

幽灵交易:一种可实施的大规模以太坊交易池拒绝服务攻击

Ghost transactions: a practical and large scale denial of Ethereum transaction pool service attack

期刊：	中国科学: 信息科学
稿件ID：	SSI-2025-0419.R3
稿件栏目：	论文
作者提交日期：	2026-04-08
参与作者列表：	苏申, 孙群洪, 王畅, 张晓洁
关键词：	拒绝服务攻击, 内存池, 区块链, 交易驱逐机制, 以太坊客户端
英文关键词：	denial of service attack, memory pool, blockchain, transaction eviction mechanism, Ethereum client
学科领域：	网络安全
专题：	

Accepted
For Review Only

https://www.sciengine.com/doi/10.1360/SSI-2025-0419

中国科学:信息科学 在审文章 onlinelatex

SCIENTIA SINICA Informationis

论文

幽灵交易：一种可实施的大规模以太坊交易池拒绝服务攻击

苏申[1]，孙群洪[2,3]，王畅[1]，张晓洁[1*]

1. 网络空间安全学院, 广州大学, 广州市 510006
2. 新型网络部, 鹏城实验室, 深圳市 518055
3. 计算机科学与工程系, 南方科技大学, 深圳市 518055
* 通信作者. E-mail: xjzhang@gzhu.edu.cn

国家自然科学基金 (批准号:62572138) 资助项目、数字化学习技术集成与应用教育部工程研究中心 (批准号:1331005) 资助项目以及广东省自然科学基金 (批准号:2024B1515020105) 资助项目.

摘要 禁用交易池服务将对以太坊网络造成严重破坏. 本文通过调研主流以太坊客户端源代码, 提出一种名为“幽灵交易”的拒绝服务攻击方法, 可禁用支持“非顺序交易缓冲”的交易池服务. 该方法的核心逻辑是利用多个账户向以太坊网络先插入一系列低燃料价格交易, 再插入一系列高燃料价格交易, 这类交易序列能在目标客户端中传播, 并以“幽灵”形式滞留于交易池中难以移除. 网络靶场实验结果表明, 该攻击方法可在 2 分钟内破坏 100 个节点的以太坊网络交易池服务并实现恶意驱逐交易; 在 Sepolia 测试网的实验进一步证实, 该攻击方法能让真实以太坊测试网环境的交易池服务失效. 通过分析以太坊主网非顺序交易, 本文总结出受攻击影响的去中心化应用行为 (如代币交易、区块链钱包操作等), 此类受影响行为规模可观且覆盖广泛的区块链应用场景.

关键词 拒绝服务攻击, 内存池, 区块链, 交易驱逐机制, 以太坊客户端

1 引言

作为规模最大、影响力最广的可编程区块链, 以太坊 [1)]承载着数千亿美元的资产, 也因此成为恶意攻击的重点目标 [2)]. 现有攻击多聚焦于合约漏洞利用与外部账户私钥盗窃, 而以太坊交易的共识与传播过程被认为相对安全: 一方面, 共识算法经专门设计可抵御女巫等恶意攻击; 另一方面, 现有攻击方法实施的前置条件较难满足 [1,2]. 目前, 除“燃料上限” (gas limit) 外, 交易传播与打包过程并无其他安全防护机制 [3)]. 这种防护机制缺失意味着以太坊交易的传播与共识过程存在安全隐患, 可能对区块链应用生态构成威胁. 在此背景下, 本文提出疑问: 能否以低成本、高影响且自动化的方式, 对以太坊交易传播与共识过程发起切实可行的攻击?

1) https://etherscan.io
2) https://hacked.slowmist.io/statistics/?c=ETH&d=all
3) https://ethereum.org/en/developers/docs/gas/

引用格式: 苏申, 孙群洪, 王畅, 张晓洁. 幽灵交易: 一种可实施的大规模以太坊交易池拒绝服务攻击. 中国科学: 信息科学, 在审文章
Shen SU, Qunhong SUN, Chang WANG, Xiaojie ZHANG. Ghost transactions: a practical and large scale denial of Ethereum transaction pool service attack. Sci Sin Inform, for review

https://www.sciengine.com/doi/10.1360/SSI-2025-0419

针对以太坊交易传播与共识过程的攻击研究已有初步探索, 文献 [3] 提出的 DETER 攻击方法最具代表性. 在该方法中攻击者以节点身份接入矿工网络, 并向目标矿工节点发送“未来交易”(无法被打包成区块但会驱逐正常交易的特殊交易), 实现对矿工节点交易池的拒绝服务 (Denial of Service, DoS) 攻击. 但在本文聚焦的“攻击整个以太坊网络”场景中,DETER 攻击存在明显可实施性缺陷, 具体表现为四点: 一是矿工通常对节点连接设置严格限制, 攻击者难以获取发送“未来交易”的接入权限 [4]; 二是收集矿工节点列表需耗费大量时间与成本, 且防御方仅需断开异常节点连接即可轻松抵御; 三是以太坊网络已升级至 2.0 版本, 矿工角色被提议者取代, 而提议者的动态性导致定位难度极大; 四是 DETER 攻击仅针对单个矿工节点有效, 无法实现对整个以太坊网络的攻击覆盖.

尽管 DETER 攻击无法解决本文关注的问题, 但其“未来交易”的思路为研究提供了方向, 即利用高燃料费非顺序交易占用存储资源. 根据交易打包规则, “未来交易”暂不满足区块打包条件 [5], 但其会占用正常交易的内存空间. 要实现攻击目标, 需解决三个核心问题:1) 攻击交易需能在目标以太坊节点间自动广播;2) 攻击成本需较低;3) 攻击发起后即使被检测到也难以防御.

为解决上述问题, 本文深入研究以太坊客户端 (Geth [6]、Nethermind [7]) 的源代码, 重点分析交易处理细节. 研究发现, 以太坊客户端仅会将具备打包条件的待处理交易 (Pending 交易) 转发给对等节点 (这也是 DETER 攻击仅能实现点对点攻击的原因), 而其他非顺序的交易仅在本地存储, 且存储数量不影响 Pending 交易. 基于此, 本文提出一种称为“幽灵交易”的攻击交易序列. 为实现攻击交易传播, 将幽灵交易设计为同一账户按顺序发送的 Pending 交易序列; 为降低攻击成本, 攻击序列采用“低燃料费交易 + 高燃料费交易”的结构, 低燃料费确保攻击序列打包的优先级较低, 交易池填满时低燃料费交易优先被驱逐, 使高燃料费交易成为非顺序交易, 在目标节点占用存储空间; 为提升防御难度, 将交易序列拆分为多个短序列, 可灵活调整攻击组合, 使基于节点 IP 或账户地址的过滤规则失效.

为验证方法有效性, 本文在网络靶场部署的 100 节点以太坊网络中发起攻击, 研究网络延迟、攻击账户数量、攻击节点数量、交易序列编排方式及交易池大小等因素对攻击效果的影响. 实验结果显示, 该攻击可在 90 秒内劫持目标网络的交易池, 且增加攻击账户与节点数量能显著加速攻击进程. 此外, 在以太坊测试网中发起实际攻击, 通过一组远程过程调用 (RPC) 服务注入交易, 观察到交易池的排队池 (Queued 池) 被占据. 本文还对被攻击的交易展开研究, 数据分析表明, 受攻击影响的排队交易 (Queued 交易) 涉及稳定币交易、去中心化交易所、区块链钱包、二层网络 (Layer2) 等各类去中心化应用 (Dapp) 行为, 进一步证实该方法对当前区块链应用生态具有重大影响.

本文的贡献主要包括三方面:

第一, 提出“幽灵交易”序列这一实用型攻击方法, 只需接入以太坊网络即可发起自动化攻击;

第二, 通过网络靶场大规模区块链网络实验与测试网实际攻击, 验证了该方法能高效瘫痪以太坊网络的非顺序交易缓冲功能, 且在以太坊 1.0 和 2.0 版本中均有效;

第三, 对被瘫痪的 Queued 交易进行研究, 发现 Queued 交易的需求规模可观, 覆盖当前大量 Dapp 的业务场景.

全文结构如下: 第 2 章介绍相关预备知识; 第 3 章详细阐述攻击方法并给出运行示例; 第 4 章通过网络靶场实验评估攻击方法; 第 5 章在测试网中发起实际攻击, 验证方法有效性; 第 6 章对以太坊主网收集的 Queued 交易进行数据分析, 证实攻击对 Dapp 服务的影响; 第 7 章介绍相关研究工作; 第 8 章总结全文.

4) https://docs.bloxroute.com/gateway/adding-the-gateway-as-a-trusted-peer

5) https://academy.binance.com/en/glossary/nonce

6) https://github.com/ethereum/go-ethereum

7) https://github.com/NethermindEth/nethermind

https://www.sciengine.com/doi/10.1360/SSI-2025-0419

2 背景介绍

以太坊客户端应用程序均采用开源软件架构模式. 据公开数据统计,Geth 与 Nethermind 两类客户端合计占据以太坊全网节点总量的近 80%, 本文重点对这两类主流客户端的源代码展开调研, 以此剖析以太坊交易池的运行机制.

2.1 交易生命周期

图1展示了交易的完整生命周期. 在以太坊网络中, 交易需经发送者签名后, 通过 RPC 服务发送至网络. 以太坊客户端 (本地节点) 接收到交易时, 会先完成验证流程, 验证通过则将交易插入本地交易池. 每笔交易都带有一个交易序号 (nonce), 用于界定同一账户的交易顺序: 若两笔交易来自同一账户, 序号为 n 的交易必须先上链, 序号为 n+1 的交易才能被处理. 交易插入本地交易池时会根据情况被放置到 Pending 池或 Queued 池: 若一笔交易 (nonce=n) 的前置交易 (nonce<=n-1) 已上链或已在 Pending 池中, 则称当前交易满足打包条件, 满足打包条件的交易进入 Pending 池,Pending 池的交易可被转发至其他节点; 若不满足打包条件, 交易进入 Queued 池进行本地缓冲. 当交易到达矿工节点 (以太坊 1.0) 或提议者节点 (以太坊 2.0) 后, 节点会选择 Pending 池部分交易打包进下一个区块, 随后将打包好的区块广播至整个以太坊网络.

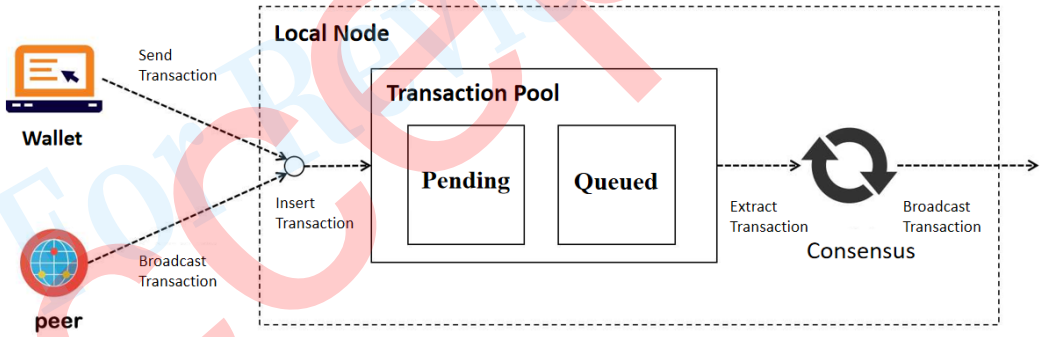


图 1 交易生命周期

Figure 1 Transaction lifecycle.

2.2 燃料费

以太坊引入“燃料”(gas) 的概念, 用于衡量执行一次交易所消耗的系统资源. 在固定的运行环境下, 单笔交易的 gas 消耗量是固定的. 燃料价格 (gas price) 指发送者为每单位 gas 支付的以太币数量 (通常以“Gwei”为单位). 矿工 (或提议者) 会优先选择 gas price 更高的交易打包以获得高收益. 燃料上限 (gas limit) 指交易或合约执行允许消耗的最大 gas 量. 若操作消耗的 gas 超过该上限, 交易将执行失败, 但已消耗的 gas 仍会被扣除. 交易总费用为 gas price 与实际消耗的 gas 量的乘积.

