March 25th tl;dc (too long, didn't call)

Note

: These call digests are not really interesting enough to warrant publication on the official blog, and will be from now on published here on the ethresearch forums. More substantial Stateless Ethereum posts will continue to be put out on the blog.

The main topics of this call were:

- Recapping the major takeaways from the Stateless Ethereum summit in Paris
- Discussion of alternative sync mechanisms for state, with an emphasis on what is being called "merry-go-round" sync
- Witness Specification discussion
- "Meta" discussion about who's working on what and call scheduling

Logistics

These are strange times, but Stateless Ethereum calls will still proceed as normal, around once per month.

It was suggested that time be made for a second sync-specific bi-weekly call, but the global stay-at-home trend actually means less time

for many researchers who are at home with their kids now. If there is to be a sync-specific call, it will be done in an ad-hoc fashion rather than a recurring scheduled call.

James will be sending around a survey to help us paint a more accurate picture of what everyone is working on, and what needs to be worked on. ETH2 folks included!

If you're focused on something specific, or have idle brainpower and are interested to dig in here with 1.x stuff, you are encouraged to post a short description of interest on the ethresearch forums.

Technical Discussion

Merry-Go-Round

There was substantial discussion about the syncing approach outlined in Piper's post here:

"Merry Go Round" sync [

Eth1.x Research

](/c/eth1x-research)

One concept that I think will probably apply well across any number of different methods for enumerating the state is how to package up and transmit the data for that state. This approach depends on the witness spec. For a given key prefix, we need a standard way to package up the data under that prefix. We assume that we know nothing about how much data is under the prefix meaning that this mechanism needs to allow for chunking of the data. We also need to be sure that each chunk of the dat…

The general idea behind merry-go-round sync is to pass the entire state around the network in a predetermined order, so that all participants in gossiping are, at any given time, passing around the same pieces of the state trie. To sync the whole state, one must complete a full "revolution" on the merry-go-round, covering all parts of the state.

This design has some useful properties. First, it allows new nodes joining to contribute immediately to state propagation, rather than only becoming useful to the network after a completed sync. Second, it inverts the current model of 'leecher-driven sync' whereby those with no data may request pieces of state from full nodes at will. Rather, new syncing nodes in merry-go-round sync know what parts of state are being offered at a given time, and adjust accordingly.

The overarching goal of an alternative sync method is to disallow the creation of unsustainable Stateless Ethereum nodes, which have the potential to overwhelm a network that relies on the getNodeData

primitive to sync the state trie. This new sync would be valuable for both Stateless Ethereum and for ETH2, but it needs to be developed further.

SNAP sync

Martin gave a brief run-down ofa new sync strategy called SNAP that the geth team is experimenting with that breaks the state into a handful of large chunks and proofs (on the order of 10,000 trie nodes) that can be re-assembled into the full state.

A syncing node would request a sub-section of the state from multiple nodes, and in a short amount of time have an almost

valid picture of the state stitched together from ~100 different similar state roots. To finish, the client ‘patches up’ the chunk by switching back to fast sync to build the valid current state.

Hardfork strategies

Opinions vary about how hard forks should be approached with changes for Stateless. One option is to execute all approved and endorsed changes in a single hardfork, e.g. adding code merkleization at the same time as transitioning to a binary trie. The other option is to compartmentalize the changes and perform one hard fork for each isolated change.

The latter approach is generally considered wisest, as it limits unknowns and makes each improvement less likely to get stuck in proverbial comittee.

Testing and Tooling

Alexey has submitted a revised and updated tooling roadmap, which outlines some of the necessary software tools for the Stateless Ethereum effort.

One need highlighted during the call is for statistical analysis using historical blocks. This yet-unbuilt tool would serve to simulate the live Ethereum network in order to test how a Stateless Ethereum network behaved using real transactions and network conditions from Eth1. Then various proposed changes to the witness spec could be activated and analyzed against a realistic and consistent data set, which might help in analyzing chain re-orgs and uncle rates, in particular.

Collaboration on these tools is essential; all researchers should be documenting and sharing their work on related tools to avoid a situation in which every team is rolling their own toolset specific to their work.

Witness Rollout

Given that Trinity and Nethermind are already working with beam sync, the thinking is that rolling out an early prototype version of witnesses sooner rather than later would provide a much-needed platform for experimentation.

This would be move the network-related (sync) work forward in a meaningful way without much downside for alternative client teams. Effectively this would create a second sub-network protocol for witnesses upon which experiments could be run using real live network data.

Side-Quests and the Critical Path

A few items were mentioned that belong to a class of things dubbed ‘side quests’, as they are not on the critical path but are nevertheless potentially valuable and interesting to the Stateless Ethereum effort.

These include but are not limited to:

- Stateless Mining and how it affects the network
- Accumulators and Historical Witness compression
- Formal Semantics of EVM execution

Additionally, there are some items that are

on the critical path but which currently do not have a “champion”, or lead researcher committed to driving the work forward:

- Code Merkleization
- EVM gas changes and analysis

It is not the case that these items are problems being ignored, it’s just that the work done on them so far is exploratory and early. Ideally, these items would full-time commitment from one or more capable researchers to work on over the coming months as a main quest.

Please send any questions, comments, or feedback in the thread below!