Routing Attacks in Ethereum PoS: A Systematic Exploration

Constantine Doumanidis and Maria Apostolaki

Princeton University

Abstract. With the promise of greater decentralization and sustainability, Ethereum transitioned from a Proof-of-Work (PoW) to a Proof-of-Stake (PoS) consensus mechanism. The goal of this paper is to investigate the security of Ethereum's PoS system from an Internet routing perspective. To this end, this paper makes two contributions: First, we devise a novel framework for inferring the distribution of validators on the Internet with minimal disturbance to the real network. Second, we introduce a class of network-level attacks on Ethereum's PoS system that jointly exploit Internet routing vulnerabilities with the protocol's reward and penalty mechanisms. We describe two representative attacks: StakeBleed, where the attacker triggers an inactivity leak, halting block finality and causing financial losses for all validators; and KnockBlock, where the attacker increases her expected MEV gains by preventing targeted blocks from being included in the chain. We find that both attacks are practical and effective. An attacker executing StakeBleed can inflict losses of almost 300 ETH in just 2 hours by hijacking as few as 30 IP prefixes. An attacker implementing KnockBlock could increase their MEV expected gains by 44.5% while hijacking a single prefix for less than 2 minutes. Finally, we discuss countermeasures practical for validator operators, consensus and P2P protocol designers.

1 Introduction

Ethereum and many other blockchain systems rely on Proof-of-Stake (PoS) consensus to maintain a globally consistent ledger. Validators propose and attest to blocks via a peer-to-peer (P2P) network, but because this communication traverses the public Internet, it remains exposed to routing attacks such as BGP hijacking which allows adversaries to redirect Internet traffic through their own infrastructure.

Practicality: BGP hijacking remains a practical threat; despite the growing adoption of RPKI [1], thousands of hijacks [2–5] are observed each month that even affect IP prefixes protected by RPKI [6]. Critically, as we discuss in this paper, 40.85% of Ethereum nodes are not (properly) protected by RPKI. Beyond the lack of a comprehensive prevention mechanism, two factors make BGP hijacks a particularly practical attack. First, resolving hijacks after they have been detected is a slow manual process, taking at least 2-3 hours even for expert teams [6, 7]. Second, legal repercussions are rare for the attacker, as hijacks are indistinguishable from misconfigurations, which are common [7, 8]. Recent incidents targeting the Celer Bridge [9] and MyEtherWallet [10] further underscore the practicality of BGP hijacking

in general and against blockchains. These attacks lasted hours, resulted in multi-million dollar gains despite RPKI, and incurred no consequences for the attackers.

Investigating the effect of BGP hijacking on Ethereum PoS requires contributions in two key directions: *(i)* understanding how PoS consensus properties such as liveness and safety can be threatened by such network vulnerabilities; and *(ii)* inferring Ethereum's network-level topology to assess how key contributors of the protocol are distributed and exposed to such attacks.

Attacking Ethereum PoS with BGP hijacks: We introduce two representative attacks against Ethereum's Proof-of-Stake (PoS) protocol: StakeBleed and KnockBlock. In both cases, the attacker controls a BGP-speaking router on the Internet—either through a malicious Internet Service Provider (ISP) or by compromising the infrastructure of a legitimate one (as in [11, 12]) or by setting up their own Autonomous System (AS) [13]. The two attacks differ in their target selection, level of stealth, and potential impact.

In StakeBleed a network adversary seeks to isolate a set of validators from the rest of the network, preventing them from contributing to consensus. In the worst case, StakeBleed can stop block finality, leaving the chain in an uncertain state where transactions may be reverted and critical applications stall. Even when the attack is not fully successful, it incurs financial losses at both isolated and non-isolated validators.

In KnockBlock the network attacker aims to prevent a specific block proposer from broadcasting its block. If the attacker controls the next block proposer, she directly benefits from the attack, as the missed block creates additional MEV opportunities and increases the attestation rewards for that block. This attack is highly stealthy, requiring the hijack to be effective for only a few seconds.[1]

Novelty: While prior work has leveraged network-layer vulnerabilities—particularly BGP hijacks [14–17]—their impact has largely been confined to PoW-based systems. To our knowledge, this is the first work to exploit Ethereum's PoS-specific characteristics, enabling both direct financial gain and inflicting financial damage —whereas prior PoW attacks focused primarily on reputational harm [17–19]. By contrast, recent attacks on PoS Ethereum [20–24] do not exploit Internet routing vulnerabilities.

Inferring network-level topology of validators: We design a multi-stage algorithm to infer the network-layer locations of Ethereum validators, enabling a realistic assessment of Ethereum's susceptibility to network-layer attacks and helping operators understand what behaviors may inadvertently reveal their validators. Our algorithm combines a scalable collector (SuperNode) with a multi-stage inference pipeline: (i) SuperNode passively accepts inbound connections, maintaining them just long enough to determine whether the peer hosts a validator. It *selectively* logs the timing of various types of attestations —along with auxiliary signals such as eagerly pushed or unsolicited messages—to reliably infer validator identities while minimizing resource overhead. (ii) High-confidence mappings are bootstrapped from validators that are activated on the blockchain in sequence and co-located, reflecting a common deployment pattern by a single operator. (iii) These mappings seed a machine learning model that generalizes timing patterns to the broader validator set, allowing

[1] Approximately 2 minutes when also accounting for BGP advertisement propagation time.

us to map it in its entirety. This is possible because, while sequential activation and co-location of validators is not a uniform pattern (*e.g.,* not all validators are activated at the same time, and even those that are might be distributed across locations), the propagation pattern of these nodes is representative of that of any validator node as it reflects protocol-level behavior rather than operator-specific idiosyncrasies. We validate our methodology by verifying that co-located validators are not common in non-consecutive validator IDs, and by observing that validators belonging to distinct Lido operators are often co-hosted with their operators' websites.

Novelty: Similar analyses exist for PoW-based Ethereum and Bitcoin, but their methodologies do not carry over to PoS. Prior research identified miners—similarly influential nodes—either through the websites of their public Stratum servers [19] or by analyzing block propagation delays using simple heuristics [18], neither of which translates to our setting. The difficulty of constructing a comprehensive validator-to-node mapping in PoS Ethereum is further underscored by recent work, which was only able to infer $\approx$15% of validators [25]. This work is the first to provide a full mapping.

Countermeasures: While securing the entire Internet is the ideal solution to such attacks, it is not within the capabilities of the Ethereum and broader blockchain ecosystem. Hence, we propose countermeasures at the deployment, consensus, and peer-to-peer layers, enabling validator operators, protocol designers, and P2P developers to respond within their respective domains.

Ethics No real Ethereum clients or IP prefixes were targeted in this study. We have disclosed our findings to the Ethereum Foundation. For details, refer to Appendix F.

2 Preliminaries

Ethereum's Proof-of-Stake (PoS) system requires validators to deposit 32 ETH and maintain high availability to perform time-sensitive duties that secure the network. In exchange, validators earn rewards proportional to their stake. Validators operate through a set of interconnected clients: the execution client handles transactions and smart contracts, the consensus client manages the PoS logic, and the validator client submits attestations and proposes blocks. Each node in the network is identified via an Ethereum Node Record (ENR), and node participation in the network is dynamic, with frequent churn—the process of nodes entering and exiting the network. At the protocol level, validators contribute to network security and liveness by issuing attestations (votes on block validity and network state) and occasionally proposing blocks that extend the blockchain. Validators are also assigned to committees, which determine which parts of the network they interact with and which messages they are expected to sign.

libp2p is a modular P2P networking stack used by Ethereum. It includes a pubsub system that allows nodes to subscribe to and publish messages in specific topics. Validators subscribe to topics based on their committee assignments, ensuring efficient and relevant message dissemination. Ethereum primarily uses Gossipsub [26], a gossip-based pubsub protocol that maintains a mesh of connections per topic. Gossipsub distinguishes between long-lived (mesh) and short-lived (fanout) peers, and uses heartbeats to maintain mesh health. It relies on several control messages: GRAFT and PRUNE ad-

just peer connections, while IHAVE and IWANT help peers exchange metadata about unseen messages, ensuring reliable propagation without overwhelming the network.

The Border Gateway Protocol (BGP) [27] is the Internet's de facto interdomain routing protocol, governing how IP packets are forwarded across Autonomous Systems. ASes exchange routes for IP prefixes, each of which is originated by a single AS and propagated through the network. Routers independently select next hops based on routes advertised by their neighbors. BGP does not verify route authenticity by default, allowing any AS to advertise false routing information. Such false advertisements, known as BGP hijacks, can misdirect traffic to unintended destinations and are a powerful mechanism for traffic interception attacks.

