Quantstamp

Berachain Native Smart Contracts

Executive Summary

This audit report was prepared by Quantstamp, the leader in blockchain security.

Type	Staking, Stablecoin, and Governance
Timeline	2024-11-18 through 2025-01-14
Language	Solidity
Methods	Architecture Review, Unit Testing, Functional Testing, Computer-Aided Verification, Manual Review
Specification	Proof-of-Liquidity (Protocol Level) Proof-of-Liquidity (Native Smart Contracts) Proof-of-Liquidity (Transaction Flow Diagrams) Honey v2 Documentation
Source Code	• berachain/berachain-contracts-monorepo #97608d2 • berachain/berachain-contracts-monorepo #b9e2469 • berachain/berachain-contracts-monorepo #b1a445b
Auditors	• Joseph Xu Technical R&D Advisor • Leonardo Passos Senior Research Engineer

Documentation quality	Medium
Test quality	High
Total Findings	15 Fixed: 6 Acknowledged: 5 Mitigated: 4
High severity findings	0
Medium severity findings	2 Fixed: 1 Mitigated: 1
Low severity findings	7 Fixed: 2 Acknowledged: 3 Mitigated: 2
Undetermined severity findings	1 Fixed: 1
Informational findings	5 Fixed: 2 Acknowledged: 2 Mitigated: 1

Summary of Findings

Final Report (2025-01-14): Quantstamp has reviewed an additional commit hash `5bc04c6` containing additional fixes. The commit contains a major update to the `/src/honey/HoneyFactoryReader.sol` contract, as well as a variety of fixes for other issues and suggestions in the Initial Report. As of this commit, a majority of the salient issues identified in the Initial Report has either been Fixed or Mitigated. The remaining issues that are Acknowledged are all Low or Informational severity. These issues are also operational in nature, and should pose little risk or impact for the community if the Berachain team operates with proper safeguards.

Updated Report (2024-12-31): Quantstamp has reviewed the responses by Berachain team, including an updated commit for the `honey` directory at commit hash `71bc9b2`. The Berachain team has put in considerable thought in addressing the issues and concerns raised in the Initial Report, as well as providing extra contexts for the codebase and the ecosystem. As a result, the auditors have decided to downgrade many of the severity assessments in this Updated Report, and also remove some issues that were either False Positive or non-critical.

The Updated Report contains 15 issues, with 2 issues of Medium severity that have been either Fixed or Mitigated. The remaining 13 issues of Low, Information, and Undetermined severity are mostly Fixed, Mitigated, or with fixes planned.

Initial Report (2024-12-13): Quantstamp has conducted an audit of the Berachain native smart contracts. These smart contract functionalities include:

1. Facilitation of the Proof-of-Liquidity (PoL) consensus mechanism (`base` and `pol` directories at commit hash `97608d2`)
2. Berachain's native stablecoin called HONEY (`honey` directory at commit hash `b9e2469`)
3. Berachain governance (`gov` directory at commit hash `b1a445b`).

The auditors believe that the system is generally well-documented, well-tested, and well-implemented. However, there appears to be some gaps in the assumptions when designing the PoL consensus mechanism and a specific feature called "basket mode" for minting and redeeming the HONEY stablecoin. This has lead the auditors to classify 6 issues as High severity, and 1 issue as Medium severity, as these issues are likely to occur while negatively impacting protocol security and user funds in a way that are contrary to the intentions of the Berachain team.

In addition, the auditors have identified 12 additional issues of Low, Informational, and Undetermined severity around the PoL mechanism and HONEY stablecoin. These issues are unlikely to cause financial loss for the protocol or users directly, but may cause problems operationally or in terms of user experience.

The auditors strongly recommend addressing all findings, with emphasis on the High severity issues around the PoL consensus mechanism and around the basket mode feature.

ID	DESCRIPTION	SEVERITY	STATUS
BERA-1	**The Ecosystem Tends Towards Centralization More than Expected**	● Medium ⓘ	Mitigated
BERA-2	**Mismatch in Basket Mode Activation Condition May Cause Financial Losses to Users**	● Medium ⓘ	Fixed
BERA-3	**Changes to the `globalCap` Parameter May Block Minting**	● Low ⓘ	Acknowledged
BERA-4	**Potential Censorship of `distributeFor()` Transactions**	● Low ⓘ	Mitigated
BERA-5	**Precision Is Not Enforced in the Setter Functions that Control Block Reward Rates**	● Low ⓘ	Mitigated
BERA-6	**Asset Parameters Silently Overriden at Vault Creation**	● Low ⓘ	Acknowledged
BERA-7	**`globalCap` Parameter Has No Minimum, Which Can Lead to Denial of Service**	● Low ⓘ	Acknowledged
BERA-8	**Recapitalization Does Not Check for the Peg Status**	● Low ⓘ	Fixed
BERA-9	**`HoneyFactoryReader.sol` Is Not Fully Basket Mode Compatible**	● Low ⓘ	Fixed
BERA-10	**Deterministic Deployment Proxy May Be Tampered With**	● Informational ⓘ	Fixed
BERA-11	**Mint and Redeem Rates Produced by `HoneyFactoryReader.sol` Is Not Guaranteed**	● Informational ⓘ	Mitigated
BERA-12	**The Current Maximum Limit on the Reward Vault Incentive Rate Seems Excessively High**	● Informational ⓘ	Acknowledged
BERA-13	**Conversion Between BGT and BERA Is Dependent on the Consensus Layer**	● Informational ⓘ	Acknowledged
BERA-14	**Missing `__*_init()` Calls to Openzeppelin Library Contracts Within Various Initializers**	● Informational ⓘ	Fixed
BERA-15	**Inconsistent Indexing of Token Addresses in the Return Array of `previewRequiredCollateral()` Function**	● Undetermined ⓘ	Fixed

Assessment Breakdown

Quantstamp's objective was to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices.

Disclaimer

Only features that are contained within the repositories at the commit hashes specified on the front page of the report are within the scope of the audit and fix review. All features added in future revisions of the code are excluded from consideration in this report.

Possible issues we looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Mishandled exceptions and call stack limits
- Unsafe external calls
- Integer overflow / underflow
- Number rounding errors
- Reentrancy and cross-function vulnerabilities
- Denial of service / logical oversights
- Access control
- Centralization of power
- Business logic contradicting the specification
- Code clones, functionality duplication
- Gas usage
- Arbitrary token minting

Methodology

1. Code review that includes the following
   1. Review of the specifications, sources, and instructions provided to Quantstamp to make sure we understand the size, scope, and functionality of the smart contract.
   2. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
   3. Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to Quantstamp describe.
2. Testing and automated analysis that includes the following:
   1. Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.
   2. Symbolic execution, which is analyzing a program to determine what inputs cause each part of a program to execute.
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarity, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.
4. Specific, itemized, and actionable recommendations to help you take steps to secure your smart contracts.

Scope

Files Included

Proof-of-Liquidity

```
Repo: https://github.com/berachain/contracts-monorepo/97608d28b2b36777bbbd2616423e99a65dfca383
Files: src/base/*; src/pol/*
```

HONEY stablecoin

```
Repo: https://github.com/berachain/contracts-monorepo/b9e24693e31c577f4281bb652afd7936d7b95acb
Files: src/honey/*
```

Governance

```
Repo: https://github.com/berachain/contracts-monorepo/b1a445b8373a652a71984eef640345530d28ed70
Files: src/gov/*
```

Findings

BERA-1
The Ecosystem Tends Towards Centralization More than Expected

● Medium Mitigated

Update

Berachain team has indicated that this issue has been thoroughly studied by their internal quant team and a specialized economic auditor. The analysis includes both heuristic assessment as well as a comprehensive agent-based simulation.

In addition, concrete measures have been taken to mitigate the centralization risk, such as: (1) using a concave reward curve that favors validators with little BGT boost to incentivizes BGT holders to split their boost among different validators, (2) introducing a cap on the maximum quantity of BERA staked per validator, (3) intending to support smaller validators through the Berachain Foundation, and (4) possibly using Governance to blacklist Reward Vaults that are exploiting PoL.

Given these contexts, the auditors have decided to downgrade the severity to Medium.

File(s) affected: `src/pol/rewards/BlockRewardController.sol`, `src/pol/rewards/Distributor.sol`, `src/pol/rewards/RewardVault.sol`

Description: The current Proof-of-Liquidity (PoL) consensus mechanism works as follows:

- Validators stake the native BERA gas token to be able to produce blocks
- Block producing validators receive the reward and governance BGT token as block reward
- The validators can choose to distribute BGT tokens to liquidity providers or users, who can then choose to stake their BGT token for a validator ("boost" a validator), which would increase the future BGT reward for the boosted validator
- BGT tokens can also be redeemed 1:1 with the BERA token

While the PoL mechanism currently models validators, liquidity providers, and users as separate entities, they can be the same entity in reality. Given the current financial incentives, playing multiple roles within the Berachain ecosystem to cycle the BGT tokens internally would maximize the reward earnings. Rich liquidity providers, for instance, may act as validators. Once these validators earn BGT block rewards, they can direct these to their own Reward Vaults. Those BGT tokens, in turn, can be used to further boost their own validator, and consequently, increase the future BGT rewards when producing blocks.

The described cycle can continue indefinitely. Liquidity providers or validators who succeed on this strategy will likely draw more boosting from others (assuming they are reliable validators), as well as more liquidity to the pool(s) invested by the liquidity providers. The speed at which BGT whales emerge may be faster than expected, and centralization can occur at both the governance and protocol level. The consequences of centralizations include:

- Berachain becoming an oligarchy, as governance decisions is directly related to the amount of BGT token holdings.
- Deployment of new Reward Vaults become restricted, as the oligarchy in place will likely censor outsiders willing to approve new Reward Vaults different from the ones they have directly invested in.
- Block production may also become restricted, due to the censorship risk described in the "Potential Censorship of `distributeFor()` Transactions" issue.

