In light of the recent Reddit AMA with the Eth 2.0 research team, there has been a lot of excitement around the idea that there are no outstanding unsolved research challenges remaining that would block implementing and deploying phases 0–2 of Eth 2.0. As a counterpoint, this post aims to summarize potential open research questions needed before phase 2 can actually be deployed responsibly. This is intended to drive conversation and stimulate discussion around these issues to hopefully resolve them.

Relayer System

Due to the use of stateless clients (which is a necessity in a sharded system, as having every client storing and updating state would make the system isomorphic to a single chain), transactions require a witness

attached to them, proving the pre-state and post-state of the transaction.

Unfortunately, it turns out that providing these witness without having to do a full-sync of a shard for each transaction requires keeping state around. The actors in Eth 2.0 that have this job are relayers

(or state providers

). An excellent summary of the history of the multitude of designs that have been considered for a relayer network can be found here.

Agreement among various researchers for this relayer network is not yet unanimous. This is for good reason: existing proposals make a number of differing tradeoffs w.r.t. centralization/monopoly potential, incentives, performance, etc. An excellent summary of such tradeoffs can be found here.

This topic has been the subject of active back-and-forth among the research community and it appears that there are resources allocated towards it, so hopefully a resolution is forthcoming.

Light Client Support

Light clients, known as SPV clients in a Bitcoin context, only need to sync block headers and can be provided with Merkle proofs for their transactions or balances. In a Proof-of-Work system, this is trivial to do, as block header validity can be done by a single hash of the header and comparison to a known difficulty function.

In Proof-of-Stake, it’s not so simple. While we can certainly see that an aggregate signature contains a sufficient number of attestations, we don’t know if those attestations actually come from validators that actually have stake. The only way to know this is if we know the balance (i.e.

, state) of every validator…which is what a full node does!

Work on this is in progress but still preliminary, and both beacon chain and shard chain light clients need to be designed (with shard chain light clients potentially being easier, as the committee size is substantially smaller). While it may be certainly be possible to “deploy” Eth 2.0 by requiring everyone to run full nodes, it would be a Pyrrhic deployment, as it would require all users to run beacon chain full nodes to have any degree of trust.

Real-World Runtime and Other Costs

There are a number of real-world costs that need to be accounted for at the research level that are currently up to implementers to optimize.

As a thought experiment, it should be obvious that if a certain algorithm used in the consensus protocol cannot be implemented in less-than-exponential asymptotic runtime, it would never be usable in practice, while being sound in theory, regardless of implementation optimizations. It is very important to ensure that all components of the system are implementable with reasonable runtime, memory, and other costs.

Running a Beacon Node

Given that light clients in a PoS system are still the subject of active research, the only choice for validators is to run a beacon chain full node. The cost of running a beacon node hasn’t been fully benchmarked at this time, though given the communication costs for attestations may end up being non-trivial.

Stateless Verification

While the stateless client model certainly removes the need to do state reads/writes in order to process transactions, it does have substantially higher 1) bandwidth and 2) processing requirements (as many cryptographic hashes must be performed in order to validate Merkle proofs). The costs of this are also unknown.

Finding Slashable Attestations

In order to provide the guarantees it claims to have, Casper requires that validators actively seek out slashable attestations. The costs of doing this may be quite high, as in the worst case every attestation must be compared against every other attestation in the previous ~6 months. Preliminary work on developing an optimized implementation certainly look exciting, but more research needs to be done in order to ensure this component of the consensus protocol does not cause nodes to choke up.

Adversarial Model

The assumptions around the chosen adversarial model are quite strong in my opinion: a never-changing super-majority of validating stake is running default-configuration client software.

It does not consider adaptable corruption of validators (unlike other protocols such as Algorand, though unsuccessfully) through bribing, potentially with Dark DAOs. As the committee size was chosen specifically with this assumption in mind, if the assumption happens to not hold the performance of the system is unknown as it has not been analyzed. We ideally want to ensure that the chain degrades gracefully rather than catastrophically.

This is especially problematic given that an enormous amount of coins are held in centralized exchanges and in the future, DeFi contracts and layer-2 contracts. Should any of these be hacked (almost a certitude given enough time), the resulting stolen coins can be used to attack the system at virtually no cost.

Privacy Considerations

Even though work has been done with respect to gas and account abstraction and EEs, Eth 2.0 doesn't have sufficient privacy guarantees in order to be conducive for everyday transactions. Even though one can implement various privacy-preserving smart contract environments described in academia in execution environments, this will just guarantee small anonymity sets (which may even be counter-productive).

Moreover, without some form of privacy at the network layer, validators can be vulnerable to a number of attacks, such as DDoS or bribing.

Formal Proofs and Justifications

Last but not least, the entire system has not had any formal analysis of its properties and its correctness—and without formal proofs, we have nothing. Even more, proofs should be mechanized in a proof assistant so as to make all assumption explicit. It's quite common to realize that many things were missed when going through either implementation or formal proving.