EthDA Lightpaper

version 0.9

Oct 2023

Contents

Abstract

EthDA is an Ethereum layer 2 rollup with a permissionless set of decentralized sequencers based on a proof-of-stake consensus mechanism. Its main goal is to scale Ethereum's storage capability by leveraging blob-carrying transactions, which are proposed by EIP-4844 and generalized for storing any type of data. Blobs are sharded and permanently stored by sequencers using a Data Availability Sampling mechanism that mirrors the one used by Ethereum layer 1, in accordance with Ethereum's Danksharding scaling roadmap. Moreover, EthDA provides a set of application level protocols that allow users to store and retrieve files of arbitrary sizes based on Blobs, making it not only a suitable Data Availability network for emerging rollups, but also an innovative fully-on-chain decentralized storage solution for the Ethereum ecosystem.

Motivation

Rollups

Rollups are widely regarded as the most promising trustless scaling solution for Ethereum in the short and medium term, and possibly in the long term as well [3]. Rollups increase Ethereum's throughput and lower transaction costs by processing transactions off-chain and posting compressed transactions to Ethereum for verifiers to check the state and initiate fraud proof on demand. In other words, they use Ethereum as their Data Availability layer.

According to Ethereum's rollup centric roadmap, Danksharding is the ultimate data sharding solution that will provide enormous space on Ethereum for rollups to store their compressed transaction data [1]. However, this solution will take a significant amount of time to implement and deploy, so Proto-Danksharding, or EIP-4844, is proposed as an intermediate step that introduces blob-carrying transactions to scale Ethereum in a simple and forwards-compatible manner [2].

Currently, most rollups use Ethereum as their Data Availability layer and post compressed layer 2 transactions to layer 1 as calldata. This approach has its limitations in scalability, so some rollups opt for using some third-party Data Availability solutions such as Celestia, Avail or EdgenDA as alternatives or supplements to Ethereum [7][8]. This helps them overcome some of the scalability constraints of Ethereum, but also poses challenges for them to migrate to Proto-Danksharding and Danksharding in alignment with Ethereum roadmap, and introduces drawbacks such as being unable to pay rollup fees with native Ether, having different technical stacks, etc.

On the other hand, migrating along with Danksharding is not an easy task for existing rollups like Arbitrum One or Optimism Mainnet, due to reasons like:

- **Interface Change**. Rollups have to adapt their interfaces to use EIP-4844 blob-carrying transactions instead of regular transactions with calldata, otherwise they will not benefit from Ethereum data sharding.

- **Blob Storage Period Limit**. Blobs are only persisted in Ethereum beacon layer for a short period of time such as 4 weeks, and rollups have to either build their own solutions or rely on some third-party solutions for permanent storage. This will remain true even after Danksharding is fully implemented.

Decentralized Storage

Decentralized storage is another key component of Ethereum and the wider Web3 ecosystem. Unlike a conventional storage system that is run by a single entity or organization, decentralized storage system distributes large amount of data across a peer-to-peer network of operators located in different regions, creating a robust, trustless, and efficient file storage sharing system.

Ethereum itself is an example of a decentralized storage system, and perhaps the most ideal one for DApps of Ethereum ecosystem. However, it's only designed to store small amounts of data such as smart contract bytecode, and the cost is high due to gas fees. For larger amounts of data such as DApp websites, NFT high-resolution images, or even bigger Web2 files, some alternative solutions such as Arweave, Filecoin or Storj are needed. While these are all valuable dStorage solutions, from an Ethereum ecosystem perspective, they have some drawbacks such as lack of on-chain storage attestations, heterogeneous application interfaces and storage fees payment, etc.

