Note

Writing this I incorrectly assumed the data for which custody would have to be proven is very large (10 GB). This does not seem to be the case, as only the blocks have to be included, not the state. I will leave it out here because the idea might still make sense in case a) we need a proof of custody for larger amounts of data or b) for small amounts of data, like 2MB as Justin mentioned (Bitwise XOR custody scheme), it could still make sense to execute this over say 1000 rounds, assuming that it is very inefficient to query a server for the data 1000 times and much easier to just download the complete data instead.

TL;DR

: A “proof of locality” is a special kind of proof of custody which is designed such that, for efficient computation, the data has to be stored close to the key, which I argue is what we actually want to achieve using the proof of custody. Replacing the phase 1 proof of custody with a “proof of locality” allows to significantly reduce the amount of data that has to be hashed (recuding Merkle tree depth, an advantage for the challenge game) and the number of calls to the “mix” function (this means it can be replaced with a more computationally expensive function, e.g. a BLS signature, making it more MPC friendly).

Background

: In Ethereum 2.0, each time a validator signs a cross link from a shard to the beacon chain, they provide a single “custody” bit showing that at the time of signing the cross link, they actually had the state data of that shard available. Providing an incorrect bit can be challenged by other validators and will result in that validator being penalised. For the state of the current proof of custody game design, see this PR: https://github.com/ethereum/eth2.0-specs/issues/568 :

For every crosslink they participate in during that period, if that crosslink has data with chunks D[0] … D[size(D)-1], for each D[i] they compute C[i] = mix(rk[p], D[i]) where mix is some function where for any rk, mix(rk, x) depends on x. They compute R = custody_root(D) as Merkle root of the C[i] values and sign R mod 2 along with the crosslink (this is their “custody bit”)

Currently, the suggestion for the mix function is to be $\text{mix}(a,b) = \text{sha256}(a \text{ xor } b)$

. If we can instead replace it by a signature scheme, it would be easy to prove that mix has been computed correctly, allowing participants of a staking pool to safely assign the task to one of their members or use an MPC to compute the proof (note in the “proof of locality” construction below, the latter only works if the pool participants have a low latency connection; we can still make this MPC-able by allowing for several secrets to be used in the mix function).

Suggested change

Let’s call N

the number of rounds of the proof of locality. Let $C_0 = \text{mix}(\text{rk}_p, D_0 \text{ xor } 0)$

. Then for $0\leq i < N$

compute

$\displaystyle a_{i+1} = C_i \text{ mod }(\text{size}(D))$

$C_{i+1} = \text{mix}(\text{rk}_p, D_{a_{i+1}} \text{ xor } i+1)$

Afterwards, we do the same as in the original suggestion, i.e. compute R

, the Merkle root of the C_i

and the custody bit equals $R \text{ mod } 2$

.

This means that each round requires random access to one data chunk. Given a sufficiently large number of rounds N

, network latency would make it too slow to compute the “proof of locality” without having the data available locally.

Analysis

Let’s look at this construction using a sketch of typical access speed using different storage/access media:

(Let’s assume a typical home internet connection that can be bought in a major city today with 200 MBit/s; There are two different scenarios here, the case where data is stored in a Content Distribution Network (CDN) like Cloudflare, or where the data is actually remote on a server which is often in another country or on another continent)

	Throughput	Random access Latency
HDD	100 MB/s	10 ms
SSD	500 MB/s	100 µs
RAM	10 GB/s	100 ns
Internet (CDN)	20 MB/s	10 ms
Internet (Worldwide)	20 MB/s	100 ms

Note that HDDs are already uncomfortably close to networking in both throughput and latency – we can exclude them from this analysis, any proof of custody that can be computed using an HDD can be computed over a network connection without storing the data locally (this is true for both the original suggested proof of custody and the proof of locality).

If the shard state is stored in either SSD or RAM, we actually get a much more differentiation between local and network storage in terms of latency rather than throughput. This suggests that the proof of locality actually gives more "design space" than the throughput-bound version of the proof of custody.

CDNs – good or bad?

I included the CDN latency above, because an interesting scenario might be where shard data is actually made available "relatively" locally for validators to compute the proof of custody, in which case latency of 10 ms might be achievable for most validators by replicating the data some 10s of times across the globe. I actually think we can allow this to happen, because in this case the goal of ensuring the data was available redundantly would have been achieved. So I believe we can design the proof of locality (i.e. choose the number of rounds N

) to allow it be computable in a reasonable time at 1–10 ms of random access latency, but make it fail at much more than that.

If we want to exclude this possibility, we should choose N

in such a way that the proof of locality can only be reasonably computed at 1 ms latency or less.

Choosing N

Given the above analysis, we can now choose a reasonable parameter N

. Let's assume that the maximum time a validator has to compute a proof of custody is one epoch, i.e. 384 s according to the current spec. Then we can choose $N=384\text{ s} / 10\text{ ms} = 38400$

to guard effectively against central internet storage but not CDN storage, and $N = 384\text{ s} / 1\text{ ms} = 384000$

to also prevent CDN storage and force local storage.

Pros:

- Less data to merkleise, shallower Merkle trees reduced custody game length

- By replacing the mix function with e.g. a BLS signature, the construction becomes more MPC friendly
- Closer to what proof of custody wants to achieve – data is available locally, i.e. physically close to the validator as defined by the one owning the secret
- While network bandwidth can theoretically improve indefinitely and is often already within an order of magnitude of locally, latency is bounded by physical limits, making this construction more future-proof
- While network bandwidth can theoretically improve indefinitely and is often already within an order of magnitude of locally, latency is bounded by physical limits, making this construction more future-proof

Cons

- After computing the proof of custody bit, it is enough for the validator to keep around the actually accessed chunks in order to be sure of winning the custody game