Recommendation: We recommend refining the economic model on the Proof-of-Liquidity (PoL) consensus mechanism to discourage the same entity to play multiple roles.

To bring a better understanding of how much such an attack would cost, we also suggest studying the risk of centralization using a model of PoL consensus mechanism that allows the same entity to play multiple roles. We suggest the following metrics to measure the risk of possible centralization:

- The amount of time/blocks it would take for two or three entities to gain a supermajority of the staking or governance power, given a static validator/liquidity provider set.
- The financial resource needed for two or three entities to gain a supermajority of the staking or governance power within a given amount of time/number of blocks, given a static validator/liquidity provider set.

BERA-2
Mismatch in Basket Mode Activation Condition May Cause Financial Losses to Users

● Medium ⓘ Fixed

Update

Berachain team has added an additional argument `bool expectBasketMode` to the functions `HoneyFactory.mint()` and `HoneFactory.redeem()` that will force users to signal their awareness of the basket mode activation. This prevents unexpected mint or redeem while the basket mode is active.

Given this change and the surrounding usage context, the auditors have decided to downgrade the severity to Medium.

File(s) affected: `src/honey/HoneyFactory.sol`

Description: When collateral assets (assumed to be stablecoins) start getting depegged or blacklisted, the `HoneyFactory.sol` contract will require transfer of all collateral assets when minting or redeeming the HONEY token. This is referred as "basket mode". The basket mode is enabled on mint if all collateral assets are depegged or blacklisted, while the basket mode is enabled on redeem if at least one collateral asset is depegged or blacklisted. The minting and redeeming operation in basket mode is performed in a way such that the relative weights of the collateral assets held by the system remain fixed.

While the basket mode is a security feature to avoid arbitrage opportunities in the face of depegging, it may cause severe losses to users minting and redeeming HONEY. A sample scenario is described in the Exploit Scenario section. This loss is caused by the fact that the basket mode is activated by different conditions for mint and redeem, and it may cause a user to trade a good collateral asset for a basket of good and bad collaterals.

Exploit Scenario:
Suppose that there are two collateral vaults of stablecoin assets, say X and Y, with weights of 90% and 10% respectively.

1. X depegs, while Y does not. When X depegs, its is quoted at 90 cents; Y is quoted at 100 cents.
2. Bob mints HONEY tokens using using 10,000,000 Y tokens (the pegged asset), equivalent to the same amount in dollars.
3. Since basket mode is not enabled on mint (there is at least one good collateral asset), Bob receives 9,980,000 HONEY, assuming a mint/redeem rate of 0.998.
4. Bob then decides to redeem his HONEY. In the case of redeem, basket mode is enabled whenever at least one asset is depegged (X in this case).
5. Bob will then receive two assets with the following valuation:
   - X: 9,980,000 * 0.9 = 8,982,000 * 0.9 * 0.998 = $8,067,632.4
   - Y: 9,980,000 * 0.1 = 998,000 * 0.997 * 0.998 = $993,015.99

1. Bob is now left with $9,060,648.4, almost $1,000,000 less than what he had initially used for minting.
2. Bob is confused as why he is receiving a depegged asset (X), since he minted HONEY using a pegged one (Y).

Recommendation: The basket mode is an intelligent approach for preventing arbitrage against the HONEY stablecoin system. However, one has to make sure that users are aware of its potential side effects once some assets start depegging. Therefore, we recommend improving the existing user-facing documentation to make users aware of all the risks involved. We also recommending documenting and making transparent policies and action plans in case of depegging events, which may include a pause of the mint/redeem functionality.

We also suggest allowing users to specify whether they want the transaction to fail if basket mode is activated, or how much the rate can deviate from a certain output value, on mint and redeem functions.

BERA-3 Changes to the `globalCap` Parameter May Block Minting

• Low ⓘ Acknowledged

Update
Berachain team has indicated that this procedure is intended to prevent the `globalCap` threshold breach due to a redemption operation. The same procedure has been used for the mint operation based on the design specification. Berachain team has indicated that this issue can be circumvented by choosing the appropriate `relativeCap` and `globalCap` parameter.

Given these contexts, the auditors have decided to downgrade the severity to Low.

File(s) affected: `src/honey/HoneyFactory.sol`, `src/honey/VaultAdmin.sol`

Description: The `HoneyFactory.sol` contract has a parameter called the `globalCap`, which ensures that no single collateral asset would exceed a certain percentage of the total collaterals backing the HONEY token.

This check is performed in the function `_isCappedGlobal()`. However, this function will return `false` and stop mint on encountering the first collateral asset that violate the condition while it is looping through the `registeredAsset[]` array. Even though the intention of the `globalCap` parameter is to prevent minting if it causes the collateral asset used in minting to exceed the collateral backing percentage threshold, it can cause mint to fail on all assets if the first asset in the `registeredAsset[]` array violates the `globalCap` threshold.

Exploit Scenario:

1. There are three tokens supported as collateral assets, X, Y, and Z. HONEY token is currently backed by 46%, 45%, and 9% of the three tokens respectively. These tokens are listed in this specific order in the `registeredAssets[]` array.
2. The Manager of the `HoneyFactory.sol` contract sets `globalCap` to 10%, with the intention of increasing the collateral backing weight of token Z but not exceeding 10%.
3. A user tries to mint HONEY with token Z. This should be OK because token Z has not exceeded the `globalCap` value.
4. In the `_isCappedGlobal()` function, the first token checked is token X.
5. Since token X is 46% of the collateral backing weight, the mint transaction reverts even though minting with token Z would have improved the overall collateral backing weight.

Recommendation: Add an argument for `_isCappedGlobal()` function to only check the collateral backing percentage for the asset that is used for minting.

BERA-4 Potential Censorship of `distributeFor()` Transactions

• Low ⓘ Mitigated

Update
Berachain team has conducted an analysis of the censorship risk based on the Comet proposer selection mechanism. While the risk of censorship exists for small border cases, in most circumstances the likelihood of a complete reward loss for a censored validator is rather low. In order for such censorship attack to be feasible, the censoring validator set needs to control almost the entire BERA stake, or nearly all validators need to launch a coordinated attack to censor only one validator.

Given the result of the analysis, the auditors have decided to downgrade the severity to Low.

File(s) affected: `src/pol/Distributor.sol`

Description: Berachain operates under a novel Proof-of-Liquidity (PoL) consensus mechanism. This consensus mechanism relies on the native smart contracts to mint and distribute the BGT token as the block reward to the block producing validator. This is done by the block producing validator submitting (or someone else submitting on behalf of the block producing validator) a transaction calling the `distributeFor()` function in the `Distributor.sol` smart contract. The `distributeFor()` function will process the proof that the validator is indeed the block producer, and will mint and distribute the BGT token accordingly.

The issue is that if a certain validator A produces a block, the transaction with the `distributeFor()` function call for validator A must be included in future blocks which might be produced by other validators. In general, validators have no incentive to include a transaction that would lead to block rewards for others and not themselves, which creates censorship risk in one of the most critical component of the PoL consensus mechanism.

There are two possible mitigations to this censorship risk: (1) the original block producing validator can increase the amount of transaction fee to incentivize the `distributeFor()` transaction to be included in a block; or (2) the original block producing validator can include the `distributeFor()` transaction for itself when it can propose a block again.

Mitigation (1) is not ideal, as it leads to excessive value extraction from an honest validator. This can also lead to a centralization risk, as large validators can wield significant power over smaller validators. Mitigation (2) is also not perfect for a few reasons. First, it delays the distribution of block reward by many blocks for most validators. Second, validators may not have a chance to propose for a long time because only the top 256 validators by the amount of BERA token staked can produce a block. Finally, the `distributeFor()` transaction must not be processed before the Beacon Root is updated. These factors can combine to cause lost block rewards for honest validators due to either a validator with larger stake coming online or from simple bad luck even if mitigation (2) is attempted.

Exploit Scenario:
1. Validator A produces a block and submits a `distributeFor()` transaction to the network.
2. The `distributeFor()` transaction is not included in any of the immediate future blocks.
3. An entity with sufficient capital (possibly another large validator) joins the network to knock validator A out of the top 256 validator before it gets a chance to produce another block.
4. Validator A is unable to rejoin the top 256 validator before the Beacon Roots update, leading to a loss of its block reward for producing a block in Step 1.

Recommendation: We recommend including some form of incentive at the consensus mechanism level for validators to include the `distributeFor()` transaction that would distribute reward to other validators. Alternatively, one could consider some form of penalty for acts of censorship at the consensus mechanism level.

BERA-5 Precision Is Not Enforced in the Setter Functions that Control Block Reward Rates

• Low ⓘ Mitigated

Update

Berachain team has indicated that the precision for these parameters are well-documented, and that small values are still acceptable. The setter functions will be called by governance proposals, which will go through community and guardian scrutiny. This process will also involve "proposal result simulation" through Tenderly.

Given these contexts, the auditors have decided to downgrade the severity to Low.

File(s) affected: `src/pol/rewards/BlockRewardController.sol`

Description: The `BlockRewardController.sol` contract uses the following five parameters to determine the block reward rate that is used to emit the BGT reward tokens:

- `baseRate`
- `rewardRate`
- `minBoostedRewardRate`
- `boostMultiplier`
- `rewardConvexity`

All these parameters should be in WAD precision (i.e., 18 decimals), as indicated in the default values. However, the corresponding setters do not enforce WAD precision, which introduces the risk of entering and accepting incorrect values. This would lead to incorrect reward calculations for the block rewards, which could be highly detrimental to the incentive system that underpin the Proof of Liquidity consensus mechanism.

