Zero-Knowledge Proof for Attack Prevention in The Ethereum Blockchain

Anders Malta Jakobsen*, Oliver Holmgaard†

Abstract—This is a placeholder abstract test. The whole template is used in semester projects at Aalborg University (AAU).

1 Introduction

TODO

2 Background

In this section, we will go through some of the concepts that will be used in the rest of the paper as well as some surrounding context like attacks performed.

2.1 Ethereum and Proof of Stake

Ethereum is a blockchain platform that allows developers to create decentralized applications using smart contracts. Previously operating with a Proof of Work (PoW) consensus algorithm, Ethereum transitioned to a Proof of Stake (POS) consensus algorithm in 2022. This transition was done to reduce the energy consumption of the network and to increase the scalability of the network. The transition was done in a series of upgrades called the Ethereum 2.0 upgrade. POS is a consensus algorithm that is used to secure blockchain net by helping to create new blocks and confirm transactions. It works by creating validators based on the amount of cryptocurrency they have staked. Then it selects some of these validators as proposers to be the ones to create new blocks, which then are confirmed by the rest of the validators. The proposers then get rewarded for creating a valid block and the validators get rewarded for confirming the block. Block proposing happens within epochs of 32 blocks per epochs and for each epoch a group of validators is selected and from them a proposer is chosen. the way the proposer is chosen is through a random proses that is weighted by the amount of cryptocurrency the validator has staked, and by using the publicly available random number generator (RANDAO) algorithm to simulate the random selection. The blocks also have a time limit of 12 seconds to be created and confirmed else the block will be discarded and the proposer is penalized. If a fork happens the validators have to choose which fork to follow. This is done by using the Latest Message Driven Greedy Heaviest Observed Subtree (LMD-GHOST) algorithm which chooses the fork with the greatest weight of attestations in its history [1].

- *All authors are affiliated with the Dept. of Computer Science, Aalborg University, Aalborg, Denmark*
- *E-mails: *amja23, †oholmg20 @student.aau.dk*

2.2 Proposer DoS Attack

In this subsection, we will be describing the attack that we will be using as a basis for our experiment in section 4. The attack is a Denial-of-Service (DoS) attack that aims at hitting the proposers selected for creating blocks in the chain. Ethereum themselves have mentioned it as a potential attack, and with the current implementation of the consensus algorithm, it seems that this attack is possible to perform [2, 3].

It has been our interest to research the feasibility of this attack and the ones mentioned in section A. This has proven to be a difficult task, given that most of our researched attacks happen in the consensus- or execution layer. Therefore, as a result of the blockchain algorithm, we are not able to clarify the feasibility of the attacks that we have found. For this reason, we have chosen the *Proposer DoS attack* as it seems exciting, has not been mitigated yet, and a potential solutions seems to include a Zero-Knowledge Proof (ZKP).

Does it make sense to mention # skipped blocks pr day even though we are unsure if any of these are attacks

The attack possible is because the consensus mechanism uses a publicly known function for choosing the upcoming block proposers. The adversary is therefore able to compute this in slight advance of the blockchain, s.t. each proposer is now known. After this, the adversary can map the proposer's IP addresses and overload their connection. A successful attack would leave a proposer unable to propose their block in time.

Should possibly be explained in more detail

3 Related Work

The usage of ZKPs in Ethereum is not a new concept. In fact, it currently uses them both on- and off-chain. The following provides a short overview of some of the already existing solutions as well as one still being in development.

3.1 dos attack

Some of the instances of dos attacks that is seen on ethereum ranges from attacks on the proposers to attacks that seek to slow down the network itself. A known attack aims to slow down the network by using underpriced opcodes to create a block that is hard to process [4, 5]. Another way to slow down the network is to create empty accounts that are hard to process [6, 7]. This attack however is outdated and has been mitigated by making at near impossible to create empty accounts in the network.

4 Experimental protocol

4.1 ENR

A Ethernet node record (ENR) is a record that contains information about a node in the network [8]. Ethereum uses ENRss as a way to package the information that is being sent from node to node during the discovery protocol, where nodes discover each other. The package contains information like the node's IP address, port, and public key. Because of the nature of the discovery protocol, if you where to also be a node in Ethereum, you would be able to see the ENR of all the nodes that you have discovered. And since the ENR contains the IP address and the public key of the node, you would be able to see the corresponding IP addresses and public keys of all the nodes that has been discovered by the node.

How do we simulate the attack? What variables do we want to control? Severity of attack - Slow down node or make node disappear?

THOUGHT: I think it would make sense to test both a slow node, and a frozen/dead node. Linux kernel Traffic Control (tc) and Netem (part of tc) seems like the best choice for simulating network issues.

