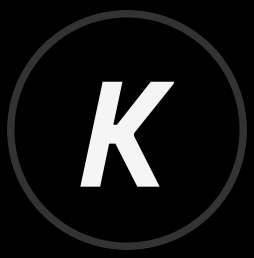

What is KYVE?

KYVE is a decentralized archival network that reconstitutes data streams as permanent resources. Any generated data stream — for example, a sequence of blocks from a blockchain — is standardized, proven valid, and stored permanently. By leveraging Arweave, KYVE secures the scalability, immutability, and availability of these resources over time.

The Problem with Data Streams

Data streams are sequences of related data elements. From a temporal perspective, retrieving arbitrarily long sequences of data can be burdensome, particularly when elements are computationally dependent on each other, or when a stream becomes non-trivial in length. Synchronization times, for instance, are a growing point of concern when nodes first enter blockchain networks. On larger chains, this can already take weeks, approaching months. Without intervention, syncing threatens to become a multi-year endeavor. The problem is not data streams as such, but that the systems they are embedded into may require their lengths be unbounded.

Similarly, streams with elements related by computation must be recovered serially. This is a step-by-step process. It can be slow. For example, computing a large number of smart contract state transitions — to "catch up" with the most recent state — has been demonstrated to negatively impact metrics like page-load times for decentralized applications. When a computation cannot begin without its antecedent, calculating the most recent state becomes increasingly more expensive and unscalable.

Enter KYVE

KYVE offers a solution in its decentralized archival framework. The framework compacts any configurable data stream into readily retrievable "snapshots". Leveraging Arweave to create permanent backups, KYVE ensures the longevity of this data over time. Naturally, data is useful only to the extent that it can be proven valid, especially when it is to be stored immutably. Invalid data can be the result of faulty transmission or storage processes, or a result of interception by malicious actors. With this in mind, any errors that may have occurred at previous junctions along a data element's lifecycle are gracefully caught and handled by KYVE's built-in and fully configurable validation step.

Generalized Data Retrieval

In recent years, decentralized systems enabled by technologies like blockchain have experienced tremendous growth, a movement known more generally as Web 3.0. There are now countless blockchain

protocols, each extending specific capabilities. It is useful to recognize that no one chain solves everything. The potential created by protocol composability is therefore enormous. Interconnecting protocols — composing one source with another — gives way to healthy and resourceful ecosystems.

Without adequate standardization, however, interoperability suffers. Across blockchains, for example, no formal standards exist to describe how data should be stored. Addressing this, KYVE automatically pipes upstream sources through a simple standardization step. By doing so, KYVE can generalize the process of data retrieval and offer a query interface for users to pull from.

On Different Terms

KYVE is deeply generalizable and therefore deeply versatile. In fact, its universality makes the mechanism's core concepts challenging to anchor outside of purely logical terms.

Let us consider the pages of a book. Now consider reading these pages one by one over the course of a day. Brains are by nature solely additive — so too is KYVE. Each page must be interpreted (or "computed") before accumulating with the last. This takes time. Given sufficient time, all pages are aggregated. The result is, effectively, an "in-memory" replication of the book. Hereafter, the book's contents may be recalled at will, as a single unit. Reading and interpreting one page at a time is no longer necessary, for pages have already been processed. At this point, pages can be thought of as having been *aggregated into a single expression*.

Specifically, KYVE is a network through which discrete elements of information enter; they are then standardized, combined, stored permanently, and validated. Arbitrary requests for the aggregate expression of these elements are possible, and data remains available indefinitely.

What is Arweave?

Arweave's fundamental inventiveness must be acknowledged. For the first time, a protocol that offers permanent, decentralized storage. The web as we know it is impermanent and prone to erasure *as a strict function of its centralization*. Corollary: hard drives fail, services are taken offline, information is lost. Web 2.0 systems are by constitution not configured to encourage collaboration amongst a collective — nor are they designed to leverage data's replicability. As a result, single points of failure are commonplace. The Arweave protocol, on the other hand, is conceived at base to maximize rates of data replication across a distributed network of participating resources, alongside a clever endowment structure to insure the cause in perpetuity. Pay now, store forever. The implications of a permanent, storage-first web are rangy, no doubt enabling KYVE's vision

The KYVE Protocol

KYVE's architecture is fully decentralized and features two main components.

The first is an autonomous governance system that handles "breaches of contract". Governance functionality is enabled by way of staking and slashing $KYVE, ensuring participants behave in harmony with the network's objectives.