3 Overview

3.1 Threat model

Our threat model follows prior routing-adversary work [19, 28]. The attacker controls a BGP-speaking router—either by compromising [11, 12], acquiring, or setting up an AS [13, 29]. She performs BGP hijacks (more-specific/subprefix or same-length) targeting prefixes hosting Ethereum nodes. More-specific / forged-origin subprefix announcements succeed because routers prefer longest-prefix match; same-length hijacks bypass origin-based RPKI checks and can still divert a large fraction of traffic depending on location. We find 40.85% of Ethereum nodes are in prefixes lacking RPKI protection: 26.98% are vulnerable due to permissive maxLength settings and 13.87% lack ROAs and have short prefixes. Even RPKI-protected prefixes remain practically vulnerable unless downstream ISPs consistently filter forged advertisements—a responsibility often unmet in practice [6]. Same-length hijacks also remain effective, typically diverting on the order of 50% of a victim's connections on average [30, 31].

To reduce visibility, the attacker limits hijack duration and scope, with specifics depending on the attack goal. Critically, though, unless the attacker stops the hijack (as in [32]), it will be effective for at least 2-3 hours, as resolving it is a human-driven process [6, 7, 10, 33] by the network operator (ISP) rather than affected applications.

3.2 StakeBleed

In StakeBleed, the adversary aims to isolate a set of validators P from the rest of the network, preventing them from contributing to consensus. If P controls more than one-third of the total stake, the attack halts block finality, leaving the chain in an uncertain state in which transactions may be reverted, critical applications stall, and all validators incur significant financial losses from missed rewards, and penalties.

Description: The attacker first maps validator keys to IP addresses, e.g., using the algorithm in §4, and then launches BGP hijacks targeting the most specific prefixes of some of those IPs to divert traffic to a selected set of nodes P. To maintain stealth, she prioritizes prefixes with higher validator density, adding them to P until the 33% stake threshold is met. After initiating the hijack, the attacker monitors the chain

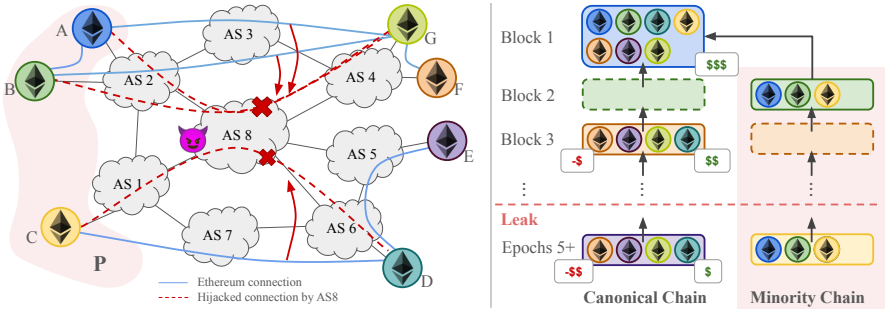


Fig. 1: **Network view (left)**: A StakeBleed adversary controlling a router in AS 8 isolates nodes in P, from the rest of the Ethereum network by diverting their traffic using a BGP hijack and dropping their connections to nodes outside P.
Blockchain view (right): Nodes in P have a minority chain view of the blockchain. Before the inactivity leak is triggered, validators in P lose proposer rewards and incur penalties because they do not produce attestations in the canonical chain, while validators outside P also lose some of their rewards from proposing and attesting. After 4 epochs of P being isolated, the inactivity leak is triggered, causing validators in P to incur inactivity penalties, while validators outside P stop receiving attestation rewards.

for attestations from targeted validators; if some appear due to mapping errors or ineffective hijacks, she expands P iteratively.

Once on-path, the attacker inspects Ethereum traffic (e.g., via known ports) and selectively drops packets to enforce the partition. Hijacks must persist for at least 4 epochs (128 slots / $\approx$26 minutes) to trigger an inactivity leak, a mechanism in Ethereum PoS for handling absent stake without compromising consistency or safety. During a leak, blocks are not finalized, and attestation rewards cease. The attack tolerates imperfect mappings or partial failures, as P can be dynamically adjusted. For a detailed walk-through of StakeBleed on Fig. 1, see §A.

Impact If $|P| > 33\%$, block finalization stops, putting the blockchain in a state of uncertainty where transactions—even those included in blocks—are not guaranteed to be irreversible. This undermines user trust, halts critical applications (e.g., DeFi protocols that rely on finalized state), and may trigger cascading effects like stalled bridges, paused staking services, and halted validator exits.

Furthermore, validators both in P and outside P are harmed financially. During the attack, validators in P miss all rewards, and during a leak accrue inactivity scores, leading to inactivity penalties that grow over time. For $|P| = 35\%$ and a two-hour attack, losses total 110.6 ETH in penalties and 61.8 ETH in missed rewards. Validators outside P are also harmed, as their attestations are delayed or omitted, and blocks include fewer attestations – for $|P| = 35\%$, this leads to an additional 114.8 ETH in aggregate losses. Even without reaching the inactivity leak threshold, the attack remains damaging: a two-hour hijack with $|P| = 20\%$ causes 214 ETH in total damages.

Practicality This attack is enabled by three key factors—two stemming from Ethereum's PoS design and one from its Internet-level deployment. First, an inactivity leak not only halts finality but also imposes financial penalties on both the

targeted and unaffected parts of the network, which could provide more incentives for the attacker. Second, validators are heavily concentrated in a small number of Internet-hosting locations (i.e., ASes), making it feasible for an attacker to isolate them by hijacking just a few IP prefixes. Indeed, we find that 33% of validators are hosted within only 29 prefixes, while 40.85% of Ethereum nodes are hosted in prefixes that are not protected by RPKI. Critically, BGP hijack attacks involving even thousands of prefixes still happen [5]. This centralization is not an artifact of our methodology; even applying simpler heuristics, similar to those developed for Bitcoin, yield similar numbers (see appendix D). Third, the leak persists regardless of which specific validators are absent. This flexibility allows the attacker to dynamically adjust the targeted validator set P, either to compensate for restored connectivity or to correct inaccuracies in IP-to-validator mappings (see §3.2).

3.3 KnockBlock

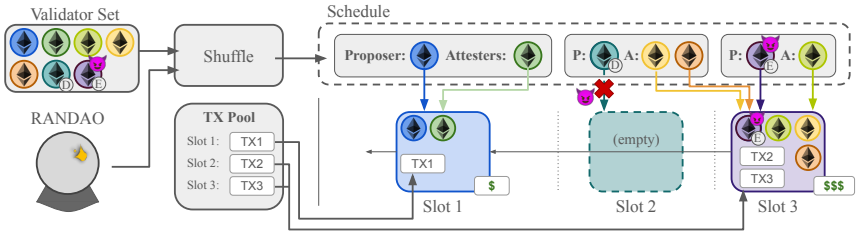


Fig. 2: A KnockBlock adversary controls the validator node E that is set to propose in Slot 3. The attacker calculates the proposer schedule, finds that validator D is scheduled to propose before her in Slot 2, and prevents node D from proposing by performing a BGP hijack. She then preferentially includes some transactions and attestations that were intended for Slot 2 in her own block, while denying D its proposer rewards.

In this attack, the adversary described in §3.1 seeks to isolate a single, carefully selected validator for a very short period, just enough to prevent it from proposing a block. The attacker owns one validator and seeks to benefit from the Maximal Extractable Value (MEV) expected to be contained in the block.

Description The attacker first calculates the proposer schedule. She identifies the validator scheduled to propose a block in the slot immediately preceding her own. Using the mapping obtained via the strategy outlined in §4, the attacker launches a BGP hijack targeting the validator's IP prefix. To increase her success probability the attacker might select the timing of her attack as well as the prefixes to hijack following strategies described in §7. Once on-path, she selectively drops all proposer connections for just one slot (12 seconds). Immediately after, she withdraws the BGP advertisement, restoring normal connectivity. Her own validator then proceeds to propose the next block as scheduled, naturally incorporating all unincluded attestations and potentially reusing some of the previously seen transactions to increase her MEV. For an example walk-through of KnockBlock on Fig. 2, see §B.

Impact The attack deterministically increases the revenue for the attacker's validator. We estimate that KnockBlock on average yields 44.5% higher MEV, and can also

lead to an increase in proposer rewards of up to 453.6%. In addition, the attacker deprives the hijacked proposer of its expected rewards. In §7, we show that targeting proposers yields the highest damage per unit of hijack duration.

Practicality Two factors make this attack practical. First, the attacker can reliably and promptly predict the proposer schedule up to 2 epochs in advance (12.8 minutes) under stable validator balances [34]. This prediction window not only provides sufficient time to initiate a targeted hijack, but also enables the attacker to strategically select a proposer whose suppression directly benefits her—specifically, the one scheduled to propose immediately before her own validator. Second, isolating a proposer requires only a brief, single-prefix BGP hijack. Since the proposer only needs to be unreachable for 12 seconds, the attacker's hijack can be short-lived and indistinguishable from routine Internet misconfigurations [35]. Further, since validators are required to operate from a single machine [36] (unlike miners who typically operate distributed infrastructures) the attacker needs to only hijack a single IP prefix, substantially reducing both complexity and risk of detection.