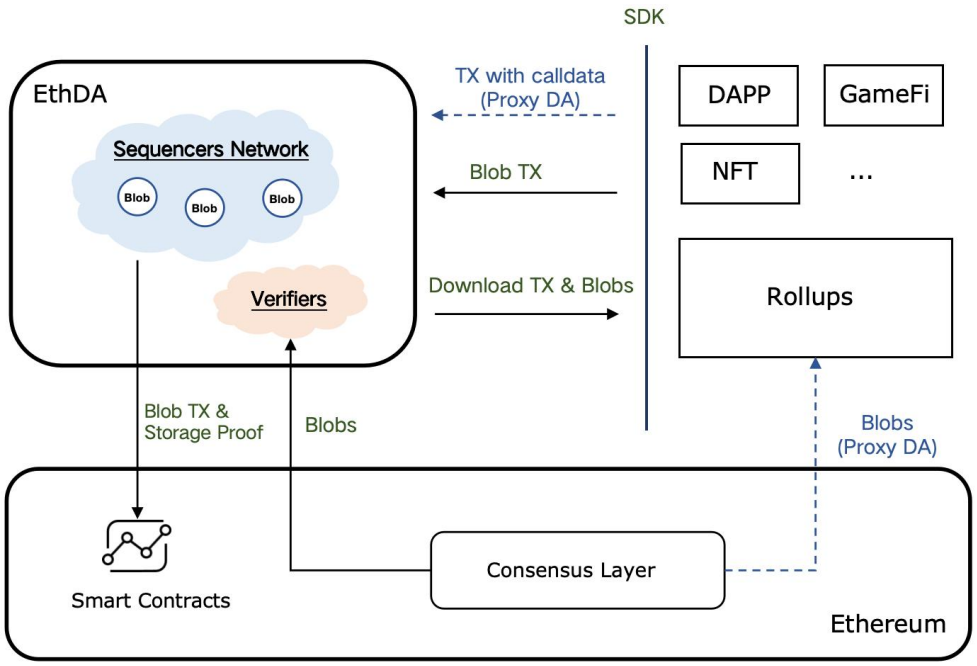
Danksharding could be seen as an attempt to scale up Ethereum's data storage capacity, but it is only aimed at storing compressed transaction data of rollups for a limited period of time. This, however, could be extended to support a decentralized storage solution that can store data of any size permanently in an Ethereum native way.

System Overview

EthDA is an Ethereum layer 2 scaling network with native blob-carrying transactions support, aiming to serve as:

- **Alt-DA** for layer 2 rollup chains, which have the option to roll up to EthDA using either transactions with calldata or blob-carrying transactions, similarly to how they would roll up to Ethereum layer 1

- **Decentralized Storage** chain for fully-on-chain DApps, or potential replacements for traditional centralized cloud storage



Protocol Overview

EthDA is designed and built based on a set of principles or philosophies.

- **Ethereum Compatibility**: EthDA aims to be an extension or scaling solution for Ethereum, not a totally different technical stack. We will keep maximal compatibility with Ethereum at both the protocol layer and the application interface layer.

- **Simplicity**: We prefer to use existing battle-tested codebase and infrastructures in Ethereum ecosystem whenever feasible, and concentrate on innovating new features on top of that. For instance, we may opt for using OP Stack to start EthDA as an Ethereum rollup chain, and may reuse Ethereum client modules if feasible.

- **Evolvability**: EthDA will evolve in alignment with Ethereum ecosystem for long-term sustainability.

The whole protocol stack consists of a collection of protocols that could be divided into two layers:

- **Core Protocols**: The optimistic rollup protocol that makes EthDA as a canonical Ethereum rollup chain using Ethereum as its data availability and settlement layer, along with a set of protocols that enables EthDA to be an alternative Data Availability solution for other rollups. Blobs can be stored on EthDA with EIP-4844 blob-carrying transactions, and will be sharded and permanently stored. The storage commitment will be rolled up to Ethereum for fraud proving.

- **Peripheral Protocols**: A set of application level protocols that equips EthDA with decentralized storage capabilities for applications. Infrastructures and toolset will be devised to facilitate large sized file storage based on Blobs.

Core Protocols

The core protocols define EthDA as an Ethereum layer 2 optimistic rollup with a permissionless set of decentralized sequencers based on a proof-of-stake consensus mechanism, as well as a novel type of blob asset that can be created and owned by accounts, and are permanently stored by the sequencers network.

Rollup Protocols

Similar to other optimistic rollups, EthDA scales Ethereum by processing transactions off-chain, and publishing transaction results and transaction data on-chain for fraud proving.

The architecture of EthDA consists of a set of on-chain contracts and a group of rollup nodes with different roles.

