Pando (formerly Sidechain Metadata Service)

Ensuring access to notarized metadata

Abstract

There are a number of emerging streams of data, including testing of miners by dealbots, reputation services, availability and demand for data, which while not directly hashed or part of the filecoin chain are foreseen as important to keep associated and available for the ongoing health of the network. The historic state of these data streams are useful for comparative analysis and understanding of the dynamics of the network, and deserve to both be preserved, and made available.
The sidechain metadata service takes these streams of data, aggregates them into CAR archives, backs them up to filecoin, and provides a lightweight query interface for consumers to read specific points of metadata with higher assurance of their global consistency and veracity.

Background

There are several mechanisms we directly foresee being used for incentivization feedback loops based around the metadata measurements of entities in the filecoin ecosystem. For instance, reports of miner behavior can be used by reputation systems ranking miners, which then are used by clients to select higher quality miners near them. This data doesn't make sense to directly embed within the filecoin chain for a few reasons: It is produced by independent entities, so the data itself does not need to meet the same 'consensus' bar as what we would expect in a global chain, and likewise aspects of reputation and measurements may have aspects of subjectivity. It is also expected that there is diversity of data and that experimentation is a good thing.
However, there are nice properties of having this sort of metadata ecosystem more tightly linked to the chain that seem desirable to encourage, and this leads to the goals for the sidechain metadata service:

- Keep included metadata consistently available
- Provide light-weight, unbiased access to metadata
- Discourage historical revisionism.

Production Interface

A data provider (here we consider as a reputation service), produces data to be included in the sidechain metadata service through this interface:

- Data is formatted as an IPLD 'DAG'

- Providers are identified by a libp2p peerID keypair
    - This keypair may also be used as a filecoin actor address, in which case the access may be made through references to the filecoin actor address in addition to the underlying peer identity.
- Providers notify the sidechain service by publishing a message over gossip sub to advertise a new head of their dag for inclusion.
- The sidechain service will implement a policy for what data it accepts from publishers for inclusion. This policy may include:
    - A rate limit on the total accepted dag size for an unknown /unpermissioned source
    - A rate limit on the total accepted dag size for a source that corresponds to an identity that has been seen as a participant on the filecoin chain (for instance, the key is that of a known filecoin account). This rate limit may vary based on the account balance of the participant
    - A rate limit on the total accepted dag size for explicitly permissioned identities that have been white-listed for inclusion.
- At a regular interval at the discretion of the sidechain service, and likely at hourly or daily granularity, the sidechain will advance it's IPLD dag chain to have a new 'head' including new provider data.

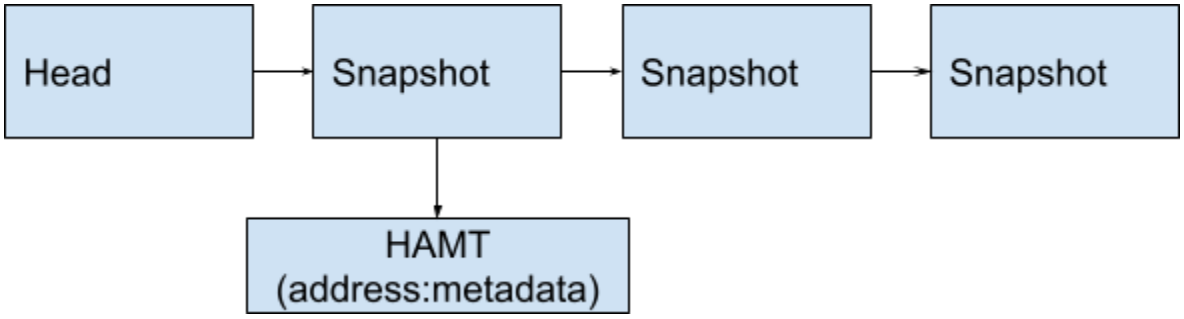
Access Interface

A client can access the sidechain metadata store in 3 distinct ways:

- Based on the known filecoin address of the sidechain metadata service, it can look at the filecoin chain to find the latest deal made by the service account, make its own retrieval deal to fetch it from an archival location, and parse the dag.
- It can `findPeer` of the known filecoin identity of the sidechain metadata service to find a current multiaddr for the service, and then fetch the IPLD data from the service directly. The most recent head can be learned from the IPNS record published at the identity address. It may use selectors to fetch only the subset data of interest if only partial data is needed.
- It can connect to one of the HTTP endpoints, listed in the metadata of snapshots, which may be cached and distributed in a 'best-effort' manner, and issue graphql queries for specific slices of metadata.

Design & Storage

The data structure of inbound data as aggregated and stored by the MDS is:



A ‘snapshot’ will contain a pointer to the current HAMT of data at that point, a timestamp of when the snapshot was generated, and metadata about the participants and locations for retrieval of the sidechain metadata service itself. This will be a list of multiaddrs that are actively operating as the MDS itself.

The HAMT is a mapping for identity account keys to the dags published by those providers. When an identity key is known in the filecoin chain by a miner or f0 account address, the dag will also be accessible under this alias.

This map structure acts as a namespacing where each publisher gets their own space of the overall metadata dag that they control the content of.

TODO(willscott): Define top level metadata object schema (pointer to cid, pointer to previous head for that provider, maybe timestamp, signed record)

Economics

Open Questions

(mike) Could we find a development partner who wants to operate the Sidechain Metadata Service as a product? (needs economic model)

- We have better success with this model vs pure dev shops

Next Steps

AIs to from 5/19 Design Review

- **(Will) Timestamps** - deal attempt data won’t compress well if records differ only in a timestamp. (Will, did I understand this right?)
- **(Will) Garbage collection and data bloat**
    - Juan’s observation was that writers to pando are going to use their own formats and anyone can use it, so the HAMT could get large fast. And searching through a HAMT is slow.

        - Idea proposed: HAMT snapshots only point to the data from last 6 months. Concerns about slowness of walking all the way back from latest snapshot.
- **(Will) Review use cases we care about**
    - Miner's wanting to understand their score, so want to retrieve all records about their specific minor
    - RSVs wanting to get all the data from 5 dealbots
- **(Will) Layer on top?**
    - Idea: Users are submitting signed objects to a layer above the datastore itself
- **(Will) Should RSVs and Dealbots also expose their data directly without going through Pando?**
    - Wasn't discussed much, but the observation was that it's the same data in the same format and it may be more efficient/simpler to listen to a Dealbot directly
    - Not a Pando replacement
- **(Mike) Name change to Pando**
    - Need to propagate this everywhere (diagrams + this doc)
- ✅ **(Mike/Will/Pooja)** What is the cost model here? How do you prevent it from filling with random data?
    - JB: yes, eventually we want to enable competing Pandos
        - In the next 1.5 years, new consensus mechanisms may emerge - like could use CRDTs here
- ✅ **(Will/Mike) Could we rate limit this by just having writers burn some filecoin for each write? (Or pay it somewhere)**
    - Conclusion was to *punt on this* until it becomes a problem

Next step after AIs: make some decisions

- Which of the above AI's are MVP vs can be deferred to later?
- Who is going to build this? Who is going to operate?
    - I think consensus is that PL builds and PL Infra operates.
- What date are we targeting for a release that unblocks reputation system work?
    - Maybe we need to scope out an MVP feature set to figure out how large the project is first?