表的节点地址范围越广[16]。在具体的客户端实现过程中, 节点逻辑距离 $dis<240$ 的节点存储在 bucket[0]中, 逻辑距离 $240 \leqslant dis<256$ 的节点存储在 bucket[dis–239],

而且对数的逻辑距离度量方式使得落入bucket中的节点呈现倾斜分布, 落入特定 bucket 中的概率随着相关距离呈现指数级衰减, 因此大多数编号较低的 bucket 中所存储的对等节点数量十分少。

假设感知节点 $node_1$ 为了请求目标节点 $node_2$ 的路由表中编号为 $max(0.16–i)$的 bucket, 根据节点发现协议, 则 FindNode 信息需要生成一个特定目标值 *target* 实现指定 bucket 的映射, 其形式如下:

$$dis(target, node_2)–239=max(0, 16–i)$$

其中, 如果感知节点想要请求 bucket[0]中的数据, 仅需要将 *target* 值设置为请求节点的ID即可, 这是由以太坊节点发现协议所定义的。

根据逻辑距离计算公式可知, 在进行目标值 *target* 计算时需要使用 keccak256 哈希算法对节点全局标识符进行处理, 而考虑到 keccak256 算法具有不可逆性, 无法直接对目标值 *target* 逆推计算, 为了实现 *target* 的去随机化, 只能够通过生成随机值, 并验证其结果是否满足条件。由于根据以太坊的对等节点存储机制, 每个 bucket 多表示的节点 ID 范围呈现指数变化, 因此在随机映射生成指定目标值时, 不同编号的 bucket 的映射概率有所不同, 其概率分布如下所示:

$$p_i=\begin{cases}2^{-16}, & i=16\\ 2^{-(i+1)}, & 0\leqslant i<16\end{cases}$$

根据上述的映射机制, 假设以太坊的一个节点试图将整个网络的 N 个节点插入到其路由表中, 则其路由表中所容纳的节点总数 n 的期望值如下:

$$E(n)=\sum_{i=0}^{16}\min\left(16,\ N\times p_i\right)$$

根据 ethernodes.org 于 2022 年 5 月份统计的以太坊节点总数量 N 在 6500 个左右, 通过代入上式可以得到 n=152, 而一个节点实际可以存储的最大对等节点数量为 272 个。因此, 从理论上来看, 通过迭代感知以太坊的节点路由表的方式, 能够获取整个以太坊中节点的路由表信息。

基于以太坊节点发现协议所实现的节点路由表感知方法能够通过迭代生成目标值 *target* 的方式获取目标节点路由表中所有 bucket 中存储的对等节点信息。以太坊节点发现协议可以以任何节点作为网络入口, 并将其用于对等网络中寻找其他的参与者。在节点路由表感知方法中, 感知节点首先进行初始化工作, 通过客户端硬编码的引导节点加入到以太坊网络, 能够获取到引导节点的邻居节点, 并将邻居节点作为后续请求过程的目标节点。同时感知节点也将自身添加到引导节点的路由表中, 以供其他节点发起连接请求。初始化过程完成之后, 感知节点可以对目标节点的路由表进行感知, 节点之间的交互过程如下:

(1)感知节点通过握手协议检测目标节点的活跃性。感知节点通过向目标节点发送 Ping 数据包以期许获得响应, 当目标节点收到 Ping 消息并对其验证之后, 会向感知节点发送 Pong 数据进行响应。

(2)感知节点选定目标节点的某个特定 bucket 构建目标值 *target*, 然后向目标节点发送包含当前 *target* 的 FindNode 消息, 表示希望获取目标节点当前 bucket 中的邻居节点。

(3)目标节点在接收到 FindNode 消息之后, 以 Neighbor 消息进行回应, 并在该消息中携带满足 *target* 的邻居节点集合。

感知节点在完成当前请求过程后, 即可获得目标节点的某个指定 bucket 中的对等节点信息。但是, 以太坊的节点拥有 17 个 bucket, 如果感知节点想要获取目标节点的整个路由表信息, 则还需要构建 16 个特定的目标值 *target*, 然后通过 FindNode 消息映射到不同的 bucket 中。通过对 Neighbor 信息中所携带的对等节点信息进行筛选、过滤, 即可得到目标节点的所有对等连接, 具体步骤如算法 1 所示。

算法 1.目标节点路由表感知算法.

输入: 目标节点的公钥 *remote_key*

输出: 目标节点的 *target_key* 集合 *Value*

1. 初始化 *flag* 用于记录当前请求轮次, 且 *flag*=1
2. 初始化 *limit* 用于记录 bucket 是否已经成功请求, 其中第 *i* 位为 1 表示目标节点的第 *i* 个 bucket 已获取, 为 0 则相反, 且 *limit*=(1 ≪ 17)–1
3. *Value*=*new Set*( )//生成新的空集合
4. WHILE*flag*!=*limit* DO
5. *target*=*randbits*(521)
   //*randbits* 函数生成的 521 位随机值
6. *target_key*=*keccak*256(*target*)
   //由 keccak256 对随机值进行哈希
7. *d*=*dis*(*target_key*,*remote_key*)
8. *index*=*heighest_bit*(*d*)–239
   //根据最高位推测落入那个 bucket
9. IF NOT(1 ≪ *index* & *flag*)THEN
   //如果当前编号的 bucket 尚未请求
10. *FindNode*(*target_key*)
11. *flag*=*flag*|(1 ≪ *index*)
12. *Value.add*(*target_key*)

k–bucket.update(*remote_key:Value*)