4 Identifying Ethereum Stakeholders

Inferring Ethereum's validator topology requires mapping validators to their nodes (and network locations). Prior methods for identifying PoW miners—based on block propagation timing [18, 37] or mining pool network information [19]—do not work for validators because validator messages are sent via Gossipsub, which introduces bursty, asynchronous message propagation. Furthermore, connecting *on-demand* to any node is not always possible, and pooled staking interactions happen on the blockchain level, rather than the network level. For more details on these challenges, see §C.

4.1 Collecting P2P information

To overcome these challenges, we design SuperNode, which passively accepts connections and logs validator attestations and auxiliary signals rather than blocks, since attestations are frequent and tied to specific validators. We explain the challenges this approach entails and the insights that make SuperNode achieve its goal.

SuperNode turns attestation timings into a reliable signal. Without being connected to all nodes simultaneously, our SuperNode cannot calculate relative timing advantage; that is, our node cannot determine whether one of its peers disseminated an attestation unusually early compared to all other nodes in the network (as it is not connected to all or enough of them). Worse yet, because Gossipsub does not force nodes to propagate the attestations they get immediately to all their peers (unlike mined blocks in PoW blockchains), our SuperNode can receive attestations with a substantial delay relative to when they were first known to the senders.

To address these challenges, we leverage the fact that each validator produces an attestation once per epoch (6.4 minutes) in a fixed 12-second slot. This allows SuperNode to establish relative latency (`rx_ts - slot_ts`) which provides a robust signal for validator identification. SuperNode avoids misleadingly late attestations by classifying incoming attestations and keeping only eagerly pushed ones. Eagerly pushed

attestations are those that have not been already advertised via an IHAVE message, and furthermore satisfy one of two conditions: (i) their sender is currently subscribed to the attestation topic and its connection with SuperNode is grafted (mesh), or (ii) they were sent over a non-grafted (fanout) connection and the sender was not subscribed to the topic. We refer to attestations of the latter case as *out-of-scope* (OOS). This method allows us to extract meaningful latency information for every validator for a given peer and compare it across nodes without requiring concurrent global connectivity.

SuperNode minimizes per-node connection duration without sacrificing mapping accuracy. Unlike PoW mining, where there were only a handful of sources to trace back (5-6 major pools), Ethereum has over one million validators. While longer SuperNode connections to peers would in theory give a better signal for validator identification, arbitrarily long connections are not feasible because they waste Ethereum network resources, count against the finite connections that SuperNode can make, and nodes experience significant network identity (IP, node IDs) churn.

We circumvent this using the observation that a node reveals its hosted validators when it eagerly propagates their attestations. Unlike mining pools, where blocks spread through multiple gateways, validators must disseminate their own attestations. Since SuperNode records only eagerly pushed messages, the hosting peer is expected to exhibit the lowest relative latency. Accordingly, SuperNode disconnects and blacklists the peer once it has observed from it at least one attestation for each validator, thereby *finalizing* it. To avoid suppressing attestations from lagging nodes, SuperNode never advertises attestations. This adaptive finalization greatly reduces overhead: 60% of nodes finalize within 41 hours, while some take much longer (up to 860 - see Fig. 7b). Without tailoring connection duration, achieving comparable accuracy would require over 20x more resources.

SuperNode waits for connections but aggressively grafts them. Instead of actively selecting peers, SuperNode passively accepts incoming connections, allowing it to capture nodes that are unreachable due to NATs or restrictive settings. Once a new connection is established, SuperNode grafts its connection to the peer on its subscribed topics to observe eagerly pushed attestations without biasing peer selection. To handle dynamic subscriptions and maximize attestation throughput, SuperNode applies a *graft-on-subscribe* strategy: immediately grafting any new topics for which the peer advertises its subscription. Finalized peers are blacklisted and disconnected.

4.2 Mapping network entities to validators

Having finalized all nodes, the goal is to map Ethereum's validator IDs to them.

SuperNode collects multiple in-network signals. To map nodes to validators, we leverage several signals, including relative latency, attestation counts, and out-of-scope (OOS) attestations statistics. Relative latency of received attestations is the strongest signal for three key reasons. First, validators originate attestations from a single node [36] - hence receiving an attestation through a relaying node will always be slower than learning about it from the source. Second, validators are incentivized to publish their attestations as quickly as possible to increase their chances of timely inclusion in a block, and thus acquire their full rewards. Finally, it is physically

impossible to artificially reduce network delay: Ethereum nodes cannot make attestations appear to have arrived any sooner than the sum of gossip communication and processing times. Attestation and OOS counts provide valuable auxiliary information, but are less reliable on their own, as not all nodes send OOS messages consistently, and counts depend on connection duration. In particular, OOS attestations should only originate from nodes hosting a validator (as observed by [25]), yet not all such nodes send them—either because they subscribe to all topics (common for nodes with many validators) or the SuperNode is outside the peer's fanout.

SuperNode leverages latency and consecutive validator IDs to map a subset of the validators Because a validator's host is expected to deliver its attestations with lower latency, SuperNode ranks peers by latency for each validator and selects the top 10 candidate IPs. Direct latency comparisons across candidates are unreliable, as measurements occurred at different times for different attestations. Instead, we make the following observation: some validators with consecutive IDs were likely activated in sequence by the same entity staking multiples of 32 ETH at a time, and are hosted under the same ISP, potentially sharing IP prefixes. By computing intersections of candidate node IP prefixes across consecutive validator IDs, we find that 50% of validators have at least one top candidate node sharing a common prefix. As Figure 3a shows, consecutive IDs typically share either one common prefix or none. Cases with no common prefix may correspond to validators hosted in different locations, no longer active, or hosted on nodes unobserved by our SuperNode. To confirm this mapping is meaningful, we shuffled validator IDs and repeated the analysis, finding almost no common prefixes—demonstrating that our initial mapping is not coincidental or dominated by a few highly connected nodes.

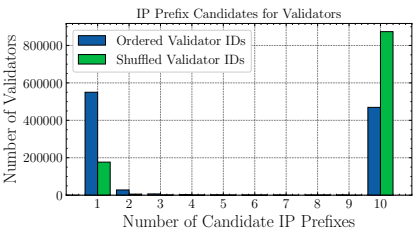
SuperNode relies on a small verifiable dataset to generalize inference. Although latency is a strong signal, it can be noisy, making manual heuristics insufficient to map all validator-to-node assignments. To leverage complex patterns in timing and auxiliary data, we train a machine learning model to identify which node in a candidate set hosts a given validator. Our intuition is that per-validator statistics for the host node have distinctive validator-specific features compared to other nodes, such as markedly lower latency. The model takes as input the behavior of the ten lowest-latency nodes for a validator and predicts the host. Importantly, we do not expect a learnable pattern in per-node behavior, as this is confounded by factors such as the number of validators a node hosts.

For training, we further filter our previously mapped validators using the intuition that validators sharing the same deposit keys may be co-located in the network. This filtering does not bias the model toward recognizing only co-located validators, as the patterns it learns—e.g., attestation timing, OOS counts—are independent of ownership. The step primarily reduces potential false positives at the cost of some false negatives (*i.e.,* mappings that were correct but we are not using them to train).

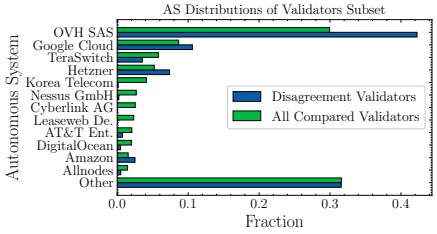
To implement this, we cluster validators by deposit key and examine their distributions across ASes, scoring each distribution using Efficiency (normalized Shannon Entropy). Near-zero efficiency indicates a skew toward a single AS, i.e., largely homogeneous prefix assignments. While some owners may distribute validators across multiple ASes, we train only on low-efficiency assignments for greater confidence in

training quality. Our model is a simple MLP with four hidden layers (3×128, 1×64 neurons) that uses ReLU activations, cross-entropy loss and the Adam (LR=0.001) for 1,000 epochs. To improve generalization, we randomly permute the order of node behavior inputs each epoch. The trained model achieves 91% classification accuracy, which we use to map all remaining validators to their prefixes.

We validate SuperNode's mappings. Validating all mappings is inherently difficult due to the absence of ground truth. However, the lack of common prefixes among shuffled validator IDs and the high accuracy of our ML model increase confidence in the inferred associations. To further support our results, we compare our mappings to those of Heimbach et al. [25], analyze the hosting environments of Ethereum staking pool operators, and evaluate an alternative mapping methodology.



