This is a summary of our paper on “Scalability limitations of Kademlia DHTs when Enabling Data Availability Sampling in Ethereum

”

We would like to thank @dankrad and @djrtwo for many discussions and constructive conversations.

One of the directions explored to deal with data availability sampling (DAS) on Ethereum was to use a distributed hash table (DHT) in order to distribute the sections of a block (e.g., rows, columns) to the nodes in the network and allow nodes to sample blocks to test for their availability. The Codex team worked on testing the performance of a standard Kademlia DHT to be used in the context of Ethereum DAS. To do so, we developed an open-source DHT simulator capable of replicating the behaviour of a p2p network with tens of thousands of nodes, all running a standard DHT. There are two important steps where the DHT plays a critical role in the DAS construct:

1. Seeding the DHT from the block builder/proposer to all the nodes in the DHT.
2. Using the DHT for random sampling

We tested these two questions with our simulator, but before that, we verified the accuracy of our simulator by performing experiments in the IPFS DHT. We executed multiple lookup experiments in IPFS, and, in parallel, we tuned the parameters of our simulator to match the observations on IPFS.

Then, we started with the sampling part (the second step). The results showed that with a standard DHT, any node was able to query 70~80 samples within a couple of seconds, depending on network latency, number of nodes, and a few other parameters.

Then, we tested seeding the DHT (the first step) using standard provide

calls through the DHT. Assuming a block size of 256x256 (~262K samples), seeding the DHT from the block builder/proposer to all the nodes in the network was problematic in the timeline of one slot (12 seconds), with any of the configuration parameters we tested. In fact, just seeding tens of thousands of samples (not even half of the block) through this method already took several minutes, given the large number of DHT seeding operations from a single point (block builder/proposer). Obviously, this depends on available resources, but it is easy to see that this would be a potential bottleneck.

Based on these results, it seems that distributing the samples through a standard DHT is not very efficient. Other approaches, such as using Gossipsub, should be considered, and this is the current plan under the PeerDAS approach. This does not mean that using a DHT for DAS is a bad idea. It is definitely useful for sampling, and there are other techniques (e.g., optimistic provide, full routing tables) and other types of DHTs (e.g., multi-layer secure Kademlia DHT, multidimensional DHTs) that could be capable of distributing the data in the time required to guarantee the chain liveness. This is part of our future work.

For more details, please read the paper and do not hesitate to contact us if you have questions/comments.