I've just read up on Ethereum's sharding documents (and am currently reading all the threads here), and hope to contribute some ideas. Given the recent threads on cross-shard locking schemes, I thought it would be a good time to jump in.

I'd like to introduce Sharded Byzantine Atomic Commit (S-BAC), a correct inter-shard consensus algorithm for atomic commitment. Unlike previously proposed algorithms, S-BAC does not rely on the client being honest to guarantee liveness, and thus no lock time-out period for clients is required to prevent deadlocks. The liveness property depends on shards being honest.

For the full details and security proofs, see page 7 of ourChainspace paper.

Here is an overview of the protocol:

[

image

744×306 35.6 KB

](https://ethresear.ch/uploads/default/original/1X/10b114b093c3313c564a00c8103d289941435f5a.png)

The protocol is agnostic to the actual BFT protocol algorithm used - that is, it doesn't matter if you use PBFT or a blockchain + proof-of-work/proof-of-stake. For the evaluation of the protocol in the paper, we used PBFT.

There are some optimisations that can be made when using a blockchain + proof-of-work/proof-of-stake, however. For example, if a shard includes a prepared(accept, T) message on the blockchain for a transaction T, but T eventually gets aborted, then that can be considered to be a waste of space on the blockchain. In another post, I will propose a way to safely and permanently prune such messages from the blockchain, even from archival nodes, so that they are not needed to bootstrap a full node.