当感知节点完成对目标节点的整个路由表的感知

之后, 通过筛选即可获得尚未得到的对等节点。然后感知节点可以与新获得的对等节点构建连接, 同时这些节点又可以分享它们的邻居节点给感知节点, 于是感知节点再与这些邻居节点连接, 如此不断发现和连接, 感知节点最终能够获得整个网络的对等节点信息。

3.2 活跃连接数量推测

比特币的消息转发协议采用的是通知模式, 即当节点在收到新消息并验证其合法性后, 会向其邻居节点发送 INV 消息, 收到 INV 消息的节点会判断自身是否已经收到当前消息。如果没有, 则向该节点发送 GETDATA 消息请求完整信息, 反之则直接忽略该消息。尽管通知模式下的传播方式尽可能的避免了消息在网络中的冗余传播, 但是仍旧造成了不可避免的消息延迟, 无法保证消息在网络中的传播效率。为解决这一难题, 以太坊采用了结构化的组网方式, 并且为了保证新消息在网络中的快速传播, 采用了一种改进的消息转发模式[17]。

在以太坊的消息传播过程中, 一旦收到来自对等节点的 NewBlock 消息并验证区块头和难度检查的有效性之后, 节点通过 NewBlock 消息将新区块转发给一部分已经连接的对等节点, 如图 3 所示。当节点执行完包含在区块中的所有事务后, 区块则被认定为是有效的, 节点将会与之前没有通知的对等节点发送一条关于新区块的 NewBlockHashes 消息。如果这些对等方未通过 NewBlock 消息从其他节点接收到完整的新区块, 则会向该节点请求完整的区块。除此之外, 事务在其传播过程中也遵循上述的传播方式。在以太坊的消息传播过程中, 节点不会将区块或者事务转发给它确认已经知道该消息的对等节点。在消息传播过程中, 完整信息转发的邻居节点数量的选择是消息传播的复杂性和传播协议的健壮性之间的权衡。

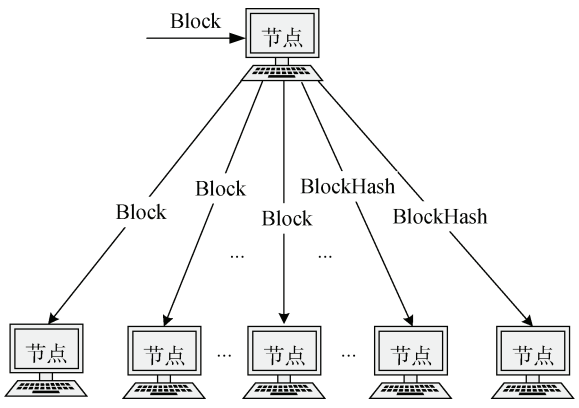


图 3 网络中信息广播过程

Figure 3 Information broadcasting procedure in network

以太坊在选择消息转发节点时, 无论是将消息本体还是消息哈希转发给对等节点, 都取决于节点自身所记录的对等节点列表。节点从其对等节点列表中依次选择节点进行转发, 而对于每一个接收到的消息, 对等节点列表中的邻居节点的顺序总是随机的, 即使对于同一组邻居节点, 它们在对等节点列表中的顺序也会因为消息的不同而不同。这意味着每次的消息转发节点都会以随机的方式选择转发节点。在具体实现过程中, 对于每一个连接的节点, 感知节点都可以从其邻居节点接收节点传播记录, 该记录存储了节点转发的消息本体数量和转发的消息哈希数量, 分别记为 *MsgFull* 和 *MsgHash*。假设在一段时间内节点的活跃连接数量不会发生变化, 而每次节点在进行消息转发时的节点选择总是随机的。根据统计学原理, 该时间段内感知节点所收到的消息本体数量与消息哈希数量保持着一定的比例关系。在以太坊有线协议 eth/66 中, 节点每次会将消息本体随机转发给平方根个已连接的节点[18]。假设当前节点存在 M 个活跃连接, 则节点的活跃连接数量可由如下公式进行推测:

$$\frac{\sqrt{M}}{M-1}=\frac{Ms\ gFull}{Ms\ gHash+Ms\ gFull}$$

其中节点会默认将消息传递给该节点的对等节点排除在外, 因为节点不会将消息转发给该对等节点。根据公式来推测活跃连接的数量从理论上是可行的, 换言之, 感知节点只需要收集有关消息本体和消息哈希的数据, 并通过简单地计算即可得到节点的活跃连接数量 $M\approx(\frac{Ms\ gHash+Ms\ gFull}{Ms\ gFull})^2$。

3.3 活跃连接验证

与比特币所使用的未使用输出模型不同, 以太坊采用账户余额的方式对事务进行记录, 并通过内存池存储待确认的事务。在以太坊内存池中, 以太坊将事务发送方账户绑定到接收方账户, 对于每一个未确认的事务, 发送方账户使用一个单调递增的 nonce 值与其相关联, 而接收方账户将未确认的事务存储在其未确认事务缓冲池中, 并通过所属账户对事务进行 nonce 值的升序排列, 如图 4 所示。在节点内存池中, 如果新事务的 nonce 值等于相同发送方在内存池中最大的 nonce 值加 1, 则称该事务为待办事务。否则, 如果新事务的 nonce 值严格大于最大 nonce 值加 1, 则称该事务为未来事务。根据 nonce 值的不同, 以太坊节点的内存池划分为两个不同的部分, 即 pending 池和 queued 池。

