TL;DR

We present an attack on LMD GHOST called “decoy-flip-flop” attack, by which an adversary can delay the finalization for a few hours ~ days by leveraging a network failure.

This attack does not break the basic security of ETH2.0 but implies some manipulability of LMD GHOST.

(Prerequisites: Vitalik’s blog post on LMD GHOST)

UPDATE: This research is published as a peer-reviewed paper on IEEE Access (“Impact of Saving Attacks on Blockchain Consensus”).

Setup: LMD toy consensus

To make the analysis simple, we discuss a binary consensus where honest validators try to make a consensus between two colors (RED and BLUE).

In a variant of GHOST fork-choice rule, this can be thought of as a consensus between blocks at the same height or conflicting subtrees.

- There are n

validators, assume homogeneous weight (stake) for simplicity

- The time is divided into slots and epochs
- 64 slots = 1 epoch
- 64 slots = 1 epoch
- Each validator is allocated into a slot for every epoch and publishes a vote in the slot
- The slot allocation for an epoch is completed before the epoch starts
- Assume unbiasable randomness
- The slot allocation for an epoch is completed before the epoch starts
- Assume unbiasable randomness
- Honest validators vote for the color with a greater score
- The score is the number of validators who vote for the color in their latest message
- When there is a tie, voters can choose one arbitrarily.
- The score is the number of validators who vote for the color in their latest message
- When there is a tie, voters can choose one arbitrarily.

The voting rule is fundamentally the same as LMD GHOST.

Also, the slot-based voting protocol can be considered as a simplified version of ETH2.0 beacon chain.

(This is similar to “divergence game” but it’s more tricky than the games in the longest-chain rule or original (non-LMD) GHOST because scores do not grow monotonically).

Network and adversary

We consider a game between honest validators and adversary who try to prevent the consensus.

- Eventual synchrony: After some time T

, the network becomes fully synchronous i.e. a message sent by an honest validator is received by any other honest validators within 1 slot. * We assume that at T

, the winner has not satisfied the convergence condition yet.

- In practice, this is a case when the chain does not converge by an accidental or adversary-driven network failure which lasts until T
- We assume that at T

, the winner has not satisfied the convergence condition yet.

- In practice, this is a case when the chain does not converge by an accidental or adversary-driven network failure which lasts until T
- Adversary controls less than 1/3

of the total validators (by ratio).

- Rushing adversary: Adversary can receive all messages allocated at the slot and deliver messages to all honest validators before the next slot.
- In practice, for instance, adversary (and also honest validators) have this ability if the network is strongly synchronous i.e. message delay is sufficiently smaller than 1 slot
- In practice, for instance, adversary (and also honest validators) have this ability if the network is strongly synchronous i.e. message delay is sufficiently smaller than 1 slot

In full synchrony, this voting game (and fundamentally, LMD GHOST) has a convergence condition

i.e. if the set of honest validators whose latest votes support for the current winner (i.e. the color with the higher score) is larger than the half of the whole validator set, adversary cannot change the winner and hence the rest of the honest validators will also vote for the winner.

Decoy-flip-flop

We consider an adversary's strategy called decoy-flip-flop

.

Specifically,

1. The adversary keeps waiting for honest validators to vote for the winner as long as he knows the current winner does not satisfy the convergence condition after the next slot.
2. Otherwise, the adversary pivots

i.e. switches all of his votes to the loser to change the winner.

1. Continue from 1.

In 2, the adversary decides whether to pivot or not before the next slot starts after seeing the result of the current slot.

The exact condition of pivoting is:

(current honest votes of the winner) + (maximum increase of honest votes of the winner in the next slot)

\gt \lceil n/2 \rceil - 1

.

Here (maximum increase of honest votes of the winner in the next slot)

is upper bounded by the number of honest voters allocated to the next slot. If the next voters are public, this is further bounded by the number of next honest voters whose current latest votes are supporting the loser.

Savings

We assume that the adversary has earned savings

before T

i.e. adversary keeps being silent for e.g. a few epochs and use the rights to vote in the skipped slots (= savings) later for the decoy-flip-flop attack.

Also, since LMD GHOST only considers the latest messages, the adversary spends his savings from older slots during the continuous flip-flopping.

Simulation

I implemented the simulation of this attack in Python (GitHub Repo).

First, I did a simulation with an adversary controlling 33% of the whole validator set assuming a pubic allocation of voters.

Here is an animation of this attack.

[

image

914×542 16.8 KB

](https://ethresear.ch/uploads/default/original/2X/d/d5a2cf600b07b96c744e5cfd660f54ab8b93b70d.png)

This is the number of adversary's pivots in every epoch.

We can see (and will easily prove) that adversary must pivot for at least once per epoch.

Therefore, although new slots are allocated to adversary every epoch even during the attack, the adversary ends up publishing votes from all the slots he is allocated and hence this attack does not continue forever.

To keep flip-flopping, the adversary must prepare sufficient savings.

The next result shows the number of epochs (x-axis) for which the adversary can delay the finalization by savings he prepared (y-axis, the number of savings is measured by the number of epochs the adversary has skipped before T

).

This implies that for 33% attacker, 1 epoch (6.4 minutes) of saving is sufficient to delay the convergence for more than 15 epoch (= 1.6 hours) on average.

Also, we can see that the average number of saving which the adversary need to spend per epoch exponentially decreases.

In addition to these, we can observe that neither the initial state nor the total number of validators does not make a difference.

Conclusions

In summary, an adversary can extend the delay of the convergence even after a network failure, by publishing votes from the slots he skipped in the network failure.

Unlike the liveness attack on FFG by simply being silent, the adversary does not necessarily require 1/3 of the total stake.

Inactivity leak

disincentivizes these liveness attacks since the adversary's deposit is eventually slashed.

The adversary must continue to corrupt validators to keep his power and the cost of this attack increases quadratically over time.

(E.g. With the current parametrization, 33% attacker can delay the finality for a few days with the cost of 1 percent of his stake.)

In a theory of BFT consensus, this attack is more problematic since the adversary is not assumed to be rational.

CBC Casper or CBC-style enforcement of the fork-choice probably mitigate this kind of attack since an adversary must provide the appropriate evidence ("justification") to switch chains, making it difficult to prepare for this attack.

N.B. This post & simulation has not been reviewed! I'll appreciate any feedback/comments.