Layer 1 Contracts

EthDA is essentially managed by a set of smart contracts deployed on Ethereum. This enables EthDA to derive security from Ethereum, and use Ethereum as its data availability and settlement layer.

Sequencers periodically post layer 2 blocks, rollup state updates and compressed transaction data to Ethereum rollup contracts. Anyone could download all the necessary data from Ethereum and initiate fraud proof during the challenge window.

Decentralized Sequencer Network

EthDA uses a permissionless decentralized sequencer network to avoid the risk of a single sequencer defrauding the network or censoring users. A sequencer is responsible for accepting and ordering user transactions, constructing and executing layer 2 blocks, and submitting user transactions to layer 1.

Sequencers are permissionless, meaning anyone could become a sequencer as long as they provide a bond as a proof-of-stake. The bond will be slashed if the sequencer behaves maliciously such as posting an invalid layer 2 block to layer 1.

For every epoch, a sequencer will be randomly selected to bundle user transactions, execute them in layer 2 EVM, and roll up state update and transaction data to Ethereum layer 1. Verifiers monitor state updates on layer 1, and re-execute transactions using their copy of the

rollup's state. If a state mismatch is detected, they will initiate a challenge and compute a fraud proof.

Blob Asset

Blob is a new type of asset specially defined by EthDA, similar to native ETH or application level assets defined by ERC-20 or ERC-721, but is soulbound to the creator. Any user could mint a Blob by posting an EIP-4844 blob-carrying transaction to EthDA, or depositing to Blob asset contract on layer 1. That new Blob will then be synchronized to EthDA and stored permanently for the user.

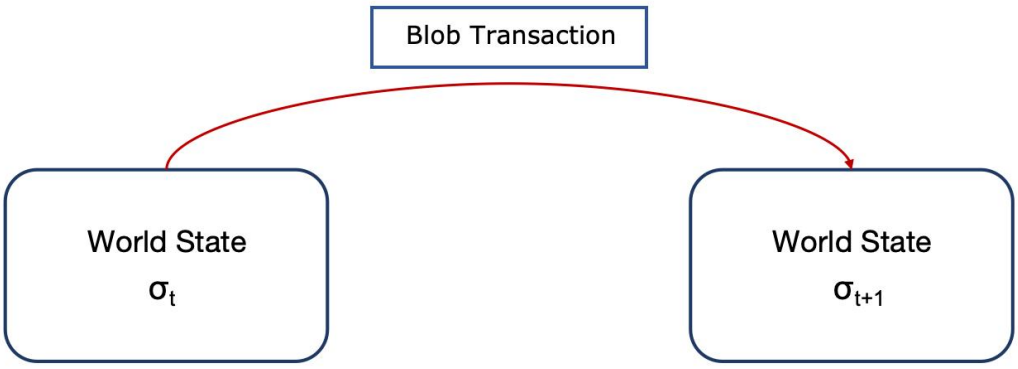
Blob assets are soulbound, meaning they are mintable only, and could not be transferred or burned once minted. Blob asset storage are guaranteed by EthDA like other assets.

From user perspective, each user owns a list of Blob versioned hashes or KZG commitments, which are visible to execution layer and could be managed by smart contracts. Blob data could be downloaded from sequencer network using versioned hash. For this purpose, a special JSON-RPC API like *getBlob(versionedHash)* will be provided by sequencer nodes.