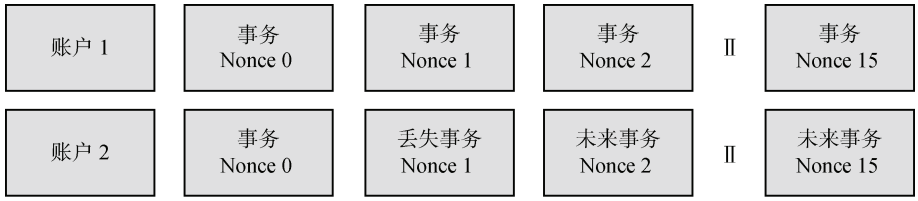


图 4　节点内存池中事务存储模式

Figure 4　Transaction storage mode in node memory pool

Pending 池维护尚未包含在区块链的块中, 但已经准备打包到新块中的挂起事务。矿工节点可以从 Pending 池中选择合适的事务, 并根据以太坊的打包规则将它们打包到新区块中。Pending 池中存储的事务严格按照 nonce 值连续且递增的规则排列, 对于同一个账户, 最多可以存储 16 个未决事务。一旦事务进去内存池中的 pending 池中, 内存池模块将通知协议管理模块, 该事务将被转发到尚未拥有该事务的邻居节点。Queued 池负责维护账户中因缺少 nonce 值而不连续且尚未准备好打包到新区块的未来事务。对于同一个账户, queued 池最多可以存储 64 个未来事务。由于以太坊客户端承认在内存池中具有潜在透支风险的事务, 所以在经过初步验证之后, 未来交易被存储到 queued 池中, 但是 queued 中的未来事务并不会被转发给相邻的活跃节点。

对于以太坊事务而言, 节点的内存池存在替换和驱逐两种事务策略, 即以太坊节点的内存池中的事务可以被后续事务以足够高的 GasPrice 替换或者驱逐, 事务的替换和驱逐是以太坊的标准功能, 受到以太坊客户端的广泛支持[19]。当从其他节点接收到传播的新事务时, 节点会对新事务进行验证并决定是否接纳事务进入内存池。一旦节点选择接纳事务进入内存池, 则可能触发事务替换或者事务驱逐。事务替换是指一个事务可以被来自同一账户且具有相同 nonce 值的高价事务所替换, 客户端默认超出原价的 10%时低价事务被替换。事务驱逐则是指内存池中已经存在的事务可以被来自不同发送方或者不同 nonce 值的其他事务驱逐, 一般发生于内存池溢出时。

根据上述的理论基础, 本文提出了一种改进的节点间活跃连接的隔离验证方法, 该方法基于以太坊中未来事务特征和事务替换策略。如图 5 所示, 通过向网络中部署一个验证节点 M, 验证节点 M 向目标节点 A 传播一个高价交易 TxA, 再向目标节点 B 传播一个低价交易 TxB, 然后向网络中其他节点传播一个中价交易 TxC。通过观察节点 B 上是否存在节点 A 中的高价事务 TxA 推测活跃连接, 其中 TxA、TxB 和 TxC 是同一账户发送的具有相同 nonce 值的不同价格的事务。直观的说, 该方法根据以太坊的交易替换策略实现强制隔离属性, 由于传播过程中的高价事务 TxA 可以代替低价事务 TxB, 但是不能代替其他节点上的中价事务 TxC, 确保隔离了事务 TxA 从目标节点 A 传播到目标节点 B 之间的其他可用通路, 如果节点 A 和节点 B 之间不存在连接, 那么

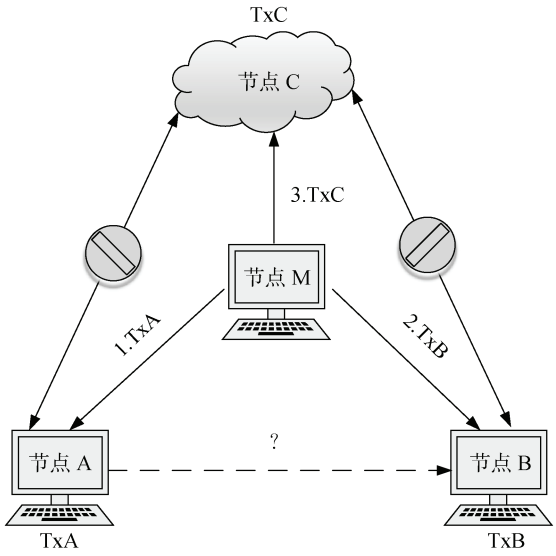


图 5　事务的隔离验证方法示意图

Figure 5　The isolation verification method of transactions

高价事务 TxA 即不会传播, 也不会到达节点 B 进而替换低价事务 TxB。

假设以太坊是一个没有网络分区的强连接对等网络, 用于检测节点间是否存在活跃连接的隔离验证方法的步骤如下:

(1)节点 M 向节点 A 发送一个未来事务 TxA, 其中 GasPrice 设置为(1+0.5R) * Y Gwei。由于发送的是未来事务, 节点 A 在收到 TxA 的时候仅进行合法性验证, 将该事务放入 queued 队列, 而不对其进行广播。同理, 节点 M 向节点 B 发送一个未来事务 TxB, 其中 GasPrice 设置为 (1-0.5R) * Y Gwei。

