

# BCM Process

This document can be found on Arweave [here](#).

**The Beanstalk Community Multisig, or BCM**, custodies ownership of the [Beanstalk contract](#). The BCM has the exclusive and unilateral ability upgrade Beanstalk. In the future, it is expected that BIPs remove governance entirely, revoking these abilities from the BCM.

The BCM is not intended to have decision making power. Its role is to enact on-chain the decisions Stalkholders make via off-chain voting and review and verify proposals to ensure the suggested changes are truthfully represented.

The BCM is deployed using [Safe](#), the most battle-tested multisig contract on Ethereum. Its m-of-n configuration started as 5-of-9 on Ethereum mainnet. Parameters m and n are each ultimately defined by Stalkholders and may evolve in the future via Beanstalk Improvement Proposal (BIP).

BCM Signers are an [anonymous](#) and diverse set of reputable community members and Beanstalk core contributors.

USDC from Fertilizer sales were custodied by the BCM between the beginning of the Barn Raise on June 6, 2022 and Replant on August 4, 2022.

- [Multisig Powers](#)
- [Snapshot Usage](#)
- [BIP Proposal and Voting](#)
- [BOP Proposal and Voting](#)
- [BIR Proposal and Voting](#)
- [Signer Best Practices](#)
- [Signer Duties](#)
- [Verifying and Signing Transactions](#)
- [Emergency Response Procedures](#)
- [Rotating Signers](#)

—  
•

- [Anonymous Signers](#)

## Multisig Powers

Beanstalk implements the [EIP-2535 Diamond Standard](#), a standard for fully-upgradeable smart contracts.

The mechanism for upgrading a Diamond is by calling `diamondCut`, which takes arguments of functions to replace and which functions to replace them with. The `diamondCut` function is only callable by the owner of Beanstalk, which is the BCM.

Upgrades should only be executed after a proposal has passed and Signers have manually reviewed the transaction. However, in the case of an emergency (like a bug or vulnerability), the BCM may execute an Emergency BIP (EBIP) to protect the Beanstalk contract. The best practices for emergency response handling are outlined in the [#emergency-response-procedures](#) section.

If a Stalkholder wants to propose a BIP, they can create a pull request to the [Beanstalk GitHub repo](#) and begin a formal proposal process outlined in the [#bip-proposal-process](#) section.

The BCM also executes the will of the [Beanstalk Immunefi Committee \(BIC\)](#) as determined by Beanstalk Immunefi Responses (BIRs). See [#bir-proposal-and-voting](#).

The following functions are only callable from the owner address:

- `diamondCut` — Add, replace and/or remove any function(s) and/or execute an `init` function with a `delegatecall`.
- `whitelistToken` — Add a token to the Deposit Whitelist.
- `dewhitelistToken` — Remove a token from the Deposit Whitelist.
- `pause` — Pause Beanstalk, which makes it such that the `gm` function cannot be successfully called.
- `unpause` — Unpause Beanstalk, which allows the `gm` function to be successfully called at the top of the 2nd hour. The TWAP oracle and Season

timer are reset as well.

- `createFundraiser` — Create a Fundraiser.
- `transferOwnership` — Transfer ownership of the Beanstalk contract to a new address.
- `addUnripeToken` — Add an Unripe token to Beanstalk.

## Snapshot Usage

The BCM is an extension of the Beanstalk DAO. As such, the role of the BCM is to (1) enact on-chain the decisions Stakeholders make via off-chain voting and (2) review and verify proposals to ensure the suggested changes are truthfully represented.

BIPs are voted on at the [Beanstalk DAO Snapshot space](#).