3a: For > 50% of validators, nodes in a single IP prefix have exceptionally low latency in delivering attestations of consecutive validator IDs.



3b: Validators in multi-region cloud environments show reduced mapping consistency, possibly due to IP prefix changes or datacenter migration.

We compare our mappings against those provided by Heimbach et al. [25], which, like SuperNode's, could not be readily verified against ground truth. Their mappings cover approximately 11% of our mapped validator set, and exclude validators hosted on nodes that subscribe to all 64 attestation gossip topics. Each validator is associated with a set of candidate countries inferred by up to four independent measurement nodes. Due to the sensitive nature of the data, these mappings do not include finer-grained network identifiers such as ASNs or IP prefixes. Consequently, our comparative analysis may be subject to artifacts introduced by IP-to-geolocation mapping. Moreover, the measurements underlying their mappings were conducted in May 2024 [25], whereas SuperNode uses data collected from November 2024 to January 2025. Although large-scale migration is unlikely, some validators may have changed IP prefixes during this interval.

For the comparison, we restrict our analysis to the subset of overlapping validators for which Heimbach et al. have high confidence in their assignment - defined as agreement across all their measurement nodes. We observe that agreement between the two mappings increases with the confidence of the provided assignments, i.e., with the number of measurement nodes in agreement. For validators with the highest confidence (four agreeing measurement nodes), the two mappings agree for 70% of validators. Finally, we analyze the validators where the two mappings disagree. Grouping these validators by the ASN assigned by SuperNode (Figure 3b), we observe that validators mapped to single-country ASes like Korea Telecom, Nessus, Cyberlink, and Leaseweb are likelier to be assigned to the same country in both

methodologies. In contrast, validators hosted on multi-region cloud providers like OVH, Google Cloud, Amazon, and Hetzner are more likely to receive disagreeing country assignments. This pattern could indicate validators in the latter category being reassigned IP prefixes in different geographic locations within the cloud environment, or migration to different datacenters.

We analyze the hosting environments of Ethereum staking pool operators collaborating with Lido [38]. Specifically, we identify the ASN hosting each operator's website and check whether any validators mapped by our methodology fall within the same ASN. For roughly half of the operators, at least one validator is co-located with the website in the same ASN, often corresponding to known IaaS or cloud providers (e.g., Amazon, OVH). For the remaining operators, the ASNs typically belong to CDNs or edge providers (e.g., Cloudflare, Fastly) or security services, which are less likely to host validator infrastructure. This contrast further supports the plausibility of our mappings.

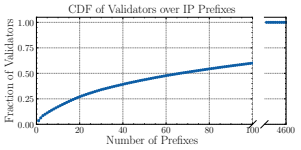
Despite our best efforts, one might still worry that more traditional mapping methodologies would prove our attacks less practical. To investigate this hypothesis, we also generate a mapping using simple heuristics similar to what was used in Bitcoin *e.g.,* by [18], and report results in appendix §D.

5 Ethereum PoS Network analysis

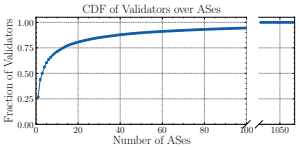
We analyze the Ethereum PoS topology inferred in §4, examining validator centralization across IP prefixes, Autonomous Systems (ASes), and geographic locations, which has direct implications for our network-level attacks and Ethereum security.

Methodology: Using our validator-to-IP mappings, we resolve routable prefixes, ASNs, ROA coverage, and countries. We obtain routable prefixes and ASNs by querying the RouteViews RIB via the pyASN library [39, 40]. Finally, we map countries using IPInfo Lite [41], and check ROA coverage with Routinator [42].

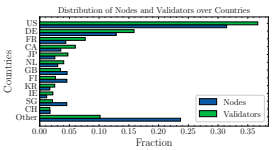
Validator Centralization High validator concentration increases an attacker's efficiency in targeting large portions of the network. Ethereum validators are highly centralized at the prefix level: 60% reside under just 100 prefixes, with individual prefixes hosting up to 37,000 validators (Figure 4a). The full set ($N = 1{,}063{,}660$) spans only 4,600 prefixes, and over a third of it is under just 29 prefixes, meaning a StakeBleed attacker could drive the network into the leak state by hijacking just these prefixes. As a robustness check on our centralization estimates, we repeated the analysis using an alternative mapping methodology (see §D), which reveals even greater centralization.



4a: Almost 60% of validators are hosted in just 100 prefixes.



4b: Just 3 ASes host $> 50\%$ of Ethereum's validators.



4c: The US hosts $\approx 37\%$ of validators and 31% of nodes.

Validators are also concentrated at the AS level: just 20 ASes host $> 80\%$ of validators, and the full set spans only 1,057 of 118,000 allocated ASNs (Fig. 4b).

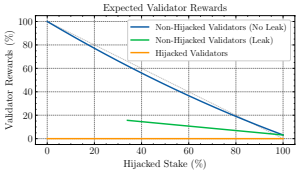
Notably, OVH alone hosts almost 27% of validators, close to the 1/3 threshold needed to disrupt block finalization.

Geographic centralization is similarly high: approximately 37% of validators reside in the U.S., and just 12 countries account for nearly 90% of stake (Figure 4c). While non-state actors are unlikely to compromise entire countries, a single country or a small coalition could threaten Ethereum consensus by surpassing the 1/3 stake threshold.

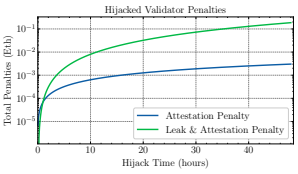
6 StakeBleed analysis

Validator rewards are multiples of the base reward rate b which scales with the inverse of the square root of the total active stake. Considering Ethereum's current validator set size of $N = 1{,}063{,}660$, we derive $b = 346$. Rewards and penalties are applied to the validator's balance, which is used to update its effective balance of 1 ETH increments every epoch. The latter is used in determining the rewards, penalties, and the probability of performing duties other than attesting. In this section we consider all validators to have their initial 32 ETH of actual balance.

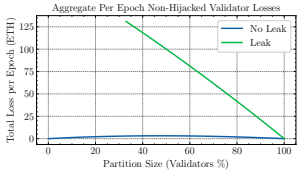
Ethereum validators earn on average 11,072 Gwei per epoch (6.4 minutes) in rewards which can be broken down to R_A (9,342 Gwei), R_P (1,384 Gwei/epoch avg.) and R_Y (346 Gwei/epoch avg.) for performing the following duties: casting attestations, proposing blocks, and participating in the sync committee. Validators cast attestations regularly in every epoch, but their selection for block proposal and sync committee participation is random and proportional to their effective balance, creating variance in earned rewards. Attestation rewards can be broken down into $R_A = R_s + R_t + R_h$ (2,422, 4,498 and 2,422 Gwei), for correct and timely votes for what the validator believes to be the *source*, *target*, and *head* blocks respectively. Finally, block proposal rewards are also divided into two parts $R_P = R_{PA} + R_{PY}$ (up to 1,335 and 49 Gwei/epoch avg.), that scale with the number of attestations and sync committee outputs that the proposer includes in the block.



5a: Non-hijacked val. rewards decrease as P grows. A leak further reduces them by 84%. Hijacked val. lose all rewards.



5b: As the attacker maintains the partition, the inactivity penalty grows quadratically over time.



5c: The attacker causes two orders of magnitude higher losses to non-hijacked validators when she causes a leak.

Non-hijacked validator rewards: if a StakeBleed attacker hijacks a p fraction of validators, non-hijacked validators lose on average $13148 \cdot p - 2422 \cdot p^2$ Gwei per epoch in rewards, which increases to $1384 \cdot p + 9342$ Gwei if the attacker induces a leak. In the absence of a leak, the losses occur because of three things: (i) attestation rewards as a whole are directly reduced by the fraction of the hijacked stake, (ii) validators voting in a slot of a hijacked proposer are unable to cast a

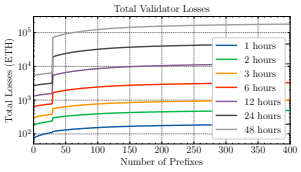
timely head vote, forfeiting that reward, (iii) non-hijacked block proposers are unable to collect attestations and sync contributions from the p fraction of hijacked validators. Putting these together, a validator's expected per-epoch reward becomes $E[R_V] = (1-p)\left[(1-p)R_h + R_s + R_t + \frac{32 \cdot R_P}{N}\right] + \frac{32 \cdot 512 \cdot R_Y}{N}$. If the attacker forces the chain to enter the leak state, R_h,R_s,R_t are set to zero, effectively removing attestation rewards. Figure 5a depicts the fraction of the their full expected rewards that validators receive as a function of the hijacked stake. Taken in aggregate, an attacker can cause losses of over 125 ETH per epoch to non-hijacked validators with a partition size of $p > 0.33$, as shown in Figure 5c. Note that all rewards R_x are multiples of the base reward rate b. A reduction in total active stake would increase b and consequently all R_x which means that our estimates serve as a lower bound for validator losses.