(2)节点 M 向节点 C 发送两个连续的 pending 事务 TxP 和 TxC, 其中 GasPrice 均为 Y Gwei。需要注意的是 TxA、TxB 和 TxC 是由同一账户发出的具有

相同 Nonce 值的事务, 且 TxP 事务作为未来事务的激活事务。

(3)节点 C 接收到 pending 事务 TxP 和 TxC 后, 会通过传播协议将 TxP 和 TxC 传播出去。当节点 A 收到 TxP 和 TxC 后, TxP 作为激活事务, 经过验证后进入 pending 池, 同时发现 queued 池中存在与 TxC 具有相同 nonce 值的 TxA, 根据事务替换策略, TxC 的 GasPrice 未达到替换 TxA 的最低门槛, 因此 TxC 不会替换 TxA。由于激活事务 TxP 的到来, TxA 本应在经过验证之后由 queued 池转入 pending 池, 但是由于节点 M 的余额为 2Y Gwei, 在预支 TxP 的消耗之后, 已经没有足够的余额去支付 TxA, 因此 TxA 无法进入 pending 池。

(4)当节点 B 收到 TxP 和 TxC 后, TxP 作为激活事务, 经过验证后进入 pending 池, 同时发现 queued 池中存在与 TxC 相同 Nonce 值的 TxB, 根据事务替换策略, TxC 的 GasPrice 未达到替换 TxB 的最低门槛, 因此 TxC 不会替换 TxB。与步骤 3 中不同的是, 此时 queued 池中的 TxB 在经过验证后由 queued 池转入 pending 池, 此时事务 TxB 会被广播到网络中, 但是由于其具有最低的 GasPrice, 因此无法对 TxC 事务进行替换。

(5)在对探测节点 M 进行账户余额调整后, 事务 TxA 会进入节点的 pending 池中并对外传播。此时考虑两种情况, 第一种情况是节点 A 和节点 B 是直接相连的, 那么节点 A 会把 TxA 传播到节点 B, 节点 B 中新到达的 TxA 将取代 TxB, 因为 TxA 的 Gas 价格高于 TxB。这种情况下, TxA 也会传播到节点 C, 但是不会替换掉节点 C 中的 TxC, 因为 TxA 的 Gas 价格未达到替换 TxC 的门槛。换言之, TxA 在节点 A 上被隔离, 在经过一定的传播延迟之后, 节点 M 可以从节点 B 接收到 TxA。另一种情况下, 节点 A 和节点 B 未连接, 节点 A 仅将 TxA 传播到节点 C, 但由于 TxA 无法替换 TxC, 且 TxC 无法替换 TxB, 因此节点 B 会保留 TxB, 节点 M 不会从节点 B 接收到 TxA。

为确保隔离验证过程的成功, 事务 TxC 应当尽可能的驻留在其余节点 C 的内存池。因此, 事务 TxC 的 GasPrice 即要保证足够低到不会被包括到下一个区块中, 又要保证足够高到不会被即将到来的事务所驱逐。对于隔离验证的测试成本而言, 由于验证节点向网络中发送的未决事务具有较小的 GasPrice, 是否会被打包进区块取决于矿工节点的内存池的状态, 因此产生的代价微乎其微。

4 实验与分析

考虑到上述所提出的方法会产生一定的 Gas 消耗, 出于实验真实性、便捷性和准确性考虑, 本文基于以太坊 Ropsten 测试网进行实验性的检测研究。Ropsten 测试网络与以太坊主网拥有出相同的节点分布、网络特征和共识机制, 仅在网络规模上呈现出不同。因此, Ropsten 测试网同样能够体现出主网的特征, 是以太坊开发者提供的优质测试环境。通过向测试网络中部署感知节点, 我们于 2022 年 6 月 3 日至 24 日对以太坊测试网络进行多次重复实验, 对网络中的可路由节点进行感知, 对可路由节点路由表的潜在活跃连接进行请求, 并通过隔离验证的方式对节点的活跃连接进行分析, 最终获取了以太坊测试网络 Ropsten 的网络快照, 并对其网络拓扑属性进行分析。

4.1 网络分析

尽管区块链网络没有统一的准入规则, 节点的随意加入和离开会对网络一致性维护产生影响, 但是在稳定运行的区块链网络中, 新加入网络中的节点总是倾向与网络中更早加入网络、更加稳定的节点相连, 一旦节点达到连接上限, 便会维持当前连接。现有研究表明[20], 区块链网络中许多 P2P 连接都是相对固定的, 在以天为单位的网络监测中, 相邻时间的网络搅动率仅为 10%, 由此可见, 区块链网络结构在短时间内不会发生较大变化。本文所提出的以太坊网络拓扑感知方法的感知周期远小于节点在网络中的在线周期长度, 因此能够实现具有时效性的网络结构的感知。