Recommendation: Enforce WAD precision checks on the setter functions of `baseRate`, `rewardRate`, `minBoostedRewardRate`, `boostMultiplier`, and `rewardConvexity`. The precision check may be substituted with a reasonable minimum on each parameter. The transaction should revert if these checks fail.

BERA-6 Asset Parameters Silently Overriden at Vault Creation

• Low ⓘ Acknowledged

Update

Berachain team is aware of this issue. Any value set before creation of the asset vault is considered not relevant by the team.

File(s) affected: `src/honey/HoneyFactory.sol`

Description: While some functions in the `HoneyFactory.sol` contract enforce that the input asset is registered before setting the value (e.g., `setDepegOffsets()`), others do not. Specifically, `setMintRate()`, `setRedeemRate()`, `setRelativeCap()`, `setRecapitalizeBalanceThreshold()`, and `setPriceFeed()` do not implement this check.

Suppose one configures the mint rate, the redeem rate, or the relative cap before creating a vault. In that case, default values for these parameters will silently override the previous values that were set when the asset vault is created.

Recommendation: Before setting any parameter related to an asset, enforce the `_checkRegisteredAsset()` check to ensure that the asset vault has been created and registered.

BERA-7 `globalCap` Parameter Has No Minimum, Which Can Lead to Denial of Service

• Low ⓘ Acknowledged

Update
Berachain team has indicated that this issue can be circumvented by choosing the appropriate `relativeCap` and `globalCap` parameter.

File(s) affected: `src/honey/HoneyFactory.sol`

Description: The `HoneyFactory.sol` contract has a parameter called the `globalCap`, which ensures that no single collateral asset would exceed a certain percentage of the total collaterals backing the HONEY token.

There is currently no minimum defined for the `globalCap` parameter, which may cause all mint operation to fail if the last mint operation that causes one asset's collateral backing weight to exceed the `globalCap` does not sufficiently decreasing the collateral backing weights for other assets. This introduces additional burden for the Manager of the `HoneyFactory.sol` contract, because the Manager must respond quickly to update the value of `globalCap`.

Exploit Scenario:
1. There are three tokens supported as collateral assets, X, Y, and Z. HONEY token is currently backed by 46%, 45%, and 9% of the three tokens respectively.
2. The Manager of the `HoneyFactory.sol` contract sets `globalCap` to 10%, with the intention of increasing the collateral backing weight of token Z but not exceeding 10%.
3. More HONEY is minted with token Z, such that HONEY is now backed just a little over 10% by token Z.
4. At this point, HONEY is 45.5% backed by token X, 44.5% backed by token Y, and 10% backed by token Z. No further mint can take place until the `globalCap` is increased because all three assets exceed the collateral backing weight threshold of 10%.

Recommendation: Define a reasonable minimum value for the `globalCap` parameter based on the number of registered and non-blacklisted collateral assets.

BERA-8 Recapitalization Does Not Check for the Peg Status

• Low ⓘ Fixed

File(s) affected: `src/honey/HoneyFactory.sol`

Description: The function `recapitalize()` in the `HoneyFactory.sol` contract allows one to add tokens to collateral vaults that contain assets backing the HONEY token. While there are checks performed on whether the asset used to recapitalize is indeed registered or still whitelisted, there is no check to see if the asset used to recapitalize is still pegged.

Recommendation: Add `_isPegged()` check on the asset used to recapitalize.

BERA-9 `HoneyFactoryReader.sol` Is Not Fully Basket Mode Compatible

• Low ⓘ Fixed

File(s) affected: `src/honey/HoneyFactoryReader.sol`

Description: `HoneyFactoryReader.sol` contract is a helper contract that contains functions to provide users with information on: (1) the amount of HONEY tokens received when minting with certain amount of collateral asset (`previewMint()`), (2) information on the amount of collateral asset received when redeeming certain amount of HONEY tokens (`previewRedeem()`), and (3) the amount of HONEY token required to redeem certain amount of collateral assets (`previewHoneyToRedeem()`).

However, all of these functions assume that the `HoneyFactory.sol` contract is not in the basket mode, which is triggered when one or more collateral asset is depegged or blacklisted. When the `HoneyFactory.sol` contract is in basket mode, these functions return information that are different from what would actually happen if these actions were to take place.

Recommendation: Update the functions listed above so that they return the correct information based on whether or not basket mode is active on the `HoneyFactory.sol` contract.

BERA-10
Deterministic Deployment Proxy May Be Tampered With

● Informational ⓘ Fixed

Update
Berachain team has indicated that the `Create2Factory` bytecode will be deployed at specified address and genesis file will be publicly available in the repository. This is already available at #499.

Given these contexts, the auditors have decided to downgrade the severity to Informational.

File(s) affected: `src/base/Create2Deployer.sol`

Description: The `Create2Deployer.sol` contract relies on a Deterministic Deployment Proxy, which is expected to be deployed at address `0x4e59...956C` on the Berachain. At the time of this audit, Quantstamp cannot confirm that the correct byte code exists at the given address on Berachain's mainnet, as the latter is not yet available.

If a malicious deployer changes the expected behavior of that proxy, then all contracts inheriting from `Create2Deployer.sol` contract would be compromised.

Recommendation: We recommend additional checks to confirm that the Deterministic Deployment Proxy bytecode on Berachain's mainnet is indeed what one expects. These checks should be performed immediately before deploying any contract inheriting from the `Create2Deployer.sol` contract.

BERA-11
Mint and Redeem Rates Produced by `HoneyFactoryReader.sol` Is Not Guaranteed

● Informational ⓘ Mitigated

Update
Berachain team has indicated that `HoneyFactoryReader.sol` is intended to be used as a preview only, and it is expected that the actual output may not match the preview. Nevertheless, the team has added additional methods and checks in the `HoneyFactoryReader.sol` to improve the UX.

Given these contexts, the auditors have decided to downgrade the severity to Informational.

File(s) affected: `src/honey/HoneyFactory.sol`, `src/honey/HoneyFactoryReader.sol`

Description: When users either mint or redeem the HONEY token, they expect a given rate, and can even preview the expected output by means of the `HoneyFactoryReader.sol` contract. However, the output of the mint and redeem transactions are not guaranteed to match these rates. This can happen through a variety of ways:

- The Manager of the `HoneyFactory.sol` contract may submit a transaction to update the mint or redeem rate parameter values
- One or more collateral assets get depegged, triggering the basket mode in the `HoneyFactory.sol` contract

These changes can happen even within the same block, if the validator orders the user's mint/redeem transactions to be processed later. While this may work either against or in users' favor, in the more extreme cases a user might unexpectedly receives a basket of tokens while trying to redeem HONEY for a single collateral asset, as described in the "Mismatch in Basket Mode Activation Condition May Cause Financial Losses to Users" issue.

Recommendation: On functions such as mint and redeem, allow users to specify whether they want the transaction to fail if basket mode is activated, or how much the rate can deviate from a certain output value.

BERA-12
The Current Maximum Limit on the Reward Vault Incentive Rate Seems Excessively High

● Informational ⓘ Acknowledged

Update
Berachain team has indicated that this limit is intended. The team has further indicated that the maximum has been very loosely set to accommodate tokens with very small value relative to the BGT token or tokens with more than 18 decimals.

Given these contexts, the auditors have decided to downgrade the severity to Informational.

File(s) affected: `src/pol/rewards/RewardVault.sol`

Description: The `RewardVault.sol` contract allows for incentive token distribution rate of up to 1e18 incentive tokens per BGT token emission (`MAX_INCENTIVE_RATE = 1e36`). However, the `bgtEmitted` variable in `_processIncentives` is expressed in 18 decimals.

This would lead to L429 `uint256 amount = FixedPointMathLib.mulDiv(bgtEmitted, incentive.incentiveRate, PRECISION);` result in 1e36 incentive token emission per 1e18 BGT token at the maximum incentive rate, which appear quite excessive in most cases and would likely empty the incentive tokens stored in the Reward Vault immediately. The problem may be even greater if the incentive token has less than 18 decimals.

Recommendation: Clarify if this limit is intended. If not, update the precision used for the maximum incentive rate.

BERA-13
Conversion Between BGT and BERA Is Dependent on the Consensus Layer

● Informational ⓘ Acknowledged

Update
Berachain team plans to improve the documentation to clarify emission and conversion mechanism between BGT and BERA.

File(s) affected: `src/pol/BGT.sol`

Description: One of the use of the BGT token is to burn it in exchange for BERA, which is the native gas token on Berachain, at a 1:1 ratio. However, this assumes that there is a sufficient balance of BERA token in the BGT contract which would allow such operation. This conversion between BERA and BGT cannot be guaranteed by the smart contracts alone.

Currently, the team has indicated that the BERA token is emitted each block to the BGT contract address at the consensus layer level. The amount of BERA emitted currently is at the maximum possible amount of BGT emitted per block. This means that the 1:1 conversion ratio of BGT to BERA can be expected to hold as of now. However, documentation for this mechanism was not available. Furthermore, any updates or bugs at the consensus layer, which was out of scope for this audit, may put this conversion solvency at risk.

Recommendation: Improve the documentation so that users are aware of the BERA emission mechanism and its impact on the conversion between BGT and BERA at the smart contract level.

