TLDR

: We suggest a cryptoeconomic gadget in a similar vein to proof of custody but for execution (as opposed to data availability). It addresses the validator's dilemma, outsourcing, self-pooling, and unfair parallelism.

Construction

Let E

be an executor (e.g. a proposer-executor or a notary-executor) voting on the validity of state roots spanning a given windback. In addition to a signature, votes now require a proof of custody of the execution trace covering the windback.

Specifically, the executor E

:

1. Chooses a secret salt s

(to be revealed at a later time)

1. Constructs a unique secret \tilde{s} = H(s || a)

where a

is the executor's address

1. Splits the execution trace into 32-byte chunks
2. Concatenates every chunk with the unique secret \tilde{s}
3. Merklelises the concatenated chunks
4. Submits the Merkle root (the proof of independent execution) with his vote

When the salt s

is revealed the proof becomes publicly verifiable and anyone can challenge a bad proof with a TrueBit-like game. Also the executor E

is slashed if the secret \tilde{s}

is leaked before the salt s

.

Notice that the execution trace and proof are "streamable". That is, the proof of independent execution can be built as execution happens, without having to store the entire execution trace in memory.

Discussion

The above construction addresses the following:

- Validator's dilemma

: An executor can't do "copycat voting" or "windback skipping" without being liable to slashing.

- Outsourcing

: An executor cannot outsource execution without leaking the secret \tilde{s}

and being liable to slashing.

- Self-pooling

: Two executors controlled by the same owner cannot reuse executions ("one CPU multiple votes") because of the uniqueness of the secret \tilde{s}

.

- Unfair parallelism

: An executor with access to "non-mainstream parallelism" (hardware like a 32-core server or FPGAs/ASICs, or software with patented/proprietary parallelism tricks) is still bound by the "inherent sequentiality" of the execution trace layout (which may allow for, say, 4-core parallelism).