Preventing Attacks Leveraging the Publicly Known List of Upcoming Proposers in Ethereum

Tereza Burianová (xburia28)
Brno University of Technology,
Faculty of Information Technology

***Abstract*—In the Ethereum PoS, the block proposers are known in advance and it is possible to obtain their IP addresses. This can lead to attacks like DoS, which can have negative impact on the network. The paper provides an overview of proposed and researched measures. The measures implemented partially or completely on the consensus layer include SSLE, like Whisk or Swap-or-Not SSLE, which generally manipulate a set of validators to hide the proposer's identity. These solutions are feasible based on the concluded research but are too complex. SnSLE, including the solution implemented in Algorand and the proposal adapted to Ethereum, select several proposers and choose one of them after their proposals. The anonymity set in these cases corresponds to the validator set, but fork-choice complications are introduced. The proposal for Polkadot is partially implemented on the network layer. It utilizes a construct called ring-VRF, which combines VRF and SNARKs. A solution implemented solely on the network layer, utilizing Dandelion++ and RLN to add a private pre-network, has also been included. Network layer solutions have too high latency based on an existing analysis and also are susceptible to attacks. The last solution is the distributed validators technology, where the obligations of one validator are distributed among several nodes. This solution is complex but increases the costs and requires a new type of client. The paper further describes the principle of each method, their drawbacks and points of further research, the methods are also briefly compared based on various properties.**

I. Introduction

In the current Ethereum consensus protocol, selected leaders (block proposers) are known in advance, which may lead to possible attacks, primarily the DoS attack, with the goal of attacker's enrichment or network limitation or shutdown. it has been shown [1] that Ethereum does not provide validator privacy and it is possible to map the leaders' IP addresses.

The goal of this paper is to summarize, review and compare existing approaches with emphasis on their suitability for Ethereum and the areas that need further research and analysis to be considered as a solution. The approaches include proposals intended for Ethereum and papers related to other blockchains, discussed as potentially feasible for Ethereum.

Most of the proposals aim to hide the proposer's identity by manipulating a larger set of validators, others select several proposers who propose their block and one of the blocks is then chosen. These measures are implemented on the consensus layer. The measures can also be partially or entirely implemented on the network layer.

The paper discusses properties like the complexity of the solution and the initial phase, anonymity set, influence on the network performance, possible vulnerabilities and others.

A. Contributions

The contributions of this survey paper are as follows:

1) Several approaches, discussed in the Ethereum Research community, were selected and their principles were shortly described. Ethereum proposals and general approaches or proposals and mechanisms of other blockchains were included.
2) Possible drawbacks and attacks on the approaches, which may need further analyses, were described.
3) The described approaches were compared based on their strengths and drawbacks.

II. Review of Approaches

- The papers were mainly selected based on various research already adapted to Ethereum, usually published on the Ethereum Research platform. Further on, several approaches of other blockchains, cited or discussed in the existing Ethereum research, were selected. Some of the sources were obtained using Perplexity AI.
- The countermeasures can be implemented either on the network layer, or on the consensus layer. The consensus layer approaches are then mainly divided into SSLE (Secret single leader election), where a single leader is elected secretly, and SnSLE (secret non-single leader election), where each validator has a change of being a valid proposer. Many of the papers provide analyses, comparisons of existing approaches and the proposed solutions.

A. Whisk: a shuffle-based SSLE protocol

Whisk is a SSLE protocol designed for Ethereum [2], inspired by the *SSLE from DDH and shuffles* construction from the article by Boneh et al. [3]. it is currently the most detailed and analyzed protocol for secret leader election.

The process starts by randomly choosing 16 384 validators using RANDAO, creating a candidate list. At this point, candidates are known to the adversary.

