Hey all,

Context

Some context here before I describe ongoing work:

1. We have made Malachite open-source, a flexible BFT consensus engine in Rust. Seehere for brief announcement and here as a good starting point for architecture.

2. We have finished implementing a major improvement to the CometBFT p2p layer in the shape of the DOG protocol, announced here.

3. We have passed the baton for maintaining CometBFT to the Interchain Labs team, announcedhere.

Given the above, we are shifting core engineering focus in Informal Systems towards Malachite, while also working with a few select teams to improve and specialize CometBFT for specific requirements.

Ongoing work

Now for the ongoing work –

Searching higher performance alternative to GossipSub

We are adapting DOG protocol as an alternative to GossipSub. This is because initial experiments with Malachite (which builds on GossipSub) were very promising, however, we are not confident we are able to push the implementation to its limits given potential limitations in the networking layer. We would like a p2p protocol for both mempool (user traffic) and vote/parts (consensus traffic) dissemination. The two networks may be separated, but p2p performance requirements are similar for both, at a high level:

- path redundancy should be dynamic and adjust to network conditions: if the peers are connected via too many paths – a parameter – , then some of those paths should not be used anymore, but if it is below, then more paths should be used

- log(n) latency for disseminating a message

The DOG protocol is the ongoing exploration we're doing in this regard.

Separating replication from ordering

This is an old idea that took many forms, will describe here in the context of a potential design as part of Malachite we would like to explore.

We would like to explore next a protocol design where the consensus engine relies on an external network for dissemination of user transactions. Roughly, this would work as follows. Let's call the external network a data provider network (DPN). User transactions go to a specific validator. The validator calls on the DPN to disseminate a batch of these transactions. The batch has an identifier, say id(b)

. The DPN provides the following guarantees: within a latency envelope – say X milliseconds, a system parameter – it guarantees that 2/3rds of validators will receive the batch with id id(b)

. We can think of this property as a Quality of Service promise (QoS). The DPN replies with a certificate cert(id(b))

. The validators then propose and decide on identifiers of these batches, avoiding the dissemination of bulky block parts that comprise user transactions.

Some open questions:

The DPN network is of interest in the P2P party category. This is similar to a content delivery network, but it's unclear to me what guarantees can the cert(id(b))

provide. Should the DPN network have BFT guarantees? Not sure. The minimal semantics I can think of is the QoS promise. We have not explored further if liveness in consensus (block building) can be affected by the fact that the DPN can only guarantee 2/3rd of validators to receive each batch.