自伦敦升级中正式采用 EIP-1559 协议后, 交易的燃料费由两部分组成: 基础费和优先费. 基础费由网络拥堵程度动态决定, 拥堵越严重基础费越高, 且交易打包时基础费会被协议自动销毁, 因此被打包交易的燃料费必须高于当前网络基础费. 优先费指额外支付给矿工 (以太坊 2.0 中为验证者) 的费用, 用于提升交易被优先处理的概率. 理论上, 优先费越高, 交易被快速打包的可能性越大, 用户可通过提高优先费加速交易处理.

https://www.sciengine.com/doi/10.1360/SSI-2025-0419

2.3 限额配置

以太坊客户端的交易池支持自定义配置, 以实现公平高效的交易服务,Geth 与 Nethermind 两类主流客户端的核心配置如下.

Geth 客户端核心配置:

全局 Pending 额度 (GlobalSlots)——限制 Pending 交易的总数量, 每个客户端的交易池最多为所有账户保留 5120 笔 Pending 交易.

单账户 Pending 额度 (AccountSlots)——限制单个账户的 Pending 交易数量. 当需要驱逐 Pending 交易时, 会优先选择交易池中交易数量超过 16 笔的账户, 无论其交易的 gas price 高低.

全局 Queue 额度 (GlobalQueue)——限制 Queued 交易的总数量不超过 1024 笔. 超出时客户端会根据交易进入 Queued 池的时间和 gas price, 保留更可能被打包的交易, 实际剔除的是最先进入 Queued 池的交易.

单账户 Queue 额度 (AccountQueue)——限制单个账户的 Queued 交易数量. 当某账户的大量交易进入 Queued 池, 最多保留 64 笔 (节点本地账户直接提交的交易除外).

Nethermind 客户端核心配置:

默认情况下,Pending 交易和 Queued 交易的总量限制为 2048 笔; 账户级别的限制与 Geth 一致.

2.4 交易池分区与调度规则

交易池由两部分组成:Pending 池和 Queued 池.Pending 池用于缓冲 Pending 交易 (已准备好打包的交易);Queued 池用于缓冲 Queued 交易 (不可执行的非顺序交易). 交易池中的所有交易 (无论 Pending 还是 Queued) 均按燃料费从高到低排序, 燃料费越高, 优先级越高.

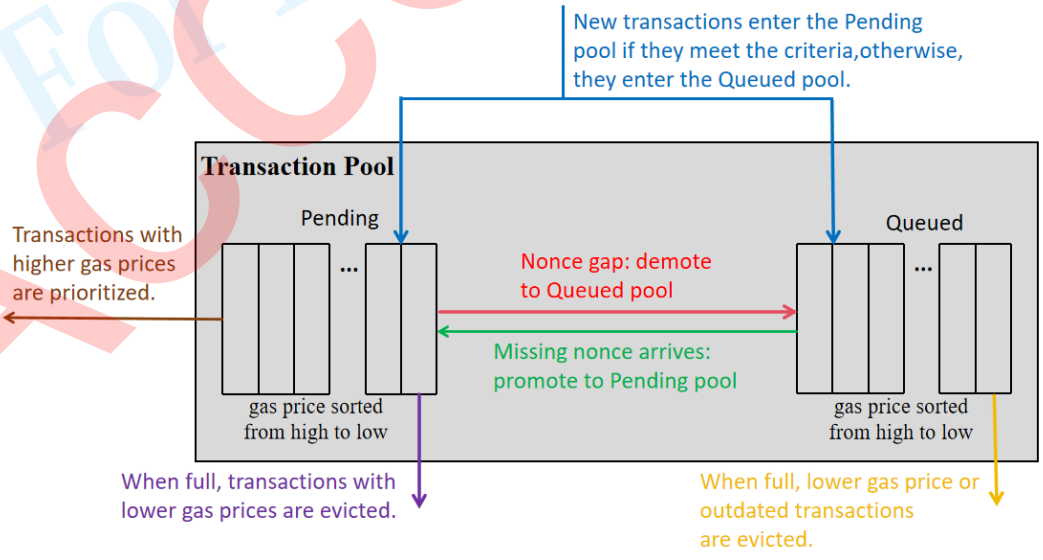


图 2 交易池分区与调度规则

Figure 2 Partition and scheduling rules of transaction pool.

当交易池满且池中没有账户的交易数量超过 16 笔 (AccountSlots 的额度) 时, 会驱逐优先级最低的交易, 为燃料费更高的新交易腾出空间. 若 Pending 交易的 nonce 出现断序 (例如前置交易未上链), 会被降级至 Queued 状态. 当缺失的 nonce 对应的交易到达后,Queued 状态的交易可被升级至 Pending 状态. 图2展示了交易池分区与调度规则.

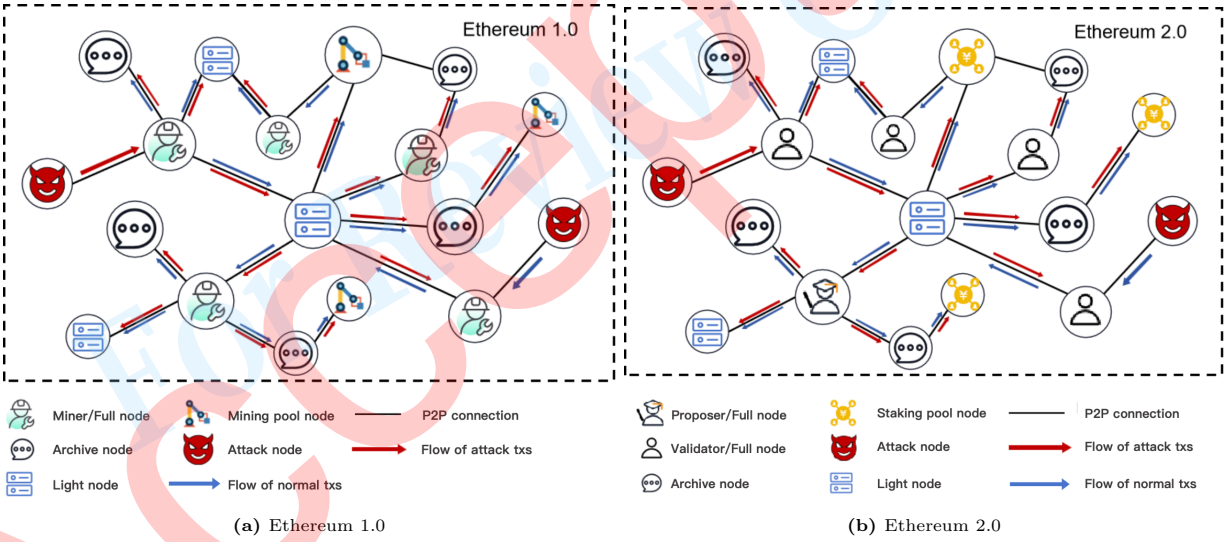
https://www.sciengine.com/doi/10.1360/SSI-2025-0419

3 基于幽灵交易的交易池拒绝服务攻击

以太坊交易池运行机制存在一处关键缺陷: 当交易池满时, 会优先驱逐燃料费最低的交易. 若被驱逐交易存在后序交易, 这些后序交易将变为非顺序交易, 需转移至 Queued 池. 若此类后序交易始终未等到前序交易重新进入交易池, 则需等待超时后才会被删除. 这类“无法打包、无法替换且占用 Queued 池资源”的交易称为“幽灵交易”. 当前以太坊的驱逐机制会催生此类具有威胁性的交易, 而本文提出的攻击方法正是基于幽灵交易设计.

3.1 威胁模型

本文的攻击目标系统为基于以太坊客户端 (1.0 或 2.0 版本) 的区块链网络, 其中存在大量使用以太坊 1.0 客户端构建的私有链或变体链, 这类网络均采用相同的交易池处理机制. 攻击者仅需接入目标区块链网络, 无需掌握网络底层拓扑结构或节点列表. 攻击者通过向网络插入交易发起攻击 (如图3所示), 最终实现以极少的攻击交易消耗, 填满所有客户端交易池 Queued 池并驱逐原先 Queued 池内所有交易的目标.



(a) Ethereum 1.0 **(b)** Ethereum 2.0

图 3 以太坊网络攻击示意图

Figure 3 Attack sketch in Ethereum network.

3.2 幽灵交易序列

定义 1 (幽灵交易序列) 若一组交易序列满足 $T_A(n), T_A(n+1), ..., T_A(n+x-1), T_B(n+x)..., T_B(n+m-1)$ 的形式, 则将其称为幽灵交易序列. 所有交易均由同一账户地址发送,n、$n+1$、...、$n+m-1$ 是交易的 nonce 号,T_A 代表低 gas price 交易 (简称 A 类交易),T_B 代表高 gas price 交易 (简称 B 类交易).

在上述幽灵交易序列中,A 类交易数量为 x,B 类交易数量为 m-x. 为自动生成幽灵交易序列, 需先定义低燃料价格与高燃料价格的阈值. 本文通过 QuickNode 提供的 RPC 服务 [8]收集了以太坊主网交易数据 (2023 年 10 月 24 小时内的约 7000 个区块), 并在 Etherscan 中查询对应交易的燃料费, 结果如图4所示. 横轴为交易燃料价格的自然对数; 纵轴为累积概率值; 基础费用指区块中交易的最低燃料价格. 数据显示, 基础费用最小值为

8) https://www.quicknode.com

https://www.sciengine.com/doi/10.1360/SSI-2025-0419

4.38Gwei,90% 区块的最高燃料价格不超过 293Gwei. 据此, 本文将低燃料价格阈值设定为 4Gwei, 高燃料价格阈值设定为 300Gwei.

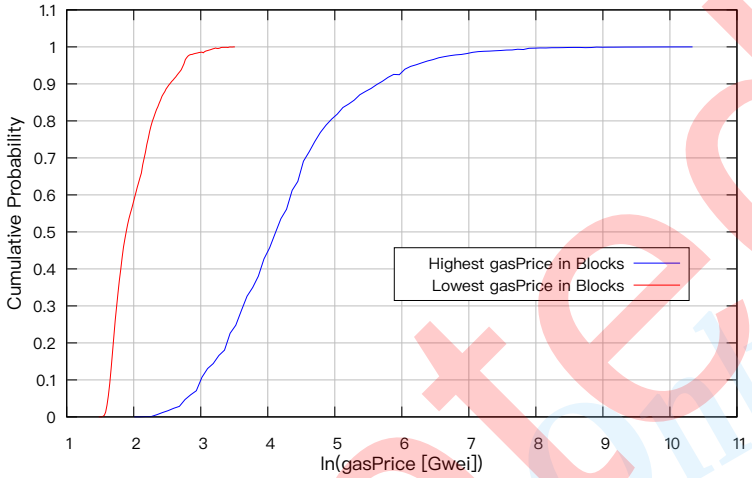


图 4 基于 7,000 个连续区块的最高与最低燃料价格累积分布图. 横轴为燃料价格的自然对数 ln(gasPrice [Gwei]), 纵轴表示小于等于该值的概率. 红色曲线代表每个区块中最低燃料价格的分布, 蓝色曲线代表最高燃料价格的分布. 例如, 在 $x = 2$ 处（对应实际价格 $e^2 \approx 7.39$Gwei）, 最低燃料价格的累积概率约为 0.6, 表明约 60% 的区块其最低燃料价格不超过 7.39Gwei; 而此时最高燃料价格的累积概率接近 0, 说明几乎无区块的最高燃料价格低于此值.