Blob Storage

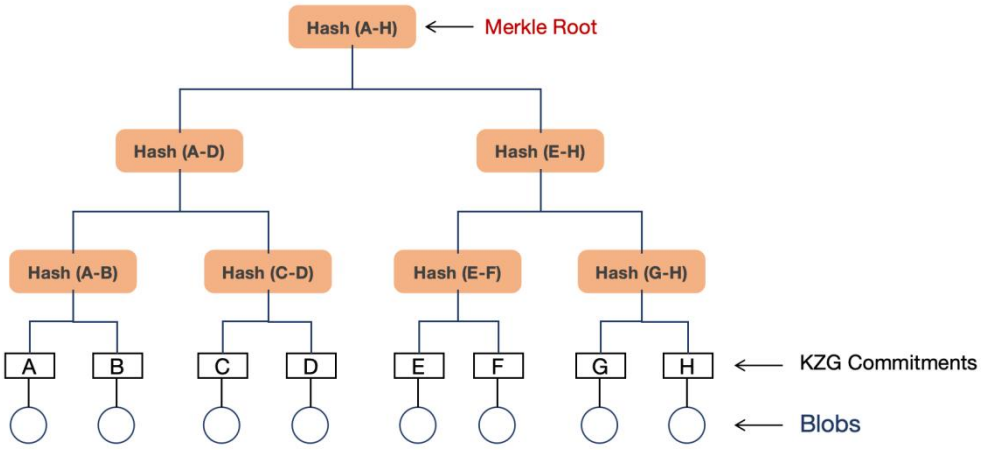
World State

Whilst Blobs are stored on a state database maintained by the peer-to-peer sequencers network using similar data sharding + DAS protocols of Ethereum Danksharding, Blob versioned hashes and KZG commitments are stored by a builtin smart contract, making them part of the world state.



Blob Merkle Tree

In state database, Blobs are organized into a Merkle tree, whose root hash is updated by blob-carrying transactions and stored as part of the world state.



Each blob-carrying transaction will append a new leaf node to the tree and cause the Merkle root to be updated. For every transaction batch (aka layer 2 block), attached Blobs compose a Merkle subtree, which will be appended to the global tree. Root hash of the subtree and the updated global tree, will be rolled up to layer 1 contracts together with world states. Also, for newly added Blobs, Merkle proofs will be generated and posted to layer 1, so that verifiers could verify Blob storage as part of the fraud proving.

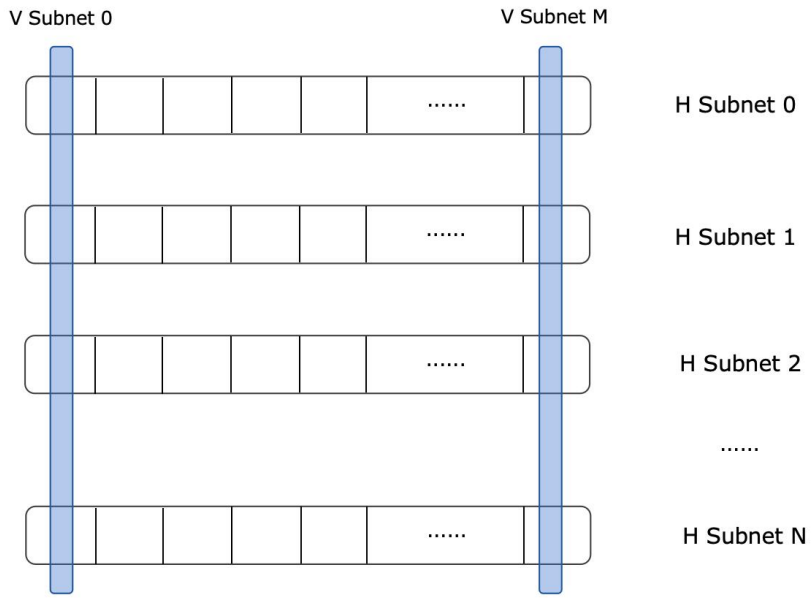
Data Availability Sampling

Data Availability Sampling on EthDA is a mirror of DAS on Ethereum, only that Blobs are permanently stored instead of being purged after a short period of time like 4 weeks.

Sequencers are randomly shuffled and split into N subnets. Since a maximal number of 64 Blobs might be attached to a block per Danksharding, imagine that N = 64. Each Blob is published to a subnet based on their index, and that subnet will be responsible for storing the Blob and ensuring its availability as a whole.

Sequencer node, however, does not need to store the whole Blob assigned to the subnet it belongs to. Instead, a Blob is split into multiple slices and sequencers will randomly sample each other to make sure the Blob is stored by the whole subnet.