BERA-14
Missing `__*_init()` Calls to Openzeppelin Library Contracts Within Various Initializers

● Informational ⓘ Fixed

File(s) affected: `src/pol/BGTStaker.sol` , `src/pol/FeeCollector.sol` , `src/pol/rewards/BeraChef.sol` , `src/pol/rewards/BlockRewardController.sol` , `src/pol/rewards/Distributor.sol` , `src/pol/rewards/RewardVault.sol` , `src/pol/rewards/RewardVaultFactory.sol` , `src/honey/Honey.sol` , `src/honey/HoneyFactoryReader.sol` , `src/gov/TimeLock.sol`

Description: Many contracts inherit from OpenZeppelin abstract contracts that have initialization functions (`__*_init()` functions). Sometimes calls to the initialization functions in OpenZeppelin contracts are skipped when initializing the Berachain contracts. Even though some of these initialization calls result in no-op, it is still a good practice to ensure that initialization calls are made for every contract that is being inherited.

Specifically:

- `BGTStaker.sol` does not call `__UUPSUpgradeable_init()`
- `FeeCollector.sol` does not call `__Pausable_init()` , `__AccessControl_init()` , and `__UUPSUpgradeable_init()`
- `BeraChef.sol` does not call `__UUPSUpgradeable_init()`
- `BlockRewardController.sol` does not call `__UUPSUpgradeable_init()`
- `Distributor.sol` does not call `__ReentrancyGuard_init()` , `__AccessControl_init()` , and `__UUPSUpgradeable_init()`
- `RewardVault.sol` does not call `__Pausable_init()` and `__ReentrancyGuard_init()`
- `RewardVaultFactory.sol` does not call `__AccessControl_init()` and `__UUPSUpgradeable_init()`
- `Honey.sol` does not call `__AccessControl_init()` and `__UUPSUpgradeable_init()`
- `HoneyFactoryReader.sol` does not call `__AccessControl_init()` and `__UUPSUpgradeable_init()`
- `TimeLock.sol` does not have an initializer - it inherits from both OpenZeppelin's `UUPSUpgradeable` and `TimelockControllerUpgradeable` , and `__UUPSUpgradeable_init()` will not be called if the initialization function of `TimelockControllerUpgradeable` is used.

Recommendation: We recommend adding the appropriate `__*_init()` calls to the OpenZeppelin library contracts being inherited in the initialization functions.

BERA-15 Inconsistent Indexing of Token Addresses in the Return Array of `previewRequiredCollateral()` Function

● Undetermined ⓘ Fixed

File(s) affected: `src/honey/HoneyFactoryReader.sol`

Description: `HoneyFactoryReader.sol` contract is a helper contract that contains a function to provide users with information on the amount of collateral asset required to mint a desired amount of HONEY token (`previewRequiredCollateral()`). However, the array returned by this function has inconsistent indexing of the reference of the assets when basket mode is on vs off.

Specifically, when the `HoneyFactory.sol` contract is not in basket mode, the return array has [target asset amount, 0, ..., 0], with the 1st item referencing the amount needed in the target asset as specified by the arguments. However, when the `HoneyFactory.sol` contract is in basket mode, the return array has [asset_0 amount, asset_1 amount, ..., asset_n-1 amount], with each array index referencing the amount needed for the assets in the same order as the `registeredAssets[]` array.

Recommendation: Modify the `previewRequiredCollateral()` function so that the return array has consistent index references to the assets when basket mode is on or off.

Auditor Suggestions

S1 Unlocked and Inconsistent Pragma

Fixed

✓ **Update**
Berachain team uses the Foundry toolchain, which can fix the Solidity version.

Related Issue(s): SWC-103

Description: Every Solidity file specifies in the header a version number of the format `pragma solidity (^)0.8.*` . The caret (`^`) before the version number implies an unlocked pragma, meaning that the compiler will use the specified version *and above*, hence the term "unlocked". The pragma used in the repositories are all unlocked and inconsistent, ranging from `^0.8.4` all the way to `^0.8.24` .

Recommendation: For consistency and to prevent unexpected behavior in the future, we recommend removing the caret and standardizing the Solidity version across all smart contracts in the repositories.

S2 Missing Chain ID Check

Fixed

✓ **Update**
This function has been removed.

File(s) affected: `src/honey/HoneyFactory.sol`

Description: There is a function `initializeBasketMode()` that is commented as: `/// @dev Used to upgrade from HoneyFactoryV1 to HoneyFactoryV2 in testnet.` . However, this function is still callable on the mainnet due to the lack of chain ID check.

Recommendation: Add a `chainId` check in the function `initializeBasketMode()` so that it is not callable on the mainnet.

S3 Additional Comments to Clarify Decimal Units May Be Helpful

Acknowledged

✓ **Update**
The comment has been updated to indicate that the `recapitalizeBalanceThreshold` variable is expressed in token decimal units.

File(s) affected: `src/honey/HoneyFactory.sol`

Description: There are places where additional comments on decimal units and precision would be more helpful in understanding the code. Most notably, the `recapitalizeBalanceThreshold` variable in `HoneyFactory.sol` contract should be commented so that developers and users know that this is expressed in token decimal units as opposed to the vault decimal units.

Recommendation: Improve the documentation and code comments on the number of decimal units, or try to standardize to 18 decimal units as much as possible.

S4 Missing Input Validation in the Setter Function `_setPayoutAmount()` Acknowledged

Update
Berachain team has indicated that this is not an issue as the `_setPayoutAmount()` function is only called once. The team plans to address this if `_setPayoutAmount()` function will be used elsewhere in the future.

File(s) affected: `src/pol/FeeCollector.sol`

Description: In the `FeeCollector.sol` contract, the `payoutAmount` variable should not be set to zero in general, and this is checked for when setting new values for this variable. However, this check `queuedPayoutAmount != 0` is done in the `claimFees()` function as opposed to in the `_setPayoutAmount()` internal function. This introduces the risk of setting `payoutAmount` to zero if the `_setPayoutAmount()` function is called elsewhere.

Recommendation: Perform the check for `payoutAmount != 0` in the `_setPayoutAmount()` internal function.

S5 `globalCap` Value Is Not Validated in the Setter Function Acknowledged

Update
Berachain team has indicated the lack of validation here is due to the code size limit.

File(s) affected: `src/honey/HoneyFactory.sol`

Description: The `HoneyFactory.sol` contract has a parameter called the `globalCap`, which ensures that no single collateral asset would exceed a certain percentage of the total collaterals backing the HONEY token.

The Manager of the `HoneyFactory.sol` contract, inadvertently or not, can set the `globalCap` value to be higher than 100%. The manager can also set it to zero. However, the value of the `globalCap` parameter should always be between (0, 100]%. While there is no obvious side effects of excessively high `globalCap`, it is still a good practice to ensure reasonable values to be used in these parameters.

Recommendation: Add checks to ensure that the `globalCap` parameter will be between 0% and 100%.

Definitions

- **High severity** – High-severity issues usually put a large number of users' sensitive information at risk, or are reasonably likely to lead to catastrophic impact for client's reputation or serious financial implications for client and users.
- **Medium severity** – Medium-severity issues tend to put a subset of users' sensitive information at risk, would be detrimental for the client's reputation if exploited, or are reasonably likely to lead to moderate financial impact.
- **Low severity** – The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low impact in view of the client's business circumstances.
- **Informational** – The issue does not post an immediate risk, but is relevant to security best practices or Defence in Depth.
- **Undetermined** – The impact of the issue is uncertain.
- **Fixed** – Adjusted program implementation, requirements or constraints to eliminate the risk.
- **Mitigated** – Implemented actions to minimize the impact or likelihood of the risk.
- **Acknowledged** – The issue remains in the code but is a result of an intentional business or design decision. As such, it is supposed to be addressed outside the programmatic means, such as: 1) comments, documentation, README, FAQ; 2) business processes; 3) analyses showing that the issue shall have no negative consequences in practice (e.g., gas analysis, deployment settings).

Appendix

File Signatures

The following are the SHA-256 hashes of the reviewed files. A file with a different SHA-256 hash has been modified, intentionally or otherwise, after the security review. You are cautioned that a different SHA-256 hash could be (but is not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of the review.

Files

- `ce0...a15 ./src/gov/TimeLock.sol`
- `889...f6b ./src/gov/BerachainGovernance.sol`
- `605...bf3 ./src/gov/GovDeployer.sol`

- `0a5...014 ./src/base/IStakingRewardsErrors.sol`
- `534...c2e ./src/base/Create2Deployer.sol`
- `b01...0a9 ./src/base/IStakingRewards.sol`
- `91f...fda ./src/base/StakingRewards.sol`
- `ac9...452 ./src/base/FactoryOwnable.sol`
- `aaf...9dc ./src/honey/IHoneyErrors.sol`
- `773...184 ./src/honey/IHoneyFactory.sol`
- `0fe...9fb ./src/honey/HoneyFactoryReader.sol`
- `ee1...ed4 ./src/honey/Honey.sol`
- `2f5...6f8 ./src/honey/HoneyFactory.sol`
- `e46...d36 ./src/honey/CollateralVault.sol`
- `ca6...eb5 ./src/honey/VaultAdmin.sol`
- `bc7...260 ./src/honey/HoneyDeployer.sol`
- `918...906 ./src/pol/POLDeployer.sol`
- `2ab...ddb ./src/pol/BeaconRootsHelper.sol`
- `68d...a4f ./src/pol/FeeCollector.sol`
- `81f...930 ./src/pol/BGTStaker.sol`
- `aec...ddb ./src/pol/BeaconDeposit.sol`
- `45c...6a3 ./src/pol/BGT.sol`
- `d15...b0d ./src/pol/BGTFeeDeployer.sol`
- `914...9fe ./src/pol/interfaces/IBGT.sol`
- `708...111 ./src/pol/interfaces/IRewardVaultFactory.sol`
- `4c8...51f ./src/pol/interfaces/IBGTStaker.sol`
- `f87...9fa ./src/pol/interfaces/IBeaconDeposit.sol`
- `704...6e1 ./src/pol/interfaces/IRewardVault.sol`
- `688...cd7 ./src/pol/interfaces/IFeeCollector.sol`
- `bbb...668 ./src/pol/interfaces/IBeraChef.sol`
- `269...800 ./src/pol/interfaces/IPOLErrors.sol`
- `8d9...7c3 ./src/pol/interfaces/IDistributor.sol`
- `8cb...66f ./src/pol/interfaces/IERC165.sol`
- `ec1...bd9 ./src/pol/interfaces/IBlockRewardController.sol`
- `c9f...cd5 ./src/pol/rewards/RewardVault.sol`
- `bc8...afe ./src/pol/rewards/BlockRewardController.sol`
- `ad4...85c ./src/pol/rewards/BeraChef.sol`
- `3a0...f01 ./src/pol/rewards/Distributor.sol`
- `328...030 ./src/pol/rewards/RewardVaultFactory.sol`

