Governor Dispute Guidelines

The governance process contains:

- The *Governance Dashboard*, where votes are held. This can be found on Snapshot https://snapshot.page.
- The *Governor* interface. This interface includes all disputed Lists. This can be found at https://governor.kleros.io.

The role of the governor is to implement, validate and execute the decisions taken by the communities' Governance process. Each submission to the Governor contains a list of transactions to be executed. A dispute arises when more than one list of transactions is submitted for execution.

In the event of a dispute, jurors should select the list that best produces the unenforced smart contract updates for this session.

Dispute Evaluation Process:

1) Proposals are voted on in the Snapshot interface. Navigate to Snapshot and find the project whose governance list is in question. (e.g. Kleros).

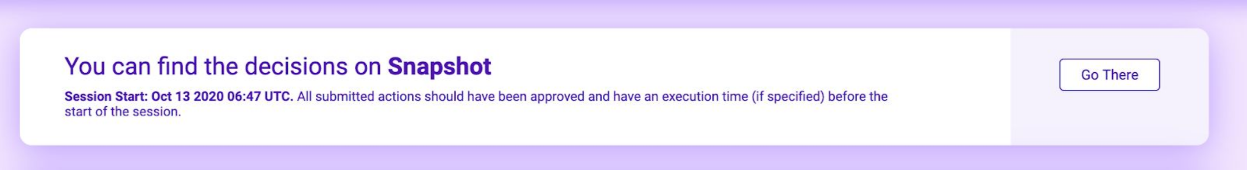
kleros
1 space(s)
Create space
Kleros
PNK

2) Find the proposals that: require **smart contract enforcement,** are **Closed** and have been **accepted before the start of the current session** of the Governor. Take note of which votes you expect to be enforced.

   In order to determine if a vote is eligible to be enforced it must meet the following criteria:

   1) The Vote passed and closed before the start of the current Governor session. You should check the Closed vote to ensure that the vote passed and therefore should be enforced. In addition Votes that close within the current Governor session should be enforced in the next Governor session. To determine when the

current Governor session started, navigate to the Governor Interface, find the matching projects Governor application and take note of the Date and Time next to “Session Start”. Any unenforced vote before this date that passed should be expected to be enforced.
E.g. The “Session Start” is Oct 7 2020, 12:00 UTC. Vote 15 Closed at Oct 7 2020, 12:01 UTC. This vote is ineligible to be enforced in the current Governor session.
E.g. Vote 16 Closed before “Session Start”, however the Vote failed to gain a majority. This vote is ineligible for enforcement.

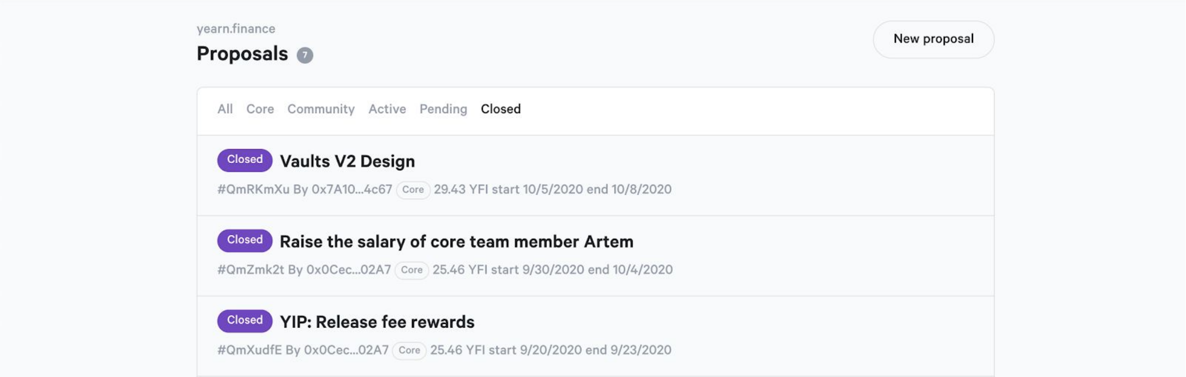


You can find the decisions on **Snapshot**

Session Start: Oct 13 2020 06:47 UTC. All submitted actions should have been approved and have an execution time (if specified) before the start of the session.

