Decentralized Gasless DAO Governance for TON

ton.vote by Orbs Network

Author: Shahar Yakir

TON.Vote is a completely decentralized, on-chain DAO governance platform built exclusively for the TON ecosystem, developed by the Orbs team. The core user base is anticipated to be TON DAOs, projects, and other TON-related communities requiring governance activities.

This white paper will describe the design and technical aspects of the platform.

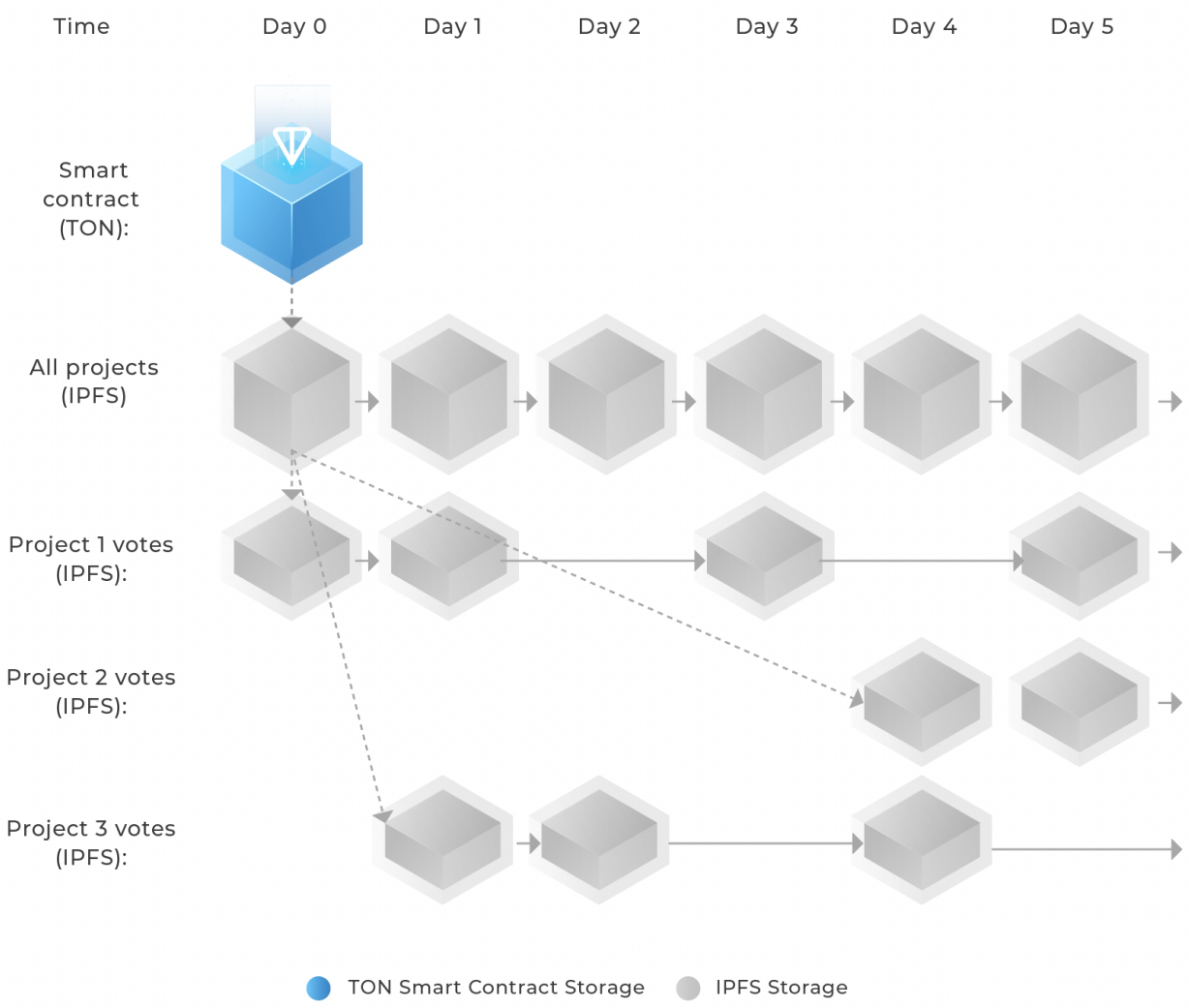
Design goals

- Decentralized infrastructure, allow all end users to audit the integrity of all votes
- Gasless voting to increase engagement (users often do not want to pay a few cents to vote)
- Snapshot token holdings at a specific historic block to prevent vote manipulation
- Support a variety of governance strategies (NFT/jettons/delegations/hierarchy/etc)
- Be inspired by snapshot.org, who built a very successful product for EVM ecosystem

Storage

The system relies on two types of decentralized storage for all of its data:

1. **TON smart contract** - Expensive storage as contract persistent state data for headers only. Stored on TON mainnet (updating this storage costs gas).
2. **IPFS -** Cheaper immutable storage for the raw data (holds the actual votes). Will be migrated in full to TON Storage as soon as TON Storage is released to production.



The system acts as a secondary logical “blockchain” on top of TON mainnet. Blocks are closed daily (every 24 hours at 23:59:59 UTC).