The next phase, which takes the next 8 192 slots, is shuffling. Due to the performance of zero knowledge proof, explained later, this phase has to be executed in smaller shuffles. To achieve this, the Randshuffle algorithm is proposed, where each of the proposers, in addition to proposing a block, chooses 128 random indices of the candidate list and shuffles the candidates by permuting and randomizing them. To simplify the proposal, Randshuffle replaced the previously

used Feistel shuffle, where proposers sequentialy shuffled all rows of a 128x128 candidates matrix and coordinates were mapped to different ones using Feistel cipher each time all of the rows have been shuffled.

To ensure honest shuffling, meaning the output is indeed a permutation of the input, each proposer has to prove the honesty using a zero knowledge proof. For this purpose, the Curdleproofs shuffle argument protocol has been proposed [4], which *relies solely on the discrete logarithm assumption* and *does not require a trusted setup*. The aim is to preserve discrete logarithm relations between pairs of group elements, allowing it to work with the representation of candidates, explained later. The shuffling phase stops one epoch before the next phase to prevent malicious shuffling based on the future RANDAO results.

After the shuffling phase, the 8 192 slots long proposal phase is launched. First, 8 192 winning candidates are selected as the next block proposers, using RANDAO. The future proposers are sequentially mapped to the corresponding beacon chain slots during the proposal phase, deciding the block proposers for the next 8 192 slots.

For the purpose of the algorithm, validators' identities are represented by trackers. Trackers are tuples (rG, krG), which allow validators to commit to a long-term secret k. Trackers can be randomized by a third party using a random secret g: $(zrG, zkrG)$, making various versions of the tracker unlinkable and still trackable by the owner.

To bind the identity of the validator to the tracker, a deterministic commitment $com(k) = kG$ is provided during the tracker registration and stored in the validator's record. Then, by providing a proof of knowledge of a discrete log, that proves the knowledge of k, such that $k(zrG) == zkrG$, and also by comparing the k in the tracker and in the commitment stored in the validator record, the owner of the winning tracker mapped to a slot in the beacon chain can prove the ownership of the tracker and propose a block in the corresponding slot.

Due to its complexity and therefore possible complications in future protocol improvements, Whisk is now being reviewed for possible simplifications. The current anonymity set is 8 192 validators, corresponding to the number of unselected candidates. The anonymity may be improved by increasing the number of validators in the candidate list, which would require an analysis of the shuffling algorithm performance. in terms of security, RANDAO attacks need to be further investigated, as the attacks prove to be more profitable than in the current implementation. it has been proposed to utilize a VDF beacon in the future.

B. Swap-or-Not SSLE

Swap-or-Not is a maximally simple prototype of SSLE in Ethereum. it utilizes the size-2 blind-and-swap primitive, which proves that two output blind commitments, (OL1, OR1) and (OL2, OR2), are re-encryptions of two input blind commitments, (IL1, IR1) and (IL2, IR2), without revealing which input the re-encrypted output refers to.

In this approach, an array *blinded_commitments* of size 2048, containing blinded commitments, is stored in the state. For each slot, random values of *proposer_blinded_index*, *fresh_validator_index* and *shuffle_offset* are chosen using RANDAO. The owner of blinded commitment at *proposer_blinded_index* of *blinded_commitments* can propose their block and their blinded commitment is replaced by a not yet used blind commitment from a list of validators at *fresh_validator_index* of *validators*.

To shuffle the validators in *blinded_commitments*, the shuffling tree mechanism is used. As shown in Fig. 1, the swap is performed between the index of the proposer x and another validator at the index set by the offset value and the current tree level. The resulting indices are wrapped around the *blinded_commitments* array of size 2048. To keep the algorithm robust in the case of malicious validators, who may reveal their swaps, the individual layers of the tree are sequentially preformed in the slots following the initial slot in which the shuffle tree was initiated. This provides an anonymity set of $2^{(D-M)}$, where D is the depth of the shuffling tree and M is the number of malicious shufflers in the process [5].