5 Discussion

This is a potential discussion section.

6 Conclusion

This is a potential conclusion section.

References

[1] ethereum.org, "Proof-of-stake (pos)," 2024, Accessed: 23-10-2024.

[2] ethereum.org, "Secret leader election," 2024, Accessed: 22-10-2024.

[3] ethereum.org, "Ethereum proof-of-stake attack and defense," 2024, Accessed: 22-10-2024.

[4] H. Chen, M. Pendleton, L. Njilla, and S. Xu, "A survey on ethereum systems security: Vulnerabilities, attacks, and defenses," *ACM Comput. Surv.*, vol. 53, no. 3, Jun. 2020, ISSN: 0360-0300. DOI: 10.1145/3391195.

[5] A. H. H. Kabla, M. Anbar, S. Manickam, T. A. Al-Amiedy, P. B. Cruspe, A. K. Al-Ani, and S. Karuppayah, "Applicability of intrusion detection system on ethereum attacks: A comprehensive review," *IEEE Access*, vol. 10, pp. 71 632–71 655, 2022. DOI: 10.1109/ACCESS.2022.3188637.

[6] V. Buterin, "A state clearing faq," November 2016, Accessed: 18-10-2024.

[7] G. Wood, "Eip-161: State trie clearing (invariant-preserving alternative)," *Ethereum Improvement Proposals*, no. 161, October 2016, Accessed: 18-10-2024.

[8] F. Lange, "Eip-778: Ethereum node records (enr)," *Ethereum Improvement Proposals*, no. 778, November 2017, Accessed: 15-11-2024.

[9] R. Sarenche, E. N. Tas, B. Monnot, C. Schwarz-Schilling, and B. Preneel, "Breaking the balance of power: Commitment attacks on ethereum's reward mechanism," 2024. arXiv: 2407.19479 [cs.CR].

[10] 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, pp. 560–576, ISBN: 978-3-031-18283-9.

[11] V. Buterin, "Eip-150: Gas cost changes for io-heavy operations," *Ethereum Improvement Proposals*, no. 150, September 2016, Accessed: 18-10-2024.

[12] V. Buterin, "Secret non-single leader election," 2024, Accessed: 22-10-2024.

[13] G. Kadianakis, "Whisk: A practical shuffle-based ssle protocol for ethereum," 2024, Accessed: 22-10-2024.

[14] 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, Los Angeles, CA, USA: Association for Computing Machinery, 2022, pp. 43–52, ISBN: 9781450398794. DOI: 10.1145/3560829.3563560.

[15] 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. DOI: 10.1109/DSN58291.2024.00020.

[16] ethereum.org, "What are zero-knowledge proofs?," 2024, Accessed: 15-10-2024.

[17] K. Charbonnet, "A technical introduction to maci 1.0," 2022, Accessed: 15-10-2024.

[18] M. Zhang, R. Li, and S. Duan, *Max attestation matters: Making honest parties lose their incentives in ethereum PoS*, Cryptology ePrint Archive, Paper 2023/1622, 2023.

Appendix A
Attacks on Ethereum

In the following section is an explanation of different attacks that either can or could happen to the ethereum blockchain. There is a description of both the attack and its possible mitigation. If found, there will be a link to a Proof-of-Concept (PoC) of the attack and/or the proposed/implemented mitigation.

A.1 Reorg

One of the reorganization attacks is *Commitment Attacks on LMD GHOST* [9]. It works by using financial incentives to convince validators to vote for a prior block in the chain, by saying it is the block you will be commiting to, to try and exploit there laziness. Possible because around 90% of validators use software called ”MEV Boost”, which tries to earn you the most money.

Another reorg attack is Short-range reorg, which uses short-range reorgs of the blockchain stipulating consensus to delay the finality of consensus decisions [10]. Such short-range reorgs also allow validators to increase their earnings from participating in the protocol. It does this by withholding a block and then releases it timed with the next honest block to orphan it. This attack requires a large amount of stake to be held by the adversary, with 30% being the aim and everything below only reducing the chance of success.

A reorg attack that has been inspired by the last attack is Low cost long-range reorg attack[10]. This attack works by the adversary avoids competing directly with honest validators of $(k-1)$ committees, as done in the short-range reorg attack. Instead, the adversary uses the technique of balancing attacks to keep honest committee members split roughly in half by ensuring they have different views on what the current head of the chain is. This makes honest nodes work against each other to maintain a tie which the adversary can tip to their liking at any point using only a few votes. A possible mitigation is the recently proposed Transaction Encapsulation approach demonstrates that some cross-layer attacks can be alleviated or prevented by dynamic transaction .