2) The Vote does not specify an enforcement date, or the enforcement date occurs before the “Session Start”. All dates and times should be assumed to be UTC unless otherwise specified.
E.g. The “Session Start” is set to Oct 7 2020, 12:00 UTC. Vote 15 Closed at Oct 7 2020, 8:00 UTC and there is nothing in the proposal about an enforcement date. This vote is eligible to be enforced in the current Governor session.
E.g. The “Session Start” is set to Oct 7 2020, 12:00 UTC. Vote 16 Closed at Oct 5 2020, 8:00 UTC. However in the proposal it states “This vote should not be enforced before Oct 8, 2020”. This vote is ineligible and should not be enforced in the current Governor session.

3) The Vote is currently unenforced and requires a smart contract call.
E.g. A vote has passed that approves a new design. There is no smart contract call needed for this action. This vote is ineligible and you should not see any enforcement.
E.g. A vote has passed that sets a contract parameter to 0. The parameter is already set to 0 and therefore does not require a contract call. This vote is ineligible and you should not see it enforced.



yearn.finance

Proposals 7

Closed **Vaults V2 Design**
#QmRKmXu By 0x7A10...4c67 Core 29.43 YFI start 10/5/2020 end 10/8/2020

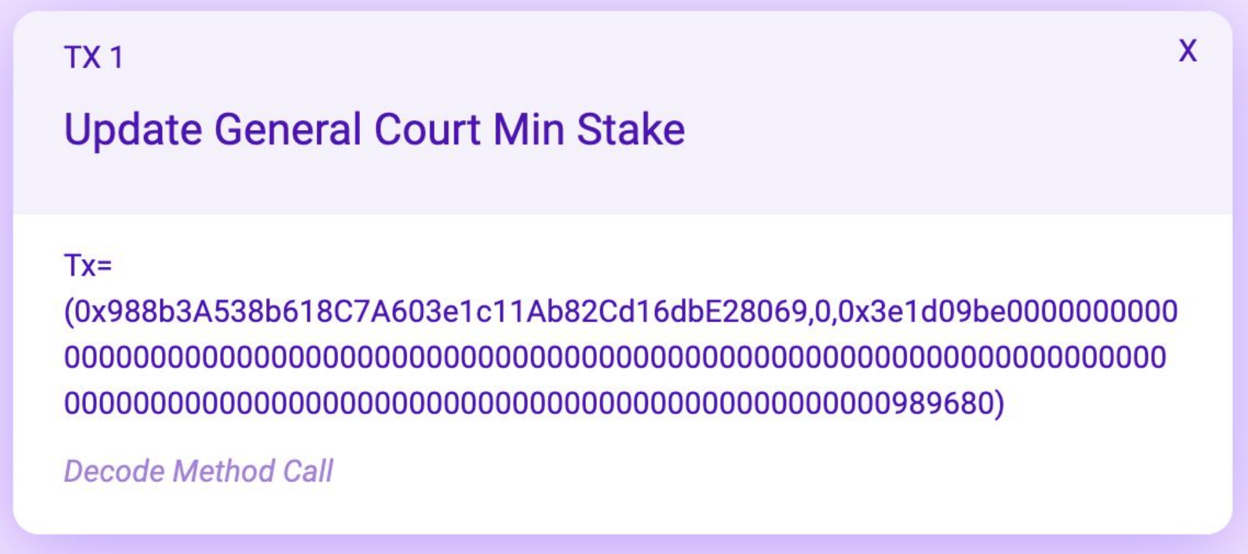
Closed **Raise the salary of core team member Artem**
#QmZmk2t By 0x0Cec...02A7 Core 25.46 YFI start 9/30/2020 end 10/4/2020

Closed **YIP: Release fee rewards**
#QmXudfE By 0x0Cec...02A7 Core 25.46 YFI start 9/20/2020 end 9/23/2020

3) Navigate to the Governor Interface. You should see at least two lists. The List Numbers correspond to the ruling options in Kleros Court. The list that ultimately wins in the dispute process is the list whose transaction calls will be executed.

4) Out of the given lists, your task is to select the list that:
    a) Accurately provides the method calls to enforce the rulings of governance votes.
    b) Most completely enforces the governance rulings.