Figure 4 Cumulative distribution map of highest and lowest gas prices based on 7000 consecutive blocks. The x-axis shows the natural logarithm of gas price, ln(gasPrice [Gwei]), and the y-axis represents the cumulative probability. The red curve depicts the distribution of minimum gas prices per block, while the blue curve shows that of maximum gas prices. For instance, at $x = 2$ (corresponding to an actual price of $e^2 \approx 7.39$ Gwei), the cumulative probability for the minimum gas price is approximately 0.6, indicating that about 60% of blocks contain at least one transaction with gas price no higher than 7.39 Gwei. In contrast, the cumulative probability for the maximum gas price at this point is nearly zero, meaning almost no block has a maximum gas price lower than this value.

3.3 两阶段攻击

对以太坊网络发起攻击时, 仅需通过 RPC 服务将生成的幽灵交易序列注入网络即可. 幽灵交易序列会在目标以太坊客户端间自动广播, 感染幽灵交易的客户端会出现交易池溢出的情况, 进而实现攻击交易进入并驱逐 Queued 池内所有其他交易. 整个攻击过程可分为两个阶段.

广播阶段: 在此阶段, 幽灵交易序列在目标客户端间传播. 若以太坊客户端的交易池有足够空间, 则幽灵交易序列会顺利通过该客户端; 若空间不足 (例如剩余空间仅能容纳 P 笔交易), 则到达的幽灵交易会触发驱逐阶段, 且仅有前 P 笔交易能传递至下一个节点.

驱逐阶段: 当交易池满时, 会触发驱逐机制. 由于交易池始终优先驱逐 gas price 最低的交易, 因此低燃料费的 A 类交易会被优先驱逐. 当某笔 A 类交易被驱逐后, 其所有后续交易将变为非顺序交易, 进而转移至 Queued 池, 占据 Queued 池内空间. 随着攻击的持续, 最终实现 B 类交易对 Queued 池内原有交易的驱逐.

3.4 运行示例

下文来通过一个示例来演示攻击过程. 如图5所示, 存在两个对等的以太坊客户端 C_1 与 C_2, 二者交易池均包含 3 个 Pending 槽位与 2 个 Queued 槽位. 其中 C_1 交易池为空,C_2 有 1 个 Pending 槽位已被占用. 攻击时, 通过两个账户向 C_1 注入 2 个幽灵交易序列, 分别为 $T_A(0)T_B(1)T_B(2)$ 与 $T'_A(0)T'_B(1)T'_B(2)$.

https://www.sciengine.com/doi/10.1360/SSI-2025-0419

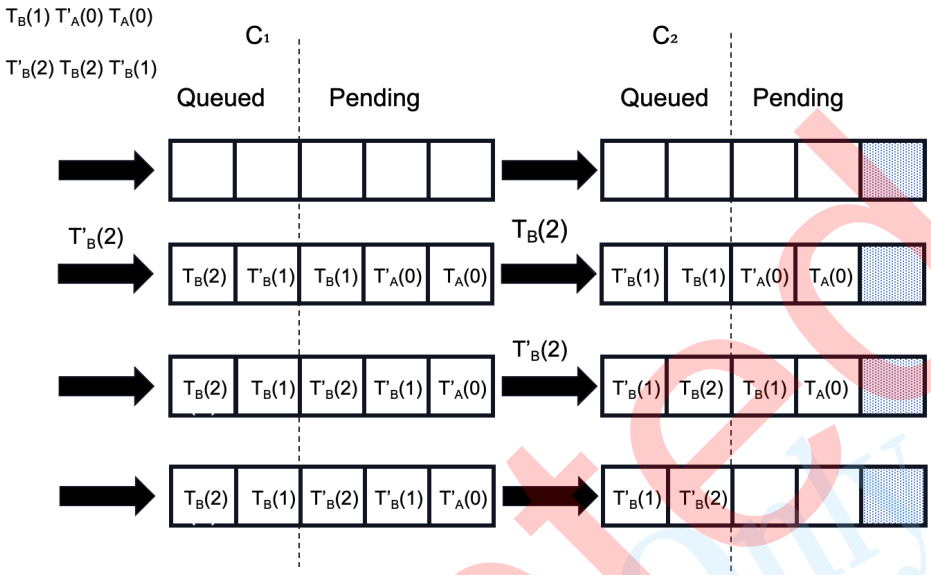


图 5 幽灵交易攻击的运行示例

Figure 5 Running example of ghost transaction attack.

当 C_1 接收到前 5 笔交易 (即 $T_A(0)$ $T'_A(0)$ $T_B(1)$ $T'_B(1)$ $T_B(2)$) 时, 会先缓冲存储, 再将其依次转发至 C_2;C_2 有空闲槽位容纳 $T_A(0)$ $T'_A(0)$ $T_B(1)$ $T'_B(1)$, 并对这些交易进行缓冲. 随后两客户端均进入驱逐阶段: 对 C_1 而言, 接收 $T'_B(2)$ 时, 因 $T_A(0)$ 燃料费最低而将其驱逐,$T_B(1)$、$T_B(2)$ 变为非顺序交易, 需转移至 Queued 池; 对 C_2 而言,$T_B(2)$ 到达后驱逐 $T'_A(0)$, 导致 $T'_B(1)$ 转移至 Queued 池. 最终, 当 $T'_B(2)$ 经 C_1 传递至 C_2, 交易池出现溢出并驱逐最低燃料费的交易 $T_A(0)$, 导致其后续交易变为不可执行交易, 节点更倾向于保留更可能被打包的交易,$T_B(1)$、$T_B(2)$ 被驱逐.

由于 $T_A(0)$ 的重发由攻击者控制 (实际攻击中不会重发),$T_B(1)$、$T_B(2)$ 将永远无法再次进入 Pending 池. 交易序列 $T'_A(0)$、$T'_B(1)$、$T'_B(2)$ 最终也无法执行, 因为交易池中存在其他正常交易, 可能触发新一轮驱逐.

3.5 跨客户端一致性

为了验证本文提出的攻击模型在异构网络环境下的普适性, 我们对当前以太坊网络中占据主导地位的三种客户端实现——Geth (Go-Ethereum)、Nethermind 和 Hyperledger Besu——进行了系统性剖析, 如表 1 所示. 据 Ethernodes 平台统计数据显示, 这三类客户端合计占据了全网节点总数的约 86%, 对比分析揭示了它们在核心交易处理逻辑上的高度一致性.

4 攻击策略研究

为探索适用于以太坊主网的攻击策略, 本文将幽灵交易注入以太坊私有链网络, 通过监控私有链网络中各节点交易池状态, 分析不同攻击策略的有效性.

4.1 实验设置

为观察以太坊各节点的运行状态, 本文在网络靶场中开展实验. 网络靶场是用于网络安全训练与技术研发的虚拟环境, 它提供增强网络基础设施与 IT 系统稳定性、安全性和性能的工具, 便于构建大规模区块链实验.

https://www.sciengine.com/doi/10.1360/SSI-2025-0419

本实验使用的网络靶场由企业 [9]开发, 该企业专注于网络靶场的全栈建设.

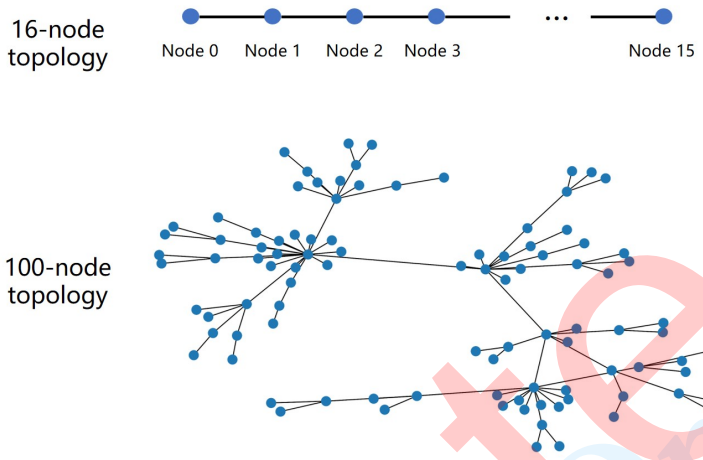


图 6 对等节点网络拓扑结构

Figure 6 Peer-to-peer network topology.

实验场景包含两种拓扑结构, 分别是 16 节点拓扑与 100 节点拓扑, 如图6所示. 每个场景均以其中一种拓扑作为点对点 (Point To Point, P2P) 覆盖网络, 构建包含攻击者、背景交易生成器和矿工的以太坊测试网[10].16 节点拓扑为线性结构;100 节点拓扑基于 B-A 模型 [4] 生成. 背景交易生成器每秒发送 1 笔背景交易; 攻击者每秒发送 4 笔幽灵交易. 每个以太坊节点虚拟机配置如下: 矿工节点配置为 8 核 CPU、80G 磁盘、16G 内存; 背景交易节点配置为 4 核 CPU、20G 磁盘、6G 内存; 其他节点配置为 1 核 CPU、20G 磁盘、2G 内存. 区块链使用的以太坊客户端版本为 1.11.0-unstable, 工作量证明 (Proof of Work, PoW) 客户端 (以太坊 1.0). 选择以太坊 1.0 是因为其与 2.0 采用相同的交易池运行机制, 且 1.0 版本更适合构建 100 节点规模的实验环境

9) https://www.range-soft.com

10) 在实验过程中, 我们的相关的代码及环境设置可见:https://github.com/cones-x/Ghost-Transaction-Attack

表 1 主流以太坊客户端交易池机制对比

Table 1 Comparison of transaction pool mechanisms across mainstream Ethereum clients

	Geth (Go-Ethereum)	**Nethermind**	**Hyperledger Besu**
Transaction pool capacity	Fixed slot limits: Pending and Queued pools are separately bounded.	Elastic total memory limit: more flexible than Geth but still bounded.	Unified capacity limits: all transactions share a single limit while executability is distinguished.
Replacement & Eviction	Replacement requires a 10% price bump in gasPrice; lowest-priced transactions are evicted when full.	Same as Go-Ethereum.	Same as Go-Ethereum.
Nonce gap handling	Immediate demotion: higher-nonce transactions are demoted to Queued and excluded from packing.	Transaction isolation: transactions are held in a Future structure until the nonce gap is resolved.	Out-of-order caching: non-contiguous transactions are buffered until preceding ones are confirmed.
Queued pool overflow	Account-based eviction: remove highest-nonce transaction from the most slot-consuming account.	Global price-based eviction: remove lowest gasPrice transactions across the pool when reached global limit.	Global price-based eviction: remove lowest gasPrice transactions to admit higher ones when full.

Accepted Only

https://www.sciengine.com/doi/10.1360/SSI-2025-0419

(第 6 章将讨论该方法在以太坊 2.0, 即 Sepolia 测试网中的有效性). 客户端配置参数如下:AccountSlots=64、GlobalSlots=240、AccountQueue=16、GlobalQueue=32. 该配置参数远小于以太坊主网配置, 目的是加速攻击过程, 并为更大的区块链网络实验节省系统资源.

网络延迟是评估攻击效果的关键因素, 严重的网络延迟会导致传播路径上的目标节点处于不同攻击阶段. 例如, 在 16 节点拓扑中 (图6), 从节点 0 发起攻击时, 前几个节点 (节点 1、节点 2 等) 会先进入驱逐阶段, 由于网络延迟, 幽灵交易可能无法同时到达节点 15. 但是, 当某节点开始驱逐某一账户发送的交易时, 该账户的所有交易都会变为非顺序交易, 存储于 Queued 池无法转发, 导致攻击范围难以覆盖节点 15. 为使实验更贴近真实场景, 本文部署了一个以太坊主网归档节点, 测量其 47 个对等节点的往返时延 (RTT). 测量结果如图7, 数据显示, 链路延迟中位数为 292ms, 且大部分数据集中于此区间. 因此, 实验将网络靶场的网络延迟设定为 150ms.

