Feb 27, 2023 5 min read

Governance Contract in Solidity

Updated: Mar 7, 2023

The pattern of governance many DeFi applications follows is heavily inspired by Compound Finance's implementation. Although, there isn't an Ethereum Improvement Proposal related to create a standard governance interface, most DeFi governance implementations generally follow the same set of principles.

Ultimately, governance contracts tend behave like multisignature wallets with votes weighted by token balances of the voters.

A proposal is an Ethereum transaction: an address or list of addresses, and a calldata or list of calldatas. The community (holders of tokens that give them the right to vote), propose Ethereum transactions and based on the outcome of the vote, the transaction is executed on chain, or defeated if it doesn't pass the election.

For actions that are not done on chain (such as changing the legal license of a piece of software), a message granting the rights is simply signed.

Useful terms

Before we start explaining the contracts, it's helpful to know the technical terms of the governance contracts.

Proposal

Every vote begins with a proposal, which was described earlier. It is always an Ethereum transaction that can be signed, I.e. it has a target address(es) and calldata(s).

To prevent proposal spam, contracts usually have some kind of a filter for who can create the proposal, usually an adddress that must hold a certain percentage of the total supply of the governance token.

Under the hood, a proposal is usually a solidity struct with some flags about its current state, the votes applied to it, and what transactions will be executed if the proposal passes.

Vote

Unsurprisingly, a vote is an ethereum transaction where the voter votes for or against a proposal. The vote is usually weighed by the amount of tokens the address held at the relevant snapshot.

Quorum

If no action could be taken unless 100% of the token holders voted, then it is very likely nothing would ever be accomplished, as the system would grind to a halt if only one token holder decided not to participate. On the other hand, if only 1% of the votes were required for an election to be valid, it would be too easy pass undesirable proposals.

For the fate of a proposal to be decided, it must reach a quorum threshold (a percentage of the total possible votes) within the voting period.

Voting period

Proposals don't wait around indefinitely waiting for the quorum to be reached. Otherwise, the governance proposal might be executed at a time when the circumstances that prompted the proposal change. This countdown starts as soon as the proposal is created and if quorum isn't reached within the time limit, the proposal is defeated.

Queued and Execution

If enough votes passed the quorum threshold in favor of the proposal, before the voting period expired, then the proposal is considered the have passed. For safety reasons, there is usually a time delay between when a proposal succeeds and when it actually gets executed.

Timelock

Not to be confused with the voting period, this is a delay between when a proposal has been approved and when the action is actually executed.

Consider a situation where a controversial proposal is in the governance contract, and a subset of dissenting users will withdraw liquidity if the proposal is enacted. The timelock gives them a window to leave after they see the lost the vote.

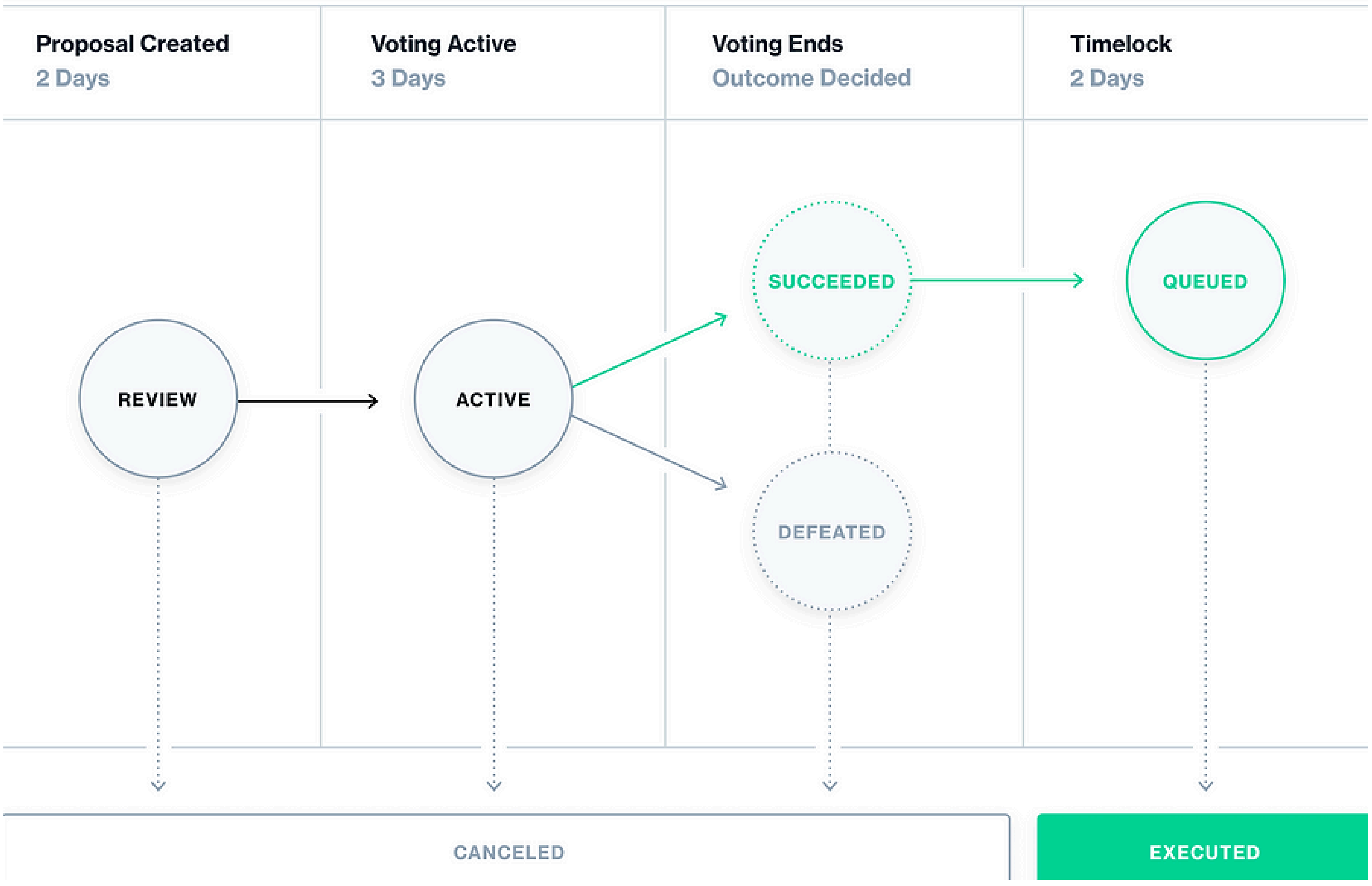
Giving users a chance to take action against unfavorable proposals incentives proposers to only include proposals that won't cause a revolt.