Hijacked validator rewards: regardless of the leak, hijacked validators miss all rewards since, when they rejoin the network, they adopt the majority view of the chain that non-hijacked validators have as canonical. Instead, hijacked validators incur an attestation penalty that scales as $0.625 \cdot n \cdot b$, with n being the validator's effective balance increments of 1 ETH. For our given b, and 32 effective balance increments, this amounts to 6,920 Gwei per epoch. If the attacker triggers a leak, hijacked validators also pay a $\frac{s_i \cdot B_i}{4 \cdot 1677216}$ Gwei penalty on every epoch, where s_i is the validator's inactivity score at epoch i (increases by 4 per epoch hijacked during the leak), and B_i is the validator's effective balance at epoch i (initially $32 \cdot 10^9$ Gwei). These penalties are depicted in Figure 5b as a function of the hijack length. Depending on the duration of the hijack, the inactivity penalty can continue to apply even after the network exits the leak state. Validators that have their effective stake balance reduced to the 16 ETH threshold are ejected from the active validator set - this takes $\sim$ 3 weeks with a leak for a validator that has a 32 ETH initial balance.

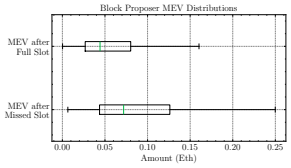
Ethereum's blockchain throughput in Transactions Per Second (TPS) decreases proportionately when an attacker hijacks a fraction f of validators. This is because block proposers are selected pseudorandomly in proportion to their effective balance. Thus, hijacking fraction f of validators prevents the same fraction of block proposals.

StakeBleed's costs in practice: We estimate the aggregate losses that a realistic adversary can inflict on validators through penalties and missed rewards. Network operators surveyed on BGP hijacks reported the majority of hijacks lasted over an hour, and their effects spanned for hours or more [43]. We thus consider an adversary with a hijacking time budget of a handful of hours during which she prioritizes hijacking prefixes with the most hosted validators to maximize her impact. As shown in Figure 6a, the attacker significantly improves her ability to inflict losses by crossing the threshold of 29 prefixes required to trigger a leak (see §4), causing nearly 300 ETH in damages with just a 2-hour hijack. Our findings also show that even if the attacker does not trigger a leak, prolonging the attack is an effective way to increase her impact, with a 3-hour hijack causing an aggregate of 379 ETH in losses.

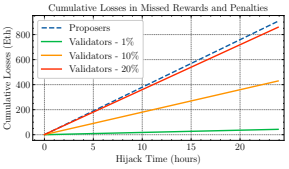
Experimental validation: We experimentally validate StakeBleed in a simulated Ethereum network. Our simulations take as input the validators distribution, and the changes to the network connectivity graph when the attacker performs a hijack. Our results confirm StakeBleed's effectiveness, as described in §6 (see §E for details).



6a: A 30 prefix hijack for 2/3 hours can cause $\sim$ 300/379 ETH losses with/w.o a leak.



6b: Blocks proposed after an empty slot had 44.5% higher average MEV.



6c: Targeting proposers yields similar damages as hijacking 20% of validators.

7 KnockBlock

Maximizing attack success: To maximize success the attacker targets proposers for which she has high mapping confidence (e.g., those seeded via consecutive validator IDs) and prefixes that are easy to hijack (e.g., prefixes $<$ /24 without an ROA). She may also hijack additional candidate prefixes for the same validator to hedge mapping or filtering failures. If the immediate preceding proposer cannot be suppressed, the attacker can instead target proposers/attesters in the few preceding slots; because blocks include a bounded number of prior-slot attestations, this means the attacker still has a high probability of including attestations intended for previous slots in her block.

Proposer predictability: KnockBlock requires accurately predicting the proposer schedule with enough lead time for a BGP announcement to take effect. Effective validator balances change only rarely (only 0.0033% of blocks include a slashing [44]), and the RANDAO-based schedule is available two epochs in advance, giving an attacker a practical prediction window of $\approx$12.8 minutes. BGP announcements can take effect in less than 90 seconds [18], so the attacker can reliably prepare short, targeted hijacks.

Impact analysis: Successful KnockBlock attacks lead to a measurable increase of up to 453.6% in the attacker's revenue in the form of block proposal rewards because of the inclusion of additional attestations and sync committee contributions. Using the protocol specification the expected block proposal reward is $E[R_P] = \frac{1}{7} \cdot \left[\frac{N}{32} \cdot R_A + 512 \cdot R_Y\right]$. In the event that there exist valid attestations issued in the preceding 32 slots that have not been included in another block, the block proposer can include these for an additional reward. In particular, new attestations receive the full reward of $\frac{54 \cdot n \cdot b}{64}$ Gwei. Attestations delayed up to 5 slots lose the head portion of their reward, meaning that they receive $\frac{40 \cdot n \cdot b}{64}$ Gwei. Attestations delayed 6 to 32 slots also lose the source portion of the reward and only receive the target portion: $\frac{26 \cdot n \cdot b}{64}$ Gwei. A proposer can include up to 8 slots worth of attestations [45] in their block, increasing their reward by up to 453.6%. A single block proposal is worth 46,003,328 Gwei, while a block proposed by a KnockBlock adversary is worth up to 254,674,423 Gwei. Using mevboost payment data [46], we observe that blocks following a missed slot yield, on average, 44.5% higher MEV (Fig. 6b). As a result, by forcing a proposer to miss its slot, a KnockBlock adversary also increases her expected MEV by 44.5%.

Attack costs: A KnockBlock adversary can use her profits from running the attack to amortize her costs in setting up her own AS (RIR fees [47], transport providers [13], hardware [29], etc.) if she does not run or has already compromised one. Given that a validator proposes on average $\sim$ 2.47 times per year, and that

KnockBlock attack results in an average increase of 0.05 ETH in MEV and 0.03 ETH in attestation rewards, the validator's annual revenue increase because of the attack will be 0.19 ETH (848 USD at the time of writing). By estimating annual AS operating costs to be approximately $2,500, the attack is profitable if the adversary owns more than 3 validators. This is a very pessimistic estimate for the attacker, considering that she can increase her revenue at no additional cost by collaborating with other validator owners and targeting the most profitable MEV opportunities.

Comparative impact A focused KnockBlock campaign can cause aggregate network damage comparable to larger StakeBleed partitions while being substantially stealthier (Fig. 6c). For example, with a 24-hour active hijack budget, both attack types can impose >800 ETH in losses, but KnockBlock achieves similar damage via many brief, single-prefix hijacks rather than sustained multi-prefix hijacks.

8 Countermeasures

Automate migration on local detection: A key challenge in mitigating BGP hijacking is that its impact cannot be addressed at the application layer, nor can it be resolved locally. Recovery requires intervention by the affected network provider, who must coordinate with upstream and peer providers to filter the bogus advertisement *at their networks*—a process that often takes several hours [6, 7]. This makes it essential for validator operators to implement migration strategies, meaning automated processes that relocate validator operations to alternative (cloud) locations, under different providers and separate jurisdictions to minimize shared dependencies. This can be triggered by observable signals, such as changes in peer connectivity, unexpected reward loss, or increased message delays, which might indicate malicious reroutes [48].

Reducing predictability: StakeBleed is feasible because the attacker can monitor validator activity and adapt the partition accordingly. Knowing when each validator is expected to attest, she can verify whether she has successfully isolated them. KnockBlock relies on the adversary's knowledge the block proposer schedule in advance, reducing the BGP hijacking set to even a single prefix. Removing the predictability in both cases would significantly strengthen the protocol against such attacks. For hiding the proposer schedule, one can use one of the several efficient SSLE constructions [49, 50]. In fact, integrating SSLE into Ethereum is actively being discussed [51].

Obfuscating validator locations: Our attacks depend on accurately mapping validators to IP addresses using P2P timing signals. We found that naive countermeasures are ineffective; adding exponentially distributed latency ($1/\lambda = [50,500]$ms), for instance, barely reduces our mapping model's accuracy (1-3%) and risks harming liveness. This finding is consistent with prior work in Bitcoin [52]. A more effective approach is to disrupt the identifying signals at the source. Disabling "eager push" and OOS attestations is highly effective, slashing our inference model's accuracy from 93% to 38%. Alternatively, validators could use temporary proxy nodes to broadcast attestations, effectively serving as decoys to obscure their true location.