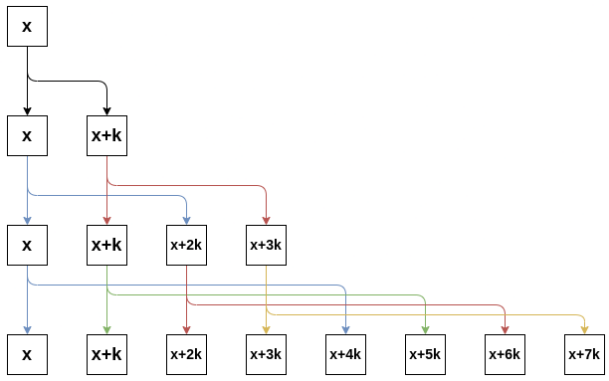

Fig. 1: The mechanism of a shuffling tree in Swap-or-Not SSLE.

In the existing analyses, it has been noted that the anonymity set may be too small, especially when considering offline or malicious validators, and more shuffling iterations may be needed to make the security comparable to Whisk. There are also several types of attacks that have not yet been explored, like various RANDAO attacks or the backtracking attacks, which could possibly reveal information about the future proposers based on the limited number of shuffle paths [5], [6].

C. Secret non-single leader election

The secret non-single leader election, further referred to as SnSLE, is a proposal, that brings the protocol closer to the PoW approach, giving validators an independent random chance to be a proposer.

The article suggests to give each validator between N active validators the chance of $\frac{5}{N}$ to be chosen as the valid proposer. To ensure this, the RANDAO reveal value must be kept under $\frac{2^{256}*5}{N}$. This approach creates a Poisson distribution

$\Lambda = 5$, which guarantees a high chance of having at least one proposer, while also having a high chance of choosing more than one proposer.

To ensure fairness, the proposer with the lowest RANDAO reveal hash is chosen from the set of proposers from the previous phase. At this point, all of the proposers have already proposed. This approach works for the cases where all of the proposers are honest or some of the proposers with a higher hash are dishonest. in the case of the proposer with the lowest hash being dishonest, not all attesters see the proposal in time, resulting in a split in votes or occasionally an empty slot. The recovery is performed using proposer boosting in the same way as in the status quo, applying the boost only to the lowest-hash proposal, as seen in Fig. 2. Contradictory to proposer boosting in the status quo, this might result in boosts in different directions, reducing the maximum safe size of a boost [7].

The proposal has not been thoroughly analyzed, although it has been suggested that to prevent adversary from attacking or taking advantage of SnSLE, it would best work combined with the proposer-builder separation [8]. it has also been remarked that the approach complicates the fork-choice significantly, making possible improvements difficult [2], therefore another approach would be preferred to SnSLE.

D. Algorand's VRF and cryptographic sortition

To prevent DoS attacks, verifiable random functions (VRFs) have been utilized by protocols like Algorand, DFINITY or Ouroboros Praos [9]. VRFs are a type of pseudo-random oracle, which allows the owner of the seed to prove that a value is correct, without making all values predictable by revealing the seed, in a non-interactive way [10].

In the process of proposer selection, cryptographic sortition is used, which randomly selects a group of users based on their account balance, including information about their priority and a proof of the priority. To discover if a user is included in the selected group, they compute a VRF, obtaining a proof of their membership. Due to the non-interactive approach, group members are unknown to a potential adversary until they start participating. Cryptographic sortition is used to select both proposers and committee members.

To minimize possible attacks once the identity of the proposer candidates becomes known, all of the candidates from the selected group gossip their proposed block. To reduce communication cost, the proposer is selected based on candidates' priorities, using small messages containing information about priorities and proofs of candidate group membership. The messages containing blocks by candidates with lower priorities can then be immediately discarded by users.

In the case of malicious proposers, inconsistencies can appear if the adversary has the highest proposer priority. These may cause Algorand to reach consensus on an empty block. Based on Algorand's assumption, the described situation is unlikely and should not cause difficulties in the protocol.