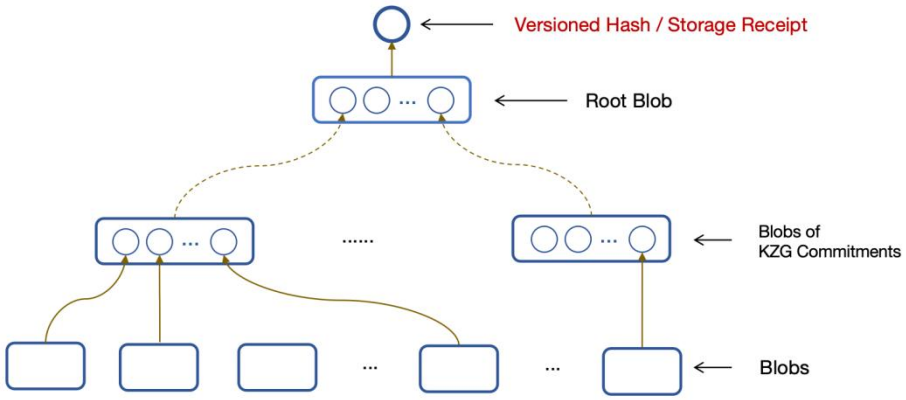
The whole subnet architecture is a grid structure of N horizontal subnets and M vertical subnets.



Peripheral Protocols

Blob Tree

An arbitrary sized file could be split into Blobs and organized into a Blob tree. The versioned hash of the root Blob could be used as the storage receipt of the file.



blob:// Protocol

A Blob access protocol could be defined as below:

blob://[blob-versioned-hash]:[chain-id]

Blob data could be auto-detected to see whether it contains KZG commitments of child Blobs, and if it does, the whole Blob tree could be parsed recursively to compose the whole chunk of data.

Infrastructures

A set of infrastructures and toolset are defined and provided to facilitate developers and users of dStorage users.

dStorage Contracts

Smart contracts to record and verify all storage orders.

Client SDK

A set of client SDK implemented in various languages (JavaScript, Python, etc) to support file storage and retrieval.

A complete e2e scenario comprises of the following stages:

- **Preprocess**: Split file into Blobs, compute intermediary Blobs of KZG commitments, and the top Blob's versioned hash as the file's storage receipt.
- **Upload**: Upload Blobs one by one to EthDA, each via a blob-carrying transaction. And post storage receipt to dStorage contract.
- **Retrieval**: Parse and download Blobs recursively, and compose the file on the client side.

Retrieval Gateway

An infrastructure to facilitate users to retrieve or render Blobs or multi-Blob files using https protocol. Imagine it like IPFS Gateway. Anyone could download the precompiled image and operate their own gateways, for scenarios like:

- File download.
- DApp rendering. DApp website could be hosted on EthDA, and be accessed via https URL in a browser with the help of a retrieval gateway.

Browser Extension

An alternative to retrieval gateway, but enable Blobs to be accessed directly via blob:// protocol.

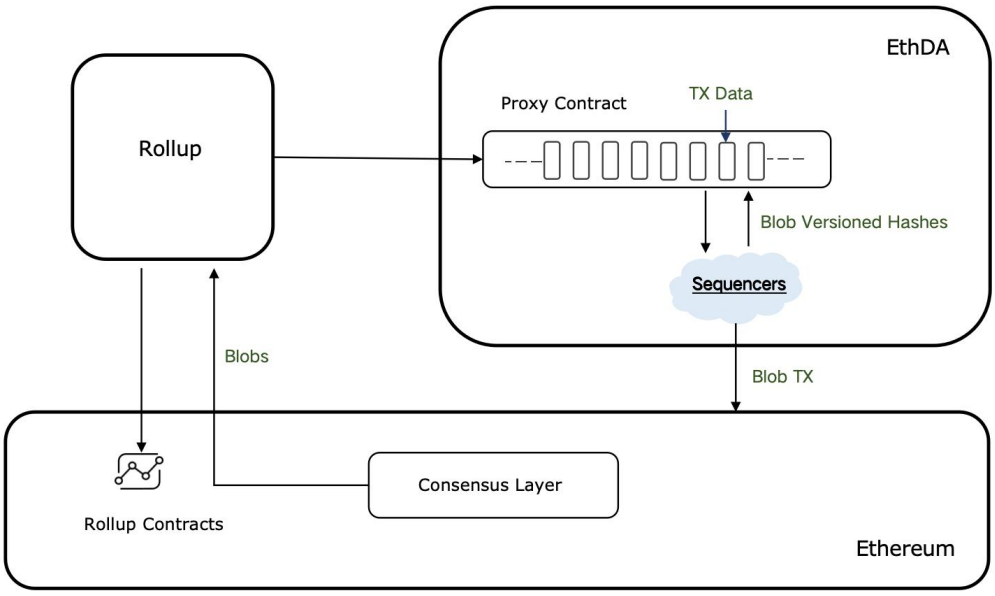
Alt-DA

EthDA is specially designed for rollups as an alternative Data Availability layer. Rollups should

store transaction data to EthDA using Blobs to maximize the benefits, but could also using calldata as a transitional solution.