The second is a computational layer that allows for participants to run customizable nodes. In exchange for $KYVE, nodes run various tasks. Inheriting from core KYVE logic, these nodes validate, standardize, and archive data streams.

Pools

Generally, pools can be described as discrete entities arranged around specific data sources. They can be created by anyone, configured to fetch data from anywhere, and orchestrate day to day operations amongst network participants. Indeed, each pool takes the form of a decentralized autonomous organization (DAO) that is powered by SmartWeave, the Arweave smart contracting language.

Node operators are synonymous with pool participants, who are organized into receiving data streams, running computations on them, and writing results to Arweave. If certain criteria are met, pools also distribute $KYVE tokens to designated uploaders and validators.

Instantiating a pool requires two instructions:

1) How to retrieve data from a data-source
2) How to validate this data

These instructions inform a given pool participant's task. Whether this task is to upload or to validate, a user must stake $KYVE in order to participate. If both the uploader and validators cooperate and remain in harmony with the pool's objectives, participants receive a payout periodically. This payout is determined by the pool creator and naturally remains in equilibrium over time. If a user's participation is deemed counter to the objectives of the pool, however, stakes are slashed. This staking mechanism encourages cooperation toward community goals.

Uploader

Only one uploader is chosen per pool. Uploaders fetch data from a source, execute instructions that may include a computational step, and write this data to Arweave. If validators find that an uploader is in violation of their terms, the uploader's stake is slashed. In this case, a validator is selected to take the uploader's place. The same process occurs if an uploader node is, for whatever reason, taken offline.

Validator

After joining a pool, validators are given a test to administer. They run

this test against the work of an uploader to determine the integrity of data elements. Though chronology is configurable, validations are by default run optimistically, *after* data is uploaded, in order to optimize for processing speed. There is no limit on how many validators may participate in each pool.

$KYVE

To the community, KYVE introduces the $KYVE token. This token enables decentralized governance within the protocol. Distributed to successful network participants, $KYVE serves as a reward for objectives met.

KYVE In Action

It is increasingly apparent that a variety of data streams — particularly those significant to Web 3.0 — benefit from being archived while remaining durable and available over time. In such cases, KYVE's usefulness shines. As it is already a major barrier, it is not difficult to imagine a blockchain community concerned with initial sync times for those wishing to spin up new nodes. A community such as this understands the implications of sync times as they relate to future network health. Blockchain data can only grow without a solution like KYVE, new nodes face greater friction upon entry, contributing to network attrition and increased centralization over time.

The first step in using KYVE to improve this scenario is to configure a pool. By doing so, a decentralized governing entity is formed to handle administration and organizational efforts moving forward. At this point, logic must also be provided to instruct future pool participants in executing their tasks.

The first piece of logic describes how to retrieve data from the data-source of interest. In our scenario, the data-source is a blockchain, with an instruction that will detail protocol-specific fetching logic to facilitate block retrieval by a pool's uploader.

The second piece of logic describes how to validate these blocks once they arrive and have been stored. This is an instruction specific to validator nodes. Because KYVE does not rely on any single central party — and is therefore trustless — validators are in place to ensure the integrity of the pool uploader's behavior going forward.

Once these first steps are complete, the pool is "open for business". Typically, the creator of the pool will join the pool itself and begin to fetch data from the defined data-source, thus becoming the designated uploader. It is important to note that staking a quantity of $KYVE against the pool is necessary for entry and acts as an expression of good faith.

Pools are flexible. Any node can switch roles between being an uploader and a validator if need be. As additional pool participants wish to join the pool and become an uploader or a validator, they too must stake $KYVE to enter,

ensuring skin-in-the-game and aligned incentives. Staked tokens ensure that uploaders and validators must now behave honestly and ensure availability, or else staked tokens will be slashed.

With everything in place, the pool is set in motion. Blockchain data is fetched and stored permanently on Arweave. Thereafter, Arweave ensures the durability and availability of this data. Uploaders and validators receive a payout of $KYVE at set intervals and continue to accumulate and validate new data elements as the blockchain state grows over time.

Now, nodes wishing to join the blockchain network and sync with its latest state can do so orders of magnitude more quickly than before. No longer must operators fetch blocks one by one; instead, data streams are reduced into snapshots with guaranteed availability. Additionally, a query interface is included out of the box to simplify requirements around syncing.

Welcome to KYVE.