9 Related Work

Network attacks and defenses on PoW Routing attacks on cryptocurrencies have previously been discussed in the context of partitioning the Bitcoin network [18, 53]. Tran et al. demonstrated that it is possible to attack PoW mining pools by tampering with Stratum connections intercepted via BGP hijacks [19]. Researchers have also attempted to partition multiple cryptocurrencies at once by inferring Bitcoin's and Ethereum's network topologies through Ripple's [28]. Neither of these works leverages a PoS-specific mechanism, nor do they infer the Ethereum PoS topology. Eclipsing attacks on blockchain peer-to-peer networks [14–17], are not routing attacks per se as they do not perform a BGP hijack and can be resolved with a single whitelisted connection. Defenses against routing attacks include SABRE [54], which, however, contributes to the network's consolidation and offers little incentive for newcomers.

Attacks on Ethereum PoS consensus Recent research in Ethereum PoS has focused on manipulating its fork choice rule and finality gadget to degrade the blockchain's performance, yet without exploiting network routing. These methods allow an attacker to perform reorgs [20, 22], and split validators across different chains [21]. In response, Ethereum introduced countermeasures and patches, such as proposer boost, but these have been shown to still be vulnerable [23, 24]. Finally, prior work has investigated ways to further exploit Ethereum's inactivity leak [55] after it has been triggered if the attacker holds stake. Such attacks can be used atop StakeBleed to increase its impact.

10 Conclusion

This paper presents the first comprehensive analysis of Ethereum's PoS from the Internet routing perspective, making two key contributions. First, we uncover a new class of network-layer attacks that exploit routing vulnerabilities and PoS mechanisms. We demonstrate two representative attacks: StakeBleed, which can halt finality and causes losses of almost 300 ETH in 2 hours by hijacking as few as 30 prefixes, and KnockBlock, which can boost MEV gains by 44.5% with a hijack lasting under 2 minutes. Second, we design a novel SuperNode that scalably logs timing information with minimal disruption to the live network, and introduce a multistage algorithm to infer the Internet-level distribution of validators from this timing data. We also propose a set of readily deployable countermeasures that move beyond idealistic calls to secure the entire Internet and focus on practical, actionable, adopter-facing defenses.

11 Acknowledgments

This work was supported by the Princeton Innovation Fund Award and the Princeton DeCenter. We thank the authors of Heimbach et al. [25] for sharing their mappings, which enabled our comparison.

References

1. Y. Gilad, A. Cohen, A. Herzberg, M. Schapira, and H. Shulman, "Are we there yet? on RPKI's deployment and security," Cryptology ePrint Archive, Paper 2016/1010, 2016. [Online]. Available: https://eprint.iacr.org/2016/1010
2. Q. Labs, "Q1 2023 ddos attacks and bgp incidents," https://blog.qrator.net/en/q1-2023-ddos-attacks-and-bgp-incidents_171/, 2023, accessed: 2025-06-03.
3. ——, "Q2 2023 ddos attacks statistics and overview," https://blog.qrator.net/en/q2-2023-ddos-attacks-statistics-and-overview_177/, 2023, accessed: 2025-06-03.
4. ——, "Q1 2025 ddos, bots and bgp incidents statistics and overview," https://blog.qrator.net/en/q1-2025-ddos-bots-and-bgp-incidents-statistics-and_211/, 2025, accessed: 2025-06-03.
5. Mutually Agreed Norms for Routing Security (MANRS), "Bgp security in 2021," https://manrs.org/2022/02/bgp-security-in-2021/, 2022, accessed: 2025-06-03.
6. B. Herdes, M. Zhang, and T. Ryan, "Cloudflare 1.1.1.1 incident on june 27, 2024," https://blog.cloudflare.com/cloudflare-1111-incident-on-june-27-2024/, 2024, accessed: 2025-06-03.
7. C. Martinho and T. Strickx, "Understanding how facebook disappeared from the internet," https://blog.cloudflare.com/october-2021-facebook-outage/, 2021, accessed: 2025-06-03.
8. R. Mahajan, D. Wetherall, and T. Anderson, "Understanding bgp misconfiguration," in *Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications*. ACM, 2002, pp. 3–16.
9. "Celer bridge incident analysis," https://www.coinbase.com/blog/celer-bridge-incident-analysis.
10. "A $152,000 cryptocurrency theft just exploited a huge 'blind spot' in internet security," https://www.forbes.com/sites/thomasbrewster/2018/04/24/a-160000-ether-theft-just-exploited-a-massive-blind-spot-in-internet-security/?sh=35607bc85e26.
11. L. Abrams, "Hacker hijacks orange spain ripe account to cause bgp havoc," https://www.bleepingcomputer.com/news/security/hacker-hijacks-orange-spain-ripe-account-to-cause-bgp-havoc/, 2024, accessed: 2025-06-03.
12. A. Greenberg, "Hacker redirects traffic from 19 internet providers to steal bitcoins," https://www.wired.com/2014/08/isp-bitcoin-theft/, 2014, accessed: 2025-06-06.
13. D. Swer, "How i set up my own autonomous system," *APNIC Blog*, 2022, online; accessed June 5, 2025. [Online]. Available: https://blog.apnic.net/2022/07/01/how-i-set-up-my-own-autonomous-system/
14. E. Heilman, A. Kendler, A. Zohar, and S. Goldberg, "Eclipse attacks on bitcoin's peer-to-peer network," in *24th USENIX Security Symposium, USENIX Security 15, Washington, D.C., USA, August 12-14, 2015.*, 2015, pp. 129–144. [Online]. Available: https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/heilman
15. H. Heo, S. Woo, T. Yoon, M. S. Kang, and S. Shin, "Partitioning ethereum without eclipsing it." in *NDSS*, 2023.
16. M. Tran, I. Choi, G. J. Moon, A. V. Vu, and M. S. Kang, "A stealthier partitioning attack against bitcoin peer-to-peer network," in *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2020, pp. 894–909.
17. M. Saad, S. Chen, and D. Mohaisen, "Syncattack: Double-spending in bitcoin without mining power," in *Proceedings of the 2021 ACM SIGSAC conference on computer and communications security*, 2021, pp. 1668–1685.
18. M. Apostolaki, A. Zohar, and L. Vanbever, "Hijacking bitcoin: Routing attacks on cryptocurrencies," in *2017 IEEE symposium on security and privacy (SP)*. IEEE, 2017, pp. 375–392.

19. M. Tran, T. von Arx, and L. Vanbever, "Routing attacks on cryptocurrency mining pools," in *2024 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, 2024, pp. 190–190.
20. M. Neuder, D. J. Moroz, R. Rao, and D. C. Parkes, "Low-cost attacks on ethereum 2.0 by sub-1/3 stakeholders," *arXiv preprint arXiv:2102.02247*, 2021.
21. J. Neu, E. N. Tas, and D. Tse, "Ebb-and-flow protocols: A resolution of the availability-finality dilemma," in *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2021, pp. 446–465.
22. C. Schwarz-Schilling, J. Neu, B. Monnot, A. Asgaonkar, E. N. Tas, and D. Tse, "Three attacks on proof-of-stake ethereum," in *Financial Cryptography and Data Security*, I. Eyal and J. Garay, Eds. Cham: Springer International Publishing, 2022.
23. J. Neu, E. N. Tas, and D. Tse, "Two more attacks on proof-of-stake ghost/ethereum," in *Proceedings of the 2022 ACM Workshop on Developments in Consensus*, ser. ConsensusDay '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 43–52. [Online]. Available: https://doi.org/10.1145/3560829.3563560
24. U. Pavloff, Y. Amoussou-Guenou, and S. Tucci-Piergiovanni, "Ethereum proof-of-stake under scrutiny," in *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*, ser. SAC '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 212–221. [Online]. Available: https://doi.org/10.1145/3555776.3577655
25. L. Heimbach, Y. Vonlanthen, J. Villacis, L. Kiffer, and R. Wattenhofer, "Deanonymizing ethereum validators: The p2p network has a privacy issue," *arXiv preprint arXiv:2409.04366*, 2024.
26. D. Vyzovitis, Y. Napora, D. McCormick, D. Dias, and Y. Psaras, "Gossipsub: Attack-resilient message propagation in the filecoin and eth2. 0 networks," *arXiv preprint arXiv:2007.02754*, 2020.
27. Y. Rekhter, T. Li, and S. Hares, "A border gateway protocol 4 (bgp-4)," Tech. Rep., 2006.
28. M. Saad and D. Mohaisen, "Three birds with one stone: Efficient partitioning attacks on interdependent cryptocurrency networks," in *2023 IEEE Symposium on Security and Privacy (SP)*, 2023, pp. 111–125.
29. N. Bouwhuis, "Setting up a personal asn," *nick.bouwhuis.net*, 2023. [Online]. Available: https://nick.bouwhuis.net/posts/2023-02-12-setting-up-a-personal-asn/
30. Y. Sun, M. Apostolaki, H. Birge-Lee, L. Vanbever, J. Rexford, M. Chiang, and P. Mittal, "Securing internet applications from routing attacks," *Communications of the ACM*, vol. 64, no. 6, pp. 86–96, 2021.
31. S. Goldberg, "Why is it taking so long to secure internet routing?" *Commun. ACM*, vol. 57, no. 10, p. 56–63, Sep. 2014. [Online]. Available: https://doi.org/10.1145/2659899
32. A. Siddiqui, "Klayswap – another bgp hijack targeting crypto wallets," *MANRS*, 2022, accessed 16 August 2024. [Online]. Available: https://manrs.org/2022/02/klayswap-another-bgp-hijack-targeting-crypto-wallets/
33. D. Goodin, "How 3 hours of inaction from amazon cost cryptocurrency holders $235,000," *Ars Technica*, 2022, accessed 16 August 2024. [Online]. Available: https://arstechnica.com/information-technology/2022/09/how-3-hours-of-inaction-from-amazon-cost-cryptocurrency-holders-235000/
34. Dapplion, "EIP-7917: How often does it help?" 2025, accessed April 11 2025. [Online]. Available: https://ethereum-magicians.org/t/eip-7917-deterministic-proposer-lookahead/23259
35. R. Mahajan, D. Wetherall, and T. Anderson, "Understanding bgp misconfiguration," vol. 32, no. 4, p. 3–16, Aug. 2002. [Online]. Available: https://doi.org/10.1145/964725.633027
36. Ethereum Foundation, "Staking launchpad: Validator checklist," accessed April 18 2025. [Online]. Available: https://launchpad.ethereum.org/en/checklist