The BCM shall not execute transactions until an associated Snapshot proposal successfully passes, apart from the exceptions outlined in the table below:

Transaction	Snapshot?	Voting Period
Executing BIP	Yes, requires a passed BIP	Up to 7 days as outlined in <a href="#">#bip-voting</a>
Non-emergency change to the m-of-n BCM configuration	Yes, requires a passed BIP	Up to 7 days as outlined in <a href="#">#bip-voting</a>
Executing BIR	Yes, requires a passed BIR	3 days
Executing EBIP (emergency hotfix, etc.)	No, but requires public notification via Discord	N/A
Rotating BCM Signers	No	N/A
Emergency change to the m-of-n BCM configuration	No	N/A

Transaction	Snapshot?	Voting Period
to remove rogue Signers		
Cancel transaction	No	N/A

## BIP Proposal and Voting

Beanstalk's governance is designed to be as permissionless as possible. Any Stalkholder with 0.1% of total Stalk may propose BIPs. If a Stalkholder wishes to propose a BIP they must complete a public proposal process before the BCM will submit a Snapshot proposal and an on-chain transaction to the BCM on their behalf.

Past BIPs can be found [here](#).

## BIP Proposal Content


BIPs can propose to (1) change the Beanstalk protocol, (2) mint Beans and/or (3) change Beanstalk governance. BIPs consist of a pull request on the Beanstalk GitHub repo that includes the following:

- Code implementing the proposed changes (or simply a change to the `PROPOSALS.md` file if there are no associated code changes);
- A written proposal of the changes that would be implemented in the BIP, which must include:
  - A **Proposer** section that includes an [Etherscan verified message](#) (or an Arweave upload if the proposing wallet is a multisig) confirming the proposer of the BIP;
  - A **Contract Changes** section that includes the facets and the `init` contract address in the `diamondCut` call, if applicable; and
  - A **Beans Minted** section that describes the number of Beans minted by the BIP, if applicable.

BIPs may have multiple transactions if made explicit in the written proposal.

# BIP Proposal Process

The following are the processes in place for a Stalkholder to propose a BIP:

1. Submit a pull request on the public Beanstalk GitHub repo with a written proposal of the changes that would be implemented in the BIP.
2. Tag the **@BCM Liaison** user in the (**# • governance**) channel in the Beanstalk Discord to create a dedicated discussion channel for the BIP.
3. Share a link to the GitHub PR and the written proposal in the newly created dedicated discussion channel.
4. Allow sufficient time for discussion of the proposal. What constitutes sufficient is at the sole discretion of the BCM.
5. Tag the **@BCM Liaison** user to request that the BIP be formally proposed.
6. Before the BCM formally proposes the BIP, they shall verify that the written proposal contains all the necessary content per [#bip-proposal-content](#).
7. The BCM shall then formally propose the BIP by submitting the on-chain transaction and Snapshot proposal.
8. During the Voting Period (1-7 days), every BCM Signer shall verify the transaction per [#verifying-and-signing-transactions](#). If not all Signers verify the transaction, the BCM may still continue per the process outlined in [#rotating-signers](#).
9. If the BIP passes, the Signers will sign m/n signatures and execute the on-chain transaction as soon as possible. If the BIP fails to pass, the Signers will submit and execute a cancel transaction with the same nonce as soon as possible.

## BIP Voting

Voting for BIPs takes place on Snapshot using Stalk at the beginning of the Voting Period that still exists.

Any Stalkholder can vote For, Abstain or Against on any BIP. In all instances, 1 Stalk equals 1 vote, and not voting or voting Abstain is equivalent to voting Against.

The Voting Period opens when the Snapshot proposal for a BIP can be voted on and ends after 7 days or once a two-thirds supermajority is reached.

If at the end of the Voting Period:

- Less than or equal to half of total Stalk is voting For the BIP, it fails, or
- More than half of total Stalk is voting For the BIP, it passes.

If at any time 24 hours or more after the beginning and before the end of the Voting Period more than two-thirds of the Stalk supply votes in favor of the BIP, the BCM can execute the BIP on-chain.

## BOP Proposal and Voting

Any Stalkholder with 0.1% of total Stalk may propose BOPs. If a Stalkholder wishes to propose a BOP they must complete a public proposal process before the BCM will submit a Snapshot proposal on their behalf.

