@vbuterin introduced a scaling scheme that can be used to scale transactions

On-chain scaling to potentially ~500 tx/sec through mass tx validation[

Applications

](/c/applications)

Hey, guys. Is anyone has already implemented this? we wanted to take it as a hackathon tasks for ETHSanFrancisco.

Here is an alternative implementation that does not use verifiable computation (SNARKS/STARKS). It may work a little slower than Vitalik's proposal, but should anyway deliver 10x performance benefit or may be more.

Here is a sketch how it works:

1. Every user deposits money to a smart contract and gets a unique user ID (say 4 bytes).
2. A Merkle tree of all account balances is kept.
3. Each transaction then becomes a 12 byte sequence (source, destination, 4 byte amount)
4. A user broadcasts a transaction to all miners.
5. Any miner can concatenate a set of transactions and send the concatenated list back to the included users.
6. The miner than waits for time T=5 sec

and the users respond with BLS/Boldyreva subgroup signature shares on the concatenated list

1. The miner forms BLS subroup signature of the concatenated list for those users that responded. Unsigned transactions are kept in the list and ignored.
2. The miner includes the concatenated list in the block as well as the BLS subgroup signature.
3. The miner also recalculates the new Merkle root and includes it in the mined block.
4. Once the block is mined, each non-mining node that receives it, verifies the new Merkle root as a part of the verification procedure.

Since the transaction size is only 12 bytes one will be able to include 20 times more transactions in the block that the current situation.

The downside of this proposal is that it requires some Layer 1 modifications. The upside is that no zkSNARKS are needed.

Note that you could make it even faster some users, by auctioning off short (one byte) addresses. A transaction between two short addresses with one byte amount would be yet 4 times faster.