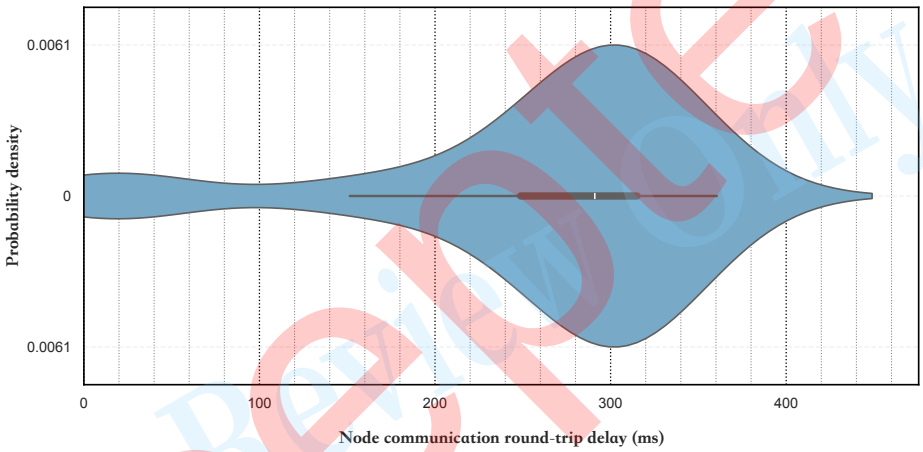


图 7 客户端对等节点的链路延迟. 白色中心代表链路延迟的中位数, 粗黑线代表四分位距 (IQR), 细黑线代表 1.5 倍 IQR.
Figure 7 The link latency of client peers. The white central dot represents the median, the thick black bar represents the interquartile range (IQR), and the thin black whiskers extend to 1.5×IQR.

为了更好模拟以太坊网络的真实环境, 我们在实验中设置了背景流量. 我们随机选择节点执行交易发送, 并依据私链出块速度严格限制了背景流量交易速率（略高于区块打包速度）, 确保在没有攻击流量的情况下, 交易池中始终有待处理交易留存. 此外, 我们统计了本地全节点 48 小时内进入交易池交易（2023/11/20-2023/11/21)的 gasPrice 范围, 并用泊松分布对 gasPrice 数据进行了拟合, 如图8所示, 图中代表了不同 gasPrice 对应的概率质量函数. 该分布的横坐标涵盖了大部分交易的价格区间约 10 至 55Gwei, 纵坐标表示特定 gasPrice 出现的理论概率, 该泊松分布呈现出良好的对称性并向两侧逐渐衰减. 最终, 我们根据泊松分布结果设置了实验中的背景流量的 gasPrice 和概率.

4.2 幽灵交易的传播过程

本节首先分析幽灵交易的传播过程细节. 网络靶场实验采用 16 节点拓扑结构, 其中节点 0 为攻击节点. 模拟开始时, 所有节点的交易池均为空; 背景交易生成器启动并生成正常交易; 节点 0 在 40 秒后发送幽灵交易. 矿工节点每秒处理约 4 笔交易 (每区块包含 50 笔交易, 出块时间为 12-15 秒), 此设置可使交易池被快速填满. 攻击节点通过同一账户每 2 秒发送 8 笔幽灵交易, 交易类型按 A 类、B 类交替排列.

图9展示了幽灵交易传播过程中节点 0、节点 1、节点 2 与节点 3 的交易池中交易数量变化. 对节点 0 而言, 其交易池在 68 秒时被填满, 共包含 104 笔幽灵交易; 随后节点 0 开始剔除交易, 至 78 秒时仅剩余 16 笔幽灵交易 (本地生成的幽灵交易开始在 Queued 池累积). 由于仍存在交易数量超过 16 笔的账户, 这 16 笔幽灵交易在 Pending 池中保留至 88 秒 (此时所有账户的交易数量均少于 16 笔);88 秒时, 交易池开始驱逐低燃料费

https://www.sciengine.com/doi/10.1360/SSI-2025-0419

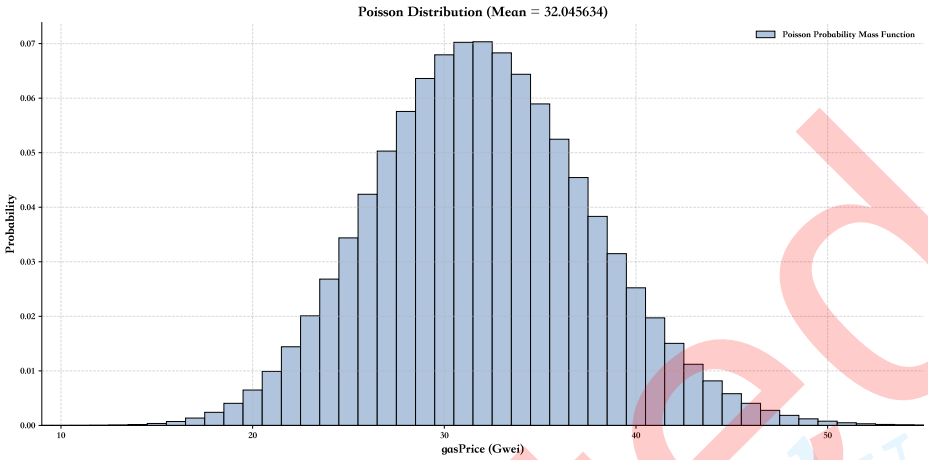


图 8　48 小时进入全节点的交易 gasPrice 分布概率

Figure 8　The probability distribution of transaction gasPrice entering the full node within 48 hours.

的交易 (即 8 笔 A 类交易), 导致 8 笔 B 类交易变为非顺序交易转移至 Queued 池. 节点 1、节点 2 与节点 3 呈现相似变化; 三者的交易池均在 80 秒时被填满, 随后开始删除交易, 至 100 秒时所有账户的交易数量均少于 16 笔; 此后驱逐机制触发,8 笔 B 类幽灵交易全部进入 Queued 池. 由该实验可得出以下结论:1) 转发至节点 1、节点 2 与节点 3 的所有幽灵交易均在 68 秒前完成发送, 即交易序列一旦断裂, 同一账户的幽灵交易便无法继续传播;2) 仅前 16 笔幽灵交易可用于攻击目标节点;3) 单一账户难以填满 Queued 池, 最终单个账户仅能占用 Queued 交易的 8 个槽位 (具体数量取决于幽灵交易序列的构造方式);4) 尽管存在 0.15 秒的网络延迟, 节点 1、节点 2 与节点 3 的实验情况仍高度相似, 且 15 个目标节点均呈现类似特征, 表明本文提出的攻击方法对网络延迟不敏感.

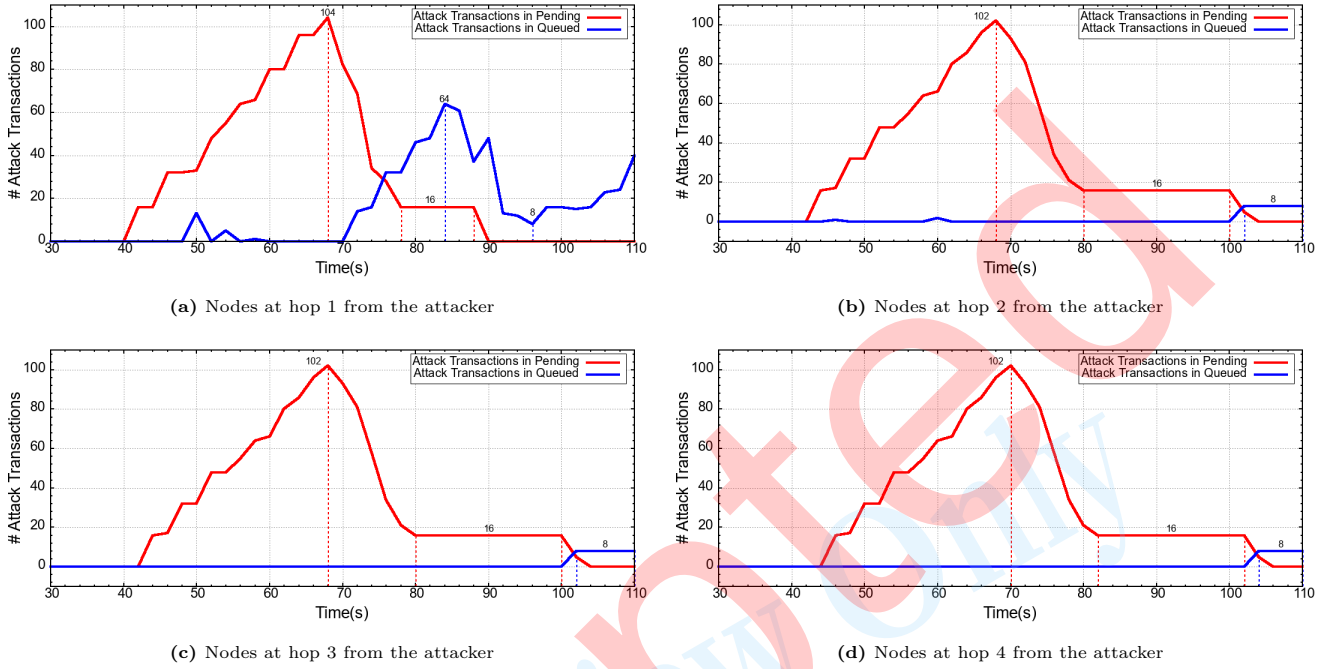
4.3　基于多账户的攻击

由上述实验可知, 可以通过多账户发送幽灵交易序列实施攻击, 每个账户发送 16 笔交易,B 类交易将以“幽灵”形式滞留于交易池的 Queued 池. 但攻击账户存在准备成本与风险 (账户资金可能被追踪, 账户本身也可能被封禁), 因此攻击者倾向于减少攻击账户数量. 为提升幽灵交易的攻击效果, 应包含尽可能多的 B 类交易, 理想的幽灵交易结构为“1 笔 A 类交易 +15 笔 B 类交易”(记为“1-15”结构).

多账户幽灵交易序列的编排方式也可能影响攻击效果. 若一个幽灵交易序列触发另一幽灵交易序列的驱逐, 导致两个序列的 B 类交易均无法完全进入目标节点的 Queued 池. 安全的发送方式是逐个发送不同账户的幽灵交易序列, 但这会消耗更多时间 (实验中用于注入交易的 RPC 服务会限制交易发送速率). 不过, 若目标节点交易池仍有大量空闲槽位, 并发发送各账户交易序列也不会产生明显差异; 仅当交易池空闲槽位少于 16 时, 编排方式才会产生影响, 但这种情况难以预测. 因此, 本研究暂不考虑不同账户间幽灵交易序列的编排问题.

我们开展了与前述实验类似的多账户攻击实验, 图10展示了节点 2 的 Queued 池内幽灵交易数量随时间的变化.5 个攻击账户:52 秒内占用 Queued 池的 78 个槽位;9 个攻击账户:46 秒内占用 Queued 池的所有 128 个槽位;13 个攻击账户:36 秒内几乎填满 Queued 池. 通常情况下, 至少需要 9 个账户才能占据整个 Queued 池. 攻击账户数量越多, 占用的 Queued 池槽位越多, 目标节点进入驱逐阶段的速度越快.

https://www.sciengine.com/doi/10.1360/SSI-2025-0419



(a) Nodes at hop 1 from the attacker

(b) Nodes at hop 2 from the attacker

(c) Nodes at hop 3 from the attacker

(d) Nodes at hop 4 from the attacker

图 9 幽灵交易在节点中的数量变化过程

Figure 9 The variation of the number of ghost transactions at nodes.

