I spent the last couple of weeks trying to wrap my head around the new sharding concept based on Beacon Chain and Validators from Casper. With this post I want to try and tie some of the concepts floating around here together while filling in the blanks with some ideas that came to my mind, so please correct any misunderstandings that I undoubtedly had and also feel free to call bullshit on any ideas I had to fill the blanks.

Mostly, my idea resolves around the role of collateralized Exectutors out of which an Executor-Proposer is chosen for each collation.

High-level View:

There are two roles: Validators who “live” on the beacon chain and Executors who “live” on shards.

For each “round”, we use RANDAO on the Beacon Chain as a randomness source to sample an Executor-Proposer (EP) from the set of Executors on each shard and a committee of Attestors from the set of Validators (with different subsets for each shard).

The sampled EP chooses transactions from the transaction pool of the shard and executes them. During execution he generates a proof of independent execution. The EP then compiles a proposed collation from the executed transactions including the execution proofs and sends it to every Attestor on the committee for the current collation. These follow the process outlined in 1-bit custody bonds to validate the collation and guarantee it’s availability. As soon as a sufficient number of attestations is broadcast for a collation it becomes a cross-link to the main chain, since every attestation is simultaneously an attestation to a beacon chain block.

Collateralized Executors:

Building on the principles from proof of independent execution, collateralizing Executors and choosing block proposers from them provides some desireable properties:

- Proposers (and therefore all Executors) are strongly incentivized for correct execution, which in turn means they are incentivized to store all relevant shard state data for as long as they act as an executor.
- A single honest executor per shard is enough to keep it secure, so Executors can be assigned to shards for long periods (or even choose shards freely and stay assigned permanently? Is there any considerbale downside to that?)
- As the possible damage caused by misbehaving Executors is significantly smaller than in the case of Validators, a smaller minimum deposit with shorter withdrawal periods should be an acceptable compromise, allowing more “smaller fish” to participate in the protocol and possibly increasing decentralization.

Open questions:

- With the delay between state transition and the possibility to challenge an execution, how do we react if a specific state turns out to be invalid in the future? This kind of makes me lean towards allowing immediate challenges based on transaction recipes and also requiring every Executor to challenge immediately if a proposed collation contains an invalid transaction / result.
- How do we handle cross-shard transactions reliably? I would assume that a correct cross-shard transaction would require an Executor to be assigned to both shards and be randomly chosen as proposer on both shards. Also: How to mitigate a situation where there is no Executor that is assigned to two specific shards A and B, which would make a tx from A to B impossible?

Okay, so in the end this post turned into me thinking out loud instead of a structured approach, but I hope we can still hear some ideas about the state transition topic in general.

Thanks for reading and have a good day!