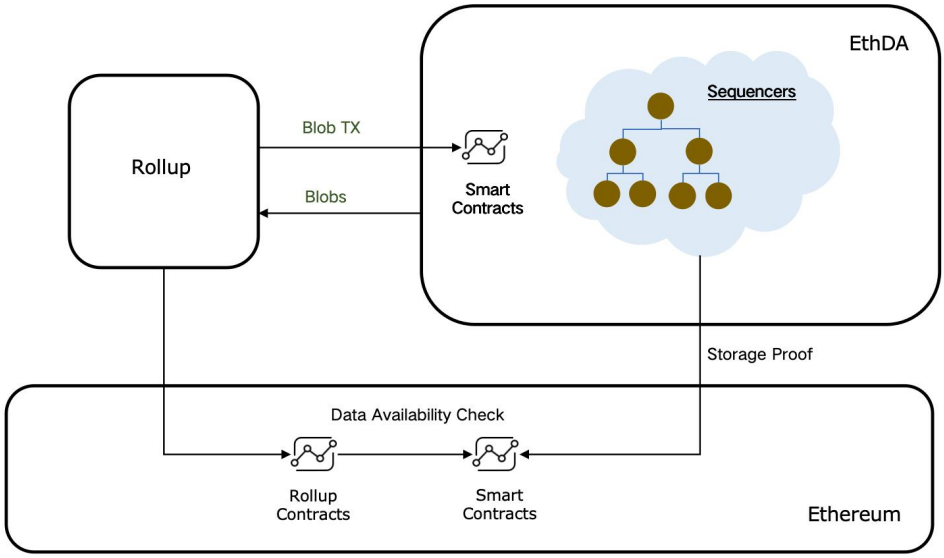
Proxy DA

In this case, rollups post compressed transactions as calldata to the rollup proxy contract deployed on EthDA. The posted data chunks are buffered in the proxy contract, until sequencers bundle them into Blobs and relay to layer 1 via blob-carrying transactions. Blob versioned hashes will be written back to the proxy contract, which could be used by rollups to look up for Blobs from layer 1 consensus network for fraud proving.



Plasma DA

Rollups could also choose to post compressed transactions via blob-carrying transactions to EthDA. In this case, EthDA acts more like a Plasma chain. Blobs are stored on EthDA in the form of a Merkle tree, and storage proofs are submitted to layer 1 smart contracts. Rollup's contracts on layer 1 could then use the storage proofs to check data availability. And in case of fraud proving, rollup verifiers need to download transaction Blobs from EthDA.



Proxy DA vs Plasma DA

Using EthDA as proxy DA mitigates the effort for rollups to migrate to Ethereum Danksharding and extracts the burden of directly posting transactions to layer 1. Essentially, rollups are still using Ethereum as their Data Availability layer. Transaction blobs are persisted by layer 1 consensus network for a short period of time, and EthDA does not take the responsibility for permanently storing the blobs.

Using EthDA as plasma DA, is a more recommended way for rollups, not only for gas reduction and throughput improvement, but also for Blob permanent persistence.

Fully-on-Chain DApp

A typical Web3 DApp comprises the following parts:

- **Smart Contracts**: Core component to keep track of user assets and operations, could be deployed on layer 1, layer 2 or any other EVM compatible chains

- **Website**: User interface to help interact with smart contracts. Could be deployed on a cloud server operated by some centralized organization, or on IPFS nodes which could be operated by some organizations like Pinata or be part of a decentralized network with incentive mechanisms like Crust Network.

- **Backend or Indexing Service**: Aggregating data from smart contracts, or performing some off-chain operations. This could be some Web2 like services, or some decentralized services like The Graph.