E. Distributed Validators

The Distributed validators (DV) proposal suggests that the responsibilities of a validator should not rely on one node, instead a validator should consist of several distributed nodes, called co-validators. in this case, each co-validator holds a share of the joint staking key. For signatures, the M-of-N threshold mechanism is used, where before signing a message, at least M of N co-validators need to reach consensus.

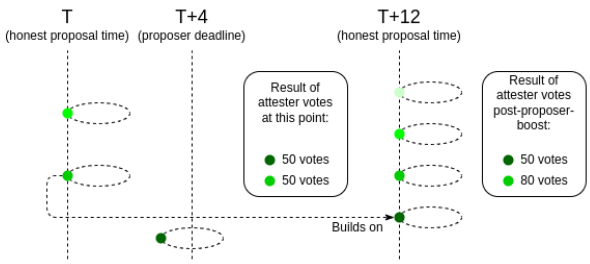

Fig. 2: Proposer boosting in the case of the lowest-hash proposer being dishonest in SnSLE.

As seen in Fig. 3, one of the possible ways to implement DV would be in the form of middleware between the beacon node and the remote signer, where both are unaware about the middleware, which provides the additional functionality without altering the communication [11], [12].

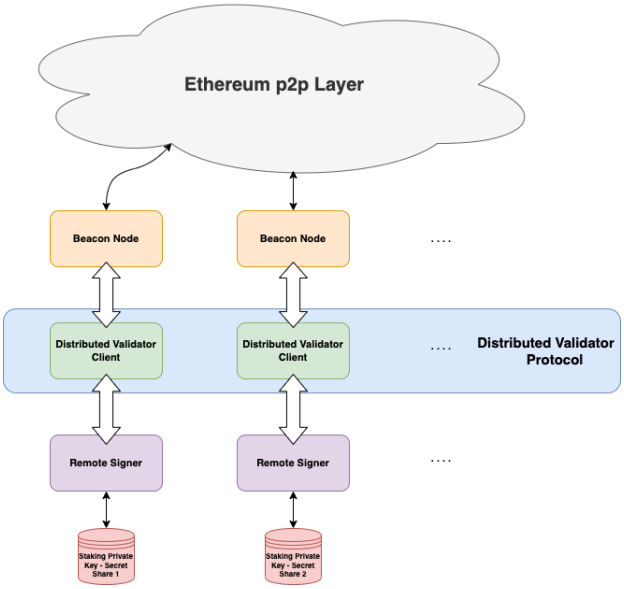

Fig. 3: The architecture of the Distributed Validators specification.

The introduction of DV would make DoS attacks on the validators more difficult, as more than $N - M$ co-validators would need to be offline for the validators not to reach consensus.

F. Network layer solution using Dandelion++ and RLN

As a solution on the network layer rather than on the consensus layer, the combination of Dandelion++ and RLN has been proposed. The motivation is the separation from the application logic and a possible opt-in approach.

A private pre-network, accessible only to validators, would be created using the Dandelion++ model. The network would obfuscate the message by propagating it through several nodes

before it reaches the consensus layer. To provide Sybil and spam protection, the proposal suggests including RLN mechanism, which requires submitting a zero knowledge proof to check the sender's identity and checking the network limits for each message. This allows for possible attackers to be removed from the private pre-network.

An analysis of this approach has suggested that it would not be suitable for this task due to the latency being too high for the current consensus layer implementation. Each hop between nodes on the pre-network adds 300 ms, with additional 2.5 s for each transaction and approx. 1 s for the RLN zero knowledge proof. This significantly restricts the possible number of hops, making the anonymity guarantee insufficient. Additionally, it has been observed that all messages, not only block proposals, would need to be protected to prevent the IP mapping, and the RLN application may introduce new vulnerabilities [13].