Test Suite Results

Test suite was checked in the commit `97608d2`, with all tests passing.

```
Compiling 333 files with Solc 0.8.26
Solc 0.8.26 finished in 12.23s
Compiler run successful!
Analysing contracts...
Running tests...

Ran 6 tests for test/berps/v0/TestOrders.t.sol:TestOrders
[PASS] testGetOpenLimitOrders() (gas: 416041)
[PASS] testGetOpenTrades() (gas: 620586)
[PASS] testStoreOpenLimitOrder() (gas: 245769)
[PASS] testStoreTrade() (gas: 340511)
[PASS] testUnregisterOpenLimitOrder() (gas: 203934)
[PASS] testUnregisterTrade() (gas: 281156)
Suite result: ok. 6 passed; 0 failed; 0 skipped; finished in 287.88ms (48.05ms CPU time)

Ran 4 tests for test/base/BeaconRoots.t.sol:BeaconRootsTest
```

```
[PASS] test_GetParentBlockRootAt_Failure() (gas: 15253)
[PASS] test_GetParentBlockRootAt_Success() (gas: 20170)
[PASS] test_IsParentBlockRootAt_Failure() (gas: 9946)
[PASS] test_IsParentBlockRootAt_Success() (gas: 14067)
Suite result: ok. 4 passed; 0 failed; 0 skipped; finished in 290.57ms (49.98ms CPU time)

Ran 2 tests for test/berps/integration/TestTradingMisc.t.sol:TestTradingMisc
[PASS] testOpenCloseSameBlock() (gas: 1681726)
[PASS] testTradingSL() (gas: 1836352)
Logs:
  honey balance: 50000000000000000000
  honey balance: 40000000000000000000
  honey balance: 40000000000000000000
  honey balance: 48099999999693243635

Suite result: ok. 2 passed; 0 failed; 0 skipped; finished in 304.79ms (47.47ms CPU time)

Ran 8 tests for test/berps/utils/TestPriceUtils.t.sol:TestPriceUtils
[PASS] testCorrectSl() (gas: 15401)
[PASS] testCorrectTp() (gas: 10058)
[PASS] testCurrentPercentProfit() (gas: 6714)
[PASS] testCurrentPercentProfitRevertsIfPriceZero() (gas: 3317)
[PASS] testMaxSlDist() (gas: 4429)
[PASS] testMaxTpDist() (gas: 4254)
[PASS] testPythTriangular() (gas: 8496)
[PASS] testSimpleTriangular() (gas: 8674)
Suite result: ok. 8 passed; 0 failed; 0 skipped; finished in 9.47ms (9.29ms CPU time)

Ran 1 test for test/berps/v0/TestReferrals.t.sol:TestReferrals
[PASS] testReferralE2E() (gas: 422037)
Suite result: ok. 1 passed; 0 failed; 0 skipped; finished in 1.36ms (807.25µs CPU time)

Ran 5 tests for test/berps/integration/TestTradingValue.t.sol:TestTradingValue
[PASS] testRefundForDelegate() (gas: 1256147)
[PASS] testRefundForDelegateWithMulticall() (gas: 1932214)
[PASS] testRefundInMulticall() (gas: 1902530)
[PASS] testRefundNoMulticall() (gas: 1383833)
[PASS] testRefundWithMulticall() (gas: 2054197)
Suite result: ok. 5 passed; 0 failed; 0 skipped; finished in 32.08ms (26.27ms CPU time)

Ran 3 tests for test/berps/integration/TestTradingFee.t.sol:TestTradingFee
[PASS] testTradingCloseMarket() (gas: 1879358)
Logs:
  honey balance: 50000000000000000000
  honey balance: 84100000003599999995

[PASS] testTradingCloseSL() (gas: 2026656)
Logs:
  honey balance: 50000000000000000000
  honey balance: 43510000000438750000

[PASS] testTradingCloseTP() (gas: 1900795)
Logs:
  honey balance: 50000000000000000000
  honey balance: 93100000004499999994

Suite result: ok. 3 passed; 0 failed; 0 skipped; finished in 28.03ms (22.63ms CPU time)

Ran 1 test for test/pol/e2e/POLGasSim.t.sol:POLGasSimulationSimple
[PASS] testGasPOLDistribution() (gas: 312097)
Suite result: ok. 1 passed; 0 failed; 0 skipped; finished in 366.45ms (1.42ms CPU time)

Ran 1 test for test/pol/e2e/POLGasSimulationAdvance.t.sol:POLGasSimulationAdvance
[PASS] testGasPOLDistribution() (gas: 1004203)
Suite result: ok. 1 passed; 0 failed; 0 skipped; finished in 36.34ms (2.27ms CPU time)

Ran 3 tests for test/base/SSZ.t.sol:SSZTest
[PASS] test_ValidatorPubkeyRoot() (gas: 3747)
[PASS] test_addressHashTreeRoot() (gas: 4723)
[PASS] test_uint64HashTreeRoot() (gas: 7109)
Suite result: ok. 3 passed; 0 failed; 0 skipped; finished in 1.52ms (1.12ms CPU time)
```

```
Ran 2 tests for test/pol/e2e/POLGasSimulationMax.t.sol:POLGasSimulationMax
[PASS] testGasPOLDistribution() (gas: 1576973)
[PASS] testGasPOLDistributionCatchUp() (gas: 3963435)
Suite result: ok. 2 passed; 0 failed; 0 skipped; finished in 410.75ms (24.01ms CPU time)

Ran 16 tests for test/pol/RewardVaultFactory.t.sol:RewardVaultFactoryTest
[PASS] testFuzz_CreateRewardVault(address) (runs: 1024, μ: 310047, ~: 310047)
[PASS] testFuzz_SetRewardVaultImplementation_FailIsNotOwner(address) (runs: 1024, μ: 840548, ~: 840548)
[PASS] testFuzz_UpgradeToFailsIfNotOwner(address) (runs: 1024, μ: 1720600, ~: 1720600)
[PASS] test_CreateMultipleVaults() (gas: 1775387)
[PASS] test_CreateRewardVault() (gas: 311500)
[PASS] test_CreateRewardVault_CreateNewVaultWithoutUpdatingMapping() (gas: 1314663)
[PASS] test_CreateRewardVault_DoesNotFailIfVaultImplChange() (gas: 1318687)
[PASS] test_CreateRewardVault_FailIfVaultImplDoesNotChange() (gas: 1024187821)
[PASS] test_CreateRewardVault_RevertsIfStakingTokenIsNotAContract() (gas: 26880)
[PASS] test_GetVaultsLength() (gas: 315706)
[PASS] test_InitialState() (gas: 40361)
[PASS] test_SetRewardVaultImplementation() (gas: 850021)
[PASS] test_SetRewardVaultImplementation_FailIfNewVaultIsNotAContract() (gas: 23939)
[PASS] test_UpgradeToAndCall() (gas: 1725819)
[PASS] test_UpgradeToFailsIfImplIsNotUUPS() (gas: 21060)
[PASS] test_UpgradeToFailsIfNotOwner() (gas: 1720320)
Suite result: ok. 16 passed; 0 failed; 0 skipped; finished in 676.61ms (602.84ms CPU time)

Ran 39 tests for test/pol/BeraChef.t.sol:BeraChefTest
[PASS] testFail_SetDefaultRewardAllocationWithZeroPercentageWeight() (gas: 52020)
[PASS] testFuzz_FailUpdateWhitelistedVaultsNotRegistered() (gas: 3721067)
[PASS] testFuzz_QueueANewRewardAllocation(uint32) (runs: 1024, μ: 452, ~: 452)
[PASS] testFuzz_SetMaxNumWeightsPerRewardAllocation(uint8) (runs: 1024, μ: 33239, ~: 33184)
[PASS] testFuzz_SetRewardAllocationBlockDelay(uint64) (runs: 1024, μ: 33104, ~: 32520)
[PASS] testFuzz_SetRewardAllocationBlockDelay_FailsIfDelayTooLarge(uint64) (runs: 1024, μ: 25351, ~:
25196)
[PASS] testFuzz_updateWhitelistedVaults(bool) (runs: 1024, μ: 41829, ~: 32359)
[PASS] test_ActivateARewardAllocation() (gas: 189250)
[PASS] test_DeleteQueuedRewardAllocationAfterActivation() (gas: 240260)
[PASS] test_EditDefaultRewardAllocationBeforeRemoveAVaultFromWhitelist() (gas: 1928355)
[PASS] test_FailIfCallerNotDistributor() (gas: 22900)
[PASS] test_FailIfInitializeAgain() (gas: 26449)
[PASS] test_FailIfInvalidRewardAllocationWeights() (gas: 43436)
[PASS] test_FailIfNotOwner() (gas: 3423466)
[PASS] test_FailIfNotWhitelistedVault() (gas: 54270)
[PASS] test_FailIfRemovingAVaultFromWhitelistInvalidatesDefaultRewardAllocation() (gas: 121923)
[PASS] test_FailIfTheNewRewardAllocationHasTooManyWeights() (gas: 1098031)