通过向 Ropsten 网络中部署感知节点并进行多次重复实验。实验发现感知节点平均每天收集到 6199 个对等节点, 其中超过 5500 个对等节点在网络中处于孤立状态, 所构成的对等网络最大连通分量的节点数量仅占所有感知节点的 10%左右。而相同时期下, EthScan 所提供的 Ropsten 网络节点数量为 6500 个节点。如图 6 所示, 在对等节点的采集过程中, 感知节点最初采集的对等节点数量呈现出爆发式的增长, 但是随着时间的推移, 尽管仍旧能够从网络中获得新的数据包, 但是数据中的大多数节点都是重复的, 节点数量的增长逐渐呈现出收敛的趋势。在节点感知过程中, 每当获取到新的节点, 感知节点会通过发送消息来验证新节点是否处于活跃状态。我们可以得出结论, 以太坊 Ropsten 网络中活跃节点的数量在 600 左右, 而不活跃节点的数量仍旧是未知的, 因为网络中仍旧存在无法探测到的对等节点。

由于以太坊允许节点随意加入和离开网络, 而无约束的协议策略促使网络中节点状态的多样化。本文基于以太坊的通信协议的特征, 对以太坊节点

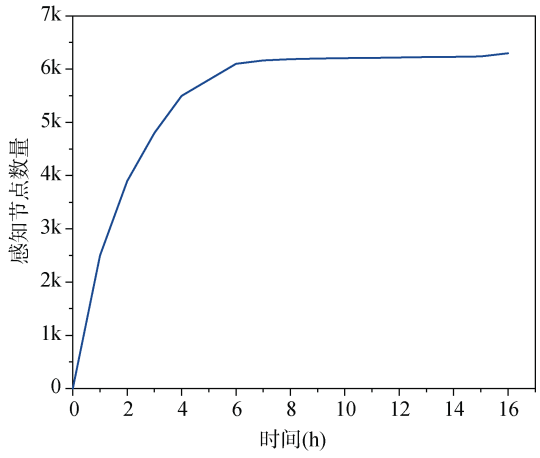


图 6 感知过程中节点的增长趋势

Figure 6 Growth trend of nodes in perception procedure

的状态进行了划分，根据以太坊节点是否在线，将节点划分为活跃节点和非活跃节点。根据节点是否可以接收外部连接请求，并与其他节点进行握手行为，将节点划分为可路由节点和不可路由节点，通常来说不可路由节点位于防火墙或者 NAT 网络之后。

本实验对以太坊 Ropsten 网络进行多次感知，在活跃节点集中，可路由节点的数量仅仅只有约 100 个，各类型节点的数量如图 7 所示。由于以太坊节点发现协议使用的是 UDP 协议，作为一种无连接的协议是无法决定一个节点是否是可路由节点。因此，通过判断节点是否启动并成功发起过 TCP 连接来推测节点是否是可路由节点，而通过节点路由表中是否存在对等节点信息恰好能够满足判断需求。通过实验分析可知，在整个网络所感知的对等节点中，活跃且可路由的节点不足 2%，而大量的无用节点充斥整个网络可能会对网络效率产生影响。

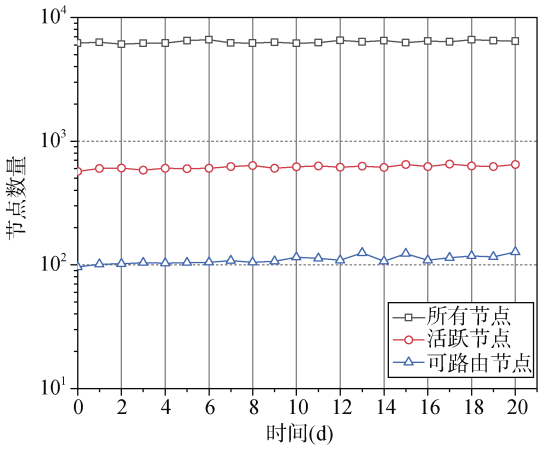


图 7 重复实验下不同类型节点的数量

Figure 7 Number of different types of nodes under repeated experiments

除此之外，根据实验感知所得的活跃节点的 IP 地址，对节点数量在不同国家的分布进行了统计。如表 1 所示，网络中近 75%的活跃节点比较集中的出现在前 10 个国家，其中位于美国和德国的活跃节点占据了 50%之多，而中国、新加坡和芬兰则是分别占据了 9.44%、5.25%和 5.09%。为探究节点 IP 和 ID 之间的映射关系，我们对感知阶段内网络中节点 ID 和节点 IP 的重叠进行了统计，其中约有 65 个节点 ID 拥有不止一个节点 IP，约有 260 个节点通过不同的节点 ID 接入到网络中。通过对重叠现象的统计，我们可以对某些节点的行为进行建模分析，一般情况下将频繁修改节点 ID 视为一种异常行为。

表 1 节点的国家分布表

Table 1 Country distribution of nodes

国家名称	节点数量	百分占比(%)
美国	201	33.17
德国	131	21.72
中国	57	9.44
新加坡	31	5.25
芬兰	30	5.09
韩国	20	3.31
法国	17	2.87
冰岛	17	2.87
英国	16	2.67
荷兰	12	2.02