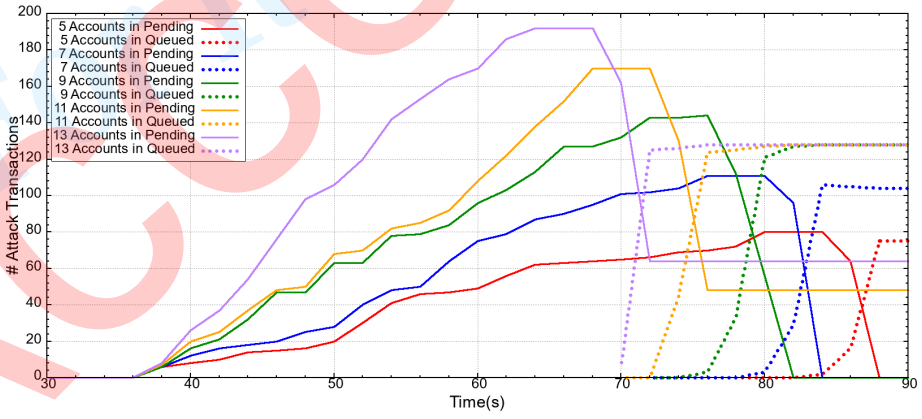


图 10 节点 2 上 Queued 池幽灵交易数量随时间的变化

Figure 10 Queued ghost transactions on node 2 over time.

4.4 短交易池引发的攻击截断

如前所述, 两个不同账户的幽灵交易序列可能因交易池空闲槽位少于 16 而相互干扰, 导致 B 类交易传播被截断, 因此有必要研究短交易池对传播路径的影响.

与第一个实验类似, 我们将节点 6 的 Pending 池大小缩减至 320, 并将攻击者的交易生成速率降至每秒 1 笔, 图11展示了交易池中的幽灵交易数量变化. 攻击发起后等待数秒, 节点 6 的 Pending 池仅剩余 80 个空闲槽位.50 秒时,4 笔幽灵交易进入节点 6 的 Pending 池, 并转发至节点 7;74 秒时, 节点 6 触发驱逐机制,3 笔幽灵交易被驱逐至 Queued 池. 由于节点 5 和节点 7 的 Pending 池较大, 没有进入驱逐阶段, 它们的 Queued 池为空. 大部分发起的幽灵交易被节点 6 截断, 因为幽灵交易需要先进入节点 6 的 Pending 池, 才能传递至节点 7.

https://www.sciengine.com/doi/10.1360/SSI-2025-0419

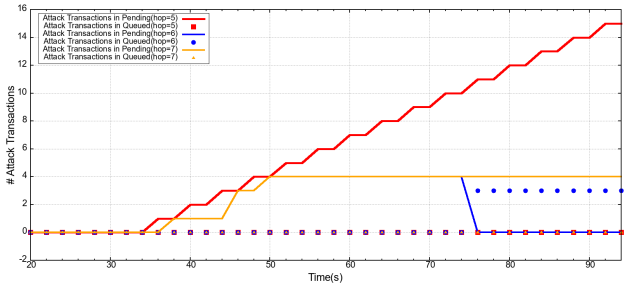


图 11 短交易池导致高 nonce 攻击交易被截断. 实线表示各跳攻击交易在节点 6 的 Pending 池中的数量, 对应标记（方块、圆点、三角）为其在 Queued 池中的数量. 当交易池空间不足时, 新到达的高 nonce 交易被移入 Queued 池而停止传播, 造成攻击链在该节点中断.

Figure 11 Truncation of high-nonce attack transactions due to a limited transaction pool. Solid lines show the number of hop-5/6/7 ghost transactions in node 6' s pending pool; corresponding markers indicate their count in the queued pool. When the transaction pool fills up, incoming high-nonce transactions are evicted to the queued pool and cease propagation, breaking the attack chain at this node.

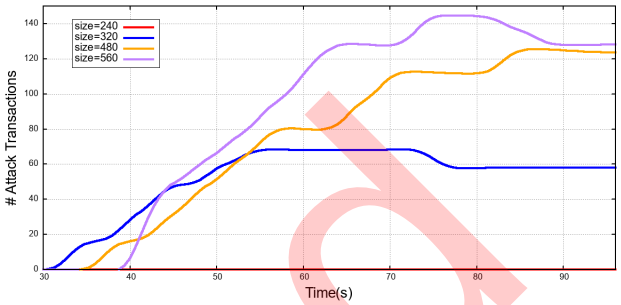


图 12 不同交易池容量对攻击传播的影响. 四条曲线分别表示节点 6 的 Pending 池允许通过 240、320、480 和 560 笔交易时, 节点 10 接收到的幽灵交易数量. 随着池容量增大, 更多攻击交易得以进入并转发, 最终到达下游节点的数量显著提升; 当容量达到 560 时, 传播趋于饱和.

Figure 12 Impact of transaction pool size on attack propagation. The four curves show the number of ghost transactions received by node 10 when node 6's pending pool allows 240, 320, 480, and 560 transactions respectively. Larger pool sizes enable more transactions to pass through, leading to higher reception at downstream nodes; saturation is observed at 560.

为研究交易池大小的影响, 我们将节点 6 的 Pending 池大小分别设置为 240、320、480、560, 观察节点 10 交易池中 Pending 池与 Queued 池的幽灵交易数量. 如图12所示, 如果将节点 6 的交易池大小视为 “阀门”, “阀门” 开启越大, 另一侧的节点 (如节点 10) 接收的幽灵交易也越多.

4.5 基于多节点的协同攻击

短交易池可能截断幽灵交易的传播, 进而限制攻击范围. 若从以太坊网络的多个入口发送幽灵交易 (指使用同一账户发送, 减少攻击账户数量), 攻击效果会如何? 为分析多节点协同攻击的有效性, 我们在节点 0 和节点 15 使用同一账户同时发起攻击, 并重复实验 20 次. 图13展示了两侧节点幽灵交易的平均占比, 结果显示攻击在 20 秒内结束.

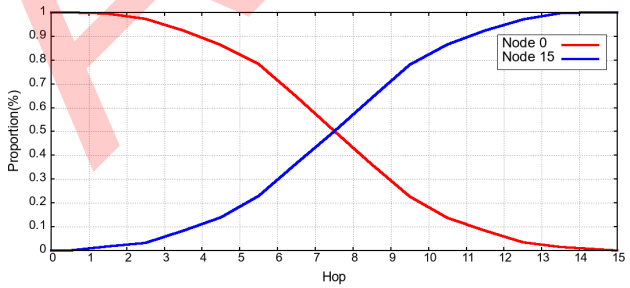


图 13 多节点协同攻击

Figure 13 Multi-node based coordinated attack

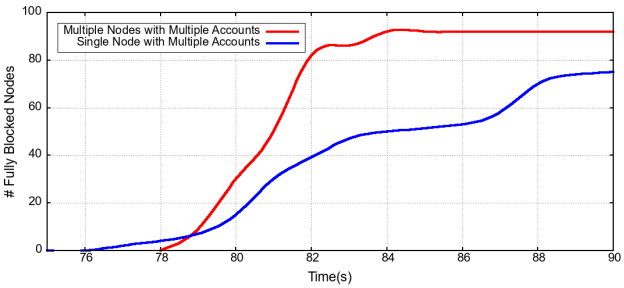


图 14 100 节点场景中阻塞节点情况

Figure 14 Blocked nodes in our 100-node scenario.

节点 0 和节点 15 的结果呈现对称性, 表明即使两侧幽灵交易相遇, 仍有机会传播至另一侧. 例如, 节点 4 可能先从更近的节点 0 接收幽灵交易, 后从节点 15 接收同一账户的幽灵交易. 此类幽灵交易可能引发 “双重

https://www.sciengine.com/doi/10.1360/SSI-2025-0419

驱逐”，因为后到达的交易可能填补缺失的顺序交易，将非顺序交易重新拉回 Pending 池；被拉回的非顺序交易要么再次被驱逐，要么在交易池满时被直接删除. 幸运的是，另一侧的幽灵交易占比不高，且当中间存在短交易池时，该占比会更低.

综上，本节探究了影响攻击效果的多项因素，包括网络延迟、攻击账户数量、攻击节点数量、交易序列编排方式及交易池大小. 实验结果表明：网络延迟与交易序列编排方式影响较小；增加攻击账户与攻击节点数量可加速攻击；交易池较小会限制攻击范围.

4.6 100 节点测试评估

为评估攻击方法的整体有效性，我们在包含 10 个短交易池节点的 100 节点拓扑中开展实验，其中 2 个节点 Pending 池大小为 240、2 个为 320、3 个为 480、3 个为 560. 实验设置两个场景:1) 单个攻击节点,9 个攻击账户;2) 两个攻击节点，每个攻击节点 9 个攻击账户.

图14展示了实验中阻塞节点的数量. 单节点攻击场景中,86 秒内阻塞以太坊测试网中 75 个节点的 Queued 池；多节点攻击场景中,90 秒内阻塞 91 个节点的 Queued 池. 两类场景中首个被阻塞的节点出现在攻击发起后的 76 秒与 78 秒，说明攻击发起后会在全网范围内快速扩散. 实验结果也再次验证了前文结论：多节点协同攻击可加快攻击进程.

4.7 幽灵攻击成本分析

幽灵攻击通过控制多个账户向区块链网络中发送“幽灵交易”，以完成对网络中节点交易池的拒绝服务攻击. 为了完成对交易池的有效攻击，幽灵攻击与其他的分布式拒绝服务（DDoS）攻击一样，均需要发送大量交易. 根据幽灵交易的可执行性、广播机制及不同交易序列的特点，每一笔交易均需要对应的成本来完成发送. 参考当前市场数据，当 ETH 价格 3000 美元/ETH，交易 gasPrice 为 5Gwei 时，一笔普通的转账交易（gas=21,000）的手续费约为 0.315 美元. 因此，当我们发送幽灵交易进行攻击时（假设 A 类交易为 2Gwei,B 类交易为 50Gwei），则每笔 A 类交易需要 0.126 美元,B 类交易需要 3.15 美元. 如果这些幽灵交易均最终上链，那么幽灵攻击的成本是每一个攻击者都难以接受的.

为了解决上述的成本问题，我们设置了一个资金回收交易 t. 对每个账户而言，交易 t 的 nonce 值等于该账户的第一笔幽灵交易的 nonce 值,gasPrice 值不小于当前网络状态下的 BaseFee 的 2 倍（为了能够尽快被打包上链），完成一笔交易的同 nonce 替换. 根据以太坊中交易的替换规则，当两笔交易 t_0 和 t_1 先后进入交易池，且具有相同的发送账户和 nonce 值，则比较两者的 gasPrice，若 t_1 交易 gasPrice 超过了 1.1 倍的 t_0 交易 gasPrice，则 t_1 交易将取代 t_0 交易. 在资金回收过程中，当交易 t 优先其他幽灵交易执行时，幽灵交易将会因为账户余额不足而执行失败，实现对幽灵交易成本的控制.

为了直观进行分析，我们假定一个主网 BaseFee 为 20Gwei 的实验环境. 攻击者控制了 $N = 100$ 个攻击账户，每个账户的攻击载荷被设计为 10 个交易组序列. 每一组包含 1 笔 2Gwei 的 A 类交易和 15 笔 50Gwei 的 B 类交易，即每个账户共计发送 160 笔幽灵交易，全网总计 16000 笔攻击交易. 在不采用资金回收机制的情况下，若所有交易均成功上链，我们可以获得单账户的攻击成本. 由于每个账户发送 10 组该序列，总共拥有 100 个账户的攻击网络，总攻击成本将高达:

$$Cost_{total} = (1 \times 0.126 + 15 \times 3.15) \times 10 \times 100 = 47376. \tag{1}$$

这一成本规模（近 5 万美元）对于单纯的 DDoS 干扰攻击而言过于高昂，在采用资金回收机制的情况下，每个账户仅需上链一笔资金回收交易 t. 在 BaseFee 为 20Gwei 的环境下，我们假定交易 t 的 gasPrice 为 60Gwei(BaseFee 的 3 倍). 此时，交易 t 将优先上链并将账户资金归集. 一旦交易 t 执行成功，该账户余额归零，

https://www.sciengine.com/doi/10.1360/SSI-2025-0419

后续的 159 笔幽灵交易（包括所有的高价 B 类交易）虽已进入交易池, 但将在执行阶段因“余额不足”而失败, 无需支付 gas 费用. 此时, 攻击者的实际支出仅为 100 笔交易 t 的手续费:

$$Cost_{actual} = 100 \times (21000 \times 60 \times 10^{-9} \times 3000) = 378. \tag{2}$$

通过对比分析, 在复杂的“A+15B”多组序列攻击场景下, 资金回收机制将攻击成本从 47376 美元降低至 378 美元, 成本缩减幅度高达 99.2%. 这进一步证实了, 通过精心的 nonce 排序与资金回收策略, 攻击者能够以极低成本维持高强度的攻击.