[PASS] test_FailIfTheNewRewardAllocationIsNotInTheFuture() (gas: 36972)
[PASS] test_FailIfZeroRewardAllocationWeight() (gas: 49584)
[PASS] test_FailQueuedRewardAllocationExists() (gas: 121484)
[PASS] test_GetActiveRewardAllocation() (gas: 198024)
[PASS] test_GetActiveRewardAllocationReturnsDefaultRewardAllocation() (gas: 2065608)
[PASS] test_GetActiveRewardAllocationReturnsDefaultRewardAllocationAfterSetMaxWeights() (gas: 2134454)
[PASS] test_GetSetActiveRewardAllocation() (gas: 319285)
[PASS] test_OwnerIsGovernance() (gas: 18469)
[PASS] test_QueueANewRewardAllocation() (gas: 123164)
[PASS] test_QueueANewRewardAllocationWithMultipleWeights() (gas: 191191)
[PASS] test_ReturnsIfTheRewardAllocationIsNotQueued() (gas: 29669)
[PASS] test_ReturnsfStartBlockGreaterThanCurrentBlock() (gas: 137288)
[PASS] test_SetDefaultRewardAllocation() (gas: 118019)
[PASS] test_SetMaxNumWeightsPerRewardAllocation() (gas: 31502)
[PASS] test_SetMaxNumWeightsPerRewardAllocation_FailIfInvalidateDefaultRewardAllocation() (gas: 140873)
[PASS] test_SetMaxNumWeightsPerRewardAllocation_FailWithZero() (gas: 19033)
[PASS] test_SetRewardAllocationBlockDelay() (gas: 29256)
[PASS] test_UpgradeTo() (gas: 3382467)
[PASS] test_updateWhitelistedVaultMetadata() (gas: 29179)
[PASS] test_updateWhitelistedVaultMetadata_FailIfNotOwner() (gas: 19373)
[PASS] test_updateWhitelistedVaultMetadata_FailIfNotWhitelisted() (gas: 24272)
[PASS] test_updateWhitelistedVaults() (gas: 54993)
Suite result: ok. 39 passed; 0 failed; 0 skipped; finished in 612.86ms (600.10ms CPU time)

Ran 3 tests for test/berps/v0/TestVault.t.sol:TestVault
[PASS] testAssetsPnL(uint48) (runs: 1024, μ: 218642, ~: 219618)
[PASS] testBalanceOf() (gas: 513716)
[PASS] testDistributeReward(uint48) (runs: 1024, μ: 284790, ~: 288680)
Suite result: ok. 3 passed; 0 failed; 0 skipped; finished in 673.56ms (672.06ms CPU time)
```

```
Ran 29 tests for test/gov/BerachainGovernance.t.sol:BerachainGovernanceTest
[PASS] testFuzz_UpgradeGovernance_FailsIfNotOwner(address) (runs: 1024, μ: 7428414, ~: 7428414)
[PASS] testFuzz_UpgradeTimelock_FailsIfNotOwner(address) (runs: 1024, μ: 3009297, ~: 3009297)
[PASS] test_AcceleratedProposal() (gas: 1047045)
[PASS] test_CancelProposal() (gas: 485392)
[PASS] test_CancelProposalFailsIfCallerNotProposer() (gas: 472570)
[PASS] test_CancelProposalFailsIfProposalActive() (gas: 491908)
[PASS] test_CreateFailedProposal() (gas: 552399)
[PASS] test_CreatePendingProposal() (gas: 495654)
[PASS] test_CreateSucceededProposal() (gas: 555232)
[PASS] test_ExecuteProposal() (gas: 810414)
[PASS] test_ExecuteProposal_FailsIfProposalNonExistent() (gas: 68639)
[PASS] test_ExecuteProposal_FailsIfProposalNotQueued() (gas: 470163)
[PASS] test_ExecuteProposal_FailsIfSucceededButNotQueued() (gas: 620326)
[PASS] test_GuardianCanCancelProposalWhenReady() (gas: 715090)
[PASS] test_GuardianCanCancelProposalWhenWaiting() (gas: 714281)
[PASS] test_GuardianCancelProposalFailsIfAddressDoesNotHaveRole() (gas: 719686)
[PASS] test_ProposeFailsIfInsufficienVotes() (gas: 76359)
[PASS] test_ProposeFailsWithInvalidDescription() (gas: 62736)
[PASS] test_QueueProposal() (gas: 700691)
[PASS] test_QueueProposalFailsIfProposalCanceled() (gas: 493037)
[PASS] test_QueueProposalFailsIfProposalNonExistent() (gas: 68547)
[PASS] test_QueueProposalFailsIfProposalNotSuccess() (gas: 470041)
[PASS] test_UpgradeGovernanceViaVoting() (gas: 8012743)
[PASS] test_UpgradeGovernance_FailsIfNotOwner() (gas: 7428181)
[PASS] test_UpgradeTimelockViaVoting() (gas: 3573069)
[PASS] test_UpgradeTimelockWithCallerAsTimelock() (gas: 3014471)
[PASS] test_VoteCastingFailsIfMissingVotingPower() (gas: 512212)
[PASS] test_state_WhenAcceleratedProposalAndCanceledByGuardianship() (gas: 1229338)
[PASS] test_state_WhenAcceleratedProposalAndQueued() (gas: 1212239)
Suite result: ok. 29 passed; 0 failed; 0 skipped; finished in 788.09ms (775.45ms CPU time)

Ran 24 tests for test/pol/BlockRewardController.t.sol:BlockRewardControllerTest
[PASS] testFuzz_ComputeRewards(uint256,uint256,uint256,uint256) (runs: 1024, μ: 28045, ~: 28379)
[PASS] testFuzz_ProcessRewards(uint256,uint256,uint256,uint256,uint256,uint256) (runs: 1024, μ: 842114,
~: 842705)
[PASS] testFuzz_SetBaseRate(uint256) (runs: 1024, μ: 49588, ~: 49150)
[PASS] testFuzz_SetBoostMultiplier(uint256) (runs: 1024, μ: 49562, ~: 49124)
[PASS] testFuzz_SetMinBoostedRewardRate(uint256) (runs: 1024, μ: 49509, ~: 49080)
[PASS] testFuzz_SetRewardConvexity(uint256) (runs: 1024, μ: 49319, ~: 48615)
[PASS] testFuzz_SetRewardRate(uint256) (runs: 1024, μ: 49585, ~: 49147)
[PASS] test_ComputeRewards() (gas: 58707)
[PASS] test_FailIfInitializeAgain() (gas: 26208)
[PASS] test_FailIfNotDistributor() (gas: 23508)
[PASS] test_FailIfNotOwner() (gas: 2329937)
[PASS] test_OwnerIsGovernance() (gas: 18358)
[PASS] test_ProcessRewards() (gas: 369826)
[PASS] test_ProcessRewardsMax() (gas: 596020)
[PASS] test_ProcessRewardsMin() (gas: 371370)
[PASS] test_ProcessZeroRewards() (gas: 72598)
[PASS] test_SetBaseRate() (gas: 48964)
[PASS] test_SetBoostMultiplier() (gas: 48850)
[PASS] test_SetDistributor() (gas: 28464)
[PASS] test_SetDistributor_FailIfZeroAddress() (gas: 18972)
[PASS] test_SetMinBoostedRewardRate() (gas: 48873)
[PASS] test_SetRewardConvexity() (gas: 48407)
[PASS] test_SetRewardRate() (gas: 48961)
[PASS] test_UpgradeTo() (gas: 2313971)
Suite result: ok. 24 passed; 0 failed; 0 skipped; finished in 2.88s (2.85s CPU time)

Ran 32 tests for test/honey/CollateralVault.t.sol:CollateralVaultTest
[PASS] testFuzz_depositIntoUSTVault(uint256) (runs: 1024, μ: 135613, ~: 136377)
[PASS] testFuzz_deposit_succeedsWithCorrectSender(uint256) (runs: 1024, μ: 135175, ~: 135936)
[PASS] testFuzz_mintFromUSTVault(uint256) (runs: 1024, μ: 137880, ~: 138695)
[PASS] testFuzz_mint_succeedsWithCorrectSender(uint256) (runs: 1024, μ: 153881, ~: 154636)
[PASS] testFuzz_redeemFromUSTVault(uint256) (runs: 1024, μ: 174370, ~: 174773)
[PASS] testFuzz_redeem_failWithInsufficientAllowance(uint256) (runs: 1024, μ: 137776, ~: 137729)
[PASS] testFuzz_redeem_failsWithInsufficientBalance(uint256) (runs: 1024, μ: 31755, ~: 31755)
[PASS] testFuzz_redeem_succeedsWithCorrectSender(uint256) (runs: 1024, μ: 173109, ~: 173356)
[PASS] testFuzz_withdrawFromUSTVault(uint256) (runs: 1024, μ: 176800, ~: 177211)
[PASS] testFuzz_withdraw_failsWithIncorrectSender(uint128) (runs: 1024, μ: 24533, ~: 24533)
```

```
[PASS] testFuzz_withdraw_failsWithInsufficientAllowance(uint256) (runs: 1024, μ: 140265, ~: 140188)
[PASS] testFuzz_withdraw_failsWithInsufficientBalance(uint256) (runs: 1024, μ: 38049, ~: 38049)
[PASS] testFuzz_withdraw_succeedsWithCorrectSender(uint256) (runs: 1024, μ: 175536, ~: 175791)