G. Polkadot Sassafras

Sassafras is a proposal that extends the BABE PoS protocol, guaranteeing the production of exactly one block, with constant-time intervals. This improves BABE, where privacy is not provided in the case of filling an empty slot with a deterministically selected validator. The approach combines the usage of VRF, described in Sec. II-D, and anonymity on the network layer.

First, validators obtain their lottery tickets that allow the protocol to distribute the slots at the beginning of each epoch. The BABE protocol provides an on-chain randomness, which is signed by all of the validators using VRF. The VRF output value serves as the lottery ticket. The validity of the tickets is verified by including a SNARK, that proves the ticket owner is among the candidates. in this way, tickets can be validated while their anonymity is preserved. The VRF variation is called ring-VRF.

To privately publish winning tickets, validators do not publish their tickets themselves, instead the tickets get relayed to a randomly selected validator, who serves as a proxy and publishes the tickets on behalf of their owner.

The published tickets are verified using SNARK and sorted using the VRF outputs. The slot is then claimed by the owner, who provides the winning ticket (their VRF output) and a non-anonymous proof that the winning ticket is indeed their VRF output. Due to the ticket system, the anonymity set is the whole validator set [14].

III. Analysis / Comparison

The selected solutions can be compared based on several properties. One of the crucial characteristics is the complexity of the solution, including the initial bootstrapping phase. The solution should not significantly limit the day-to-day functioning (time or performance wise) and possible future improvements, the operations should also not become more complicated from the participants' point of view. Another important consideration is the safety and the resistance to dishonest participants. The solutions provide various anonymity set sizes, with the ideal solution having high enough anonymity set to make an attack on the selected participant very difficult or impossible even when considering malicious participants.

In the existing analyses, it has been discovered that solutions working partially or entirely on the network layer, including the solution based on Dandelion++ and RLN described in Sec. II-F and the Sassafras proposal by Polkadot described in Sec. II-G, are mostly unsuitable for Ethereum due to the high latency and the amount of possible attacks. Implementing these solutions would require significant changes to the Ethereum networking model, primarily for a protocol resembling Sassafras, which requires mapping the validators to their p2p nodes [2]. The main strength of a solution like Sassafras, compared to consensus layer solutions, is not only lower consensus layer complexity, but also the anonymity set, which is the whole validator set. When compared to Whisk, where the anonymity set corresponds to the number of unselected validators from the proposer candidate list.

SnSLE approaches, like the SnSLE Ethereum proposal described in Sec. II-C or Algorand's version described in Sec. II-D, provide a simple way to mitigate these attacks by initially selecting several leaders. These changes would non-negligibly increase the fork-choice complexity, complicate future development and would require combination with another proposal, like the proposer-builder separation, to work safely in Ethereum. Additionally, the increased network traffic would require measures on the network layer, similarly to Algorand's solution. As previously described, network layer solutions introduce additional crucial challenges.

Another consideration for all of the solutions is the bootstrapping phase, which describes the introduction of the chosen measure in the already running ecosystem. This phase may be problematic due to the size of the network and also the possible existing safety issues. Therefore, bootstrapping requires its own research and analyses of the potential approaches. Based on the research of bootstrapping in Whisk, described in Sec. II-A, the safe solutions are generally complex [15]. Except Whisk, no other solutions in Ethereum, as well as Polkadot Sassafras described in Sec. II-G, have not yet defined a safe method for the bootstrapping phase, making the implementation of these methods possibly much more complex.

While proposed SSLE solutions at least partially solve some of these challenges, their anonymity set may be problematic. This is especially the case for the Swap-or-Not SSLE described in Sec. II-B, where the anonymity set is further decreased by introducing malicious validators in the shuffling process. For Whisk, the anonymity set corresponds to the number of the unselected candidates from the candidate list. The set can be further increased by increasing the size of the candidate set, although this may have negative effects on the performance and safety of the approach. This is a disadvantage of the existing SSLE solutions compared to solutions like SnSLE or Polkadot Sassafras, where the anonymity set corresponds to the whole validators set.