Past BOPs can be found [here](#).

## BOP Proposal Content

BOPs can propose for the DAO to agree on processes that do not involve (1) changing the Beanstalk protocol, (2) minting Beans or (3) changing Beanstalk governance. BOPs consist of a pull request on the Beanstalk GitHub repo that includes the following:

- Updates to the `PROPOSALS.md` file;
- A written proposal of the changes that would be implemented in the BOP, which must include a **Proposer** section that has an [Etherscan verified message](#) (or an Arweave upload if the proposing wallet is a multisig) confirming the proposer of the BOP.

## BOP Proposal Process

The following are the processes in place for a Stalkholder to propose a BOP:

1. Submit a pull request on the public Beanstalk GitHub repo with a written proposal of the changes that would be implemented in the BOP.
2. Tag the **@BCM Liaison** user in the (**#🏛️ • governance**) channel in the Beanstalk Discord to create a dedicated discussion channel for the BOP.
3. Share a link to the GitHub PR and the written proposal in the dedicated discussion channel.
4. Allow sufficient time for discussion of the proposal. What constitutes sufficient is at the sole discretion of the BCM.
5. Tag the **@BCM Liaison** user to request that the BOP be formally proposed.
6. Before the BCM formally proposes the BOP, they shall verify that the written proposal contains all the necessary content per [#bop-proposal-content](#).
7. The BCM shall then formally propose the BOP by submitting the Snapshot proposal.

## BOP Voting

Voting for BOPs takes place on Snapshot using Stalk at the beginning of the Voting Period that still exists.

Any Stalkholder can vote For or Against on any BOP. The Voting Period opens when the Snapshot proposal for a BOP can be voted on and closes after 7 days.

If at the end of the Voting Period:

- Less than or equal to 35% of total Stalk is voting For or the majority of participating Stalk is voting Against the BOP, it fails, or
- More than 35% of total Stalk is voting For and the majority of participating Stalk is voting For the BOP, it passes.

## BIR Proposal and Voting

The BIC determines the categorization and payout of bug bounties in accordance with the [ImmuneFi Bug Bounty Program](#) approved by the Beanstalk DAO. The BCM executes the will of the BIC as determined by BIRs.

Any BIC member may propose BIRs. Past BIRs can be found [here](#).

## BIR Proposal Content

BIRs can propose to mint Beans to reward valid bug reports on ImmuneFi. BIRs consist of a Snapshot proposal that includes information about the bug report and has a **Beans Minted** section that describes the number of Beans minted to both the whitehat and ImmuneFi.

## BIR Proposal Process

See [BIC Process](#).

## BIR Voting

The following are the processes in place for the BIC to propose and have the BCM execute a BIR:

1. The BIC shares BIR info with the BCM for them to submit a corresponding on-chain transaction.
2. The BCM shall then formally propose the BIR by submitting the on-chain transaction.
3. The BIC shall then formally propose the BIR on the [Beanstalk Bug Bounty Snapshot space](#).
4. During the 3 day Voting Period, every BCM Signer shall verify the transaction per [#verifying-and-signing-transactions](#). If not all Signers verify the transaction, the BCM may still continue per the process outlined in [#rotating-signers](#).
5. If the BIR passes (two-thirds majority of BIC members voting For), the Signers will sign m/n signatures and execute the on-chain transaction as soon as



possible. If the BIR fails to pass, the Signers will submit and execute a cancel transaction with the same nonce as soon as possible.

## Signer Best Practices

BCM Signers shall follow the best practices outlined below. It is of paramount importance that Beanstalk limits key man risk by implementing best practices with respect to multisig key custody. Signers are expected to:

1. Regularly check in with the rest of the BCM and confirm access to their wallet;
2. Maintain active communication regarding travel plans and availability in order to ensure that there are always enough Signers on call; and
3. Acknowledge their Signer duties and processes for signing off on BIPs and BIRs.