A different reorg attack is called $>= 50\%$ stake attack [3]. The attack works if the adversary has over 50% stake, the adversary would be able to have the majority vote every time in the fork choice algorithm. This would mean that all honest validators need to vote along with the adversary to not get their ETH burned due to the inactivity leak security protocol. Because of said protocol, the adversary would also eventually gain finalization. This attack also makes it possible for the adversary to easily perform a reorg of a block. It is thought that given the large amount of stake needed for the attack to work, this attack would not be feasible.

A.2 DoS

We’ve found three different kinds of DoS attacks that either were or are possible to perform on Ethereum.

One of the attacks is called *under-priced opcodes* [4, 5]. This attack works because Ethereum has a gas mechanism to reduce abuse of computing resources. Though when a contract has a lot of underpriced opcodes, they will consume many resources. Execution of contracts requires a lot of resources.

To mitigate this, Ethereum has raised the gas cost of opcodes to preserve the number of transactions-per-second [11] [1].

Another attack, which is closely related to the former, is *empty account in the state trie* [4, 5]. This attack was possible because the existence of empty accounts increases the transaction processing time and synchronization. An empty account is an account with zero balance and no code. The attack required the proposer to select only the transactions of the adversary, which could be insured by offering a higher gas price.

The mitigation is a combination of the one explained for *under-priced opcodes* as well as a mitigation for clearing empty accounts [6, 7, 11] [2].

The last example of a DoS attack is called *Proposer DoS* [2, 3]. The background to making this attack

1. EIPs/EIPS/eip-150.md at master ethereum/EIPs GitHub
2. EIPs/EIPS/eip-161.md at master ethereum/EIPs GitHub

possible is that the consensus mechanism uses a publicly known function for choosing the upcoming block proposers. The adversary is therefore able to compute this in slight advance of the blockchain, s.t. each proposer is now known. After this, the adversary can map the proposer's IP addresses and overload their connection. A successful attack would leave a proposer unable to propose their block in time.

To prevent this kind of attack, Ethereum plans to use something they call Single Secret Leader Election (SSLE) which ensures that only the selected validator knows that they have been selected [2, 12].

Specifically, a proposal has been made to use an election protocol called Whisk, which is a type of SSLE [13][3][4][5]. It works by each validator submitting a commitment to a secret shared by all validators. The commitments are shuffled s.t. no-one can map commitments to the validators, but each validator knows what commitment belongs to them. This shuffle-phase goes on for a day, 256 epochs, before using the shuffled proposer list the following day. Commitments are chosen at random, and the selected proposer will detect its commitment to know when to propose a block.

The shuffling phase requires validators to occasionally shuffle a subset of candidate proposers. Using a subset is a measure to reduce computation for the validators, as 256 epochs correspond to 8912 proposers. The shuffle requires a validator to construct a ZKP to confirm that the shuffle was performed correctly.

A.3 Balancing Attack

For this type, we found two attacks.

The first attack we call *LMD-specific balancing attack* [14]. This attack exploits the Latest Message Driven (LMD) *proposer boosting* by sending out two competing blocks but giving half the validators one block before the other and the opposite for the other half. This would create a fork in the blockchain.

Although no mitigation is mentioned, it requires $W_p/b + 1$ adversarial slots [6].

Another balancing attack has the adversary exploiting adversarial network delay and strategic voting by a vanishing fraction of adversarial validators to stall the protocol indefinitely [10] [7].

Though this attack does depend on networking assumptions that are highly contrived in practice; those being the attacker having fine-grained control over latencies of individual validators.

A.4 Finality Attack (Bouncing Attack)

For the Ethereum, there exist attacks called finality attacks, also known as bouncing attacks, which have the purpose of denying the blockchain to finalize its block, halting its functionality.

The first attack is a *double finality* attack [3, 15]. It is theoretically possible for an attacker that wants to risk 34% of the total staked ether. Two forks finalize simultaneously, creating a permanent split of the chain.

A way to see a mitigation of this is that it is practically impossible given the current value of 34% of the total staked ether. Also, voting on two different chains, called double voting, is a slashable offense in the Ethereum chain, so the adversary would get their staked ETH slashed.

Another finality attack is called *33% finality attack* [3]. Here, all adversarial ($\geq 33\%$) validators can simply go inactive, meaning that a block cannot get 2/3 attestations which is required to achieve finality.

3. Shuffle_SSLE/rust_code/src at master ethresearch/Shuffle_SSLE GitHub

4. [WIP] Introduce consensus code for Whisk (SSLE) by asn-d6 Pull Request #2800 ethereum/consensus-specs GitHub

5. GitHub - dapplion/lighthouse at whisk

6. where W_p is the proposer boost weight (fx 100 validators/slot: $W_p = 0.7 \cdot 100 = 70$) and b is the fraction of adversaries in the committee in each slot

7. GitHub - tse-group/gasper-gossip-attack

Therefore, the blockchain would not be able to go further, as it would not be able to achieve finality.

