Originally posted on tzionis.com.

A few weeks ago, I've read Vitalik's post on Single Slot Finality (SSF) and left both excited and a little disheartened. Solving fast finality with as many validators as Ethereum has is very interesting but at the same time it might be compromising either the security of the network or the 32ETH deposit and hence decentralization that already felt high to me.

I kept thinking about it since and came up with the following "scheme" that aims to give a third solution to the problem of effectively capping validator count for faster finality.

Might be worth mentioning that I haven't worked with the Ethereum consensus layer before and my experience spans other parts of the protocol.

Single Slot Finality

Single Slot Finality aims to solve a few issues the current consensus of Ethereum has such as slow finality for transactions (~15m), re-org incentives for MEV bots and other malicious actors and protocol complexity / interaction bugs between LMD-Ghost and Gasper FFG.

It does that by making Ethereum consensus closer to BFT and utilizing the ability to finalize the chain in a single slot if there are enough attestations. This new consensus could still support more validators than other BFT-like systems using efficiency of BLS signatures and won't halt if >33% are offline but would enter an inactivity leak state similar to the current consensus of Ethereum.

There's one catch, the new consensus would still require reducing the validator count from hundred of thousands to tens of thousands to efficiently operate. within the bounds of one slot.

The HackMD post discusses two ideas to get there.

Ideas from the HackMD post

Idea 1: Super-committees

Randomly select a super-committee of tens of thousands of participants for each slot.

Instead of all validators participating in each Casper FFG round, only a medium-sized super-committee of a few tens of thousands participates, allowing each round of consensus to happen within a single slot.

Super-committees allow us to not put further economic restrictions for validators but their drawback is that they reduce the total amount of ETH required to attack Ethereum as each committee has a fixed number of validators.

For example, 62.5K validators per super-committee means 1M ETH at stake and 2M ETH security as 62.5 * 32 ETH = 2M ETH which is much lower that the current economic security of Ethereum ie. ~900K * 32 ETH = 28.8M ETH.

Idea 2: Validator set size capping

Cap the total ETH deposited or the validator count.

We could try to adopt either (or both) of two kinds of caps:

1. Capping the total ETH deposited
2. Capping the total validator count

Either cap could be implemented either with an order-based mechanism (stack or queue) or with economic mechanisms.

Validator set size capping allows to not reduce the amount of ETH required to attack Ethereum but require either increasing the required amount for staking or alter the permission-less of Ethereum. Increasing the stake could also cause griefing attacks where big validators split their stake to push smaller validators out.

Blending the two ideas

The proposal is to have a one committee for each discrete amount of ETH deposit. Committees will have different chances to be selected to produce the next block based on the amount of ETH deposit (eg. the 64 ETH committee will have double the chances of the 32 ETH committee). Each committee should randomly select validators that the total number of validators for all committees is fixed (eg. 10 committees with 8K validators each).

Proposal

Allow staking with a discrete deposits D_1,...,D_n

, where D_i = 2^i

for i=m,..,n

. For example, if m=1

and n=10

the deposit values will be 2,4,…128, 512 and 1024 ETH.

Each deposit committee will have a fixed number of validators K_{committee}=K

. Since there are n

committees, the total number of validators will be K_{total}=(n-m+1)*K_{committee} = (n-m+1)*K*

. If there are more validators in any deposit committee the participating validators will be randomly selected.

In each slot, every committee votes and when the aggregated votes are weighted based on their deposit value ie. V_{weighted_votes}= V_{committee_votes}*D_i

. Vote aggregation can still utilize BLS signatures for efficient. There's just n-m+1

extra multiplications at the end of each slot to calculate aggregate vote across the all committees.

Given that validators now have uneven deposit values, there should be an adjustment to the block production selection to account for the lower and higher validator deposits. The probability each committee produces a new block is the adjusted by the committee deposit and the total deposits of all committees. Each committee has a chance of P_{committee}

to produce a new block and each individual validator has a chance of P_{validator}

.

