Overrules

Overview

Being inspired by theLido's EasyTracks , Overrules allows to reduce governance load on the main Neutron DAO. Instead of creating the special motions that allow making actions on behalf of the main DAO, Overrules allow for a subDAOs to manage things by themselves still being limited by the main DAO.

It works in the following way:

- Special-purpose subDAO is created (e.g. Security subDAO, Grants subDAO, Liquidity Mining subDAO, etc.)
- This subDAO gets allocated budget from the main DAO and also gets any permissions needed
- This subDAO can now manage its budget and permissions without the need to ask the main DAO for every action
- Still, every subDAO proposal can be overruled by the main DAO within a specified timelock period

To prevent subDAOs from experiencing operational delays, the timelock period should be reasonably brief (e.g. 3 days). At the same time, overruling should be easy enough, so the threshold should be low enough (e.g. 10 times lower than regular proposal threshold).

Technical details

This section describes the technical details of the Overrules implementation. It can be interesting for developers who want to understand how it works for using Overrules in their DAO or making the integrations with Neutron DAO.

Smart contracts design

To implement the Overrules, we add several smart contracts to the DAO:

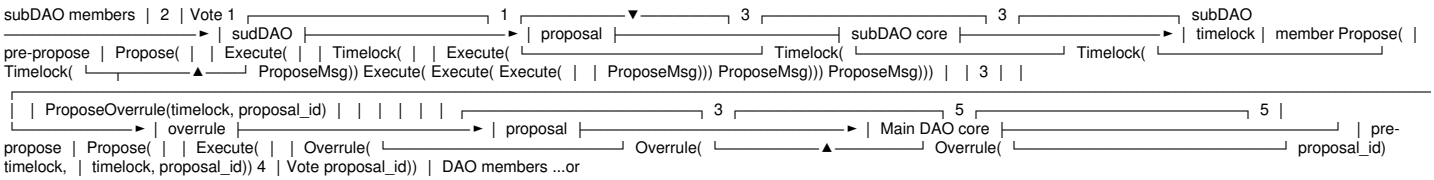
1. Overrule pre-propose module for main DAO
2. subDAO pre-propose module for subDAOs
3. Timelock contract

This design allows to implement the Overrules in a way that doesn't require any significant changes in the main DAO smart contracts. (The only change made to the main DAO is the addition of a query to check if subDAO is in the DAO's subDAOs list , also there is ExecuteTimelockedMsgs message added for SubDao Core).

How it works:

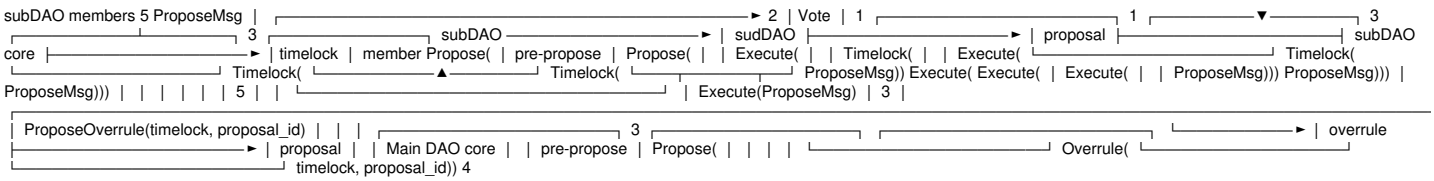
1. subDAO member submits a proposal to subDAO pre-propose module, which takes the proposal message and wraps it in aTimelockProposal
2. message.
3. subDAO members vote for the proposal and...
4. it gets executed, which means that timelock contract1. locks the subDAO proposal,
5.
    1. creates a new overrule proposal in the overrule pre-propose module of main DAO.
6. Main DAO members vote for the overrule proposal.
7. Overrule proposal is executed and subDAO proposal is rejected.

Simplified schema:



1. (same as above)
2. (same as above)
3. (same as above)
4. DAO members ignore the overrule proposal.
5. Timelocked proposal can be executed once voting period of overrule proposal is over.

