Summary & TL;DR

The ProbeLab team (https://probelab.io) is carrying out a study on the performance of Gossipsub in Ethereum's P2P network. This post is reporting the first of a list of metrics that the team will be diving into, namely, how efficient is Gossipsub's gossip mechanism. For the purposes of this study, we have built a tool called Hermes (GitHub - probe-lab/hermes: A Gossipsub listener and tracer.), which acts as a GossipSub listener and tracer. Hermes subscribes to all relevant pubsub topics and traces all protocol interactions. The results reported here are from a 3.5hr trace.

Study Description:

The purpose of this study is to identify the ratio between the number of IHAVE

messages sent and the number of IWANT

messages received from our node. This should be done both in terms of overall messages, but also in terms of msgIDs

. This metric will give us an overview of the effectiveness of Gossipsub's gossip mechanism, i.e., how useful the bandwidth consumed by gossip messages really is.

TL;DR:

The effectiveness of Gossipsub's gossip mechanism, i.e., the IHAVE

and IWANT

message exchange is not efficient in the Ethereum network.

Message ratios between Sent IHAVEs

and Received IWANTs

can reach to more than 1:50 for some topics. Suggested optimisations and things to investigate to improve effectiveness are given at the end of this report.

Overall Results - Sent IHAVEs vs Received IWANT

The plots below do not differentiate between different topics. They present aggregates over all topics. The ratio of sent IHAVEs

vs received IWANTs

does not seem extreme (top plot) with a ratio of less than 1:2, but digging deeper into the number of msgIDs

carried by those IHAVE

and IWANT

messages shows a different picture (middle plot). The ratio itself for all three topics are given in the third (bottom plot), where we see that especially for the beacon_block topic the ratio is close to 1:100 and going a lot higher at times.

[

gossip_control_msgs_sent_ihave_vs_recv_iwant_messages

1000×600 68 KB

](https://ethresear.ch/uploads/default/original/3X/2/7/27e3878da92a987de30755da843003d2390d2852.png)

[

gossip_control_msgs_sent_ihave_vs_recv_iwant_message_ids

1000×600 75.5 KB

](https://ethresear.ch/uploads/default/original/3X/1/6/169e349eb5bf6f5eae425791f895e9ae32a5a41d.png)

[

gossip_control_msgs_ration_of_sent_ihave_vs_recv_iwant_message_ids_all_topics

1000×600 76.8 KB

](https://ethresear.ch/uploads/default/original/3X/a/3/a37bfd19efcd740d899634ae0d27a92413c2cb2e.png)

Per Topic Results - Sent IHAVEs vs Received IWANT

Next, we're diving into the ratio per topic

to get a better understanding of the gossip effectiveness for each topic. We're presenting the overall number as well as the ratio per topic. The ratio of sent IHAVEs

vs received IWANTs

is more extreme and reaches an average of close to 1:100 for the beacon_block

topic, 1:10 for the beacon_aggregate_and_proof

topic and 1:6 for the sync_committee_contribution_and_proof

topic.

[

gossip_control_msgs_sent_ihave_vs_recv_iwant_message_ids_69ae0e99_beacon_block

1000×600 73.5 KB

](https://ethresear.ch/uploads/default/original/3X/0/b/0b1d31747dac95b38c72afe9858fb9211ded447c.png)

[

gossip_control_msgs_sent_ihave_vs_recv_iwant_message_ids_69ae0e99_beacon_aggregate_and_proof

1000×600 56.4 KB

](https://ethresear.ch/uploads/default/original/3X/d/5/d5ac836932182c36c9ef2f0097bc2c27a8b39beb.png)

[

gossip_control_msgs_sent_ihave_vs_recv_iwant_message_ids_69ae0e99_sync_committee_contribution_and_proof

1000×600 71.1 KB

](https://ethresear.ch/uploads/default/original/3X/a/a/aa5d777e133a0cf20446b3e978d7060fbc4a8086.png)

It is clear that there is an excess of IHAVE

messages sent compared to the usefulness that these provide in terms of received IWANT

messages. There's at least a 10x bandwidth consumption that we could optimise for if we reduced the ratios especially for the beacon_block

and beacon_aggregate_and_proof

topics.

The beacon_aggregate_and_proofs

topic sends hundreds of thousands of message_ids

over the wire in a minute, with very few IWANT

messages in return. The ratio of sent IHAVE

msgIDs

to the received IWANT msgIDs

stays around 10 times bigger.

Overall Results - Received IHAVE vs Sent IWANT

The situation is even more extreme for the case of Received IHAVE

vs Sent IWANT

messages in terms of overhead. We include below the overall results only, as well as the ratios per topic. We consider that the ratios are even higher here because our node is rather well-connected (keeps connections to 250 peers) and therefore is more likely to be included in the GossipFactor

fraction of peers that are chosen to send gossip to (i.e., IHAVEs

). This in turn means that we must be receiving lots of duplicate msgIDs

in those IHAVE

messages. Digging into the number of duplicate messages are subject to a different metric further down in this report.