The P_{committee}

and P_{validator}

are given by:

P_{committee} = \dfrac{D_{i}}{S_{total}} = \dfrac{2^{i}}{\sum_{i=m}^{n} 2^i} = \dfrac{2^i}{2^{n+1} - 2^{m+1}}

P_{validator} = \dfrac{P_{committee}}{K_{committee}} ≈ \dfrac{P_{committee}}{K}

where

i

is the index of the committee and m<i<n

S_{total}

is the total deposit given by \sum_{i=m}^{n} 2^i = 2^{n+1} - 2^{m+1}

K_{committee}

is the committee validator count and should approach K

The total amount of ETH T_{stake}

deposited in all committees is \sum_{i=m}^{n} 2^i * K = K * (2^{n+1} - 2^{m+1})

ETH.

Picking the parameters

Deciding the maximum index n

for fixed m = 1

and K=8k

:

n=8

\implies

T_{stake}=4.06M

ETH, P_{min}=0.19\%

and P_{max}=50.2\%

n=10

\implies

T_{stake}=16.3M

ETH, P_{min}=0.05\%

and P_{max}=50.0\%

n=12

\implies

T_{stake}=65.5M

ETH, P_{min}=0.01\%

and P_{max}=50.0\%

Deciding the minimum index m

for fixed n = 12

and K=8k

m=1

\implies

T_{stake}=65.5M

ETH, P_{min}=0.01\%

and P_{max}=50.0\%

m=3

\implies

T_{stake}=65.4M

ETH, P_{min}=0.08\%

and P_{max}=50.1\%

m=5

\implies

T_{stake}=65.0M

ETH, P_{min}=0.38\%

and P_{max}=50.4\%

Deciding committee validator count K

for fixed m = 1

and n=12

K=6k

\implies

T_{stake}=49.1M

ETH, P_{min}=0.01\%

and P_{max}=50.0\%

K=8k

\implies

T_{stake}=65.5M

ETH, P_{min}=0.01\%

and P_{max}=50.0\%

K=12k

\implies

T_{stake}=98.2M

ETH, P_{min}=0.01\%

and P_{max}=50.0\%

Note:

P_{min}

and P_{max}

are the probabilities the smallest and largest committees produces a block respectively.

Source code for calculation:

const m = 1, n = 12, k = 8_000

const Tstake = k * (2**(n+1) - 2**(m+1)) const Pmin = (2**m - 1) / (2**(n+1) - 2**(m+1)) const Pmax = (2**n - 1) / (2**(n+1) - 2**(m+1)) console.log(Tstake, Pmin, Pmax)

A numerical example

Let’s take m=1

, n=12

and K=8k

as an example. To simplify the maths I’ll only consider a perfectly balanced system where each stake value is filled with the exact number of validators it’s cap is but that definitely won’t be the case in practice.

In this example, there would be 8k validators at 1 ETH, 8k validators at 2 ETH and so on until 4096 ETH. The total number of validators is K=8k*12= 96k

.

As mentioned before, to calculate the block production chances for each committee we need to calculate the adjusted probability each committee has based on their stake. So the sum of 1 + 2 + ... + 4096 = 8191

. That means that the 1 ETH committee has a probability of \dfrac{1}{8191} \approx 0.012\%

to produce a block and the 4096 ETH committee has a probability of \dfrac{4096}{8191} \approx 50.0\%

.

To get the probability of a single validator in the committee producing a block we just have to divide that committee probability with the committee validator count ie. 8k

. This means 1 ETH validators has a chance of \dfrac{0.012\%}{8k} \approx 0.0000015\%

of producing a block that turns out to be years and 4096 ETH validators has a chance of \dfrac{50\%}{8k} \approx 0.00625\%

which is a few days.

In this configuration the total amount of ETH stake is 8k*(1 + 2 + ... + 4096) = 8k*(8191) = 65.5M

ETH.

Open Questions

1. Picking the correct implementation parameters ie. the validator count cap per committee K

and the range of discrete deposit values m

and n

?

1. Investigating the dynamics when smaller deposit values get filled out? Are griefing attacks still meaningful and should be protected against?

References

[1] https://notes.ethereum.org/@vbuterin/single_slot_finality

[2] https://eth2book.info/capella/part2/building_blocks/committees/

[3] https://github.com/ethereum/consensus-specs