Accuracy: Each transaction should be vetted to ensure that the target address, value and method call are correct.


TX 1

Update General Court Min Stake

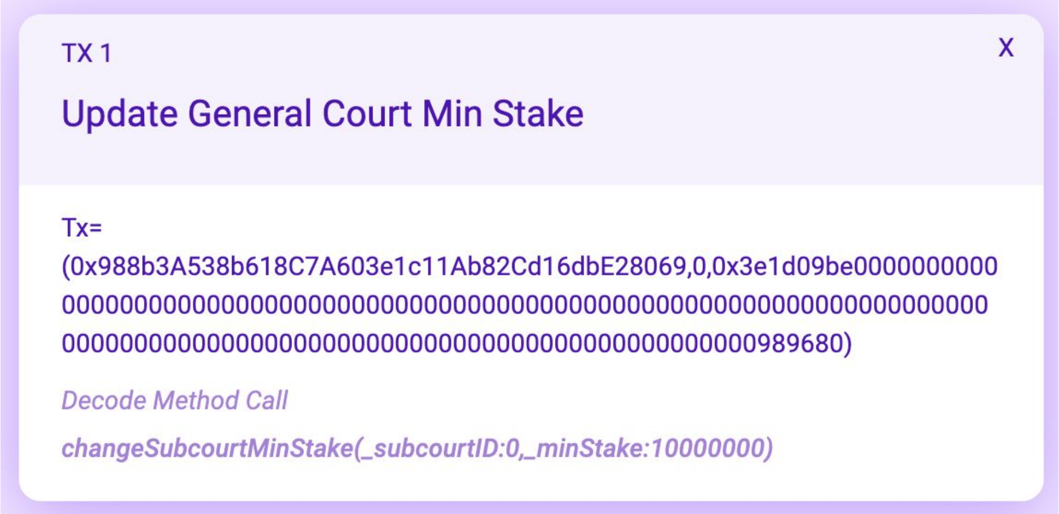
Tx=
(0x988b3A538b618C7A603e1c11Ab82Cd16dbE28069,0,0x3e1d09be0000000000
0000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000989680)

- A transaction should not be rejected solely because of the **title**. Titles are meant for informational purposes.
  E.g. A list that mixes up the titles of the transactions is still acceptable as long as the transactions themselves accurately enforce the vote.

- The first field in the transaction is the target **contract address**. This address should be carefully checked to ensure the call will be made to the correct smart contract. A list that contains a transaction with the incorrect contract address should never be accepted.
  E.g. A list containing a transaction that incorrectly targets the ERC20 contract instead of the Proxy should be immediately disqualified.

- The second field is the **value** (in WEI). In many cases this will be 0. If it is not 0, ensure that the value is needed as part of the vote and that the correct amount of WEI is specified. A list that contains a transaction with an incorrect value should never be accepted.
  E.g. A list contains a transaction that incorrectly sends 10 ETH to a function that isn't payable. This list should be immediately disqualified.

- The last field is the **bytecode data** that specifies the function call and parameters. If the smart contract source code has been verified on Etherscan, you can use *Decode Method Call* to convert the method and the parameters to a readable format in the Governor interface. Ensure that this is the correct function signature and that the parameters are accurate. The raw bytecode data should also be verified manually. This can be done by recreating the transaction and comparing the bytecode. A list containing a transaction with malformed bytecode data should never be accepted.
  E.g. A list contains a transaction that incorrectly serialized one of the parameters. This transaction is guaranteed to fail. This list should be immediately disqualified.



Update General Court Min Stake

```
Tx=
(0x988b3A538b618C7A603e1c11Ab82Cd16dbE28069,0,0x3e1d09be0000000000
0000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000989680)
```

Decode Method Call

changeSubcourtMinStake(_subcourtID:0,_minStake:10000000)

- A list that contains a transaction that makes an inaccurate contract call should never be accepted.
  E.g. A list containing a transaction with a parameter that is different than what was voted for should be immediately disqualified.

- A list that contains transactions that are not an enforcement of a vote, or are not eligible to be enforced should never be accepted.
  E.g. A list should be immediately disqualified if it contains a transaction that is an enforcement of a vote that passed, but specifies enforcement to be some time in the future.
  E.g. A list should be immediately disqualified if it contains transactions enforcing a vote that passed during the current governor session.
  E.g. A list should be immediately disqualified if it contains a transaction that takes an action that was not voted upon, or was an action for a vote that failed to pass.