[

gossip_control_msgs_recv_ihave_vs_sent_iwant_messages

1000×600 59.3 KB

](https://ethresear.ch/uploads/default/original/3X/e/8/e8bfe2c7e1e5061cc1e6189027c61cb561e5ebff.png)

[

gossip_control_msgs_recv_ihave_vs_sent_iwant_message_ids

1000×600 76 KB

](https://ethresear.ch/uploads/default/original/3X/d/7/d7221f99eab26664812e8a7b8a5af32da2457f1a.png)

[

gossip_control_msgs_ration_of_recv_ihave_vs_sent_iwant_message_ids_all_topics

1000×600 112 KB

](https://ethresear.ch/uploads/default/original/3X/9/3/93bb45e0850316dea9fa50186848459a21d1d3bd.png)

Anomalies

Gossipsub messages should always be assigned to a particular topic, as not all peers are subscribed to all topics. Having a topic helps with correctly identifying invalid messages and avoiding overloading of peers with messages they're not interested in.

We have consistently seen throughout the duration of the experiment both IHAVE

and IWANT

messages sent to our node with an empty topic. Both of these are considered anomalies, especially given that the IWANT

messages we received were for msgIDs

that we didn't advertise through an IHAVE

message earlier.

Digging deeper into the results, we have seen that 49 out of the 55 peers that we received messages with an empty topic were Teku nodes. We have started the following Github issue to surface the anomaly: Possible Bug on GossipSub implementation that makes sharing IHAVE control messages with empty topics · Issue #361 · libp2p/jvm-libp2p · GitHub which has been fixed: Set topicID on outbound IHAVE and ignore inbound IHAVE for unknown topic by StefanBratanov · Pull Request #365 · libp2p/jvm-libp2p · GitHub.

Takeaways

- The average effectiveness ratio of the gossip functionality is higher than 1:10 across topics, which is not ideal.
- Messages that are generated less frequently (such as beacon_block

topic messages) are primarily propagated through the mesh and less through gossip (IHAVE/IWANT

messages), hence the higher ratios, which reach up to 1:100 for this particular topic.

- GossipSub control messages are relevant, but we identify two different use-cases for GossipSub that don't benefit in the same way from all these control messages:
- Big but less frequent messages → more prone to DUPLICATED

messages, but with less overhead on the IHAVE

control side. The gossiping effectiveness is rather small here.

- Small but very frequent messages → add significant overhead on the bandwidth usage as many more msg_ids

are added in each IHAVE

message.

- Big but less frequent messages → more prone to DUPLICATED

messages, but with less overhead on the IHAVE

control side. The gossiping effectiveness is rather small here.

- Small but very frequent messages → add significant overhead on the bandwidth usage as many more msg_ids

are added in each IHAVE

message.

Optimisation Potential

Clearly, having an effectiveness ratio of 1:10 or even less, i.e., consuming >10x more bandwidth for IHAVE/IWANT

messages than actually needed, is not ideal. Three directions for improvement have been identified, although none of them has been implemented, tested, or simulated.

1. Bloom filters: instead of sending msgIDs

in IHAVE/IWANT

messages, peers can send a bloom filter of the messages that they have received within the "message window history".

1. Adjust GossipsubHistoryGossip

factor from 3 to 2: This requires some more testing, but it's a straightforward item to consider. This parameter, set to 3 by default [link], defines for how many heartbeats do we send IHAVE

messages for. Sending messages for 3 heartbeats ago obviously increases the number of messages with questionable return (i.e., how many IWANT

messages do we receive in return).

1. Adaptive GossipFactor

per topic: As per the original go implementation of Gossipsub [link], the GossipFactor

affects how many peers we will emit gossip to at each heartbeat. The protocol sends gossip to GossipFactor * (total number of non-mesh peers)

. Making this a parameter that is adaptive to the ratio of Sent IHAVE

vs Received IWANT

messages per topic can greatly reduce the overhead seen. 1. Nodes sharing lots of IHAVE

messages with very few IWANT

messages in return could reduce the factor (saving bandwidth).

1. Nodes receiving a significant amount of IWANT

messages through gossip could actually increase the GossipFactor

accordingly to help out the rest of the network.

1. There is further adjustments that can be made if a node detects that a big part of its messages come from IWANT

messages that it sends. These could revolve around increasing the mesh size D

, or rotating the peers it has in its mesh.

1. Nodes sharing lots of IHAVE

messages with very few IWANT

messages in return could reduce the factor (saving bandwidth).

1. Nodes receiving a significant amount of IWANT

messages through gossip could actually increase the GossipFactor

accordingly to help out the rest of the network.

1. There is further adjustments that can be made if a node detects that a big part of its messages come from IWANT

messages that it sends. These could revolve around increasing the mesh size D

, or rotating the peers it has in its mesh.

For more details and results on Ethereum’s network head over tohttps://probelab.io.