Lido DAO currently has governance power over most of the protocol code and parameters.

In this post, we discuss the current scope of governance, the associated risks, and how these risks can be minimized by reducing the governance scope (via ossification) and by aligning incentives between the DAO and stakers (via dual governance structure).

Table of contents

- Protection from veto abuse
- Option: significant timelock instead of veto
- Option: expropriate stETH tokens
- Option: significant timelock instead of veto
- Option: expropriate stETH tokens
- Dual governance scope
- [Including more governance parties

](#including-more-governance-parties-17)

- Implementation complexity
- Next steps

Current governance mechanics

Let’s start with a recap on the current Lido governance mechanics.

Lido protocol is governed by the LDO token. Any change has to be approved via a DAO vote, which can be:

- A standard Aragon vote that has to gain support from at least 5% of the LDO total supply (not circulating supply) and get more support than opposition.
- An optimistic “Easy Track” motion that is strictly limited in scope, cannot upgrade protocol code, and can be opposed by 0.5% of the total LDO supply.

More detailed explanation of the current mechanics and upcoming improvementscan be found in this note.

Scope of governance

As detailed in the governance risk FAQ, Lido DAO currently has control over the following components of the Ethereum liquid staking protocol:

- Upgrades of the Ethereum liquid staking protocol code:
- Lido

at 0xae7ab96520DE3A18E5e111B5EaAb095312D7fE84

: implements stETH token mechanics, stake buffering, general coordination between other contracts. Acts as the main interface to the protocol.

- Withdrawals

at 0xB9D7934878B5FB9610B3fE8A5e441e8fad7E293f

: implements ETH undelegation (currently a stub).

- Oracle

at 0x442af784A788A5bd6F42A01Ebe9F287a871243fb

: manages the consensus layer oracle committee and ensures consensus within its members.

- NodeOperatorsRegistry

at 0x55032650b14df07b85bF18A3a3eC8E0Af2e028d5

: manages the list of node operators and new stake distribution between them.

- Lido

at 0xae7ab96520DE3A18E5e111B5EaAb095312D7fE84

: implements stETH token mechanics, stake buffering, general coordination between other contracts. Acts as the main interface to the protocol.

- Withdrawals

at 0xB9D7934878B5FB9610B3fE8A5e441e8fad7E293f

: implements ETH undelegation (currently a stub).

- Oracle

at 0x442af784A788A5bd6F42A01Ebe9F287a871243fb

: manages the consensus layer oracle committee and ensures consensus within its members.

- NodeOperatorsRegistry

at 0x55032650b14df07b85bF18A3a3eC8E0Af2e028d5

: manages the list of node operators and new stake distribution between them.

- Adding and removing node operators; changing the mechanics of node operators registry via a code upgrade.
- New stake distribution among node operators: DAO sets the upper limit on the number of validators each node operator can run.
- Protocol and node operator fees: which percentage of staking rewards goes to the DAO treasury and node operators.
- The LDO governance token mechanics and issuance.
- Protocol treasury.

Of the control levers listed above, only protocol treasury management doesn't affect Ethereum network security directly or indirectly.

An important nuance is that, though Lido DAO (LDO token and governance smart contracts) is deployed on Ethereum, LDO holders possess governance power over liquid staking protocols on other chains like Polygon and Solana. While this governance power is unidirectional (from Ethereum to other chains) and Ethereum liquid staking protocol is by far largest of all staking solutions governed by Lido DAO, the multi-chain governance setup means that LDO holders have incentives related to other chains, and these incentives are not necessarily aligned with those of Ethereum network participants.

Worst-case scenarios

Let's examine the worst-case scenarios that Lido governance capture could trigger for stakers and the Ethereum network as a whole.

stETH holders

Due to the control over stETH code upgrades, Lido DAO can upgrade the stETH contract in a way that would allow burning stETH on an arbitrary address and minting stETH to an arbitrary address. Thus, even though the DAO currently has no control over the ETH that backs stETH, it can effectively steal funds from any user by upgrading the code to burn their stETH and mint it to another address.

Post-Capella (probably 2023H1), due to the upgradeability of the withdrawal smart contract, Lido DAO would be able to steal the withdrawn ETH given that the withdrawal process is assisted by a node operator. For this, the DAO would need to maliciously upgrade the withdrawal contract to allow extracting the received ETH and then persuade or trick a node operator into initiating the exit.

After smart contract-initiated withdrawals are implemented (likely not earlier than 2024), the DAO would also be able to trigger withdrawal of an arbitrary validator and steal the withdrawn ETH without assistance from the node operator.