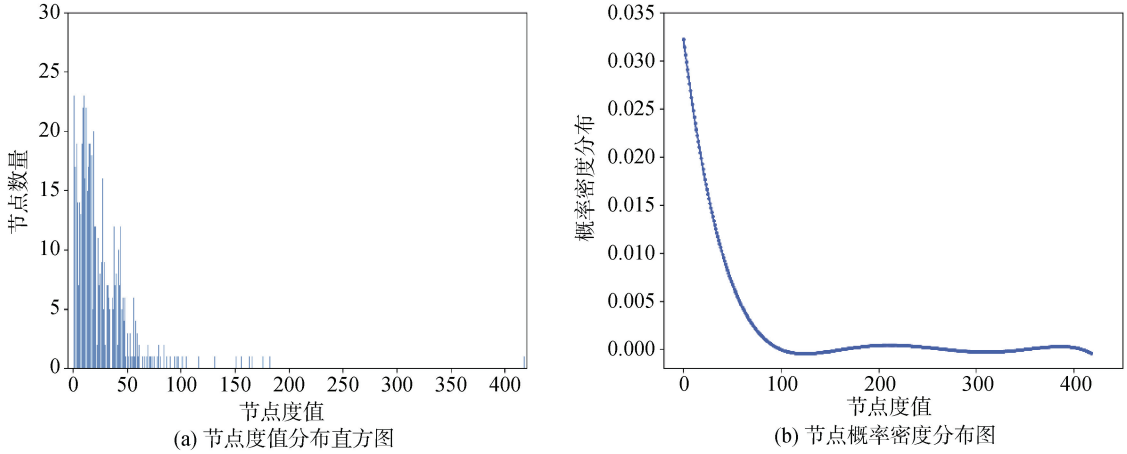
4.2 拓扑分析

在对以太坊网络的拓扑进行分析之前，通常只对网络中活跃的节点进行建模。根据以太坊的节点发现协议，新加入网络中的节点通过向可路由节点发送 FindNode 消息来获取其他对等节点信息，而离线节点或者不活跃节点往往不响应 FindNode 请求。

尽管活跃节点仅占据全网节点的一小部分，但是它们共同维护着整个网络的活跃连接并为新加入网络中的节点提供服务，同时为网络中消息的传播做出重大贡献，因此以活跃节点进行网络分析足以体现出以太坊的网络特征。本文将感知节点所收集的对等节点集合进行筛选，删除网络中的孤立点，最后通过网络中的最大连通分量子图 $G=(V,E)$来表示以太坊的网络拓扑结构。

节点的度分布是分析网络拓扑结构重要指标之一。以太坊网络结构不会因为节点的频繁连接与断开而发生变化，稳定运行的以太坊网络在短时间的网络拓扑结构并不会发生较大的波动。因此，通过对整个网络中所有节点的状态进行快照是分析网络拓

扑结构的合理方式。从节点的度值分布和拓扑结构可以对以太坊 Ropsten 测试网络的隐藏特征进行分析。针对实验采集的数据, 如图 8(a)以柱状图展示了以太坊节点的度分布。由图可知, 网络中超出 95%的节点表现出较小的度值, 而且集中的分布在 30 左右。实验分析表明, 网络中节点的平均度值为 25, 更加契合以太坊客户端的默认活跃连接数量限制的要求, 本文所提出的方法度量出节点的活跃连接, 表现出了更高的测量精度。此外, 当对等连接数量超出默认数量限制之后, 度数较高的节点数量更是微乎其微。由此可见, 以太坊测试网络中节点的连接状况具有严重的不均匀分布性, 网络中少数节点拥有大量的对等连接, 而大多数节点只拥有少量的对等连接, 这种异质性的连接方式表明网络具有无标度特性。图 8(b)则描述了以太坊测试网节点概率密度分布图, 图中曲线是由节点概率分布值进行拟合所得。文献[21]中提出以零假设显著性检验的 p 值来验证所收集数据的经验分布是否符合幂律分布, 并将零假设无效的阈值由 0.05 提升至 0.1, 即当样本数据的 p<0.1 时, 幂律分布的假设是不成立的。为了验证实验所得数据是否符合幂律分布, 本文以最大似然估计方法对节点的度数秩图进行拟合, 实验样本数据结果表明 p=0.13, 即拟合所得曲线符合幂律分布的特征。



(a) 节点度值分布直方图 (b) 节点概率密度分布图

图 8 以太坊网络拓扑分析

Figure 8 Topological analysis of ethereum test network

从图 8(a)节点度值的分布直方图中可以看出本文实验所得的节点度值普遍较小, 与文献[13]所提出的方法相比, 本文所提出的以太坊活跃网络拓扑感知方法旨在获得更高的精度的网络结构, 即推测节点间的活跃连接。尽管以太坊的节点发现协议能够实现对节点路由表的填充, 但节点通过 TCP 协议与对等节点构建可靠会话时, 并不是同时和路由表中所有节点建立连接。出于网络安全的考虑, 以太坊客户端默认最多可以发起 25 个实时连接, 而这些实时连接来源于节点路由表中存储的对等节点。在本文所提出的活跃连接验证方法根据以太坊事务内存池中未来事务的特征实现节点之间的隔离, 并根据事务流向判断节点之间是否存在活跃连接。在实验过程中, 节点的活跃连接数量推测作为限制条件, 可以加快活跃连接验证的过程, 而隔离验证阶段可以通过 Web3 所提供的 RPC 接口向节点发送 eth_ getTransactionByHash 命令查询事务是否存在于节点的内存池, 并以此为依据推测节点直接是否存在活跃连接。