In addition to the above expectations, Signers shall follow the BCM's wallet security best practices:

1. Use a reputable hardware wallet like Trezor or Ledger;
2. Use a fresh address that doesn't have any pre-existing transactions or balances on it;
3. Set up a new passphrase on their hardware wallet device when selecting a new address to be the signing address; and
4. Follow the standard self-custody best practice guide [here](#).

## Signer Duties

Signers are expected to follow best practices and maintain active communication in order to ensure that there are always enough Signers available to execute transactions in case of an emergency. Signer duties are broken down into three stages: (1) confirming access to their wallet, (2) verifying proposed transactions and (3) executing transactions on the BCM.

Once the Voting Period for a proposal begins on Snapshot, all Signers are expected to promptly review and verify that the proposed transaction accurately reflects the Snapshot proposal per [#verifying-and-signing-transactions](#). This limits blind signing

and encourages each Signer to independently verify that a proposed transaction is accurately represented. Anyone can verify that a Signer verified a proposed transaction during the Voting Period.

As soon as possible after the Voting Period:

- If the proposal passed, Signers who have verified the transaction will sign and execute the transaction; or
- If the proposal failed, Signers will sign and execute a cancel transaction with the same nonce.

A Signer shall lose their role on the BCM (by the remaining Signers removing them) if they:

- Act against Stalkholders' off-chain voting (or the BIC's off-chain voting in the case of BIRs);
- Do not follow best practices outlined in the [#signer-best-practices](#) section; or
- Get through 2 votes without performing any of their Signer duties.

## Verifying and Signing Transactions

All BCM Signers are expected to know how to verify `diamondCut` data and confirm they have verified transactions by creating an Etherscan verified message.

The following should be used as a guide for the minimum review criteria in cases where a `diamondCut` is being executed:

- Performing the `diamondCut` on a local mainnet fork and confirm that the facet addresses, function selectors and facet cut actions are correct;
- Confirming that the `_init` address and `_calldata` are correct; and
- Confirming that the deployed bytecode for changed facets matches the corresponding bytecode compiled from the GitHub PR branch.

Once Signers have verified the transaction, they shall sign a message indicating that they have verified the transaction by creating an [Etherscan verified message](#).

This process must be completed for each transaction in a BIP or BIR. See the [BCM Verification](#) page for more information on how BCM Signers verify transactions.

If a situation arises where not all Signers submit a verification, the BCM may continue with execution if the Signer:

1. Was not responsive during the draft phase of the proposal or Voting Period;
2. Notified the rest of the BCM during the draft phase of the proposal that they would not be able to verify, sign or execute the transaction;
3. Missed 2 verifications;
4. Was not responsive in 2 months; or
5. Votes against the outcome of any proposal, in which case the Signer would be subject to removal immediately.

## Problems During Verification

The BCM will not execute a transaction that was misrepresented in the Snapshot proposal.

In the case that any Signer during the verification process determines that a Snapshot proposal does not accurately represent the transaction, that Signer will sign an Etherscan verified message indicating as such with context on the issue.

The BCM will respond as follows:

- If the BCM determines that the Signer is "rogue" and attempting to censor the transaction, the BCM will indicate that by continuing to verify (and ultimately sign and execute, in the case that the proposal passes) the transaction; or
- If the BCM determines that the Signer is not rogue, the BCM will indicate that by submitting, signing and executing a cancel transaction with the same nonce.

In cases where the BCM cancels a transaction, the community will be notified via the Beanstalk Discord and the BCM will work with the proposer to resolve the issue. Once resolved, a new BIP will be proposed on Snapshot with its associated transaction.

# Emergency Response Procedures

Although the BCM's role is to enact on-chain the decisions Stalkholders make via off-chain voting, it is of critical importance that the BCM take swift action to protect Beanstalk in the event of a bug or vulnerability. Emergency upgrades to Beanstalk submitted by the BCM are known as EBIPs.

Bugs or security vulnerabilities qualify as emergencies. Emergency action will not be taken for any reason related to the economic health of Beanstalk.