Ethereum network

Lido DAO or Lido team has no direct control over validators or the node operators running them. Indirect control stems from the fact that currently, Lido governance manages a whitelist of node operators that can participate in the protocol:

1. Lido Node Operator Subgovernance Group (LNOSG) consisting of representatives of Lido team and whitelisted node operators' teams proposes changes to be made to the node operator set, e.g. adding new members.
2. No changes to the set can be made without an explicit DAO vote by LDO holders.
3. Lido DAO can vote for inclusion or exclusion of any set of node operators, not only those proposed by LNOSG.

Thus, Lido DAO (represented by LDO holders) can exclude any operator from the list, preventing them from getting new

stake.

As a result, the worst-case scenario would be a cartelization of the protocol's validator set. As discussed inThe Next Chapter for Lido, it would most probably happen gradually, by directing new stake to a subset of node operators and gradual withdrawal of other node operators (that will become possible after smart contract-initiated withdrawals are implemented).

The problem

The root cause of Lido governance introducing additional risks is a principal-agent problem between stakers (the principal) and LDO holders (the agent)

. The problem exists because LDO holders don't have the exact same incentives as stakers.

One manifestation of this problem is that, due to the governance having power over multiple blockchains, a "supranational" organization (Lido governance) currently makes unilateral decisions over a "national" protocol (Ethereum liquid staking).

We propose to deal with this problem using a dual approach:

1. Reduce the scope of governance as much as possible via ossification.
2. For the remaining governance scope, break the principal-agent problem by introducing a dispute and resolution mechanism for misaligned incentives between stakers and LDO holders.

One could argue that, even if Lido governance is aligned with stakers, this still won't protect ETH users since "ETH users" is a different set than "ETH stakers". However, we believe this property is something related to PoS design in general and thus cannot be changed by Lido (nor that is it necessary). See Hasu's explanation for a more detailed reasoning.

Reducing the governance scope

Reducing the power of governance over the Lido protocol via ossification is the most effective solution to governance risks, and is something the Lido team plans to propose as early as possible.

However, ossification only becomes an option after the ETH2 specification gets finalized and battle-tested, so we need an interim solution until then.

Also, even after ossification, some of the governance scope will still remain, namely management of the node operator set (including control over the parameters of a permissionless registry).

Solving the principal-agent problem

This brings us to the following problem: given that the current governance scope will remain for some time, and that certain aspects of node operator set management will be governance-controlled even after the ossification, can we describe a governance mechanism that satisfies the following goals?

1. Solve the principal-agent problem between Lido and stakers by preventing proposals from passing that would be hazardous for stakers.
2. Don't introduce new attack vectors, e.g. from stakers on Lido.
3. Allow Lido to recover from a "captured/stuck" state of LDO governance.

The answer we propose is a dual-governance framework between LDO ad stETH holders discussed below.

Dual governance

Let's modify the current LDO governance mechanics as follows:

- LDO holders can still propose and vote in the same way as before.
- A timelock is added between LDO vote passing and execution (LDO execution timelock

).

- By default, Lido governance is in the global state where, after a timelock, any successful LDO vote can be executed (normal state

).

- However, a quorum of stETH holders can apply a veto, switching Lido governance into the adversarial state where no LDO vote can be executed unless it's specifically un-vetoed (veto state

).

[

Gov flow chart

2044×2484 280 KB

](https://europe1.discourse-cdn.com/business20/uploads/lido/original/2X/c/cf4bcbffa9e5908c3f7dfb921a5ad7c574744613.png)

Transition to veto state:

- Any stETH holder can lock their stETH in a dedicated escrow contract.
- The governance transitions to the veto state as soon as the total amount of stETH locked in the escrow contract minus stETH pending withdrawal (see below) exceeds a certain threshold (veto threshold

).

Veto state:

- No LDO vote can be executed in veto state unless execution of this specific vote is explicitly allowed via anti-veto voting. This includes unexecuted votes started in the normal state.
- An LDO vote created in veto state cannot be executed even in normal state unless its execution is allowed by a passed anti-veto vote.
- LDO execution timelock is significantly larger than in the normal state.

Anti-veto voting:

- Any participant having their stETH locked into the escrow possesses veto power

equal to the stETH amount locked.

- While governance is in veto state, any possessor of veto power can create an anti-veto vote

for any specific LDO vote that's passed, not executed, and had no associated anti-veto vote created.

- An anti-veto vote is for the following outcome: allow execution of the specific LDO vote while the governance is in veto state.
- An anti-veto vote is active for a fixed amount of time (anti-veto voting period

) divided into two phases. During the first phase, any possessor of veto power can vote for or against the outcome. During the second phase, one can only vote against the outcome. Veto power of a participant is snapshotted at the moment they vote (not at the moment of vote creation).

