Full Call Transcript: https://notes.ethereum.org/@afhGjrKfTKmksTOtqhB9RQ/ryzGAd9Mv

Odds and Ends

The pandemic has been a general setback for the whole world, with Eth1x efforts being no exception. In addition to many people having more family responsibilities and less productivity due to that quarantine lifestyle, we could not organize another Stateless summit as we had planned. It's simply harder to make progress and especially to make decisions online and in many timezones, but this is the new normal and we'll just have to soldier on!

This call was generally about re-focusing on the priorities in the mid-term, and looking at what features are within reach for the 1x/stateless Ethereum effort. Even though "full" Stateless Ethereum is still a worthy goal, there are closer milestones that are desirable in their own right, and in the same direction as working towards the ultimate goal of Stateless:

Hexary to Binary Trie Transition

Binary tries are not only valuable for reducing the size of witnesses, but that's a big motivation@gballet has been leading on what is thought to be the most viable path toward a binary trie transition, the overlay method.

Still, performing the transition comes with risks, and ultimately each client will have to enact the change in their own way, which might be easier for some clients to do than others. Turbogeth, for example, won't have an issue because of its flat database layout, while go-ethereum might need a bit more elbow-grease to implement. It's possible that the network might need to halt transactions (but not blocks) in order to effectively perform the switch.

There is an ongoing discussion about the Merklization rules, of which there are currently a few different iterations. Settling on these rules is really the only remaining point before an EIP for the transition can be at least drafted; it's alright to leave some of the client implementation questions open for now.

Discussions are here: Binary trie format

Also this: Overlay method for hex -> bin tree conversion

Code Merkleization

@hmijail and @sandJohnson have been working on data collection for methods of code merkleization. TeamX has shuffled things around a bit, so this work has been delayed, but within a few weeks we should have some hard data on the tradeoffs and approaches to code chunking.

Very briefly, there are 4 approaches

- Strict chunking where all code is chunked into something like 128 or 64 byte segments
- Fixed chunking where the chunk size is fixed on a contract-by-contract level but flexible
- Chunking based on JUMPDEST opcodes
- Static Analysis based on Solidity ABI FunctionId's

Nothing is decided until the data can be analyzed, but the intuition for the moment is that the more complex chunking methods might not offer enough savings to justify the implementation effort. Stay tuned for more data, and potentially a draft EIP soon.

Gas repricing

Code merkleization would potentially allow for the removal of contract size restrictions, but such a change would be best accompanied by gas repricing, which has been discussed in conjunction with most of the other Stateless Ethereum efforts to bound witness sizes to sane levels.

It's important that gas price changes remain on the table even as we shift toward more comprehensive solutions such as reGenesis.

reGenesis

The main point of discussion for this call is Alexey's reGenesis proposal, which is seen as a both a valuable improvement for Ethereum in its own right as well as a stepping stone for Stateless Ethereum.

reGenesis sidesteps the (yet unsolved) challenge of pay-on-demand gas estimation for witnesses, and instead allows transaction senders to pay up-front for the cost of a witness, which is needed for any state moved to 'inactive' after a reGenesis event. If a transaction touches an 'inactive' portion of the state, it automatically elevates it to 'active' (whether or not the transaction is successful) where it remains until the next reGenesis event. This has the nice property of creating some of the economic bounds on state usage that state rent had without actually deleting any state, and allowing transaction sender unable to generate a witness to just blindly keep trying a transaction until everything it touches is ‘active’ again.

There are two flavors of reGenesis: One in which miners are still required to keep the full inactive state, and one in which they are not. In the former, miners would be able to ‘fill in’ witnesses for transactions they receive, with the transaction senders still paying for the portion of their transaction requiring miners to provide a witness. In the latter version, a transaction sender would additionally be required to provide the witness along with a payment, which has the benefit of moving some responsibility away from miners and onto transaction senders. It might be that the first version is better to introduce regenesis, and then transitioning to the second at a later hardfork.

There are still a few unanswered questions about reGenesis, such as what happens if a re-org spans across a reGenesis event, and whether or not the proposal causes the same kind of issues with guarded sub-calls (metatransactions) that were being discussed with Oil/Karma and UNGAS.