Simplified schema:



DAO members If the subDAO proposal is rejected by subDAO members, everything works the same way as without any timelocks/overrules since they're not triggered.

Overrule pre-propose module allows only overrule messages to be created, thus, it takes only timelock contract address and subDAO proposal id as parameters. Title and description are generated automatically.

The proposal module for overrule proposals is just a properly configured regular SingleChoiceProposal module.

When an overrule proposal is going to be created, the overrule pre-propose module does some checks:

1. does the timelock contract correspond to the subDAO
2. is the subDAO in the DAO's subDAOs list
3. is the proposal timelocked
4. is an overrule proposal for this subDAO proposal already created

Those checks are needed to avoid spam and duplications. It's pretty crucial since the spam proposals and duplications can mislead DAO members and make the proposal that supposed to be overruled passed by washing out voting power to wrong overrule proposals. Lack of check also could allow to create overrule proposals for the unregistered subDAOs/contracts which is not great either.

For check #4 pre-propose module stores the map from pairto overrule proposal id. It also allows one to get the corresponding overrule proposal id for given subDAO proposal viaspecial query . Reverse query (get subDAO proposal id for given overrule proposal id) is also possible, one just needs to query the overrule proposal content from proposal module and get the subDAO proposal id from the proposal message.

In general, overrule proposal creation is permissionless. Still, since it's created in the very same transaction as the proposal is getting timelocked and duplications aren't allowed, there's no moment in time when one can create overrule proposal themself.

Caveats

Current implementation has several caveats:

1. The model might be a bit confusing in terms of proposal statuses. subDAO proposal now have two phases:1. subDAO-decision phase: the proposal is created, voted and executed by the subDAO. On this phase, the proposal has
2.
    1. regular statuses (e.g.Passed
3.
    1. ,Rejected
4.
    1. , etc.). Still,Executed

5.
    1. doesn't mean that the proposal is executed, it
6.
    1. means that subDAO sent the proposal to the timelock contract.
7.
    1. Timelock phase: the proposal is locked at the Timelock contract. On this phase, the proposal has statusesTimelocked
8.
    1. ,Overruled
9.
    1. ,Executed
10.
    1. ,ExecutionFailed
11.
    1. . HereExecuted
12.
    1. means actual proposal execution.
13. The overrule proposal module should be configured in a very special way:1. Obviously, it should have lower threshold and lower voting period than regular single choice proposal module.
14.
    1. Revoting should be disabled so that once threshold is reached, the overrule message can be executed.
15.
    1. Quorum should be set to the absolute percentage type so that even if significant voting power is against overruling, it
16.
    1. would happen anyway.
17.
    1. It should have no deposit since rejection of the overrule proposal is the only way to execute the subDAO proposal
18.
    1. and should be considered normal thing, no one should be punished for creation such proposal.
19. Overrules modules require both from main DAO and a subDAO to be configured in a special way:1. Main DAO should have the overrule-compatible pre-propose module.
20.
    1. subDAO should have the subDAO pre-propose module with timelocking feature.Actually, 1st requirement can be avoided by changing the overrule proposal module in a way so that it won't create
21.
    1. Proposal message based on input parameters but will validate one. It'll make it fully compatible with regular
22.
    1. pre-propose modules and allow to use it already existing pre-propose modules for overrules. However, it's a bit tricky
23.
    1. because of title and description.

Deployment

Overrule pre-propose module is deployed pretty much the same way as any other pre-propose module and doesn't have any additional init message.

subDAO pre-propose module is also deployed the same way as regular pre-propose modules. Still, it instantiates timelock module and should have the message to instantiate it (the only thing required is the overrule pre proposal module address of main DAO: it's required so that timelock module could create overrule proposals and check their statuses).

UX

While Overrules are technically just another-proposal-type for main DAO, to avoid confusion, we should have a separate UX for them. Given that, the Overrule proposal can be voted on the subDAO proposal page. Previous Overview Next Differences from DAO-DAO