The phases of governance

There is no universal specification for how a governance proposal is created, voted on, and executed. However, you can roughly expect it to follow some approximation of these transations

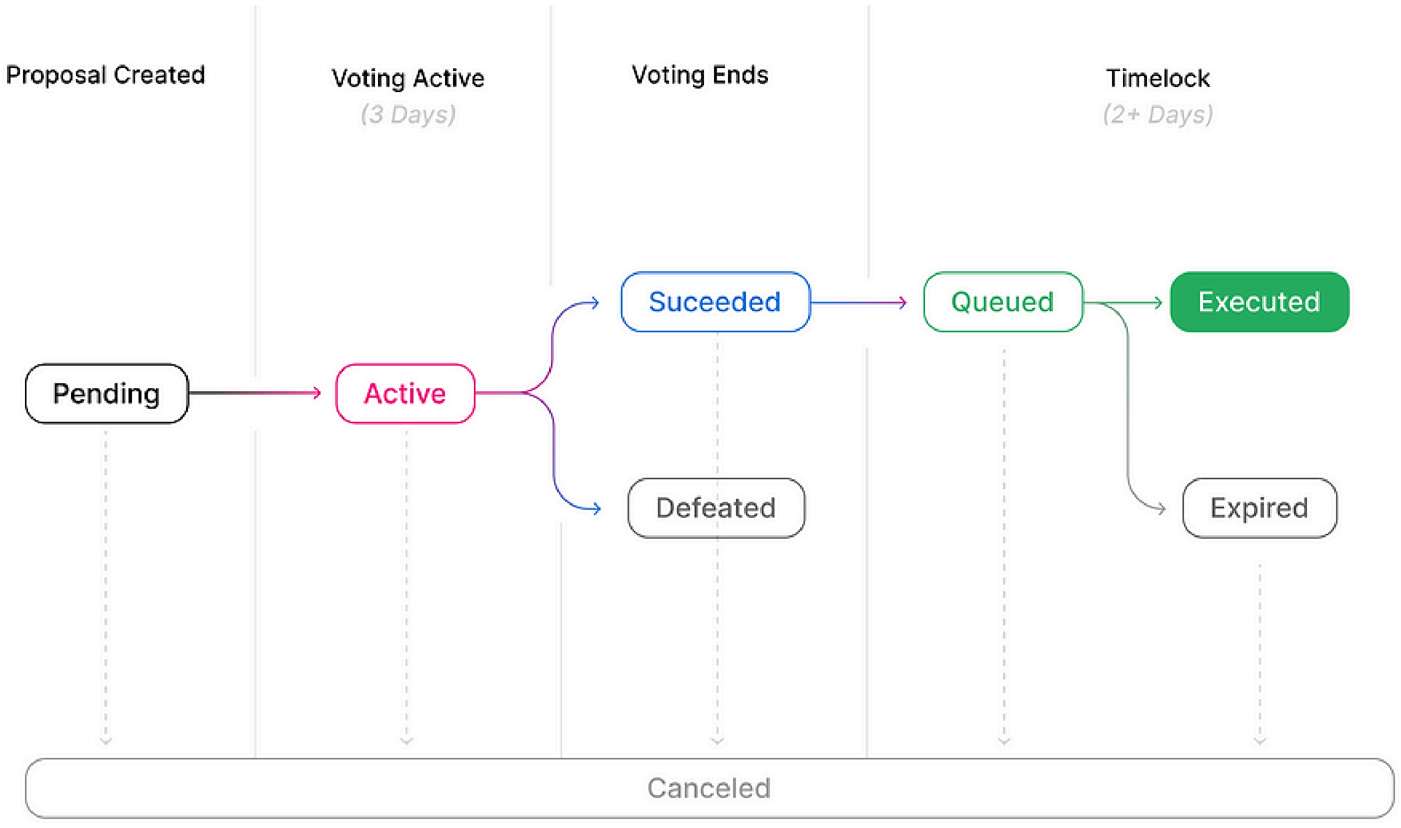
```
Pending → Active → Defeated → Canceled
                                ↳ Succeeded → Queued → Executed
                                  ↳----------↳-------↳  Expired
```