攻击成本的敏感性分析 根据上述讨论, 采用资金回收机制时, 攻击者的实际成本满足

$$Cost_{actual} \propto (N \cdot gasPrice \cdot ETH) \tag{3}$$

,N 为账户数量. 使用传统 DDoS 攻击方案而不采用资金回收机制时, 所有攻击交易均会占用资金并最终上链, 成本为

$$Cost_{naive} \propto (M \cdot gasPrice \cdot ETH) \tag{4}$$

,M 为攻击交易总数, 且通常满足 $M \gg N$. 因此, 两种策略的成本比为

$$\frac{Cost_{actual}}{Cost_{naive}} \propto \left(\frac{N}{M}\right). \tag{5}$$

由于攻击过程中仅需少量交易完成资金回收, 而需要大量交易用于填充交易池, 通常有 $N \ll M$, 从而得到 $Cost_{actual} \ll Cost_{naive}$, 表明攻击成本的主导因素由“交易规模”转变为“账户规模”, 实现数量级下降. 攻击成本对市场变化满足

$$Cost_{actual} \propto (gasPrice \cdot ETH) \tag{6}$$

, 说明成本仅随 gasPrice 和 ETH 价格线性变化, 不存在超线性放大. 攻击者可通过择时执行以近似最小化实际支出.

4.8 缓解方案

为缓解幽灵攻击对节点交易池造成的负面影响, 本文提出一种改进的交易剔除策略. 幽灵攻击的核心在于利用交易池在 Queued 区域溢出时所采用的替换机制, 当前主流实现通常依据交易的存活时间和 gasPrice 等因素决定剔除优先级. 然而, 这种机制忽视了交易发送者账户层面的分布不均问题, 使得攻击者可通过大量提交低 gasPrice 且 nonce 连续的交易, 占据 Queued 区域并挤占合法用户的交易池空间.

为此, 我们建议在交易剔除过程中引入 nonce 和账户维度的公平性考量: 当 Queued 区域达到容量上限时, 应优先从交易数量最多的账户中移除其 nonce 值最高的交易, 并重复此过程, 直至所有账户在 Queued 中保留的交易数量趋于均衡. 该策略不仅有助于抑制单一账户对交易池资源的过度占用, 还能有效削弱幽灵攻击的可行性, 同时保障普通用户的交易提交机会. 值得注意的是, 尽管主流以太坊客户端（如 Geth）已通过参数 `LegacyPool.Config.AccountQueue`（默认值为 16）限制每个账户在 Queued 区域中可容纳的交易数量, 但该阈值在实际攻击场景中仍可能过高. 攻击者仅需控制少量账户即可填满全局交易池, 从而对网络可用性构成威胁. 因此, 单纯依赖静态的账户级配额并不足以抵御此类攻击, 而动态均衡剔除机制可作为对现有策略的有效补充.

基于上述机制, 若攻击者仍试图实施幽灵攻击, 则需使用大量独立账户以规避账户级配额限制, 会导致资源成本急剧上升. 此外, 本文也对其他潜在的防御思路进行了探讨.

Accepted For Review Only

https://www.sciengine.com/doi/10.1360/SSI-2025-0419

动态自适应配额机制: 现有的 AccountQueue 阈值多为静态配置（如 Geth 默认的 16），这为攻击者提供了明确的资源边界. 引入动态阈值机制, 使其随交易池全局负载（Global Utilization）实时波动, 可以在检测到大面积低价交易涌入时自动收缩单账户配额. 这种流量压力感知防御能显著压缩攻击窗口, 迫使攻击者必须持有比预期多出数倍的账户才能维持攻击强度.

交易信誉机制: 引入信誉机制可从身份维度对交易进行过滤. 通过量化账户的历史成交频率、gas 消耗密度以及是否存在频繁被剔除的不良记录, 为进入交易池的每个账户建立信誉评分. 在资源竞争剧烈时, 节点优先保护高信誉账户的交易, 而将低信誉账户置于更易被剔除的优先级.

综合来看, 动态配额实现简单且不依赖历史数据; 而信誉机制防御精准度更高但需要额外的存储与计算开销. 这些多维度协同防御机制是未来提升区块链系统抵抗 DDoS 攻击能力的关键方向.

5 测试网真实攻击

为进一步验证方法有效性, 本文在已升级至以太坊 2.0 版本的 Sepolia 测试网中发起实际攻击（本次 Sepolia 测试网实验并未对真实用户造成经济损失）. 由于无法完全控制 Sepolia, 攻击需通过 RPC 服务发起 (如图15所示). 我们使用了 15 个来自不同网络接入点的 RPC 服务, 其中 12 个用于发起攻击,3 个用于监控攻击结果.

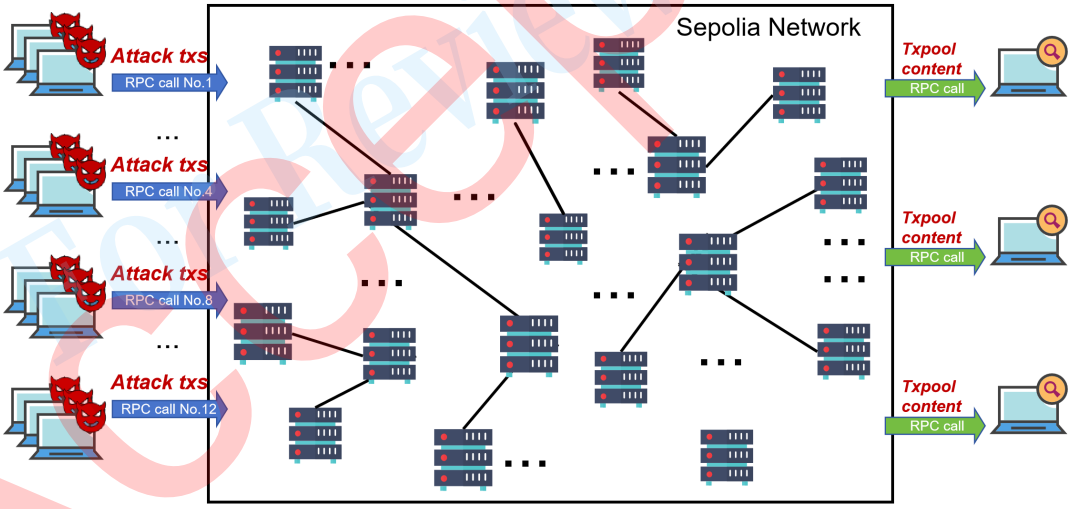


图 15 针对 Sepolia 测试网的攻击示意图

Figure 15 Schematic of our attack on Sepolia.

实验于 2024 年 1 月 10 日 15:34-16:22 期间进行, 总攻击时长 48 分钟, 我们用 300 个攻击账户发送 “8-8” 结构的幽灵交易序列 (8 笔 A 类交易 +8 笔 B 类交易). 幽灵交易通过脚本自动生成, 为便于识别, 所有交易均设置为 0ETH 的账户自转账类型. 结合攻击期间 Sepolia 测试网的燃料费标准, 将 A 类交易的燃料费设为 5Gwei,B 类交易的燃料费设为 40Gwei.

通过在 QuickNode 上创建的监控 RPC 服务[11)]观察到的结果如图16所示.

首先分析 Queued 池状态, 初始时 Queued 池包含 441 笔交易,Pending 池包含 12134 笔交易. 在 t_0(15:34) 时, 攻击开始; 在 t_1(16:22) 时, 攻击结束; 在 t_2(16:27) 时,Queued 池交易首次达到较高比例, 为 96%;t_3(16:48), Queued 池包含 642 笔交易, 其中 624 笔为攻击交易, 占比达 97.19%(最大值);t_4(19:09) 时,Queued 池中攻击交易占比大幅下降至 40%,Queued 池共 451 笔交易, 其中 182 笔为攻击交易;t_5(19:26),Queued 池中攻击交易

11) https://summer-bold-putty.ethereum-sepolia.quiknode.probc6234/

https://www.sciengine.com/doi/10.1360/SSI-2025-0419

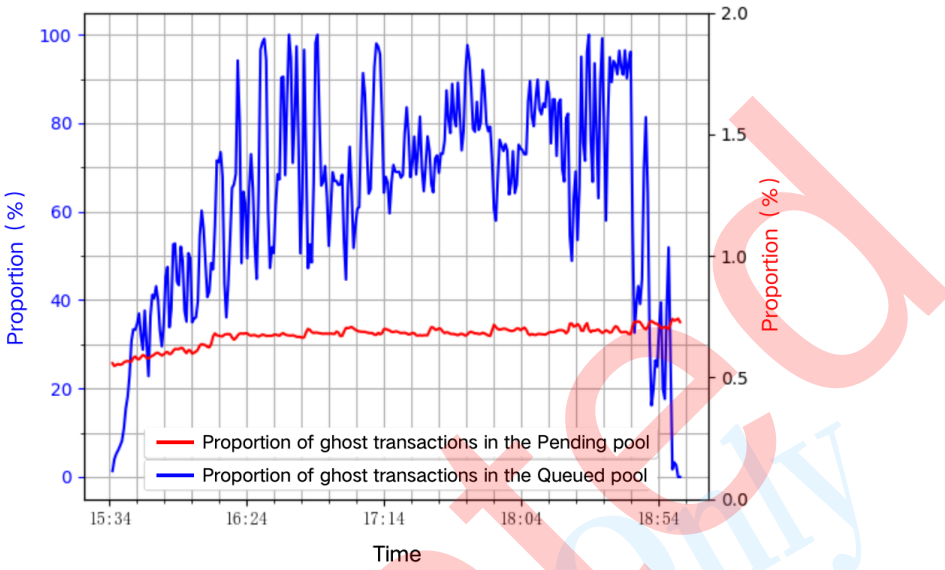


图 16 通过 QuickNode 集群监控到的测试网节点交易池中幽灵交易占比变化. 红色曲线表示 Pending 池中攻击交易的占比（稳定在 20%-40%）, 蓝色曲线为 Queued 池中的占比.

Figure 16 Variation in the proportion of ghost transactions observed via QuickNode' s cluster in testnet transaction pools. The red curve shows the share of attack transactions in the Pending pool (consistently 20%–40%), while the blue curve represents that in the Queued pool.

占比降至最低,Queued 池共 461 笔交易, 仅 17 笔为攻击交易, 占比 3.68%;t_6(19:27), 交易池中攻击交易完全清空;t_6(19:27)-t_7(22:20), 持续观察至 22:20, 交易池中攻击交易占比始终为 0. 综上, 所有攻击交易在攻击发起约 4 小时后从 Queued 池被清除.

接下来分析 Pending 池状态.t_0-t_1 期间,Pending 池中攻击交易占比开始上升; 由于攻击交易不断被驱逐出 Pending 池,t_2 时 Pending 池共 12209 笔交易, 攻击交易占比仅 0.67164%;t_2-t_7 期间,Pending 池占比趋于稳定,B 类的交易数量缓慢增加,A 类的交易稳定在 71 笔 (具体变化可参考拟合图, 数据按分钟记录, 反映整体趋势).

