Decentralized GPS Navigation Alerting Engine

Introduction

This paper proposes how a decentralized alerting layer of a GPS application might look. Most modern GPS applications can be split into two core components: routing and alerting. Routing entails dynamic path generation, and alerting entails crowd-sourced alerts, such as speed traps, construction, crashes, or heavy traffic. By splitting up a GPS application into components, we can decentralize the alert engine layer, effectively creating a shared layer that can be used across multiple distinct GPS applications.

Network Topology

When thinking about the network topology, it helps to think of the network as an ethereum clone with an additional node ID verification protocol and more focus on light clients. Nodes join peer groups based on geolocation, and alert messages are propagated through the network until they are included in a block by a block producer. The additional node ID validation protocol exists for sybil resistance. Initial node ID verification can be bootstrapped in either a centralized or decentralized manner.

If we partition our application by regional metropolitan areas, we can run multiple peer chains which are scoped to each area. This helps ease block size and timing requirements. When a node crosses network boundaries via interstate travel, they can either perform re-verification on the new chain or operate in read-only mode. The process for crossing network boundaries is sure to evolve over time as new bridging strategies emerge.

Node Reputation

Reputation is the metric that provides Sybil resistance to the network. Nodes won't accept reported alerts unless the reporting node has met minimum reputation threshold requirements. A node's reputation can increase with behavior that leads to increased network function and security, and can decrease with behavior that is detrimental to the network. Examples of behavior that would lead to a reputation increase include: having reported alerts confirmed by other verified nodes, and reporting fake alerts. Alerts can be deemed as 'fake' If the epoch delta between alert reporting & rollback is small enough, and there are enough verified nodes that report the alert as rolled back. Attempting to publish a fake alert would be an example of behavior that leads to a reputation decrease.

Presentation-layer applications can use node reputation as a filter to determine which alerts should be bubbled up to the user. These applications could be receiving chain data using a light client approach, and it's acceptable to prematurely bubble up alerts to the user depending on the specifics of its current syncing state. For example, if the node receives an alert from a verified user, but is operating using chain data that is 1 day stale; it's unlikely that the reporting users reputation would have decreased in the past day, so the node can bubble up the alert to the user. Reputation doesn't hold any monetary value, and can't be exchanged or traded. Reputation can be staked for nodes that wish to participate in network activities. Reputation is tracked using a trie included in network blocks. One area for further research is to determine if the reputation metric carries enough weight to secure the chain, or if further backing is needed.

Bootstrapping the Reputation Engine

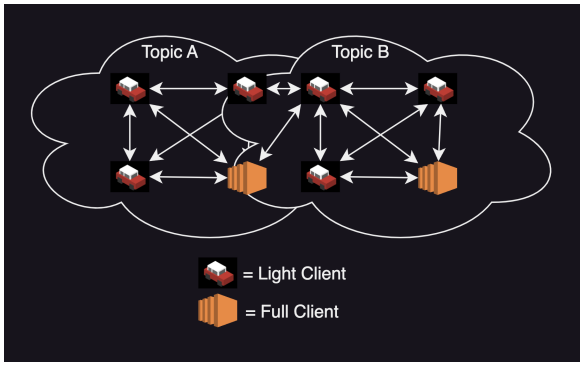
Existing user-data can be used to bootstrap reputation for nodes that have been active in the centralized protocol. Additionally, a verification protocol overlayed on top of the existing network can be created. Users can choose to periodically broadcast their location until they receive a threshold number of attestations to correlate their license plate with their pubkey. The license plate to pubkey correlation serves as the backing for the reputation metrics core sybil-resistance aspect. Attestations can be provided by any already-verified nodes. Further research can be done to determine if additional constraints are needed on the verification protocol to ensure actor-set security.

Peer Discovery

Bootstrapping the peer discovery process can be done in phases. Each step should decrease the network's reliance on centralized providers. For the first phase, DNS discovery lists can be utilized, similar to the model outlined in EIP-1459. As the network matures, the on-ramp servers can be either migrated to an distributed-hash table system or overhauled entirely as new methods emerge.

PubSub Scoping

Each pubsub channel is associated with a particular geolocation, which allows nodes to connect to channels along their routed path to collect/publish routing alerts. For example, in the diagram below, ‘topic A’ might cover several particular lights in a downtown district while ‘topic B’ covers several adjacent lights in the same district.



Message Types

Messages propagated through the pubSub network are used for alert operations and peer verification. There are two distinct types of alert messages: alert creation and alert deletion. There may be several distinct types of peer verification messages depending on the approach selected.

Block production

The reputation chain follows an opt-in approach for block production & validation, backed by reputation. Nodes are chosen in sequence from the validator set to produce blocks, which are then voted on by the other validators. Blocks with enough acceptance votes are accepted by the network, whereas blocks with rejections trigger an associated reputation adjustment for the producer.

Finality

Our main consideration with regards to selecting a finality protocol is balancing 1) the epoch delta between behavior occurring and the corresponding reputation updates being picked up by nodes, and 2) chain state-safeness by allowing a sufficient enough time for any reorgs to happen. The specifics of the finality gadget used are likely to evolve and change over time and

don’t need to be specified completely upfront.

Conclusion

In this paper, we’ve proposed a model for the decentralization of an existing GPS provider. In summary, our chain can be thought of as an ethereum clone, with more emphasis on light client support. Additionally, there’s a requirement for a validation protocol to run on top of any chain related components to provide sybil resistance for alert messages. It will be interesting to see how the evolution of decentralized protocols affects existing GPS solutions.

Appendix

Tries

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

```
reputation trie (root is signers pubkey) {
   reputation
   UID (hash of plate)
}
alert trie (root is GPS grid location) {
   alert_type
   reporter (account root in reputation trie)
   location
   reported_time
}
```

Block Contents

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

```
header{
   trie_roots
   publisher_details
}
messages[]{
   signed(report(alert_type, reporter,
location, reported_time))
   signed(negate(alert_type, reporter,
location, reported_time))
   signed(verify(target_nodes_pubkey, UID
(hash of plate), location, reported_time))
}
```