This post follows from the discussion of on-chain scaling in On-chain scaling to potentially ~500 tx/sec through mass tx validation, the issues of not having data availability (https://github.com/ethereum/research/wiki/A-note-on-data-availability-and-erasure-coding) and the proposal of solutions based on zk-snark Roll_up / roll_back snark side chain ~17000 tps) with up to 17000 tx/sec but with no data availability on-chain.

We could scale to 100 tx/sec maintaining data availability.100 tx/sec implies a 400% increase in TPS and is simple.

A user identifier (address) is composed of 4 bytes. Amounts are also represented using 4 bytes. A transaction can be formed as a tuple (to, amount, signature). This information is sent to the relayers that pack many of this transaction together and send it to the smart contract. The smart contract produces a log indicating that a batch transaction has happened.

There is no formal verification of individual transactions as this can be verified by anyone in the network just by looking at the data (also notice that the sender can be obtained from the signature which then allows us to get the 4 bytes of the user identifier )

The amount of data per transaction is 73 bytes, which in the worst case amounts to 4964 gas per transaction, assuming an overhead of 50000 gas (base transaction, log, etc.,) and a block limit of 8000000 gas, this gives us 1601 transaction per block, 106 tx/sec.

We could also include a field data in the tuple such that transactions to smart contracts can be included in the system: (to, amount, signature, data) for ether transactions data will be empty. Otherwise, the data field will allow calling smart contracts inside the batch transactions.

Note that the relayers do not need to verify the transactions, their only goal is to build the transactions. However, is in their best interest to validate them as they need to be sure that they will be able to charge the corresponding fee. This is not mean to be a formal mechanism of validation but acts as a filter in the system. Also, a crafty relayer has no more effect than any other fraudulent user (regarding the integrity of the transactions).

Everyone can then build a Merkle tree of the data and ignore transactions that are inconsistent, effectively moving the load of verification off-chain. The network acts then as an immutable register of information, determining the integrity of the data is left to applications and users that use the immutable register. This is possible because all the data is available.

The same idea can be used in general for any smart contract. Users send the data, making it available to everyone, but there is no reason to execute the contract on-chain and more important there is no need to store the results of the contract execution on the network.