37. A. Miller, J. Litton, A. Pachulski, N. Gupta, D. Levin, N. Spring, B. Bhattacharjee *et al.*, "Discovering bitcoin's public topology and influential nodes.(2015)," *Cited on*, p. 54, 2015.
38. Lido, "Ethereum node operators," *Lido Node Operator Portal*, 2025, online; accessed 5 June 2025. [Online]. Available: https://operatorportal.lido.fi/lido-operators-database/ethereum-node-operators
39. The University of Oregon, "Route Views Project." [Online]. Available: http://www.routeviews.org/routeviews/
40. H. Asghari and A. Noroozian, "pyasn: Python ip address to autonomous system number lookup module," 2014, accessed August 25 2024. [Online]. Available: https://github.com/hadiasghari/pyasn
41. IPInfo, "IPInfo Lite," 2026, accessed January 4 2026. [Online]. Available: https://ipinfo.io/lite
42. NLnet Labs, "Routinator." [Online]. Available: https://www.nlnetlabs.nl/projects/routing/routinator/
43. P. Sermpezis, V. Kotronis, A. Dainotti, and X. Dimitropoulos, "A survey among network operators on bgp prefix hijacking," *ACM SIGCOMM Computer Communication Review*, vol. 48, no. 1, pp. 64–69, 2018.
44. Rated.Network, "Rated explorer." [Online]. Available: https://explorer.rated.network/slashings?network=mainnet&timesSlashedPage=1&slashesReportedPage=1
45. Dapplion and M. Kalinin, "EIP-7549: Move committee index outside of the signed Attestation message," *Ethereum Improvement Proposals*, 2023. [Online]. Available: https://eips.ethereum.org/EIPS/eip-7549
46. T. Wahrstätter, "Mevboost.pics - open data," 2023, accessed April 12 2025. [Online]. Available: https://mevboost.pics/data.html
47. APNIC, "How much does it cost?" *APNIC*, 2025. [Online]. Available: https://www.apnic.net/get-ip/apnic-membership/how-much-does-it-cost/
48. A. Milolidakis, T. Bühler, K. Wang, M. Chiesa, L. Vanbever, and S. Vissicchio, "On the effectiveness of bgp hijackers that evade public route collectors," *IEEE Access*, vol. 11, pp. 31 092–31 124, 2023.
49. D. Catalano, D. Fiore, and E. Giunta, "Efficient and universally composable single secret leader election from pairings," in *IACR International Conference on Public-Key Cryptography*. Springer, 2023, pp. 471–499.
50. D. Boneh, S. Eskandarian, L. Hanzlik, and N. Greco, "Single secret leader election," in *Proceedings of the 2nd ACM Conference on Advances in Financial Technologies*, 2020, pp. 12–24.
51. G. Kadianakis, J. Drake, D. Feist, G. Herold, D. Khovratovich, M. Maller, and M. Simkin, "Whisk: A practical shuffle-based ssle protocol for ethereum," *Ethereum Research*, 2022, online; accessed June 4, 2025. [Online]. Available: https://ethresear.ch/t/whisk-a-practical-shuffle-based-ssle-protocol-for-ethereum/11763
52. G. Fanti, S. B. Venkatakrishnan, S. Bakshi, B. Denby, S. Bhargava, A. Miller, and P. Viswanath, "Dandelion++: Lightweight cryptocurrency networking with formal anonymity guarantees," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 2, no. 2, Jun. 2018. [Online]. Available: https://doi.org/10.1145/3224424
53. M. Saad, V. Cook, L. Nguyen, M. T. Thai, and A. Mohaisen, "Partitioning attacks on bitcoin: Colliding space, time, and logic," in *2019 IEEE 39th international conference on distributed computing systems (ICDCS)*. IEEE, 2019, pp. 1175–1187.
54. M. Apostolaki, G. Marti, J. Müller, and L. Vanbever, "SABRE: Protecting Bitcoin against Routing Attacks," in *Proceedings of the 26th Network and Distributed System Security Symposium (NDSS'19)*, 2019.

55. U. Pavloff, Y. Amoussou-Guenou, and S. Tucci-Piergiovanni, "Byzantine attacks exploiting penalties in ethereum pos," in *2024 54th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2024, pp. 53–65.
56. Ethereum Foundation, "Pooled staking," accessed 21 April 2025. [Online]. Available: https://ethereum.org/en/staking/pools/
57. D. Knopik, "Ethshadow: Discrete-event ethereum network simulator," *Ethereum Protocol Fellowship - Cohort 5*, 2024. [Online]. Available: https://github.com/ethereum/ethshadow
58. R. Jansen and N. J. Hopper, "Shadow: Running tor in a box for accurate and efficient experimentation," 2011.
59. W. Du, H. Zeng, and K. Won, "Seed emulator: An internet emulator for research and education," in *Proceedings of the 21st ACM workshop on hot topics in networks*, 2022, pp. 101–107.
60. Ethereum Foundation DevOps Team, "Ethereum package," 2022. [Online]. Available: https://github.com/ethpandaops/ethereum-package
61. Kurtosis: A platform for packaging and launching ephemeral backend stacks with a focus on approachability for the average developer. [Online]. Available: https://github.com/kurtosis-tech/kurtosis
62. M. Bailey, D. Dittrich, E. Kenneally, and D. Maughan, "The menlo report," *IEEE Security & Privacy*, vol. 10, no. 2, pp. 71–75, 2012.

A StakeBleed Example

We illustrate the StakeBleed attack in Fig. 1 where the left figure corresponds to the Internet level view and the right to the blockchain view. We assume a small representative network of 8 ASes (Fig. 1), with some of these hosting Ethereum nodes with validators. The adversary controls a BGP-speaking router in AS 8 and aims to isolate the shaded nodes on the left side of the network. To achieve this, she advertises IP prefixes that cover the IP addresses of Ethereum clients A, B, and C. As a result, she intercepts traffic destined for these nodes, effectively enabling her to drop all connections between nodes in the target set P and the rest of the network. For example, the attacker drops traffic between nodes A and G, as illustrated by the red dashed lines. The results of the attack at the blockchain level are shown in Fig. 1. If the attack had not taken place, blocks 1, 2 and 3 would have been proposed by nodes A, B, and F, respectively and attested by a subset of the nodes in the network. During the attack nodes in P and outside it have distinct views of the chain. From the perspective of nodes outside of P, which will eventually prevail (canonical chain), node B appears to have failed to propose block 2 when it was supposed to. As a result, it is deprived of its proposer rewards. From the perspective of the canonical chain, the next block (block 3) will be proposed normally as node F is not in P, although some attestations will be missing because the corresponding validators are isolated reducing F's rewards from including attestation rewards. Nodes in P set to attest in that block will see penalties (illustrated with red dollars). Four epochs (or 25.6 minutes) later, the chain enters the leak state; meaning that the validators on nodes A, B, and C are considered inactive by those on D, E, F, and G, and thus incur both the attestation and inactivity penalties. At the same time, nodes D, E, F and G stop earning attestation rewards and only earn rewards when they propose blocks or contribute to the sync committee.

