With the increase in gas prices on the current ETH 1.0 chain, developing scaling solutions in the base-layer is becoming more urgent. This proposal investigates a simple base-layer scaling solution with an earlier roll-out than the fully developed phase 2 of ETH 2.0, but as little additional client development as possible.

Several ETH 1.x shards in ETH 2.0:

The current roadmap envisions one ETH 1.x shard, the original Ethereum chain, plus several “data” shards, which will be upgraded later to fully functional shards.

But instead of having just one ETH 1.x shard, we could run with very little development work, several of the ETH 1.x VMs in different shards. Maybe starting with 3 shards would be a good choice.

The communication between these different ETH 1.x shards could be implemented via Merkle proofs of finalized block hashes in other shards. Each shard could have a shard_transfer_contract

deployed. In this contract, users can lock their tokens in one shard, and then prove in the shard_transfer_contract

on the other shard via Merkle proofs that the tokens are locked. Once this proof was successful, the tokens can be minted on the second shard. Later on, this transfer can be reversed by locking the tokens in the shard_transfer_contract

of the second shard and proving it in the shard_transfer_contract

of the first shard. Making proofs about the store stage via the blockhash is alreadypossible in solidity today.

Pros

While this technical solution won’t offer a stellar user experience - waiting times are high, and transaction Merkle proofs are relatively expensive -, it would allow dapps to be run on shards with lower gas prices. If some dapps are moving from the original ETH 1.x chain, it would reduce the load on the original ETH 1.x chain and hence, even reduce gas costs over there.

The reason why we might want to consider exactly this proposal is that its execution requires very little additional work:

- A new opcode that allows reading the blockhash of different shards additional to theexisting solidity function would be required. Optimally, this opcode would also return the finalization state of the blockhash.
- [Fully orthogonal work] An easy Dapp that allows users to migrate tokens from one chain to another.

Cons

The main drawback of this proposal is that we would create new “unoptimized ETH 1.x” shards, which will be hard to recycle to fully compliant “ETH 2.0 Phase 2 shards”.

I am inquisitive about the further pros and cons of this proposal and some sentiment, whether such an approach is a viable option.