Aggregated deal-making

Learn about aggregated deal-making on the Filecoin blockchain, where developers can combine small storage deals into larger, more attractive deals for storage providers.

Filecoin is designed to store large data for extended periods. Small-scale data (<4 GiB) can be combined with other small deals into larger ones, either on-chain or off-chain. Smart contracts can handle programmatic data storing. This article explains the process, referring to small-scale data assub-piece data .

For context, apiece of data in Filecoin refers to a unit of negotiation for data to be stored on Filecoin. A sub-piece refers to a sub-unit of that larger piece. These are typically small data like NFT images, short videos and more.

Aggregation is the process of combining multiple packages ofsub-piece data into a single package. This large package is stored on the Filecoin network instead of multiple smaller packages. Aggregation can be done off-chain or on-chain.

Process

The base interface for aggregation requires the following components:

1. A client who has data to upload.
2. An aggregator platform that clients can interact with to request to make a storage deal and retrieve Proof of Deal Sub-piece Inclusion (PoDSI) from.
3. An aggregation node to aggregate the sub-piece data into a larger file and to provide PoDSI that can be called via an API endpoint. Aggregation of data always happens off-chain. This is typically hosted by the aggregator platform
4. An optional aggregation smart contract that clients can submit an on-chain request to, to request an off-chain aggregation node to make a storage deal
5.

?

Proof of Deal Sub-piece Inclusion (PoDSI)

Proof of Deal Sub-piece Inclusion (PoDSI) is motivated by a need for sub-piece data uploads to eventually issue verification and proof that the data was included in an associated deal on Filecoin. PoDSI is heavily used in the aggregated deal-making workflow.

PoDSI is a proof construction and is generated for each sub-piece CID (within the large data segment) and stored in an off-chain database.

The proof consists of two elements:

1. An inclusion proof of a sub-tree, which contains the size and position of the sub-piece data in the larger aggregated data piece, corresponding to the tree of the aggregator's committed larger aggregated data piece.
2. An inclusion proof of the double leaf data segment descriptor, which describes the sub-piece data within the largedata segment index
3. , which is contained at the end of a deal, describing all the data segments contained within that deal.
4.

Requesting for aggregation off-chain

To request for aggregation and PoDSI off-chain, developers interact with an aggregator platform:

1. The client submits sub-piece data to an aggregator platform. The aggregator prepares the data and generates the sub-piece CID, known as pCID, and URL to download the CAR file.
2. The aggregator hosts an off-chain aggregation node, which aggregates the sub-piece CAR files into a larger aggregated CAR file.
3. Simultaneously, the aggregator aggregates indexed data segments (based on specshere
4.). It runs the proofing library and generates PoDSI proofs for each sub-piece pCID, storing them in an off-chain database.
5. The aggregator usesprogrammatic deal-making
6. ormanual deal-making
7. to make storage deals with storage providers for the aggregated larger CAR file.
8. Storage Providers download the aggregated CAR file and publish storage deals.
9. Clients can query a proofing endpoint provided by the aggregator, which will look up the sub-piece CID (pCID) in the database and return the PoDSI proof, aggregated CID, and associated deal ID.
10. Clients can use the sub-piece pCID for on-chain verification with the aggregation smart contract, which will verify the Merkle proof to ensure the sub-piece pCID (CommPc) matches the piece CID (CommPa) of the associated deal ID.
11.

Lighthouse.storage is the first aggregator platform available. You can find their docs on how to utilize their SDK for the above process .

Requesting for aggregation on-chain

On-chain aggregation and PoDSI requests go through aggregator oracle smart contracts:

1. The client prepares the data
2. and generates the sub-piece CID, known as pCID (CommPc). Here is an easy data preparation tool
3. by lighthouse.storage
4. .
5. The client submits a sub-piece CID (CommPc) with metadata (e.g. URL to download the sub-piece CAR file) directly to the aggregation smart contract.
6. The aggregator watches the aggregation contract, and when the aggregator decides there are enough sub-pieces, it downloads all sub-piece data, to generate the aggregated piece from the CAR file URL.
7. The aggregator aggregates indexed data segments into a larger data file for deal-making (based on specs here
8.).
9. The aggregator combines the sub-piece data into the aggregated CommP (CommPa) by computing within aggregator's off-chain node.
10. The aggregator uses programmatic deal-making
11. or manual deal-making
12. to make storage deals with storage providers for the aggregated larger CAR file.
13. Storage Providers download the aggregated CAR file and publish storage deals. Upon the client's request, they can find the data via sub-piece CID.
14. Clients can query the aggregation smart contract, which notifies the aggregator platform to look up the sub-piece CID (pCID) in its aggregation node's database and return the PoDSI proof, aggregated CID, and associated deal ID.
15. Simultaneously, clients can use the sub-piece pCID for on-chain verification with the aggregation smart contract, which will verify the Merkle proof to ensure the sub-piece pCID (CommPc) matches the piece CID (CommPa) of the associated deal ID.
16.

?

To build your own on-chain aggregator oracle smart contract, check out one of the implementations with Filecoin Data Tools .

Previous Cross-chain bridges Next Mainnet

Last updated4 months ago