Authors: Joachim Neu, Ertem Nusret Tas, David Tse

Special thanks to Danny Ryan and Aditya Asgaonkar for feedback and discussions.

This attack was subsequently published in:"Two More Attacks on Proof-of-Stake GHOST/Ethereum"

TL;DR:

We describe a generic attack on PoS GHOST variants. This points to conceptual issues with the combination of PoS and GHOST. PoS Ethereum as-is is not susceptible to this attack (due to LMD, which comes with its own problems, see here). Still, we think this attack can inform the fork choice design and help projects that (consider to) use similar consensus protocols.

We assume basic familiarity with GHOST (seeGasper and the GHOST paper). Knowledge of the beacon chainfork choice specification and earlier attacks won't hurt, either.

High Level

The Avalanche Attack on PoS (Proof-of-Stake) GHOST (Greedy Heaviest Observed Sub-Tree) combines selfish mining

with equivocations

. The adversary uses withheld blocks to displace an honest chain once it catches up in sub-tree weight with the number of withheld adversarial blocks. The withheld blocks are released in a flat but wide sub-tree, exploiting the fact that under the GHOST rule such a sub-tree can displace a long chain. Only two withheld blocks enter the canonical chain permanently, while the other withheld blocks can subsequently be reused (through equivocations) to build further sub-trees to displace even more honest blocks. The attack exploits a specific weakness of the GHOST rule in combination with equivocations from PoS, namely that an adversary can reuse 'uncle blocks' in GHOST, and thus such equivocations contribute to the weight of multiple ancestors. Formal security proof of PoS GHOST seems doomed.

A proof-of-concept implementation for vanilla PoS GHOST and Committee-GHOST is providedhere. By "vanilla PoS GHOST" we mean a one-to-one translation of GHOST from proof-of-work lotteries to proof-of-stake lotteries. In that case, every block comes with unit weight. By "Committee-GHOST" we mean a vote-based variant of GHOST as used in PoS Ethereum, where block weight is determined by blocks (and potentially a proposal boost). Subsequently, we first illustrate the attack with an example, then provide a more detailed description, and finally show plots produced by the proof-of-concept implementation.

A Simple Example

We illustrate the attack using a slightly simplified example where the adversary starts with k=6

withheld blocks and does not gain any new blocks during the attack. In this case, the attack eventually runs out of steam and stops. (In reality, the larger the number of withheld blocks, the more likely the attack continues practically forever, and even for low k

that probability is not negligible.) Still, the example illustrates that k=6

blocks are enough for the adversary to displace 12

honest blocks—not a good sign.

First, the adversary withholds its flat-but-wide sub-tree of k=6

withheld blocks, while honest nodes produce a chain. (Green/red indicate honest/adversarial blocks, and the numbers on blocks indicate which block production opportunity of honest/adversary they correspond to.)

Figure 1:

Once honest nodes reach a chain of length k=6

, the adversary releases the withheld blocks, and displaces the honest chain.

Figure 2:

Note that the adversary can reuse blocks 3, 4, 5, 6. Honest nodes build a new chain on top of 2 → 1 → Genesis. Once that new chain reaches length 4

, the adversary releases another displacing sub-tree.

Figure 3:

Finally, note the adversary can reuse blocks 5, 6. Honest nodes build a new chain on top of 4 → 3 → 2 → 1 → Genesis. Once the new chain reaches length 2

, the adversary releases the last displacing sub-tree.

Figure 4:

Honest nodes now build on 6 → 5 → 4 → 3 → 2 → 1 → Genesis. All honest nodes so far have been displaced. Overall, the adversary gets to displace O(k^2)

honest blocks with k

withheld adversarial blocks.

Attack Details

Selfish mining and equivocations can be used to attack PoS GHOST (using an ‘avalanche of equivocating sub-trees rolling over honest chains’—hence the name of the attack). The following description is for vanilla PoS GHOST, but can be straightforwardly translated for Committee-GHOST. Variants of this attack work for Committee-GHOST with Proposal Weights as well.

Suppose an adversary gets k

block production opportunities in a row, for modest k

. The adversary withholds these k

blocks, as in selfish mining

(cf Figure 1 above). On average, more honest blocks are produced than adversary blocks, so the developing honest chain eventually ‘catches up’ with the k

withheld adversarial blocks.

In that moment, the adversary releases the k

withheld blocks. However, not on a competing adversarial chain (as in selfish mining for a Longest Chain protocol), but on a competing adversarial sub-tree of height 2, where all but the first withheld block are siblings, and children of the first withheld block. Due to the GHOST weight counting, this adversarial sub-tree is now of equal weight as the honest chain—so the honest chain is abandoned (cf Figure 2 above).

At the same time, ties are broken such that honest nodes from now on build on what was the second withheld block. This is crucial, as it allows the adversary to reuse in the form of equivocations

the withheld blocks 3, 4, …, k

on top of the chain Genesis → 1 → 2 formed by the first two withheld adversarial blocks, which is now the chain adopted by honest nodes.

As an overall result of the attack so far, the adversary started with k

withheld blocks, has used those to displace k

honest blocks, and is now left with equivocating copies of k-2

adversarial withheld blocks that it can still reuse through equivocations (cf Figure 3 above). In addition, while the k

honest blocks were produced, the adversary probably had a few block production opportunities of its own, which get added to the pool of adversarial withheld blocks. (Note that the attack has renewed in favor of the adversary if the adversary had two new block production opportunities, making up for the two adversarial withheld blocks lost because they cannot be reused.)

The process now repeats (cf Figure 4 above): The adversary has a bunch withheld blocks; whenever honest nodes have built a chain of weight equal to the withheld blocks, then the adversary releases a competing sub-tree of height 2; the chain made up from the first two released withheld blocks is adopted by honest nodes, the other block production opportunities can still be reused in the future through equivocations on top of it and thus remain in the pool of withheld blocks of the adversary.

If the adversary starts out with enough withheld blocks k

, and adversarial stake is not too small, then the adversary gains 2 block production opportunities during the production of the k

honest blocks that will be displaced subsequently, and the process renews (or even drifts in favor of the adversary). No honest blocks enter the canonical chain permanently.

Proof-of-Concept Implementation Results

For illustration purposes, we plot a snapshot of the block tree (adversarial blocks: red, honest blocks: green) resulting after 100 time slots in our proof-of-concept implementation. The attack is still ongoing thereafter, and as long as the attack is sustained, no honest blocks remain in the canonical chain permanently.

PoS GHOST

- Adversarial stake: 30%
- Initially withheld adversarial blocks: 4

[

2823×3048 563 KB

](https://ethresear.ch/uploads/default/original/2X/4/474a4c5fe8385ec20af1eb309561cdf00653ad19.png)

Committee-GHOST

- Adversarial stake: 20%
- Initially withheld adversarial blocks: 12

[

7377×2114 997 KB

](https://ethresear.ch/uploads/default/original/2X/a/a6f1b271b62c5cb9222ee8b5721acc31fe9c5ce5.jpeg)

Applicability to PoS Ethereum

PoS Ethereum's LMD (Latest Message Driven) aspect interferes with this attack, but comes with its own problems, see here.