- An anti-veto vote is passed if, during anti-veto voting period, it attracts participation from at least some fixed percentage of total veto power snapshotted at the moment of vote start (anti-veto quorum

), and the amount of veto power supporting the outcome exceeds the amount of opposing veto power.

- The outcome of a passed anti-veto vote becomes effective as soon as the veto voting period ends: the respective LDO vote's outcome becomes executable by anyone regardless of whether the governance is in veto state.
- Failing an the anti-veto vote (either because it doesn't get the quorum or because the majority opposes) means that the respective LDO vote's outcome becomes unexecutable under any present of future circumstances.

Transition to normal state:

- Any participant possessing veto power can initiate withdrawal of their stETH from the escrow. This immediately destroys the participant's veto power, transfers the participant's stETH into the "pending withdrawal" state, and starts a veto withdrawal delay

after which the participant can withdraw these stETH. Veto withdrawal delay should be significantly longer than anti-veto voting period.

- As soon as the total amount of stETH locked in the escrow contract minus stETH pending withdrawal becomes less than veto threshold, a timelock starts lasting enough time for ETH withdrawals from the Lido protocol to be processed (veto lift timelock

).

- During the timelock period, the governance is still in veto state.
- If, after the timelock passes, the total amount of stETH locked in the escrow contract minus stETH pending withdrawal is still less than veto threshold, the governance transitions to normal state.
- A participant whose stETH are pending withdrawal can change their mind and cancel the withdrawal. In this case, veto withdrawal delay is cancelled (meaning that if the participant decides to withdraw again, they will need to wait full veto withdrawal delay again) and veto power equal to the stETH amount is assigned to the participant.

The mechanism satisfies the first condition outlined above: solving the principal-agent problem between stakers and LDO holders. Let's discuss how we can satisfy the other two conditions: allowing the governance to recover from a captured state and not introducing new attack vectors.

Unstuck mechanism

The dual-governance setup allows stakers to block any adversarial change. However, if the governance is permanently captured, there is no way for valid LDO votes to pass, and so the protocol simply stays in a gridlocked state.

That is not a bad outcome assuming withdrawals are enabled, but if possible, we would also like the ability for governance to "unstuck itself" and move forward without falling back to social intervention aka Ethereum hardfork.

Option: burn rogue LDO after successful veto

For this, the following mechanism could be implemented:

- Change the main LDO voting mechanics to use vote escrow-locked LDO (veLDO) as the source of governance power instead of pure LDO, similar to the veCRV mechanics.
- If, for a certain LDO vote, an anti-veto vote gets the quorum and gets rejected, all LDO tokens that supported the LDO vote get burned.

This effectively redistributes the LDO governance power of the attacker or incentives-misaligned DAO members between other LDO holders, allowing the protocol governance to unstuck without involving social consensus.

A variation of the mechanics is possible: instead of burning all LDO tokens supporting the vetoed vote, burn some percentage of them (from each LDO supporter) that's proportional to the amount of veto power that voted against lifting the veto.

One important effect of this approach is that, since stETH effectively gets the power of burning LDO, the governance power of LDO is mostly transferred to stETH.

Protection from veto abuse

We also want to ensure that the veto power of stETH holders cannot be abused and doesn't open new attack vectors on stakers.

Since the described dual governance mechanism cannot be used to propose any changes, the only new attack vector it could add is governance griefing attack, where someone possessing a large stETH amount uses it to prevent any governance proposal from passing for an indefinite period of time.

This attack is hard to pull off since fair participants can always obtain additional stETH, either from market-buying or by minting from ETH, and use it for lifting the veto from specific votes. Thus, the amount of capital required to stop the governance is extremely high. However, without any additional mechanics, cost of the attack is small since stETH is withdrawable for ETH and so the attacker doesn't lose any funds unless slashing happens in the meantime. At the same time, the profit from the attack generated by e.g. opening a short position on LDO and/or stETH might be significant.

Two things can be done to further disencentivise this type of attack: making it less effective and/or more expensive. Let's discuss both of them.

Option: significant timelock instead of veto

To make the governance griefing attack less effective, the dual governance mechanics could be changed so that veto state enforces a significant (say, half a year) timelock on the execution of LDO vote's outcome instead of preventing the execution altogether.

In the case of governance capture, the timelock would give stakers enough time to withdraw their funds from the protocol (or the Ethereum community to fork the chain before withdrawals are implemented). At the same time, the governance griefing attack now makes much less sence since it can only slow the governance down instead of completely stopping it.

The obvious downside of the approach is that it makes any recovery from governance capture impossible, even if the attacker is willing to negotiate. This is especially undesirable pre-withdrawals.

Option: expropriate stETH tokens

Another way of disencentivising the attacker is increasing potential cost of the attack.