[PASS] test__codesize() (gas: 81749)
[PASS] test_deposit() (gas: 135546)
[PASS] test_deposit_whileItsPaused() (gas: 50699)
[PASS] test_deposit_withOutOwner() (gas: 21465)
[PASS] test_initializeVault_failsIfZeroAsset() (gas: 144484)
[PASS] test_initializeVault_failsIfZeroFactory() (gas: 121936)
[PASS] test_mint() (gas: 137488)
[PASS] test_mint_failsWithIncorrectSender() (gas: 23647)
[PASS] test_mint_whileItsPaused() (gas: 50765)
[PASS] test_pauseVault_succeedsWithCorrectSender() (gas: 48662)
[PASS] test_pausingVault_failsIfNotFactory() (gas: 20506)
[PASS] test_redeem() (gas: 165656)
[PASS] test_redeem_whileItsPaused() (gas: 51000)
[PASS] test_redeem_withOutOwner() (gas: 21833)
[PASS] test_unpausingVault_failsIfNotFactory() (gas: 20507)
[PASS] test_unpausingVault_succeedsWithCorrectSender() (gas: 37626)
[PASS] test_vaultParams() (gas: 37936)
[PASS] test_withdraw() (gas: 167743)
[PASS] test_withdraw_whileItsPaused() (gas: 51110)
Suite result: ok. 32 passed; 0 failed; 0 skipped; finished in 4.02s (3.78s CPU time)

Ran 4 tests for test/base/Create2Deployer.t.sol:Create2DeployerTest
[PASS] test_DeployProxyWithCreate2() (gas: 3464080)
[PASS] test_DeployProxyWithCreate2_FailIfAlreadyDeployed() (gas: 1024333504)
[PASS] test_DeployWithCreate2() (gas: 3375004)
[PASS] test_DeployWithCreate2_FailIfAlreadyDeployed() (gas: 1024330101)
Suite result: ok. 4 passed; 0 failed; 0 skipped; finished in 5.63ms (4.68ms CPU time)

Ran 24 tests for test/pol/BeaconDeposit.t.sol:BeaconDepositTest
[PASS] testFuzz_AcceptOperatorChange_FailsIfNotEnoughTime(uint256) (runs: 1024, μ: 181977, ~: 182562)
[PASS] testFuzz_DepositCount(uint256) (runs: 1024, μ: 1759640, ~: 1747930)
[PASS] testFuzz_DepositNotDivisibleByGwei(uint256) (runs: 1024, μ: 62782, ~: 62927)
[PASS] testFuzz_DepositWrongCredentials(bytes) (runs: 1024, μ: 24610, ~: 24610)
[PASS] testFuzz_DepositWrongMinAmount(uint256) (runs: 1024, μ: 94605, ~: 94672)
[PASS] testFuzz_DepositsWrongPubKey(bytes) (runs: 1024, μ: 28553, ~: 28546)
[PASS] testFuzz_RequestOperatorChange(address) (runs: 1024, μ: 173990, ~: 173990)
[PASS] test_AcceptOperatorChange() (gas: 177491)
[PASS] test_AcceptOperatorChange_FailsIfNotEnoughTime() (gas: 180862)
[PASS] test_AcceptOperatorChange_FailsIfNotNewOperator() (gas: 179793)
[PASS] test_AcceptOperatorChange_FailsIfNotQueued() (gas: 138994)
[PASS] test_CancelOperatorChange() (gas: 173207)
[PASS] test_CancelOperatorChange_FailsIfNotCurrentOperator() (gas: 179611)
[PASS] test_Deposit() (gas: 131115)
[PASS] test_DepositNotDivisibleByGwei() (gas: 101129)
[PASS] test_DepositWrongAmount() (gas: 94422)
[PASS] test_DepositWrongCredentials() (gas: 23874)
[PASS] test_DepositWrongPubKey() (gas: 27866)
[PASS] test_DepositZeroOperator() (gas: 34317)
[PASS] test_Deposit_FailsIfOperatorAlreadySet() (gas: 145302)
[PASS] test_RequestOperatorChange() (gas: 175849)
[PASS] test_RequestOperatorChange_FailsIfNotCurrentOperator() (gas: 137382)
[PASS] test_RequestOperatorChange_FailsIfZeroAddress() (gas: 135236)
[PASS] test_SupportsInterface() (gas: 13176)
Suite result: ok. 24 passed; 0 failed; 0 skipped; finished in 6.91s (7.31s CPU time)

Ran 17 tests for test/pol/Distributor.t.sol:DistributorTest
[PASS] testFuzz_DistributeDoesNotLeaveDust(uint256) (runs: 1024, μ: 2381964, ~: 2382326)
[PASS] testFuzz_DistributeDoesNotLeaveDust(uint256,uint256,uint256,uint256,uint256) (runs: 1024, μ:
2452317, ~: 2457284)
[PASS] test_Distribute() (gas: 667299)
[PASS] test_DistributeAndActivateQueuedRewardAllocation() (gas: 764770)
[PASS] test_DistributeDuringGenesisNoBgtWaste() (gas: 341253)
[PASS] test_DistributeForNonReentrant() (gas: 4109745)
[PASS] test_DistributeMulticall() (gas: 898967)
[PASS] test_FailIfInitializeAgain() (gas: 26738)
[PASS] test_FailIfNotManager() (gas: 34264)
[PASS] test_FailIfNotOwner() (gas: 2660067)
[PASS] test_IsTimestampActionable_OutOfBuffer() (gas: 20545)
[PASS] test_IsTimestampActionable_Processing() (gas: 296036)
```

```
[PASS] test_OwnerIsGovernance() (gas: 18193)
[PASS] test_ProcessTimestamp_OutOfBuffer() (gas: 194596)
[PASS] test_ProcessTimestamp_Processing() (gas: 439000)
[PASS] test_UpgradeTo() (gas: 2639060)
[PASS] test_ZeroRewards() (gas: 314455)
Suite result: ok. 17 passed; 0 failed; 0 skipped; finished in 5.43s (5.43s CPU time)

Ran 25 tests for test/pol/FeeCollector.t.sol:FeeCollectorTest
[PASS] testFuzz_Donate(uint256) (runs: 1024, μ: 220916, ~: 221270)
[PASS] testFuzz_Donate_FailsIfAmountLessThanPayoutAmount(uint256) (runs: 1024, μ: 19540, ~: 18847)
[PASS] testFuzz_Donate_FailsIfNotApproved(uint256) (runs: 1024, μ: 56269, ~: 56142)
[PASS] testFuzz_Donate_FailsIfPaused(uint256) (runs: 1024, μ: 169389, ~: 169314)
[PASS] testFuzz_PayoutAmountChangeAfterClaim(uint256) (runs: 1024, μ: 190850, ~: 190857)
[PASS] testFuzz_PayoutAmountDoesntChangeIfClaimFails(uint256) (runs: 1024, μ: 111874, ~: 111867)
[PASS] testFuzz_QueuePayoutAmountChange(uint256) (runs: 1024, μ: 52576, ~: 52569)
[PASS] test_ClaimFees() (gas: 174181)
[PASS] test_ClaimFees_FailsIfNotApproved() (gas: 69091)
[PASS] test_ClaimFees_FailsIfPaused() (gas: 117615)
[PASS] test_Donate() (gas: 220048)
[PASS] test_Donate_FailsIfAmountLessThanPayoutAmount() (gas: 18662)
[PASS] test_Donate_FailsIfNotApproved() (gas: 55346)
[PASS] test_GovernanceIsOwner() (gas: 18148)
[PASS] test_Initialize_FailsIfGovernanceAddrIsZero() (gas: 2038609)
[PASS] test_Initialize_FailsIfPayoutAmountIsZero() (gas: 2040921)
[PASS] test_Initialize_FailsIfPayoutTokenAddrIsZero() (gas: 2038684)
[PASS] test_Initialize_FailsIfRewardReceiverAddrIsZero() (gas: 2038641)
[PASS] test_Pause() (gas: 44088)
[PASS] test_Pause_FailIfNotVaultManager() (gas: 18949)
[PASS] test_QueuePayoutAmountChange() (gas: 52318)
[PASS] test_QueuePayoutAmountChange_FailsIfNotOwner() (gas: 19510)
[PASS] test_QueuePayoutAmountChange_FailsIfZero() (gas: 19223)
[PASS] test_Unpause() (gas: 33856)
[PASS] test_Unpause_FailIfNotVaultManager() (gas: 51364)
Suite result: ok. 25 passed; 0 failed; 0 skipped; finished in 3.05s (3.05s CPU time)

Ran 32 tests for test/honey/Honey.t.sol:HoneyTest
[PASS] testFuzz_Approve(address,uint256) (runs: 1024, μ: 43384, ~: 43540)
[PASS] testFuzz_Burn(uint256,uint256) (runs: 1024, μ: 83361, ~: 83751)
[PASS] testFuzz_Burn_FailIfInsufficientBalance(uint256,uint256) (runs: 1024, μ: 74764, ~: 75769)
[PASS] testFuzz_Mint(uint256) (runs: 1024, μ: 70640, ~: 71146)
[PASS] testFuzz_Mint_TotalSupplyOverflow(uint256,uint256) (runs: 1024, μ: 69122, ~: 69136)
[PASS] testFuzz_Transfer(address,uint256) (runs: 1024, μ: 86710, ~: 87021)
[PASS] testFuzz_TransferFrom(address,uint256) (runs: 1024, μ: 104817, ~: 105099)