在具体的实验过程中, 通过本文所提出的方法和文献[13]所提出的 NodeFind 方法对以太坊 Ropsten 测试网络进行多次重复测试, 实验结果如图 9 所示。本文所提出的方法所对测试网络进行度量后发现, 网络中节点的平均度值为 25.5, 保持相对稳定的状态。这与以太坊客户端默认节点最多可以保持 25 个对等连接相吻合。不可否认的是, 网络中确实存在少量度值较大的节点, 其原因是用户可以根据自身需求更改节点的连接限制, 一般认为此类对等连接无上限的节点为超级节点。文献[13]所提出的基于 NodeFind 的网络感知方式以整个节点路由表信息构建网络拓扑, 其节点度值存在较大的波动, 其原因在于以太坊自由的准入策略。而文献[13]实验结果的节点平均度值远大于本文所提出方法的度量结果, 其原因在于 NodeFind 方法仅对节点路由进行感知, 而并未对节点的活跃性进行检测, 必然存在大量已经离开以太坊网络但尚未被清除出路由表的对等节

点。尽管以太坊节点路由表可以存储 272 个对等节点，而实验展示的对等节点数量维持在 80 左右，其原因一方面在于 bucket 的映射机制，位于上游的 bucket 有很大概率被选中，而编号较小的 bucket 往往仅存储少量对等节点。另一方面在于以太坊 Ropsten 测试网络规模有限，无法达到路由表数量的阈值。

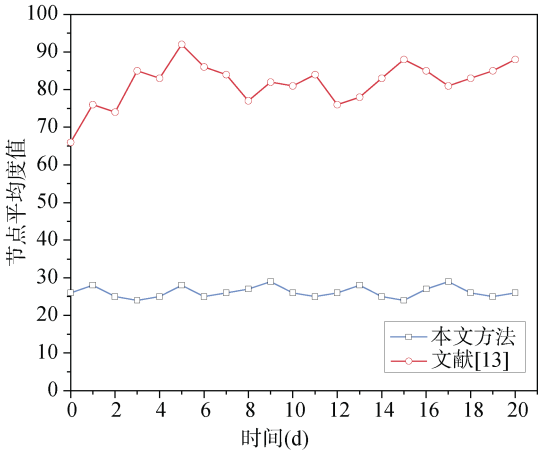


图 9　不同方法在重复实验下节点的度值变化

Figure 9　Node degree variations under repeated experiments with different methods

以太坊的活跃网络拓扑具有较小的节点度值，且节点的度分布遵循幂律分布。由于幂律分布的优先附着原理，新加入网络的节点很容易连接到连接度较高的节点上，进而很快被整个网络所知，因此幂律分布的网络结构在信息传播和链路利用率上表现出较好的性能。但是，一旦网络中发生蓄意攻击，即使大多数节点仍旧是可达状态，网络将不能利用节点之间的最短路径进行信息传播，网络的传播延迟将剧烈增长。

小世界网络则是指网络中大多数节点彼此之间并不连接，但是通过极少的跳数就可以到达其他对等节点的网络，其中平均最短路径长度和聚类系数是评价小世界网络的重要指标[22]。从安全和公平的角度来看，理想中的以太坊网络应该拥有一个均匀的节点度值和紧密的中心分布，从而避免网络的碎片化，同时较大的聚类系数和较小的平均最短路径确保信息能够迅速的在网络中传播。

以太坊选择 Kademlia 机制存储对等节点的主要原因是这种结构化的索引存储方式会产生一个较小最短路径的网络拓扑结构，从而确保更强的消息传播能力和数据同步能力。因此，从网络设计的角度来看以太坊网络拓扑结构本身具有小世界特征，所以实验的主要目的是对感知得到的拓扑结构进行验证其是否具有小世界特征。

为了验证以太坊网络是否真的具有小世界网络特征，实验通过多次收集 Ropsten 测试网络的数据计算其特征，为了保证实验的真实性，同时生成一个相同大小的随机网络进行对比[23]。如表 2 所示，以太坊 Ropsten 网络的聚类系数约为 0.23，而同等规模的随机网络的聚类系数远低于该数值。由于以太坊 Ropsten 网络规模远小于主网，所以 Ropsten 网络的网络直径大小为 7，网络的平均最短路径长度约为 3.6。因此，实验结果表明以太坊网络具有很明显的小世界效应，符合区块链网络分布式设计的初衷。

表 2　相同规模下以太坊与随机图的网络属性对比

Table 2　Comparison of network properties between ethereum and random graphs under same scale

	Ropsten	Random Graph
网络直径	7	3
外围节点数	36	292
平均最短路径	3.6	2.27
网络聚集系数	0.23	0.044

5　结论

作为以太坊的重要组成部分之一，对等网络对以太坊生态系统的方方面面都具有影响。本文基于以太坊网络协议特征和内存池中事务处理策略提出了一种以太坊网络拓扑结构的感知方法，与传统以太坊网络测量方法不同，本文所提出的感知方法所探测的节点间连接是节点间的活跃连接，而非节点路由表中的潜在连接。在本文中，我们对以太坊测试网的节点分布和网络拓扑结构进行了分析，揭示了以太坊网络的隐藏特征。

对于未来的工作，我们将针对方法进行优化，进一步提高方法的感知效率，并验证攻击者是否能够通过网络的拓扑结构发起蓄意攻击以谋取更多的利益。

参考文献

[1] Squarepants S. Bitcoin: A Peer-to-Peer Electronic Cash System[J]. *SSRN Electronic Journal*, 2008.

[2] Maymounkov P, Mazières D. Kademlia: A Peer-to-Peer Information System Based on the XOR Metric[M]. Peer-to-Peer Systems. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002: 53-65.