Completeness: The list of transactions should completely cover all enforceable votes.

- Enforcements that require multiple transactions should never be accepted if they are missing one or more transactions that would make the remaining transactions lead to a state that is counter to the vote.
  E.g. An eligible vote requires a state update that requires a transaction to activate the new state, as well as a transaction to set a parameter that moderates how the activated feature can be used. A list that only makes the state active should be immediately disqualified because without the parameter change, activating the state by itself leads to a different outcome than what was voted on.
  E.g. An eligible vote requires 4 separate unrelated state updates. A list that contains transactions that make 3 of these state updates is still potentially eligible, but it should be noted that it is a partial enforcement.

Choosing Between Lists: Follow these steps to select the best list.

Step 1: Disqualify all lists that produce inaccurate enforcements. Follow the criteria in the Accuracy and Completeness sections to disqualify lists. If only one list remains, select that list. If there is more than one list continue to Step 2. If there are no remaining lists, choose “Refuse to Arbitrate”.
E.g. List 1 enforces Vote 25 and Vote 26 but contains a transaction with a bad parameter which disqualifies it. List 2 enforces Vote 25 accurately but is missing enforcement for Vote 26. Select “List 2”.
E.g. List 1 contains a transaction with a bad parameter. List 2 contains a partial enforcement that produces a state counter to a vote. Select “Refuse to Arbitrate”.

Step 2: If there remains more than one eligible list, select the list that produces the most complete enforcement. A list that contains enforcements that fully fulfill a vote are considered more complete than lists that contain partial enforcements. Count the number of full enforcements in each remaining list. If one list fully enforces more votes than the others, select this list. If more than one list fully enforces the highest number of votes, disqualify the other lists with fewer full enforcements and continue to Step 3.
E.g. Vote 25, 26 and 27 are eligible for enforcement. List 1 fully enforces Vote 25 and partially enforces Vote 26 and 27. List 2 fully enforces Vote 25 and 26 but does not enforce Vote 27 at all. Select “List 2”.

Step 3: Choosing between incomplete lists.

Find the list partially enforcing the most votes and disqualify lists that partially enforce less votes.
E.g. Vote 25, 26 and 27 are eligible for enforcement. List 1 fully enforces Vote 25 and partially enforces Vote 26 and 27. List 2 fully enforces Vote 25 and partially enforces Vote 26. It does not partially enforce Vote 27. Disqualify “List 2”.

If all lists contain the same number of votes being partially enforced, but one list *clearly* is partially enforcing the vote more than another, disqualify the other list. Do not disqualify either list if there is ambiguity.
E.g. Vote 25 and 26 are eligible for enforcement. List 1 fully enforces Vote 25 and partially enforces Vote 26. Vote 26 requires 3 transactions for a complete enforcement and List 1 provides 1 of the 3 transactions. List 2 fully enforces Vote 25 and partially enforces Vote 26. List 2 provides 2 of the 3 transactions required for Vote 26. Disqualify "List 1".
E.g. Vote 25, 26 and 27 are eligible for enforcement. List 1 fully enforces Vote 25 and partially enforces Vote 26. Vote 26 requires 2 transactions for a complete enforcement and List 1 provides 1 of the 2 transactions. List 1 does not enforce Vote 27. List 2 fully enforces Vote 26 and partially enforces Vote 27. Vote 27 requires 3 transactions for complete enforcement and List 2 provides 2 of 3 transactions. It does not enforce Vote 25. There is ambiguity as to which list is more complete, do not disqualify either list.

If only one list remains, select that list. If there are still multiple eligible lists continue to Step 4.

Step 4: At this stage there is no objective way to determine which list of transactions is more correct. No remaining list should contain any transactions that are not accurate or would produce a result counter to anything voted upon. However each list is not providing complete enforcement of the eligible votes. In this situation, the List that was submitted first should be selected.
E.g. List 1 and List 2 both contain the same number of accurate full enforcements and similar amounts of partial enforcements. List 1 was submitted before List 2. Select "List 1".