Past EBIPs can be found [here](#).

Depending on the severity of a given emergency, BCM members shall swiftly decide the best course of action:

- If a bug or vulnerability is severe and requires significant code changes to fix, the BCM may remove functions from Beanstalk and take any necessary extra action to mitigate further damage; or
- If a bug or vulnerability is minor and does not require significant code changes to fix, a hotfix may be implemented by the BCM.

In cases where functions must be removed immediately to protect Beanstalk, BCM Signers are not required to submit an Etherscan verified message as outlined in [#verifying-and-signing-transactions](#).

After emergency action is taken, the BCM shall swiftly issue a summarized report to the community via the Beanstalk Discord detailing:

1. The administrative permissions that were used (*e.g.*, which functions were removed);
2. The context and severity of the issue; and
3. The next steps and decisions, if any, that the community must agree on in order to proceed.

## EBIP During Voting Period

In the event that an EBIP must be executed during the Voting Period of another proposal, the on-chain transaction for the outstanding proposal must be canceled.

In order avoid the need to revote on a proposal in this situation, the BCM will repropose the on-chain transaction for the outstanding proposal after the successful execution of the EBIP.

## Hypernative

[Hypernative](#) actively detects and responds to high confidence pre-exploit and exploit-in-progress detections on Beanstalk. Hypernative custodies [an EOA](#) that has the ability to call functions on [the Hypernative module](#) added to the BCM. This module has the ability to remove all functions from Beanstalk except for `diamondCut`, `facetAddress`, `facetAddresses`, `facetFunctionSelectors` and `facets`. Removing all other functions protects the protocol in the event of an attack.

The BCM continues to be the owner of and the sole address capable of adding and updating functions on Beanstalk. Instances where Hypernative removes all non-diamond functions will be considered EBIPs, i.e., they will be documented as such and follow the DAO-approved procedures outlined above.

## Rotating Signers

In the event that one or more Signers are compromised, vote against the outcome of any proposal, voluntarily choose to be removed from the BCM, or are otherwise removed in accordance with [#signer-duties](#), the BCM will rotate them out of the multisig and replace them with another Signer. In no instance shall a majority of the BCM keys be held by Beanstalk Farms contributors, nor shall more than 1 key be held by Publius.

## Anonymous Signers

Off-chain governance introduces significant risks related to security and censorship. The BCM is designed to mitigate as many of those risks as possible by distributing the multisig keys across reputable community members and Beanstalk core contributors, and collectively adhering to [#signer-best-practices](#).

The most significant risk associated with off-chain governance is the potential corruption of the BCM from an outside party. In order to minimize the chances of this, the Signers are anonymous. The anonymous Signers are selected by Publius. Signers will be anonymous to each other as well, apart from Publius.

In no instance shall a majority of the BCM keys be held by Beanstalk Farms contributors. Publius will collectively hold at most 1 key. The remaining keys will be held by reputable members of the Beanstalk community.

## Malicious Key Holder Risk

Under this structure, it's important to acknowledge the risk of anonymous key holders conspiring to attack Beanstalk. Because only Publius knows the identities of the anonymous Signers, Publius would be the main attack vector. If a malicious actor were to compromise Publius before conspiring to attack Beanstalk, they could be reasonably sure that their identity would never be revealed.

In order to mitigate this attack vector, the BCM institutes the following process whenever the m-of-n configuration of the BCM is changed:

- Publius will publish a hash of the list of Signers and their corresponding wallets on Etherscan (you can find these under the Signer Hashes section of the [BCM Dashboard](#)); and
- Publius will share the list of Signers and their wallets with their personal legal counsel, to be released in the event that Publius is compromised such that they cannot publicly share the list themselves. This makes Publius and their personal legal counsel the only parties with access to the list.

In the event that Publius' counsel publicly shares the list of Signers and their wallets, anyone could verify that it's the correct list by hashing it and comparing it with the hash on Etherscan. This creates accountability for the anonymous Signers.