[3] Kiffer L, Salman A, Levin D, et al. Under the Hood of the Ethereum Gossip Protocol[C]. *International Conference on Financial Cryptography and Data Security*, 2021: 437-456.

[4] Decker C, Wattenhofer R. Information Propagation in the Bitcoin Network[C]. *IEEE P2P 2013 Proceedings*, 2013: 1-10.

[5] Maeng S, Essaid M, Lee C, et al. Visualization of Ethereum P2P Network Topology and Peer Properties[J]. *International Journal of*

Network Management, 2021, 31(6): e2175.

[6] Faramondi L, Panzieri S, Setola R, et al. Assessing Node Criticality in Dynamical Distributed Systems[C]. *2019 18th European Control Conference*, 2019: 1537-1543.

[7] Ethereum DevP2P Protocol. August. 2022. [Online]. Available: https://github.com/ethereum/devp2p.

[8] Hsiao H C, Liao H, Huang C C. Resolving the Topology Mismatch Problem in Unstructured Peer-to-Peer Networks[J]. *IEEE Transactions on Parallel and Distributed Systems*, 2009, 20(11): 1668-1681.

[9] Franzoni F, Salleras X, Daza V. AToM: Active Topology Monitoring for the Bitcoin Peer-to-Peer Network[J]. *Peer-to-Peer Networking and Applications*, 2022, 15(1): 408-425.

[10] Tian G H, Hu Y H, Chen X F. Research Progress on Attack and Defense Techniques in Block-Chain System[J]. *Journal of Software*, 2021, 32(5): 1495-1525.
(田国华, 胡云瀚, 陈晓峰. 区块链系统攻击与防御技术研究进展[J]. *软件学报*, 2021, 32(5): 1495-1525.)

[11] Delgado-Segura S, Bakshi S, Pérez-Solà C, et al. TxProbe: Discovering Bitcoin's Network Topology Using Orphan Transactions[C]. *International Conference on Financial Cryptography and Data Security*, 2019: 550-566.

[12] Essaid M, Park S, Ju H T. Bitcoin's Dynamic Peer-to-Peer Topology[J]. *International Journal of Network Management*, 2020, 30(5).

[13] Kim S K, Ma Z E, Murali S, et al. Measuring Ethereum Network Peers[C]. *The Internet Measurement Conference 2018*, 2018: 91-104.

[14] Gencer A E, Basu S, Eyal I, et al. Decentralization in Bitcoin and Ethereum Networks[C]. *International Conference on Financial Cryptography and Data Security*, 2018: 439-457.

[15] Miller A, Litton J, Pachulski A, et al. Discovering bitcoin's network topology and influential nodes[J]. Coinscope: Discovering Bitcoin's Network Topology and Influential Nodes (Website), 2015: 1-17.

[16] Ethereum Node Discovery Protocol. August. 2022. [Online]. Available: https://github.com/ethereum/devp2p/blob/master/discv4.md.

[17] Ethereum Wire Protocol. May. 2022. [Online]. Available: https://github.com/ethereum/devp2p/blob/master/caps/eth.md.

[18] Wang T T, Zhao C H, Yang Q, et al. Ethna: Analyzing the Underlying Peer-to-Peer Network of Ethereum Blockchain[J]. *IEEE Transactions on Network Science and Engineering*, 2021, 8(3): 2131-2146.

[19] Li K, Tang Y Z, Chen J Q, et al. TopoShot: Uncovering Ethereum's Network Topology Leveraging Replacement Transactions[C]. *The 21st ACM Internet Measurement Conference*, 2021: 302-319.

[20] Li R G, Zhu J W, Xu D W, et al. Bitcoin Network Measurement and a New Approach to Infer the Topology[J]. *China Communications*, 2022, 19(10): 169-179.

[21] Neudecker T, Andelfinger P, Hartenstein H. Timing Analysis for Inferring the Topology of the Bitcoin Peer-to-Peer Network[C]. *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing*, *Advanced and Trusted Computing*, *Scalable Computing and Communications*, *Cloud and Big Data Computing*, *Internet of People*, *and Smart World Congress*, 2016: 358-367.

[22] Ferretti S, D'Angelo G. On the Ethereum Blockchain Structure: A Complex Networks Theory Perspective[J]. *Concurrency and Computation*: *Practice and Experience*, 2020, 32(12): e5493.

[23] Baumgart I, Heep B, Krause S. OverSim: A Scalable and Flexible Overlay Framework for Simulation and Real Network Applications[C]. *2009 IEEE Ninth International Conference on Peer-to-Peer Computing*, 2009: 87-88.

白销东 于 2019 年在长江大学计算机科学与技术专业获得学士学位。现在南京理工大学电子信息专业攻读硕士学位, CCF 学生会员。研究领域为网络安全、区块链安全。研究兴趣包括: 区块链。Email: 120106222721@njust.edu.cn

魏松杰 于 2009 年在美国特拉华大学计算机网络专业获得博士学位。现任南京理工大学计算机科学与工程学院副教授。研究领域为计算机应用技术、网络空间安全。研究兴趣包括: 区块链、软件定义网络、网络攻防。Email: swei@njust.edu.cn

刘代东 于 2021 年在南京理工大学软件工程专业获得学士学位。现于南京理工大学电子信息专业攻读硕士学位。研究领域为区块链、区块链应用。研究兴趣包括: 区块链、区块链应用。Email: 917106840328@njust.edu.cn