There are two types of blocks (see diagram above):

- “Project N votes” block for day D - A JSON file containing all the votes and proposals submitted by the community of DAO project N for a specific day D.
- “All projects” block for day D - A JSON file which contains the list of all existing DAO projects, each pointing to their own latest “Project N votes” block, updated for a specific day D.

The TON smart contract points to the latest “All projects” block (it contains the IPFS URL of the latest “All projects” block in its state data). A new “All projects” block is written at the end of every day.

For “Project N votes”, we only store new blocks on days with activity (a day that had new votes or new proposals for the project submitted in it). The blocks contain:

- Proposals - Besides the description, voting duration in days and available answers (all votes are multiple choice), proposals include the logical timestamp on TON mainnet of when holdings snapshot is taken, and the state root of relevant mainnet chains. Proposals are signed off-chain with the public key of the proposer.
- Votes - Besides the selected answer and the proposal this vote belongs to, a vote includes a Merkle proof of the holder’s TON mainnet token balance (matching the state root published in the proposal). Votes are signed off-chain with the public key of the voter.
- Proposal results - When a proposal voting period ends, the votes can be tallied and the winning result can be published. See “Tallying Votes” below for the specifics of the calculation. Proposal results are signed off-chain with the public key of the operators.

Tallying the votes

Voting for a proposal is open for a limited time (for example 7 days). To tally the votes one needs to retrieve all votes belonging to this proposal, for example by reading the 7 consecutive “Project N votes” blocks from when the proposal was published.

Each vote is first authenticated by checking the off-chain signature of the voter. The token balance of the voter is authenticated by verifying the included Merkle proof using the state root appearing in the proposal.

Votes are tallied by running a standalone “strategy” (this term comes from snapshot.org) TypeScript class over the balances. Community members will be able to PR new strategies, which will enable all sorts of custom governance models - for example, NFT votes weighted by rarity.

Trustless proofs for users

End users can be sure that vote results are authentic in a completely trustless manner:

- How can a user know that their vote wasn’t censored? The user’s client will check that the user’s vote was added to an immutable block at the end of the day. The client will retrieve the latest “All projects” block from the TON smart contract, read this block from IPFS to see the latest project block and read this block from IPFS to see that the vote appears in it.
- How can a user audit the voting result of a proposal? Any user can perform the process under “Tallying Votes” since the blocks containing all required information are immutable and public. To retrieve the votes, the user client will retrieve the latest “All projects” block from the TON smart contract, read this block from IPFS to see the latest project block and start reading from this block backwards until the relevant date period is reached. Advanced clients will include this logic to satisfy suspicious users that the voting results of a certain proposal are indeed true.

- How can a user be sure that nobody was impersonated? When tallying the votes, the off-chain signature of the voter is verified. This guarantees that nobody can impersonate the real voter and vote in their stead.

Appending new blocks

As a secondary logical “blockchain” on top of TON, the system requires a set of validators to add new blocks. The system relies on Orbs Network for this purpose. This role is exactly what Orbs is designed for, Orbs is a decentralized blockchain infrastructure network operating in L3 and running on top of L1 blockchains like TON (a layer above TON itself).

Orbs has two dozen permissionless validators running in mainnet since March 2019 - IP addresses of the validators are here - https://status.orbs.network. Much like TON itself, the nodes are operated by independent parties that are all staked with the ORBS token. The total stake of the network is over $100 million. Orbs validators are elected using a Proof-of-Stake mechanism that is very similar to TON’s but is implemented over Ethereum mainnet (for historic reasons, Orbs was already live before TON’s mainnet).

A consensus of Orbs validators (more than 50%) is required for adding a new block to the TON smart contract. The TON smart contract checks the quorum of validators’ signatures before allowing the new block pointer to be written on-chain to TON mainnet.

Writing the new blocks on-chain costs a little gas (bounded) and pinning files to IPFS has a small cost involved. Orbs network validators will cover this cost as they are writing both.

Submitting proposals and votes

When an end user submits a new proposal or a new vote under a proposal, they sign the payload with an off-chain signature using their wallet. This mechanism is currently supported by ton-x but not yet supported by TonConnect2 (will take about a month).

The signed payload is then sent to any one of the two dozen Orbs validators. Validators broadcast these payloads among themselves (similar to a blockchain mempool).

Once the daily block is written at the end of the day, the payload will be committed to storage and become immutably part of the chain.

Since end users sign the payloads off-chain, all operations for end users are gasless.

Validator block sync

When any of the Orbs validators initializes, it queries the TON smart contract to get the latest “All projects” block and downloads the latest “Project N votes” blocks for each of the projects. It then starts retrieving all blocks backwards.

During this sync, the validator creates in-memory indexes of all projects, all active proposals and the results of all past proposals.

All validators respond to REST API allowing end users and applications to query any of these indexes easily.

Decentralization

- Voting results are fully auditable by end users, so the platform is trustless and trusting the validators in particular is not required.
- Large number of validators (two dozen) creates a lot of redundancy, almost half the nodes can be offline and the system is still fully operational.
- All validator nodes are independent and operated by independent parties since 2019.
- Nodes are incentivized for high uptime using Orbs Proof-of-Stake.