Here is Compound's governance state transition flow



https://docs.compound.finance/v2/governance

And this is Uniswap's



https://docs.uniswap.org/contracts/v2/reference/Governance/governance-reference

Who has the authority to create a proposal (and move it to pending) is protocol dependent. A common pattern is that wallets which hold enough of the governance token can create a proposal. Similarly, who can transition the state to “canceled” is protocol dependant also. There is no universal standard for who has the authority to execute these actions.

When unsure about when a state transition can occur, it’s best to simply read the governance solidity smart contract directly.

Governor Alpha and Bravo

Compound has heavily influenced how governance is carried out in DeFi. In 2021, Compound introduced new features to their original Governance smart contract design. This new designed was called Governor Bravo, and other DeFi protocols followed suit in the design changes.

Here is what is new in Governor Bravo

- Voters can explicitly “abstain” rather than voting only yes or no
- The governance contract becomes an upgradeable proxy pattern
- Voters can add a reason string to their votes

In the OpenZeppelin Bravo implementation, the added field for “abstain” can be seen in the vote struct.

An example workflow of executing a governance proposal

The website tally.xyz provides a very nice interface that shows the workflow of a governance proposal.

Let’s look at proposal 9 from Uniswap, a successful proposal to add a 1 basis point fee tier to their liquidity pools.

For those unfamiliar with Uniswap, liquidity providers earn a fee whenever someone who wants to trade tokens executes a swap against the pools liqudity providers create. This fee is determined at pool creation time, but can only be from a fixed set of fee sizes. The community wished to add a very small fee option to be competitive against other token swapping DeFi services, and the proposal passed.

Here is tally’s visualization of the vote and execution.

https://www.tally.xyz/gov/uniswap/proposal/9

The page should be largely self explanatory, but we will briefly explain here for convenience.

Each of the actions listed here are on-chain actions. The reader can click the three dot widget to access the respective transactions on Etherscan.

Here is the transaction where the proposal was actually executed. https://etherscan.io/tx/0x5c84f89a67237db7500538b81af61ebd827c081302dd73a1c20c8f6efaaf4f3c#eventlog

It is apparent the "FeeAmountEnabled" event was emitted by the contract at 0x1f98431c8ad98523631ae4a59f267346ea31f984, which is the Uniswap factory for creating pools.

Tally conveniently provides the code that the voters were voting on executing, and which was eventually executed.

Governance Attacks

Here are some example governance attacks that hackers executed successfully.

Flash loan attack

The BeanStalk protocol had a feature called "emergencyCommit" where a transaction could be executed immediately if enough voters supported the transaction. This was intended to bypass the time delay if there was an emergency proposal that needed to be pushed through. The attacker exploited this by taking out a flashloan to obtain sufficient voting power to bypass the timelocks and execute a command that drained the protocol of funds.

Low price attack

If the price of a token is low enough, then an attacker can economically obtain enough votes to pass whichever proposal they wish. Protocols that have significant assets locked in the protocol relative to the market capitalization of the governance token are especially suceptible to this kind of attack. This attack was carried out against True Seniorage Dollar, where the attacker voted to mint billions of stablecoins for himself, then sold them on another decentralized exchange. Because governance tokens have relatively low market capitalizations, the chance of a 51% attack is not negligible.

Social or political attacks

As with any democracy, votes can be socially manipulated to a certain degree. A coordinated campaign by an equipped attacker can push through undesirable proposals to execution.

Room for improvement

It hasn't been established that the current pattern of DeFi Governance, which this article describes, is actually the optimal way. Vitalik Buterin has attacked the current practices as generally bad for small voters. He proposed solutions in his blog post. One solution that gained some traction is to weigh votes by the square root of the holdings, also known as "quadratic voting." There is still an unsolved problem as an attacker can split up their holdings into multiple addresses to avoid getting their votes de-weighted by the square root function.

Tools for setting up governance

For those wishing to set up DAOs with onchain governance, here are some resources to avoid writing the contracts from scratch.

- https://wizard.openzeppelin.com/#governor
- https://www.tally.xyz/developers
- https://docs.alchemy.com/docs/how-to-create-a-dao-governance-token

Conclusion

Governance contracts are the currently accepted pattern for governance token holders to vote on what Ethereum transactions to execute. Although there is a common pattern many projects follow, conducting governance in a secure and fair way is still an open research question.

Learn More

See our blockchain bootcamp to learn Ethereum and DeFi development.