Recently, we (with Ferenc Béres and Domokos Kelen) finished a research project concerning validator privacy. Here is a summary of our project. We appreciate feedback, comments, questions, food for thought, etc.

Motivation

Validator privacy is critical to the success of Ethereum. The lack of validator privacy enables, among others, attacks on privacy or denial of service (DoS) attacks against validators. A malicious actor knowing the IP addresses of validators could knock out upcoming validators in the next slot by DoSing them (maybe to increase their MEV share or for other reasons), or try to reveal their true identity for other reasons.

The goal of validator privacy is to create efficient network privacy protocols that hide the connection between a validator's physical address on the network (e.g., IP address, Ethereum address) from its consensus messages (e.g., blocks, attestations).

In the literature, there are several network privacy protocols: one class is source routing protocols (e.g., onion routing protocols like TOR), and another notable class is hop-by-hop routing protocols (e.g., Dandelion(++)). It is known that the latter class of protocols offers only brittle privacy guarantees.

Research Questions

1. What kind of privacy features or guarantees do possible protocols have?
2. Intuitively, network privacy will incur more messages on the beacon chain. Can we quantify the tradeoff between anonymity and efficiency for a particular network privacy protocol via simulations?
3. How shall we choose the parameters of network privacy protocols to achieve a given level of anonymity? Do these parameters imply an efficient deployment, given Ethereum's proof-of-stake consensus low-latency requirements?
4. Once we identify the validator privacy scheme we want to apply, what are the considerations and design choices one needs to make to be able to roll out a viable deployment?

Summary of results

We extensively evaluated the privacy and robustness guarantees of several privacy-enhancing message routing protocols, such as Dandelion, Dandelion++, and our proposed onion-routing-based protocol. In accordance with prior work, we found that Dandelion and Dandelion++ provide little to no anonymity across all parameters (i.e., forwarding probability). Therefore, we see no point in adopting Dandelion and its variants for Ethereum. On the other hand, onion-routing-based algorithms do provide meaningful privacy guarantees. However, the implementation of such a scheme needs to pay attention to many non-trivial considerations, such as spam protection and Sybil resistance during peer discovery. Our summary of onion-routing-based schemes linked below details possible solutions for these.

We suggest proceeding with either our suggested integrated onion-routing protocol or adopting the TOR anonymity network as a standalone network anonymity solution (cf. this Ethresearch post by @kaiserd). The first option requires considerable implementation and evaluation effort. Especially the deployment needs to make sure to meet Ethereum's low-latency requirements. The second option does not require such a substantial implementation effort. However, in that case, Ethereum would rely on an independent, anonymous communication system which might be undesirable. Future work entails deploying and evaluating these two options for network-level privacy.

Links

Detailed technical summary of this project:
https://github.com/ferencberes/ethp2psim/blob/c7b5f05648490a44ae6332a9647c861dfc14047d/ethp2psim_summary.pdf

Analysis of our proposed onion-routing-like message propagation algorithm:
https://info.ilab.sztaki.hu/~kdomokos/OnionRoutingP2PEthereumPrivacy.pdf

ethp2psim network privacy simulator for Ethereum: https://github.com/ferencberes/ethp2psim

The documentation of ethp2psim: https://ethp2psim.readthedocs.io/en/latest/?badge=latest