🏍️

Product Idea: Cross-Chain Governance Using LayerZero

Table of Contents

Introduction

DAOs are evolving the decentralized ecosystem by bringing forth trustless decision-making and execution post-decision-making. Compound Finance was the leading force for building on-chain decision-making contracts. Though it started at a time when dApps and web3 were significantly on Ethereum and alternate and L2 chains were nascent in their roadmap and user base.

Over the last year, a lot of this has changed, with alternate chains having a sizeable TVL, user base, and supporting ecosystem. The large applications have gone multi-chain and should support users on their preferred chain, even when it comes to governance voting.

Need for Cross-Chain Governance Systems

Leading on-chain protocols have a DAO structure in place and many upcoming platforms share progressive decentralization. These protocols are cross-chain, with users familiar with certain tooling and infrastructure and have their governance token available for end-user purchase in the respective chains. For eg: Uniswap’s UNI token can be both in eth2 mainnet, arbitrum, polygon, etc.

On-Chain vs Off-Chain Voting

This has been the key deciding factor in governance setups so far. The larger protocols started on ethereum and last year it was common to see a $50-$60 gas fee for a simple transaction. Obviously, spending that amount per token holder for every proposal is not something most people are up to.

In such scenarios, the voting is taken off-chain by signing a message and storing it with a party trusted by reputation rather than technology. With cheaper gas fees (3x-5x in optimistic rollups and expected 10x-100x cheaper in zk-rollups), voting on-chain without involving a centralized party is feasible.

Cross-Chain Governance Implementation

With protocols going multi-chain and having their tokens in each chain, there’s a need for a clean & efficient system to reasonably support every token holder's right to participate in decision-making.

Here we propose a technical solution for implementing cross-chain governance using LayerZero’s cross-chain communication stack. We will later discuss its drawbacks and potential future improvement areas.

Tradeoff Decisions

While there are many designs possible for cross-chain governance voting and execution, our preferences at this stage are as follows:

1. **Voting Weightage Tracking:** The voting power of an address is visible to the governance proposal in the respective chain only. This means a user has to cast their vote in every chain they have voting power in, separately. This is to reduce complexity and gas fee cost (or reliability on a central system) to share the voting power at a certain point in time for each address to the controlling chain’s contract.
2. **Vote Communication Latency:** The vote cast can be communicated when a user casts their votes. This has the advantage of faster proposal execution once a certain number of decisive votes are cast. However, it is worse in terms of gas cost, as now every L2 user has to pay gas for two chains.
   The approach we take is batching all the votes till the execution deadline and sharing one message to give the chain-specific vote count to the orchestrating contract which aggregates votes across different chains. This way, users pay only the L2 gas fee and one-time message communication per chain can be borne by the DAO.
3. **Cross-Chain Governance Token Creation:** Many bridges exist and tokens in L2 are wrapped tokens staked on the mainnet. Each bridge would have a different wrapped contract created for the same token contract in the mainnet. To reduce confusion, we encourage each project to encourage bridging through a specific protocol. We use OFT20, ONFT72, and ONFT11155 interfaces to create tokens for generating governance tokens for DAOs. New DAOs will directly launch tokens with this standard, and existing ones with non-compatible tokens will wrap and launch new tokens in a 1:1 fashion.

4. **Proposal & Execution Control:** Instead of having a single chain's contract having the right to create and execute the proposal, we believe every chain's contract should be able to do so. This has certain advantages:

    a. The chain with the lowest gas fee trends at that point in time can be used for the execution

    b. The proposal can be created directly in the chain where a change needs to happen.

    This has certain disadvantages as well:

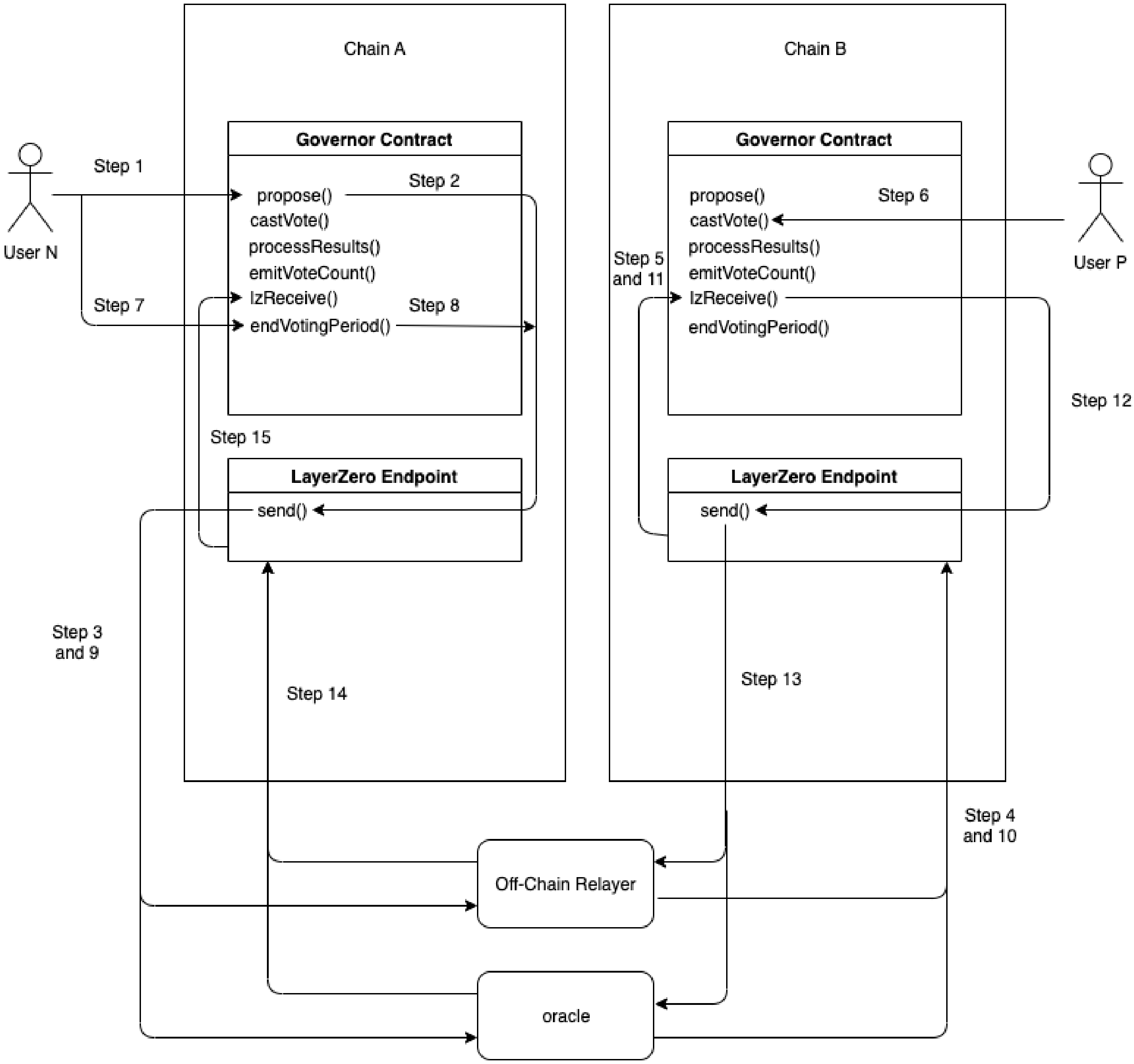
    a. More complex Code-base: The contracts, analytical tools, and indexers will become more complicated to implement over-time

    b. Addition/Modification to the contract address in a specific chain has to be communicated with all the remaining chains.

Proposal LifeCycle

There are four phases to a proposal cycle:

1. Proposal Creation
2. Cross-Chain Voting
3. Vote Reconciliation
4. Result Execution



Proposal Creation

Step 1: Anyone can trigger a proposal creation on any chain, given they have the voting power required to create a proposal on that chain.

Step 2: The contract emits a new proposal creation message with the proposal creation time (and optional end-time if it’s not fixed for every proposal) to other chains.

Step 3: The Layer Zero endpoint events are monitored by Relayer and Oracle.

Step 4: The signed message is submitted to LayerZero Endpoint in other chains

Step 5: The LayerZero Endpoint submits the message to the recipient address by invoking lzReceive() with a payload containing details for the new proposal. Here the receiving contract verifies the correctness of the payload and emitter chain & address.

Now, the voting takes place as per the governance schedule.

Cross-Chain Voting

Step 6: Once the voting period begins, any user with voting power on a chain can cast votes on the specific chain's contract.

The contract will record the vote for each address and a count of the *for*, *against* & *abstain* votes.

Vote Reconciliation

Step 7: After the voting period ends, any address can execute endVotingPeriod() in the orchestrating contract. This will emit a message to all other chains, confirming the end of the voting period and requesting voting numbers.

Steps 8-11: The cross-chain communication using Layerzero's endpoint contracts, similar to as described in steps 5-11.

Step 12: The contracts verify the emitter chain and address and verify that the stipulated voting period has in fact ended. They then emit a message containing the vote counts back to the original chain.

Steps 13-15: Using Layerzero stack to share the vote counts for each chain with the original chain.

Result Execution

After completion of step 15, the original chain has votes from all chains. Based on Timelock specifications, the desired action can be taken at any time in the future by calling **processResults()**.

Implications of the Design

Different delegation addresses in different chains

The delegation assignee is independent in each chain currently. This allows a token holder to select different delegates in different chains instead of supporting one delegate with all the voting power. Though this is not the official fractionalization of token voting rights, it is possible without distributing your token holdings across different addresses.

Different vote preferences in different chains

Each chain records votes for a user based on the voting in the respective chain. Because of this, a user can vote in favor of the proposal on certain chains and against the proposal in the other chains. Though this behavior would lead to no benefit in the proposal decision itself. Just an interesting behavior now possible.

Future Research Areas

Voting/Delegation Weight Aggregation

The biggest UX improvement would be to cast vote on a single chain for the all voting power an address has, irrespective of the chain the tokens are held in. Research in accomplishing this without utilizing gas or relying on a centralized system would be interesting to explore and a powerful addition to the platform as more chains get added to the DAO’s protocol.

Deployment to a new chain

The process of deployment to a new chain should be easy and maintenance for future updates should be possible. Research into ensuring the protocol is future-proof is always required.

Support for Voting With Staked Tokens

Many DAO tokens are eligible for earning rewards through staking. Currently, people staking the tokens lose their voting rights. While people are looking into solutions to make this possible, staking and voting on cross-chain would have to be thought through.

Supporting Tools

Tools giving insights into proposal participation or delegate statistics are something required in the current ecosystem as well. The same would be needed for cross-chain governance platforms.