6 Queued 池交易数据分析

幽灵交易能够劫持以太坊网络的 Queued 交易池, 使 Queued 池内非顺序交易被驱逐; 但对于具备打包条件的顺序交易, 攻击无影响. 因此, 需进一步分析该攻击方法对当前以太坊应用生态系统的影响.

我们从 QuickNode 集群节点收集了 2023 年 8 月 29 日 19:26 至 8 月 30 日 20:20 期间的 Queued 池交易, 共 15376 笔. 其中,633 笔交易在进入 Queued 池后重新返回 Pending 池, 并最终被打包上链; 这 633 笔交易中,531 笔与顺序交易相关, 占比约 83.89%. 对于 Queued 池中剩余未上链的交易, 我们认为其重要性较低 (因从未出现在以太坊主网), 因此重点分析最终上链的 531 笔交易, 并将其归类为以下 6 种场景, 图17展示了幽灵交易攻击的影响分布.

6.1 场景 1: 授权-交易模式

该场景包含 57 笔交易, 涉及 50 个账户, 覆盖转账前授权、跨链转账、铸币前授权等多种情况. 以转账前授权为例, 当交易发起方通过 UniswapV2 转账代币时, 需先向 UniswapV2 授予代币使用权限, 再执行转账操作.

https://www.sciengine.com/doi/10.1360/SSI-2025-0419

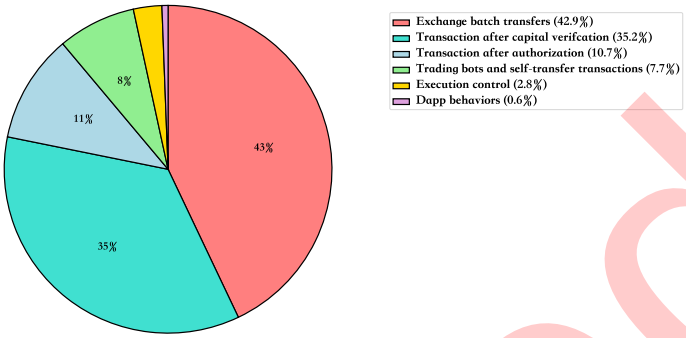


图 17 幽灵交易攻击影响分布

Figure 17 Distribution of the impact of ghost transaction attacks.

由于未授权的转账交易会失败, 且同一账户的交易需按顺序执行, 发起方必须确保转账交易的 nonce 大于授权交易的 nonce. 用户倾向于让关联交易同步广播并打包, 同时受网络不稳定性、手动发起交易导致的广播延迟等因素影响, 部分发起方会选择先发起转账交易, 待转账交易到达相邻节点后, 再发送授权交易. 此时, 相邻节点会因授权交易 (前置交易) 缺失, 将转账交易放入 Queued 池; 当授权交易到达后, 节点会将授权交易与 Queued 池中的转账交易一同转移至 Pending 池并广播, 最终确保两笔交易均上链. 例如, 在 UniswapV2 中观察到的 Queued 交易 0xe20b5af318d17c21ea6b9a4c8a661c5cda9dc435138045c45345e8404ec5ee36, 用于将 HIPVP 代币兑换为以太币, 但该交易执行需以另一笔授权交易 0xe5d9030b35337b83dc59cef514f07fd607ef98c43441
06e8d71a7cb1cc48fc33 完成为前提. 实验中可观察到代币兑换交易处于 Queued 池, 但未检测到授权交易. 当发起攻击时, 该代币兑换交易将从交易池中被移除, 进而破坏转账前授权的正常操作流程.

6.2 场景 2: 资金验证模式

该场景包含 187 笔交易, 涉及 38 个账户. 在这类交易过程中, 用户常需按顺序执行多笔交易, 例如先存款后交易、先到账后提现、交易手续费“免单”服务等. 这类交易通常先发送后续交易, 待收到转账后再补充发起前置交易, 以确保整套交易能被成功打包上链. 以交易手续费“免单”服务为例, 交易发起方先发送一笔“退还超额手续费”的后续交易, 等待费用到账; 当收到其他用户转入的手续费, 便补充发起前置转账交易. 此时, 相邻节点会将新收到的前置交易与 Queued 池中的后续交易一同广播, 确保两笔交易均上链. 例如, 交易 0x616258bf5cb116bf1791a948e9200182236b069deff3027376341041f
8953bc1 为一笔手续费退还交易, 其前置交易为稳定币转账. 当发起方收到作为手续费的转账后, 会补充前置交易来完成整个操作. 实验在 Queued 池中捕获到了这笔手续费退还交易; 发起攻击后, 该费用退还交易将被删除, 不仅会导致操作流程无法完成, 还会损害账户信誉.

6.3 场景 3:Dapp 行为

该场景虽只包含 3 笔交易, 涉及 2 个账户, 但其对相关操作的影响程度很大. 目前, 各类交易所推出自研应用管理用户账户, 并根据用户需求提供一系列服务. 为提升用户体验, 交易所钱包设计了一键服务功能, 将多个交易步骤整合, 用户只需发起简单操作, 后端自动完成后续流程. 这类钱包操作中, 顺序操作相对普遍, 如用户在 Layer2 交易前, 需先与 Layer2 交互, 再执行 Layer2 交易. 交易 0x1e298432cba9d1b7493eff47abde3a2da780b099
222e75e1378f71edffa98dc3 的前置交易设置了极低的燃料费, 确保前置交易能在全网广播, 致使该交易长期滞留于交易池中. 当账户收到交易所转账后, 使用转账资金发起一笔前置交易, 替换原低燃料费的前置交易, 完成

https://www.sciengine.com/doi/10.1360/SSI-2025-0419

与 Layer2 的交互. 攻击发起后, 该账户在交易替换前发送的交易将从交易池中被移除, 导致后续操作中断.

6.4 场景 4: 执行控制

该场景包含 15 笔交易, 涉及 8 个账户. 依据交易打包上链规则, 交易需按顺序执行, 且燃料费高的交易享有高优先级, 同时低燃料费的交易不会被主动删除. 为实现批量交易打包上链, 发起方常采用先发送一个低 nonce、低燃料费的交易, 再发送一系列正常燃料费交易的策略. 相邻节点收到这一系列交易后, 会按规则转发, 初始的低燃料费交易导致后续交易滞留在交易池中. 随后, 交易发起方会发送一笔高燃料费交易, 替换首笔低燃料费交易, 只需一次广播即可确保整个交易序列具备打包条件, 被打包进同一个区块. 例如, 交易 0x2ee1a9ec694691df9f44794e6489f72fa30a73ca4e52e704b13ffdd
05e0e4b8d 的前置交易初始燃料费设为 0.25Gwei 和 0.275Gwei, 导致后续交易滞留在交易池中. 此外, 部分节点会对交易池中的交易设置最低燃料费限制, 因此这些节点会丢弃原 0.25Gwei 的交易, 将后续交易放入 Queued 池. 攻击发起后, 交易发起方原本发送的所有交易均会丢失, 仅最后发送的交易能上链.

6.5 场景 5: 交易所批量操作

该场景包含 228 笔交易, 涉及 5 个账户. 交易所的大部分交易都是通过脚本执行批量操作, 包括利用 nonce 控制交易时机和节省成本 (通过量化计算采用最低可行燃料费) 的方式. 利用 nonce 控制交易时机的方式与场景 4 类似, 但为确保批量交易能被打包进同一个区块, 交易所设置的燃料费相对较高. 节省成本的方式有时会导致交易燃料费低于部分节点设置的最低阈值, 引发交易丢失. 例如, 交易 0xdb834fa2f174977d774bffa21cd7c0c9
82c584c519deed6021bc03644f84b080 是 HitBTC 交易所发起的一笔转出交易, 因缺失前置交易而暂存于 Queued 池. 当交易所发送前置交易后, 该交易被打包上链. 攻击发起后, 交易所的大量后续交易将被删除, 阻碍交易所业务的正常执行.

6.6 场景 6: 自动化机器人

该场景包含 41 笔交易, 涉及 29 个账户. 空投机器人 (AirdropBots) 利用批量发起交易的优势申领 NFT 空投, 新机会出现时需执行序列化操作, 确保交易快速上链. 同理, 套利机器人 (Arbitrage Bots) 也会借助自身特性实现交易间的序列化执行. 零价值自转账交易主要用于填补缺失 nonce 的交易, 同时提升账户活跃度. 当账户交易次数较少但需参与对账户活跃度有要求的项目时, 账户自转账交易便会发挥作用.

上述场景表明,Queued 交易覆盖了各类区块链应用操作. 但可能存在质疑: 这类交易仅为本地生成, 攻击对其无影响. 为反驳这类质疑, 我们进一步从 publicnode(https://ethereum.publicnode.com)、ankr(https://rpc.ankr.com/eth)、llamarpc(https://eth.llamarpc.com) 收集同一观察期内的 Queued 池交易, 分别获得 6679 笔、6843 笔和 1280 笔. 在上述 633 笔重点分析的交易中, 有 368 笔交易可被两个及以上 RPC 服务提供商检测到, 此类交易显然并非本地生成.

7 相关工作

在区块链系统中, 网络层的通信基于 P2P 网络架构, 新区块的传播采用泛洪机制. 然而, 正是网络层基础设施与 P2P 网络架构本身存在的漏洞, 催生了各类网络安全问题.

学术界对 P2P 网络层攻击展开了广泛研究, 尤其聚焦于 PoW 共识机制场景. 攻击者可通过劫持边界网关协议前缀控制网络路径, 进而干扰区块链节点间的通信, 此类攻击主要包括分区攻击与延迟攻击 [5~7]. 在分区攻击中, 攻击者通过切断部分节点与全网的通信连接, 将区块链网络分割为多个孤立子网络, 导致各子网络产

https://www.sciengine.com/doi/10.1360/SSI-2025-0419

生不同的区块数据, 破坏网络一致性; 延迟攻击刻意延缓区块在特定节点间的传播速度, 使部分节点接收区块的时间显著滞后, 增加交易确认延迟或引发数据同步偏差. 双花攻击与女巫攻击同样可在网络层实施 [8,9]: 在双花攻击中, 攻击者利用网络层数据传播的时延差, 在不同节点集群中重复使用同一笔数字资产发起交易, 由于区块同步延迟, 部分节点可能误将伪造的重复交易打包上链, 造成资产“双重花费”; 女巫攻击是攻击者通过伪造大量虚假节点 (即“女巫节点”) 接入 P2P 网络, 借节点数量优势操纵区块投票, 干扰交易验证, 破坏网络的去中心化特性. 日食攻击是另一类典型攻击形式: 攻击者通过与目标区块链节点建立大量连接, 导致目标节点的连接管理器无法处理海量连接请求为其他对等节点提供服务, 从而引发 DoS 攻击 [10,11].

尽管针对上述攻击已提出诸多防御措施, 但 DoS 攻击及其变体在分布式网络中仍十分常见 [12,13]. 在比特币区块链中, 存在内存池拥堵攻击问题, 此类攻击会导致合法用户难以完成交易, 本质上属于分布式拒绝服务 (Distributed Denial of Service, DDoS) 攻击, 攻击目标涵盖矿池、去中心化交易所等 [13]. 区块链拒绝服务 (Blockchain DoS, BDoS) 攻击则专门针对 PoW 与基于激励的共识机制, 攻击者通过获取战略优势迫使矿工停止挖矿活动 [14].MemPurge [15] 针对交易池可执行交易的缓存进行攻击, 通过前序交易改变依赖状态, 使得后续 DDoS 交易失效. 此外, 相关研究也从不同维度探索攻击与防御的优化策略: 有研究提出, 可利用比特币与以太坊交易的最小均方值动态限制最大交易价值, 以此降低大额交易被攻击利用的风险 [16]; 还有研究将不同矿池间的竞争交互建模为一般随机博弈, 提出高效纳什学习算法以获取接近最优的攻击策略 [17].