The mitigation for this is called *the inactivity leak*. The Ethereum chain penalizes the validators who resist voting or are inactive. Their ETH gets burned until the majority vote has a 2/3 majority. This makes the attack impractical for the attacker, as it is costly to have $\geq 33\%$ of the total staked ETH and their ETH gets burned as well.

A.5 Avalanche Attack

This type only includes a single attack that we call *avalanche attack on proof-of-stake ghost* [14] [8]. The attack uses withheld blocks to make wide subtrees to displace an honest chain. It does this by exploiting the reuse of uncle blocks.

The mitigation for this attack is already a part of the Ethereum blockchain [9]. It is mitigated by LMD, which works together with Greedy Heaviest Observed Subtree (GHOST). The protocol only counts a vote from a validator if the vote is strictly later than the current entry. If two equivocating votes were sent from the same validator at the same time slot, only the earlier message would be counted.

A.6 Bribery

An adversary using bribery attacks on the Ethereum chain could be interested in dictating a choice in some sort of voting mechanism. This is exactly what happens in a described *quadratic funding* attack [16]. The adversary researches votes on the chain and bribes users to vote for what the adversary wants.

In defense of a potential bribery attack, the Ethereum blockchain implements a private voting system called Minimum Anti-Collusion Infrastructure (MACI) [16, 17] [10].

What MACI does is essentially hiding what each person has voted for. It does so by demanding the voters to send their votes encrypted to a central coordinator. This coordinator constructs Zero-Knowledge Succinct Non-Interactive Argument of Knowledge (ZK-SNARK) proofs, which verifies that all messages were processed correctly, and that the final result corresponds to the sum of all valid votes.

As votes are now hidden, the adversary is not able, by oneself, to prove that the bribee voted in way of said bribery. Though the bribee could decrypt their own message and show the vote to the adversary.

MACI has fixed this problem by implementing public key switching. This means that a voter can request a new public key. In addition to this, a vote is only valid if it uses the most recent public key of the voter. Therefore, a bribee can show its first vote obeying the adversary, generate a new public key, and send a new, now honest, vote. The old vote will then become invalid as it uses a deprecated public key.

A.7 Staircase Attack

The last type of attack that we cover is called a *staircase attack*. This is a two-part attack, with a warm-up attack and a full attack [18] [11].

First, we will cover the *warm-up attack*. The attack works when the adversary, called a, has to propose the first block of an epoch. From there, the attack works in four parts:

1) The adversary withholds its block at slot t
2) The honest attestors in slot t create attestation with the last block of the previous epoch, called b.

8. GitHub - tse-group/pos-ghost-attack

9. Proposer LMD Score Boosting by adiasg Pull Request #2730 ethereum/consensus-specs GitHub

10. GitHub - privacy-scaling-explorations/maci: Minimal Anti-Collusion Infrastructure (MACI)

11. GitHub - tsinghua-cel/Staircase-Attack: This is the staircase attack implement for the paper ”Max Attestation Matters: Making Honest Parties Lose Their Incentives in Ethereum PoS.”

3) Adversary a releases block b_t and honest validators update their checkpoint to block b_t.
4) Attestations with targets different from b_t will be discarded, and honest attestors in slot t will be penalized eventually

This attack is mitigated by what Ethereum calls *honest reorg*, which should prevent intentionally withheld blocks [18].

A block proposed in slot t with fewer than 20% attestations is considered invalid. Though it can be theoretically avoided with network timing, ensuring at least 20% of the attestors get the block.

With the *warm-up attack*, the author goes on to describe a full *staircase attack*. The attack starts with the *warm-up attack*. Then the plan is to manipulate the source of the honest validators. Half the validators should get an outdated *last justified checkpoint* and be penalized. All byzantine validators withhold their blocks and publish them in the middle of the epoch. It should be able to be done as a one-time attack with probability 98.84% if adversary controls N/3 validators [18]. The byzantine validators don't get penalized, but will get a smaller reward than the fair share.

What would make this attack infeasible is that it would require a lot of controlled validators. The author states that for Ethereum to be vulnerable, there would need to be < 16384 validators, which makes the attack as good as impossible given Ethereum has $1.073.406$ daily active validators [18] [12]. Also, Ethereum released a patch fixing this attack after the author pointed it out [13].

12. According to beaconcha.in as of 22-10-2024

13. Confirmation Rule by saltiniroberto Pull Request #3339 ethereum/consensus-specs GitHub