The distributed validators approach described in Sec. II-E

is a completely different approach, which would also provide higher consensus decentralization, improved diversity, more secure validator key management and other advantages. It is a complex solution that would require another type of client, in addition to consensus and execution clients, with several implementations to avoid network problems due to client errors or vulnerabilities. It would also create additional costs related to the required higher amount of nodes and possibly introduce increased latency.

IV. Conclusion (10-15 lines)

The survey summarized several approaches to mitigating possible attacks, primarily the DoS attack, on the leader in Ethereum, which is currently known in advance. While all of the solutions are effective, some include disadvantages that can render them difficult or unusable for Ethereum. Before an approach is selected and implemented, further research of all approaches, including their bootstrap phases, is required. The goal is to choose a measure that is effective, safe and provides a sufficient anonymity set without adding excessive complexity to the process.

References

[1] J. Rhea, "Packetology: Validator privacy." [Online]. Available: https://ethresear.ch/t/packetology-validator-privacy/7547

[2] G. Kadianakis, J. Drake, D. Feist, G. Herold, D. Khovratovich, M. Maller, and M. Simkin, "Whisk: A practical shuffle-based ssle protocol for ethereum." [Online]. Available: https://ethresear.ch/t/whisk-a-practical-shuffle-based-ssle-protocol-for-ethereum/11763

[3] D. Boneh, S. Eskandarian, L. Hanzlik, and N. Greco, "Single secret leader election," Cryptology ePrint Archive, Paper 2020/025, 2020, https://eprint.iacr.org/2020/025. [Online]. Available: https://eprint.iacr.org/2020/025

[4] "Curdleproofs: A shuffle argument protocol." [Online]. Available: https://github.com/asn-d6/curdleproofs/blob/main/doc/curdleproofs.pdf

[5] V. Buterin, "Simplified ssle." [Online]. Available: https://ethresear.ch/t/simplified-ssle/12315

[6] D. Khovratovich, "Analysis of swap-or-not ssle proposal." [Online]. Available: https://ethresear.ch/t/analysis-of-swap-or-not-ssle-proposal/12700

[7] V. Buterin, "Secret non-single leader election." [Online]. Available: https://ethresear.ch/t/secret-non-single-leader-election/11789

[8] ——, "State of research: increasing censorship resistance of transactions under proposer/builder separation (pbs)." [Online]. Available: https://notes.ethereum.org/@vbuterin/pbs_censorship_resistance

[9] I. Homoliak, S. Venugopalan, D. Reijsbergen, Q. Hum, R. Schumi, and P. Szalachowski, "The security reference architecture for blockchains: Toward a standardized model for studying vulnerabilities, threats, and defenses," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 1, pp. 341–390, 2020.

[10] S. Micali, S. Vadhan, and M. Rabin, "Verifiable random functions," in *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, ser. FOCS '99. USA: IEEE Computer Society, 1999, p. 120.

[11] A. Asgaonkar, C. Beekhuizen, R. Saltini, and D. Feist, "Ethereum distributed validator specification." [Online]. Available: https://github.com/ethereum/distributed-validator-specs

[12] "Distributed validator technology." [Online]. Available: https://ethereum.org/en/staking/dvt/

[13] "Ethereum consensus layer validator privacy and feasibility analysis using dandelion++ and rln." [Online]. Available: https://www.notion.so/Ethereum-consensus-layer-validator-privacy-and-feasibility-analysis-using-Dandelion-and-RLN-4674432febdc43979a67f043961442e6

[14] J. Burdges, F. Shirazi, A. Stewart, and S. Vasilyev, "Sassafras." [Online]. Available: https://research.web3.foundation/Polkadot/protocols/block-production/SASSAFRAS

[15] dapplion, "Whisk: the bootstrapping problem." [Online]. Available: https://hackmd.io/@dapplion/whisk_bootstrapping