为对抗区块链中的恶意行为, 现有研究从多维度展开防御探索: 通过识别账户的异常身份行为来定位恶意账户 [18~20]; 探索基于元数据的恶意行为检测方法, 借助交易元数据的异常变化识别恶意操作 [21~23]; 将机器学习技术应用于区块链系统, 用于检测和防范常见的恶意交易 [24,25]. 针对 DDoS 攻击, 研究人员提出了多种防御算法, 涵盖基于交易中继优化、挖矿费用、交易时长等不同技术路径 [12,26~28];SAFERAD [29] 通过建立严格的准入费用门槛和驱逐代价, 从数学上保证攻击者的成本必须高于某个下限, 从而杜绝所有利用“低成本驱逐高价值交易”的攻击; 部分研究引入区块链防火墙-ChainGuard, 通过限制非法节点接入、阻断异常访问, 从源头解决泛洪攻击问题 [30]; 另有研究聚焦网络中的恶意数据包, 基于时间阈值对其进行检测、丢弃与流量控制, 减少恶意数据对网络的影响 [20].

表 2 DETER、MemPurge 与幽灵交易攻击特征对比

Table 2 Comparison of attack characteristics among DETER, MemPurge, and Ghost transaction.

	DETER	MemPurge	Ghost Transactions
Attack sequence	Static discontinuity: leverages initial Future transaction with discrete nonces	Static continuity: maintains consecutive nonces to ensure executability	Dynamic evolution: initially continuous, then becomes discontinuous
Mechanism exploitation	Leverages buffer reservations in direct P2P nodes for occupancy.	Exploits price-priority eviction under memory saturation	Exploits the interaction flaw between out-of-order buffering and replacement
Attack target	Single node	Low-priced transactions in the Pending pool	Network-wide out-of-order transaction buffering service
Propagation	Weak: non-sequential transactions violate propagation rules and requires direct P2P connections	Strong: propagates via normal broadcast paths but transactions are prune to be included	Extreme: achieves network-wide spread via initial continuity
Attack cost	High: requires connection permissions and incurs real fees	Medium: requires sustained high-rate transaction injection	Negligible: recovers funds via replacement mechanisms after attack takes effect.

表2对比了幽灵交易攻击与 MemPurge [15] 和 DETER [3] 在攻击序列、机制利用等方面的差异. 幽灵交易

https://www.sciengine.com/doi/10.1360/SSI-2025-0419

攻击在传播能力和实施成本方面均优于 MemPurge 和 DETER.

8 结论及未来工作

本文提出了一种针对以太坊网络的可实施性大规模交易池拒绝服务攻击方法——“幽灵交易”. 该方法利用以太坊客户端不合理的交易池驱逐机制, 通过构造特定交易序列占据交易池的有效存储槽位. 为验证攻击效果, 我们将幽灵交易序列分别注入网络靶场部署的测试环境与真实的 Sepolia 测试网, 实验结果表明: 在以太坊 1.0 与 2.0 网络环境下, 该方法均能有效攻击交易池服务. 受攻击影响, 以太坊生态内转账、智能合约交互等各类应用的正常业务将陷入拒绝服务状态.

本研究证实, 当前以太坊网络架构在抵御此类恶意攻击时仍存在明显安全短板, 设计并部署针对性的交易池防护机制已成为亟待解决的关键问题, 这也将是我们后续的核心研究方向.

致谢 本文由国家自然科学基金项目 (批准号:62572138)、数字化学习技术集成与应用教育部工程研究中心项目 (批准号:1331005) 以及广东省自然科学基金项目 (批准号:2024B1515020105) 资助.

参考文献

1 Ekparinya P, Gramoli V, Jourjon G. Double-spending risk quantification in private, consortium and public ethereum blockchains. CoRR, 2018, abs/1805.05004
2 Ye C, Li G, Cai H, et al. Analysis of security in blockchain: Case study in 51%-attack detecting. In: Proceedings of 5th International Conference on Dependable Systems and Their Applications, DSA 2018, Dalian, China, September 22-23, 2018, 2018. 15–24
3 Li K, Wang Y, Tang Y. DETER: denial of ethereum txpool services. In: Proceedings of Kim Y, Kim J, Vigna G, et al., editors, CCS '21: 2021 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, Republic of Korea, November 15 - 19, 2021, 2021. 1645–1667
4 Barabási A L, Albert R. Emergence of scaling in random networks. science, 1999, 286: 509–512
5 Apostolaki M, Zohar A, Vanbever L. Hijacking bitcoin: Routing attacks on cryptocurrencies. In: Proceedings of 2017 IEEE symposium on security and privacy (SP), 2017. 375–392
6 Apostolaki M, Marti G, Müller J, et al. Sabre: Protecting bitcoin against routing attacks. In: Proceedings of Proceedings of the 26th Annual Network and Distributed System Security Symposium, 2019. 02A1
7 Sheikh M A, Khattak F, Khan G Z, et al. Secured land title transfer system in australia using vpn based blockchain network. In: Proceedings of 2022 24th International Conference on Advanced Communication Technology (ICACT), 2022. 125–131
8 Ekparinya P, Gramoli V, Jourjon G. Double-spending risk quantification in private, consortium and public ethereum blockchains. arXiv preprint arXiv:1805.05004, 2018
9 Eisenbarth J P, Cholez T, Perrin O. Ethereum' s peer-to-peer network monitoring and sybil attack prevention. Journal of Network and Systems Management, 2022, 30: 65
10 Henningsen S, Teunis D, Florian M, et al. Eclipsing ethereum peers with false friends. In: Proceedings of 2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), 2019. 300–309
11 Zheng H, Tran T, Arden O. Total eclipse of the enclave: detecting eclipse attacks from inside tees. In: Proceedings of 2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), 2021. 1–5
12 Jin-hao H, Yuzhen D, Liang X, et al. Reinforcement learning based cache scheduling against denial-of-service attacks in embedded systems. Computer Science, 2020, 47: 282–286 [黄锦灏, 丁钰真, 肖亮, 等. 一种基于强化学习的嵌入式系统抗拒绝服务攻击的缓存调度方案 [J]. 计算机科学,2020,47(07):282–286]
13 Raikwar M, Gligoroski D. Dos attacks on blockchain ecosystem. In: Proceedings of European conference on parallel processing, 2021. 230–242
14 Mirkin M, Ji Y, Pang J, et al. Bdos: Blockchain denial-of-service. In: Proceedings of Proceedings of the 2020 ACM SIGSAC conference on Computer and Communications Security, 2020. 601–619
15 Yaish A, Qin K, Zhou L, et al. Speculative {Denial-of-Service} attacks in ethereum. In: Proceedings of 33rd USENIX security symposium (USENIX Security 24), 2024. 3531–3548
16 Akilandeswari R, Malathi S. Design and implementation of controlling with preventing ddos attacks using bitcoin by ethereum block chain technology. Journal of Transportation Security, 2022, 15: 281–297
17 Wu S, Chen Y, Li M, et al. Survive and thrive: A stochastic game for ddos attacks in bitcoin mining pools. IEEE/ACM Transactions On Networking, 2020, 28: 874–887

https://www.sciengine.com/doi/10.1360/SSI-2025-0419

18 Pham T, Lee S. Anomaly detection in bitcoin network using unsupervised learning methods. arXiv preprint arXiv:1611.03941, 2016
19 Maesa D D F, Marino A, Ricci L. Detecting artificial behaviours in the bitcoin users graph. Online Social Networks and Media, 2017, 3: 63–74
20 Sundareswaran N, Sasirekha S. Packet filtering mechanism to defend against ddos attack in blockchain network. In: Proceedings of Evolutionary Computing and Mobile Sustainable Networks: Proceedings of ICECMSN 2021, pages 201–214. Springer Singapore, 2022
21 Signorini M, Pontecorvi M, Kanoun W, et al. Advise: anomaly detection tool for blockchain systems. In: Proceedings of 2018 IEEE World Congress on Services (SERVICES), 2018. 65–66
22 Signorini M, Pontecorvi M, Kanoun W, et al. Bad: A blockchain anomaly detection solution. IEEE Access, 2020, 8: 173481–173490
23 Il-Agure Z, Attallah B, Chang Y K. The semantics of anomalies in iot integrated blockchain network. In: Proceedings of 2019 Sixth HCT Information Technology Trends (ITT), 2019. 144–146
24 Bogner A. Seeing is understanding: anomaly detection in blockchains with visualized features. In: Proceedings of Proceedings of the 2017 ACM international joint conference on pervasive and ubiquitous computing and proceedings of the 2017 ACM international symposium on wearable computers, 2017. 5–8
25 Dey S. Securing majority-attack in blockchain using machine learning and algorithmic game theory: A proof of work. In: Proceedings of 2018 10th computer science and electronic engineering (CEEC), 2018. 7–10
26 Hai-Ting Z, Ming-Gang W, Feng-Ning L, et al. Ddos attack detection and mitigation based on asynchronous personalized federated learning. Chinese Journal of Computers, 2025, 48: 808–827 [朱海婷, 魏明岗, 刘丰宁, 等. 基于异步个性化联邦学习的 DDoS 攻击检测与缓解 [J]. 计算机学报,2025,48(04):808–827]
27 Hancheng G, Haiping H. A novel distributed large-scale traffic cleaning scheme against ddos attacks. Netinfo Security, 2025, 25: 78–87 [高汉成，黄海平. 对抗 DDoS 攻击的新型分布式大规模流量清洗方案 [J]. 信息网络安全，2025，25（1）：78–87]
28 Saad M, Njilla L, Kamhoua C, et al. Mempool optimization for defending against ddos attacks in pow-based blockchain systems. In: Proceedings of 2019 IEEE international conference on blockchain and cryptocurrency (ICBC), 2019. 285–292
29 Ding W, Tang Y, Wang Y. Asymmetric mempool dos security: Formal definitions and provable secure designs. In: Proceedings of 2025 IEEE Symposium on Security and Privacy (SP), 2025. 1584–1602
30 Steichen M, Hommes S, State R. Chainguard—a firewall for blockchain applications using sdn with openflow. In: Proceedings of 2017 Principles, Systems and Applications of IP Telecommunications (IPTComm), 2017. 1–8

Accepted
For Review Only

https://www.sciengine.com/doi/10.1360/SSI-2025-0419

Ghost transactions: a practical and large scale denial of Ethereum transaction pool service attack

Shen SU[1], Qunhong SUN[2,3], Chang WANG[1] & Xiaojie ZHANG[1*]

1. *Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou 510006, China*
2. *Department of New Networks, Peng Cheng Laboratory, Shenzhen 518055, China*
3. *Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China*

* Corresponding author. E-mail: xjzhang@gzhu.edu.cn

Abstract Disabling the transaction pool service would be a critical attack on the Ethereum network. In this paper, based on the source code investigation of the mainstream Ethereum clients, we propose a DoS attack method (Ghost Transactions) to disable the transaction pool service for non-sequential transaction buffering. Our ghost transaction method is to insert a sequence of low gas price transactions followed by a sequence of high gas price transactions into the Ethereum network by multiple accounts. The sequence of ghost transactions is able to propagate among victim clients, and stored in the transaction pool as "ghost" which can hardly be removed. With experiments conducted on the cyber range, we prove that we are able to disrupt the transaction pool service of a 100-node Ethereum network within two minutes, leading to the malicious expulsion of legitimate transactions. With the launched attack on the testnet of Sepolia, we prove we can disable the real world Ethereum network as expected. With analysis on the collected Ethereum mainnet non-sequential transactions, we summarize the affected dapp behaviors like tokens, blockchain wallet etc, which affect a non-trivial scale and cover a wide range of on-chain activities.

Keywords denial of service attack, memory pool, blockchain, transaction eviction mechanism, Ethereum client

Accepted
For Review Only

https://www.sciengine.com/doi/10.1360/SSI-2025-0419