For this, escrowed stETH could be prevented from being withdrawn from the escrow (expropriated

) if they voted against the majority in an anti-veto vote that got the quorum. If the anti-veto vote didn't get the quorum, no stETH is expropriated.

That is, stETH of a participant get expropriated if:

- The participant supported an anti-veto vote, this vote got the quorum and was rejected.
- The participant opposed an anti-veto vote, this vote got the quorum and was passed.

Expropriated stETH don't contribute to the veto threshold, don't generate veto power and cannot be withdrawn from the escrow. However, LDO holders can launch a vote for releasing expropriated stETH tokens of a certain participant. This vote should also be in the scope of the dual governance.

This mechanism makes an unsuccessful attack, be it a governance capture attack or a governance griefing attack, more expensive since the capital used to apply or lift the veto can be lost.

At the same time, the ability for the fair participants to unlock the attacker's stETH (by launching an LDO vote for the unlock and lifting the veto from it via anti-veto voting) leaves room for negotiation with the attacker and gives fair participants more leverage in this negotiation.

That said, the following griefing attack is still possible:

1. Buy or mint stETH amount larger than veto threshold.
2. Lock it into the escrow => governance is transferred to veto state.
3. Immediately initiate the withdrawal.
4. After the veto withdrawal delay ends, withdraw stETH and unwrap to ETH.

The maximum cost of the attack is the opportunity cost of locking stETH for veto withdrawal delay period, and the effect is either that governance is paused for veto lift timelock period, or that fair participants are required to escrow-lock at least the same amount of stETH and actively vote for lifting the veto from all new LDO votes.

Dual governance scope

Instead of applying dual governance to any LDO vote, we propose to limit its scope to the following governance decisions:

- Upgrading the Ethereum liquid staking protocol code (see the list of contracts in the Scope of governance section).
- Managing the list of consensus layer oracle committee members.
- Managing the Ethereum node operators set and its parameters.
- Changing the total fee percentage of the Ethereum liquid staking protocol outside of the agreed boundaries (e.g. can be set between 5% and 20% by an LDO vote, anything outside this only via dual governance).
- Minting and burning LDO.

This leaves managing protocol treasury, changing protocol fee within the sane limits, and upgrading any code external to the Ethereum liquid staking protocol (i.e. not listed in the Scope of governance section) out of the dual governance scope, allowing LDO governance to execute non-critical actions witout additional friction. At the same time, any potentially hazardous decision goes through additional timelock and veto phases.

Including more governance parties

It's arguable whether aligning incentives between LDO holders and stakers is enough or more parties should be included in the governance.

One option would be to include block producers, making Lido more into an actual neutral DAO of participating node operators (in Adam Cochran's words, "Lido arguably isn't even a staking entity, but instead, a rewards incentive and protocol layer for staking entities to be able to provide staking as a standardized service"). This could be done, for example, by

requiring node operator (super)majority to approve the veto state in order for it to be applied and using quadratic weighting based on the amount of stake.

Another option would be to somehow include Ethereum users, whatever it might mean.

We think, however, that dual governance is likely enough and that Lido probably doesn't need to align more incentives than between stakers and LDO holders in order to minimize the protocol governance risks, as explained in Hasu's note.

One more argument against including more parties is smart contract risk: the more parties are involved in the governance, the more complex the interactions between them, the more the probability of unnoticed bugs in the implementation or unforseen effects in the governance mechanism design.

Implementation complexity

Discussing implementation details is out of scope of this post, but let's provide some estimates on the complexity:

- The basic dual governance mechanism

could be implemented in the form of a proxy (Aragon forwarder) sitting between the Aragon Voting contract and the governed Lido contracts. Then, all in-scope permissions together with their management rights can be re-assigned from Voting to this proxy contract. Time to ship is expected to be around 3-4 months.

- Option: burn rogue LDO after successful veto

would take extra 3–4 months on top of that.

- Option: significant timelock instead of veto

would take additional 2–3 weeks.

- Option: expropriate stETH tokens

would increase the shipment time by 2–4 weeks.

- The option of including node operators into the governance

would take additional 2–5 weeks, depending on the approach.

Next steps

Reducing the governance-related risks is the topic of importance that cannot be underestimated, especially for a protocol sitting as close to the underlying network as Lido does. At the same time, any change to the governance structure could have unforseen or second-order effects and thus should be thoroughly designed and peer-reviewed.

We invite you to contribute to the discussion and share your opinion about the proposed dual governance framework, both in general and regarding the described mechanics.

After the initial discussion, the Lido dev team will prepare and share in this thread a more technical proposal detailing the general architecture, APIs and changes required to implement the basic mechanism together with the initially-selected set of options.

Sam, Eugene, Hasu, Izzy, Vasiliy