B KnockBlock Example

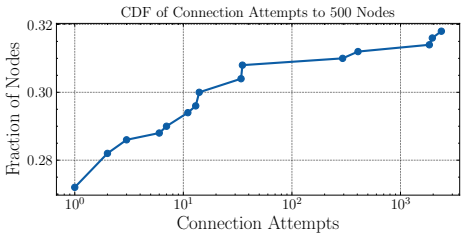
We illustrate the KnockBlock attack in Fig.2, where the adversary's validator node E is set to propose at Slot 3. The adversary calculates the proposer schedule ahead of time and identifies the validator ID that is scheduled to propose at Slot 2, namely that belonging to node D. By preventing node D from proposing during Slot 2, the attacker extends the pool of available transactions her validator can include and gains additional rewards from unincluded attestations. In the meantime, the validator of node D loses their proposer rewards.

C Miner identification vs Validator identification

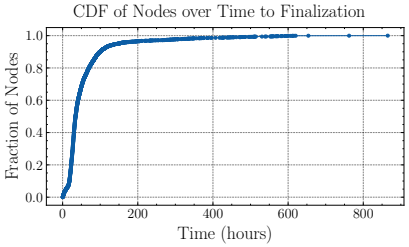
The first approach used to identify miners involves deploying multiple supernodes, namely real Bitcoin/Ethereum clients with multiple peers, which would log the timestamp when each of their peers advertised each block [18,37]. By analyzing the relative delays in advertising each block across different nodes, one could potentially identify the source of a block. This method, however, relies on two key assumptions: *(i)* that it is possible to maintain concurrent connections with a significant portion of the network to calculate relative delays, and *(ii)* that clients eagerly advertise blocks to all their peers. Unfortunately, neither holds true in Ethereum today. Because of the extremely bursty communication pattern of the Gossipsub protocol, maintaining connections to all nodes in the network (say 9,000) would require receiving, processing and logging information through the Internet at the impractical rate of 216 Gbps.[2] While this is not the average rate required, inability to keep up with incoming bursts will cause nodes to disconnect and unreliable timestamps. Beyond the scaling issue, we find that connecting to that many nodes on demand is impossible. Fig. 7a demonstrates our attempt to connect to nodes corresponding to 500 ENRs. Even after 2500 attempts and 625 hours of trying (attempting every 15 minutes), 68% of the nodes are still not connected. Additionally, Ethereum nodes do not immediately propagate messages to all their peers; instead, they only do so selectively (§4).

More recent approaches have relied on the public presence of mining pools and the predictability or advertisement of their stratum server URLs. In Proof-of-Work (PoW), independent mining is highly risky since rewards are only earned by successfully mining a block. Mining pools openly advertise their stratum servers, enabling miners to contribute their computational power. As a result, public mining pools must (directly or indirectly) disclose this information to attract participants [19]. However, this is not the case for Ethereum's Proof-of-Stake system. Anyone with 32 ETH can independently and securely be a validator, eliminating the need for public disclosure or reliance on vulnerable protocols. Even pooled stakers do not usually interact with their respective pools on the network level, but rather do so indirectly through on-chain smart contracts [56].

[2] This considers a low-estimate of 24 Mbps per node, which is what we measured in practice but much lower than the theoretic average of a fully-subscribed node.



7a: Out of 500 randomly selected node ENRs, our node succeeds in connecting to $< 1/3$ them on demand.



7b: CDF of SuperNode's node finalization time. 60% of node finalizations happened in just over 41 hours.

D An alternative assignment strategy

To validate the accuracy of our findings in Section 5, and assert that they are not negatively affected by the use of our mapping heuristics related to validator management (i.e., using consecutive validator IDs, and low-efficiency validators), we repeat the mapping process using an alternative heuristic that deliberately avoids considering validator ownership. Concretely, we consider the top 10 candidate prefixes for each validator by their relative latencies, and then select the candidate with the highest number of out-of-scope (OOS) attestations.

We compare the findings from this simplified mapping heuristic to those by our method in Section 5, and find them to be on par. As illustrated in Fig. 9, just 20 ASes host nearly 75% of Ethereum's validators, while Fig. 8 shows that nearly 60% of all validators are concentrated in 100 prefixes. These observations are similar to those made in Section 5. Where the simplified heuristic differs from our method in Section 4, is in the number of ASes and prefixes to which it makes validator assignments. Results from the simplified heuristic suggest a higher network centralization than that indicated in Section 4.

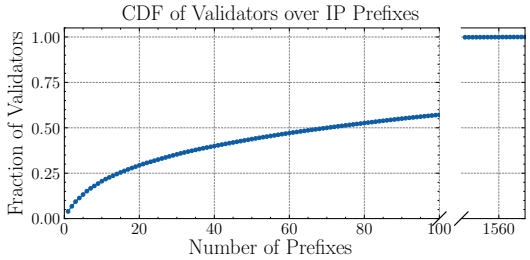


Fig. 8: Distribution of validators under prefixes using our simplified heuristic. Almost 60% of validators are hosted in just 100 prefixes.

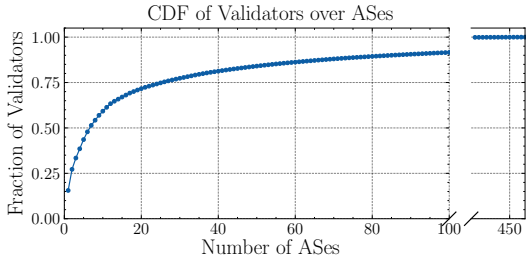


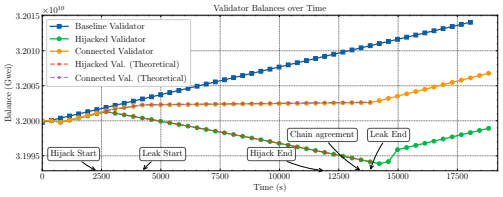
Fig. 9: Distribution of validators under ASes using our simplified heuristic. Just 100 ASes host almost 90% of Ethereum's validators.

E Simulation Setup

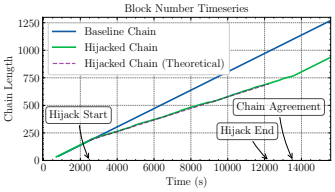
Simulation: Existing solutions for instantiating virtual devnets [57–59] either do not support simulating a given validator distribution or only offer limited configuration options for simulating our attacks. We adopt Ethereum Package [60], a package built by the Ethereum Foundation DevOps team for the Kurtosis [61] platform. We then modify Kurtosis to enable fine-grain networking control of the nodes, and write our own orchestration layer that uses a combination of firewall and traffic control rules to emulate the effects of a BGP hijack given its connectivity graphs.

Attack Scenario: We start the devnet with a strongly connected topology of 20 full Ethereum nodes, the maximum number of nodes that Kurtosis allows us to run on our server. We then assign validators to our devnet nodes by sampling the validator distribution derived in Section 4, and get a devnet of $N = 1{,}008$ validators. The attacker chooses to create a 37% partition by hijacking the connections to 3 of the nodes. The attacker initiates the hijack at second 2,496 of the simulation (Fig. 10a), and resolves the partition at second 12,228.

Observed Effects: True to the protocol, the simulation shows that the network enters the leak state after failing to finalize a checkpoint for 4 consecutive epochs. As can be seen in Figure 10a, connected and hijacked validators match their theoretically calculated balances. Connected validators initially have their rewards reduced proportionately to the hijacked partition size, and after the leak starts, they lose their attestation rewards entirely. Hijacked validators incur attestation and inactivity penalties, and even after the hijack ends and their nodes agree to follow the canonical chain, they continue to incur the inactivity penalty until their inactivity score eventually reaches zero. As also shown in Figure 10b, we confirm that the chain throughput is reduced according to the hijacked partition size, since approximately 37% of slots are missed. Note that the absolute values of penalties and rewards in our simulation differ from those of the Ethereum mainnet due to our simulation having a higher b because it hosts fewer validators.



10a: Baseline validator balances are recorded without adversarial activity. In the adversarial setup, a hijack triggers a leak 4 epochs after its start. When the hijack stops, the hijacked validator node regains connectivity, and eventually follows the canonical chain.



10b: Ethereum block chain growth slows down proportionately to the percentage of hijacked validators.

F Ethical Considerations

We discuss the ethical implications of our work using the Menlo report framework [62]. With regard to **respect for persons**, our work does not directly involve physical persons or their identities, but rather validator entities which we map to their network information. Considering **beneficience**, our mappings can be used to mount the proposed or other attacks against validators; thus, we intend to make the code and data available only upon request for review purposes or upon approval of the affected parties. We further intend to publish a high-fidelity yet anonymized version of the produced dataset for further research. To protect validators, we also discuss countermeasures to our methods. In the spirit of **justice**, our work is relevant to all validators, and benefits the entire Ethereum ecosystem. In **respect for public law and interest**, our work does not exploit any vulnerabilities (including performing BGP hijacks), uses only publicly available information, and aims to improve Ethereum's security posture.