TLDR

: By combining data availability sampling and proofs of custody we show how to make unavailability faults mass-slashable without social slashing. (This is similar to FFG safety faults where 1/3 of validators are slashable without social slashing.) We also address the data availability sampling validator dilemma.

Construction

First require that validator chunk queries as part of data availability sampling are done using deterministic randomness instead of non-deterministic local randomness. (Deterministic randomness can be achieved using BLS signatures as a VRF similarly to RANDAO mix shares in phase 0.) Next, add a proof of custody to the chunks. For example, a chunk_custody_bit

which covers all the data availability sampling chunks between source

and target

is added to [AttestationData

](https://github.com/ethereum/eth2.0-specs/blob/0f2fcac133e98e3bad43677714f5251f6d441d07/specs/phase0/beacon-chain.md#attestationdata).

Discussion

If a data availability faut occurs, i.e. a finalised beacon block points to unavailable shard data, at least 2/3 of all validators must have attested to unavailable data. At most 1/3 of all validators are controlled by an attacker performing a data withholding attack so 1/3-ε of all validators made an attestation without custody of the corresponding data availability sampling chunks (the ε accommodates for the small portion of validators the attacker can fool with data availability sampling). With a single custody bit roughly half of those validators (i.e. about 1/6 of all validators) are liable to get slashed.

Notice also that the chunk_custody_bit

prevents lazy validators from following other validators (“the herd”) and attesting without first doing data availability sampling. In other words, the chunk custody bit addresses the data availability sampling validator dilemma.