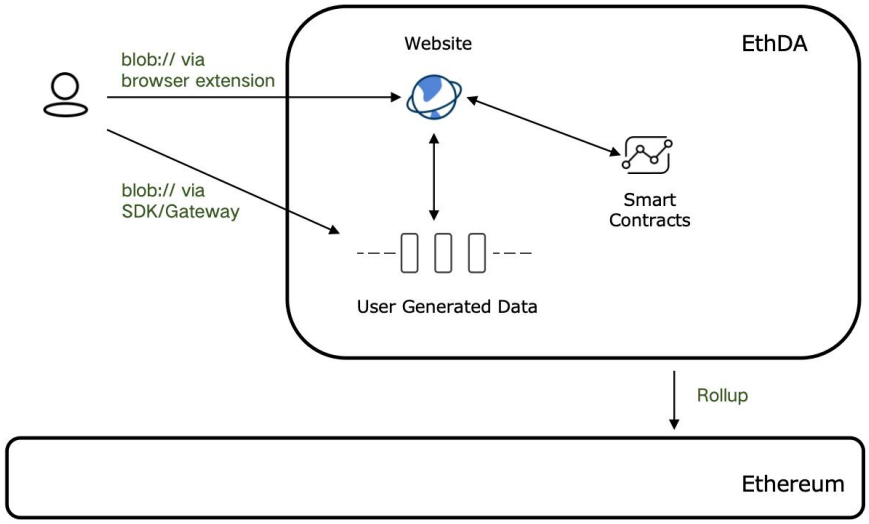
Also, some other infrastructures or points need to be considered:

- **Domain Name**: A traditional domain plus some bridging solutions like DNSLink, or some Web3 protocols like ENS name, need to be used to bind to DApp address to facilitate user access.

- **UGC**: User Generated Contents also need some storage solutions to store. It could be a S3 object storage service, some decentralized solutions like IPFS, or even blockchains like Arweave.

All these components or subsystems above, could be centralized or decentralized. And for the decentralized ones, they could be on-chain or off-chain. Nevertheless, there is no doubt a continuous effort in the community to bring DApps towards decentralization and ideally fully-on-chain from all aspects.

EthDA could help turn this vision into reality. Smart contracts and websites could be both deployed on EthDA, and user generated contents could also be stored on EthDA via the dStorage protocol. Backend service is optional, and the necessity is alleviated due to storage capacity extension. Also websites and UGC data could be accessed via blob:// protocol without binding to a domain or ENS name, making a fully-on-chain DApp totally feasible.



Conclusion

In this article, we have presented a new Ethereum layer 2 chain, EthDA, which is specially designed for scaling Ethereum as a Data Availability network for rollups and a decentralized storage system for applications. This is achieved by natively supporting blob-carrying transactions on layer 2, plus a mirror of Data Availability Sampling mechanism of Ethereum Danksharding for Blob sharding and permanent storage among a permissionless set of decentralized sequencers. Technically speaking, EthDA is a layer 2 rollup yet with some plasma characteristics for Blob storage, since Blobs are stored off-chain (from Ethereum

perspective) and only Blob storage proofs are committed on-chain.

Also a set of peripheral protocols are designed for permanently storing arbitrary sized files based on Blobs, making EthDA an ideal solution to build fully-on-chain decentralized applications.

References

[1] Danksharding. https://ethereum.org/en/roadmap/danksharding/

[2] Proto-Danksharding. https://www.eip4844.com/

[3] An Incomplete Guide to Rollups. https://vitalik.ca/general/2021/01/05/rollup.html

[4] A step-by-step roadmap for scaling rollups with calldata expansion and sharding. https://notes.ethereum.org/@vbuterin/data_sharding_roadmap

[5] An explanation of the sharding + DAS proposal.https://hackmd.io/@vbuterin/sharding_proposal

[6] Data Availability Sampling Phase 1 Proposal. https://hackmd.io/@vbuterin/das

[7] Quantum Gravity Bridge: Secure Off-Chain Data Availability for Ethereum L2s with Celestia. https://blog.celestia.org/celestiums/

[8] Avail-Powered Validiums. https://docs.availproject.org/about/introduction/validiums/

[9] Decentralized Storage. https://ethereum.org/en/developers/docs/storage/

[10] https://dnslink.dev/