[PASS] testFuzz_TransferFrom_FailsIfInsufficientAllowance(address,uint256) (runs: 1024, μ: 104282, ~:
104457)
[PASS] testFuzz_TransferFrom_FailsIfInsufficientBalance(address,uint256) (runs: 1024, μ: 104449, ~:
104800)
[PASS] testFuzz_Transfer_FailsIfInsufficientBalance(address,uint256) (runs: 1024, μ: 74505, ~: 74585)
[PASS] testFuzz_UpgradeTo_FailsIfNotOwner(address) (runs: 1024, μ: 1962789, ~: 1962789)
[PASS] test_Approve() (gas: 45254)
[PASS] test_Burn() (gas: 83334)
[PASS] test_Burn_FailIfInsufficientBalance() (gas: 35529)
[PASS] test_Burn_FailIfNotFactory() (gas: 16556)
[PASS] test_Initialize_FailsIfZeroAddresses() (gas: 2065432)
[PASS] test_MetaData() (gas: 16140)
[PASS] test_Mint() (gas: 70873)
[PASS] test_Mint_FailIfNotFactory() (gas: 16513)
[PASS] test_Permit() (gas: 101968)
[PASS] test_Permit_BadDeadlineReverts() (gas: 50534)
[PASS] test_Permit_BadNonceReverts() (gas: 50810)
[PASS] test_Permit_PastDeadlineReverts() (gas: 45109)
[PASS] test_Permit_ReplayReverts() (gas: 103606)
[PASS] test_Transfer() (gas: 88721)
[PASS] test_TransferFrom() (gas: 106604)
[PASS] test_TransferFrom_FailsIfInsufficientAllowance() (gas: 106221)
[PASS] test_TransferFrom_FailsIfInsufficientBalance() (gas: 106477)
[PASS] test_Transfer_FailsIfInsufficientBalance() (gas: 76143)
[PASS] test_UpgradeTo() (gas: 1528194)
[PASS] test_UpgradeTo_FailIfNotOwner() (gas: 1962512)
[PASS] test__codesize() (gas: 88503)
Suite result: ok. 32 passed; 0 failed; 0 skipped; finished in 1.99s (1.96s CPU time)
```

```
Ran 1 test for test/pol/POLDeployer.t.sol:POLDeployerTest
[PASS] test_DeployPOL() (gas: 20764462)
Logs:
  POLDeployer init code size 73647
  BeraChef implementation address 0x4A7683f9b9Cb9e6dC4fDE48812880aa056e0E786
  BeraChef init code hash
  0x847fafa28286ca17333f05548da95b97fb3f0733d8666af1dad2e528de859034
  BeraChef address 0x5a85409CA3f33CBf16A0fB6e417D5B57da679682
  BlockRewardController implementation address 0x3ECd186CADF4Dce44C437dDeC9739a074949E19d
  BlockRewardController init code hash
  0xe2b6823ce1eeff9697ab07014ba6c4083a6553de5d59be7233d4246fb87de012
  BlockRewardController address 0x1be03E6D1f82A605068aaA8784472620a590Ef9E
  Distributor implementation address 0x2079Fc8BA555E91D328F6153C39372ec31045630
  Distributor init code hash
  0x9115add76f7fd98c3c3ddb80eebeef83919cc535418e91cf8eb92f793808f920
  Distributor address 0xC37d652C6c0981F1Ff063c68b6CD376a6f65834d

Suite result: ok. 1 passed; 0 failed; 0 skipped; finished in 4.85ms (3.36ms CPU time)

Ran 73 tests for test/honey/HoneyFactory.t.sol:HoneyFactoryTest
[PASS] testFuzz_CollectedFeesWithDifferentFeeRate(uint256,uint256) (runs: 1024, μ: 289799, ~: 301790)
[PASS] testFuzz_InflationAttack(uint256) (runs: 1024, μ: 352898, ~: 354452)
[PASS] testFuzz_PreviewHoneyToRedeem(uint64) (runs: 1024, μ: 50349, ~: 50349)
[PASS] testFuzz_PreviewRequiredCollateral(uint128) (runs: 1024, μ: 43655, ~: 43655)
[PASS] testFuzz_mint(uint256) (runs: 1024, μ: 304802, ~: 307412)
[PASS] testFuzz_mintWithHigherDecimalAsset(uint256) (runs: 1024, μ: 298070, ~: 308270)
[PASS] testFuzz_mintWithLowerDecimalAsset(uint256) (runs: 1024, μ: 306912, ~: 308782)
[PASS] testFuzz_mint_failWithPausedFactory(uint128) (runs: 1024, μ: 53994, ~: 53994)
[PASS] testFuzz_mint_failsWithInsufficientAllowance(uint256) (runs: 1024, μ: 34555, ~: 34478)
[PASS] testFuzz_mint_failsWithUnregisteredAsset(uint32) (runs: 1024, μ: 805097, ~: 805097)
[PASS] testFuzz_redeem(uint256) (runs: 1024, μ: 354395, ~: 354587)
[PASS] testFuzz_redeemWithHigherDecimalAsset(uint256) (runs: 1024, μ: 356044, ~: 356238)
[PASS] testFuzz_redeemWithLowerDecimalAsset(uint256) (runs: 1024, μ: 355887, ~: 356081)
[PASS] testFuzz_redeem_failWithPausedFactory(uint128) (runs: 1024, μ: 51789, ~: 51789)
[PASS] testFuzz_redeem_failsWithInsufficientHoneys(uint256) (runs: 1024, μ: 45185, ~: 45159)
[PASS] testFuzz_redeem_failsWithInsufficientShares(uint256) (runs: 1024, μ: 298791, ~: 298758)
[PASS] testFuzz_redeem_failsWithUnregisteredAsset(uint128) (runs: 1024, μ: 802499, ~: 802499)
[PASS] testFuzz_setFeeReceiver(address) (runs: 1024, μ: 28723, ~: 28723)
[PASS] testFuzz_setMintRate(uint256) (runs: 1024, μ: 32659, ~: 32661)
[PASS] testFuzz_setMintRate_failsWithOverOneHundredPercentRate(uint256) (runs: 1024, μ: 22574, ~: 22551)
[PASS] testFuzz_setMintRate_failsWithUnderNinetyEightPercentRate(uint256) (runs: 1024, μ: 22551, ~:
22472)
[PASS] testFuzz_setPOLFeeCollector(address) (runs: 1024, μ: 28681, ~: 28681)
[PASS] testFuzz_setPOLFeeCollectorFeeRate(uint256) (runs: 1024, μ: 28092, ~: 28074)
[PASS] testFuzz_setRedeemRate(uint256) (runs: 1024, μ: 32470, ~: 32507)
[PASS] testFuzz_setRedeemRate_failsWithOverOneHundredPercentRate(uint256) (runs: 1024, μ: 22445, ~:
22422)
[PASS] testFuzz_setRedeemRate_failsWithUnderNinetyEightPercentRate(uint256) (runs: 1024, μ: 22551, ~:
22472)
[PASS] testFuzz_withdrawAllFee(uint256) (runs: 1024, μ: 322445, ~: 325239)
[PASS] testFuzz_withdrawFee(uint256) (runs: 1024, μ: 309603, ~: 312342)
[PASS] testFuzz_withdrawFee_failsWithAssetNotRegistered() (gas: 802407)
[PASS] test_CollectedFees() (gas: 301393)
[PASS] test_CreateVault() (gas: 1064185)
[PASS] test_InitializeFactoryWithZeroAddresses() (gas: 7120775)
[PASS] test_Initialize_ParamsSet() (gas: 38574)
[PASS] test_IntegrationTest() (gas: 682867)
[PASS] test_TransferOwnershipOfBeacon() (gas: 31545)
[PASS] test_TransferOwnershipOfBeaconFailsIfNotOwner() (gas: 23643)
[PASS] test_TransferOwnershipOfBeaconFailsIfZeroAddress() (gas: 24223)
[PASS] test_UpgradeAndDowngradeOfBeaconProxy() (gas: 2886127)
[PASS] test_UpgradeBeaconProxy() (gas: 538471)
[PASS] test_UpgradeBeaconProxyImplFailsIfNotOwner() (gas: 511985)
[PASS] test_UpgradeBeaconProxyToFaultyVault() (gas: 2842691)
[PASS] test_WithdrawFee_TestEvent() (gas: 343429)
[PASS] test__codesize() (gas: 136348)
[PASS] test_createAlreadyRegisteredVault() (gas: 24364)
[PASS] test_factoryPause() (gas: 43356)
[PASS] test_factoryPause_failsWhenAlreadyPaused() (gas: 44302)
[PASS] test_factoryPause_failsWithoutManager() (gas: 16753)
[PASS] test_factoryUnPause_failsWhenAlreadyUnpaused() (gas: 20730)
[PASS] test_factoryUnPause_failsWithoutManager() (gas: 16775)
```

```
[PASS] test_factoryUnpause() (gas: 33386)
[PASS] test_mint() (gas: 277593)
[PASS] test_mint_failsWithBadCollateralAsset() (gas: 57984)
[PASS] test_pauseVault() (gas: 66141)
[PASS] test_pauseVault_failsWithoutManager() (gas: 19392)
[PASS] test_redeem() (gas: 298250)
[PASS] test_setCollateralAssetStatus() (gas: 51573)
[PASS] test_setCollateralAssetStatus_failsWithSameState() (gas: 26606)
[PASS] test_setCollateralAssetStatus_failsWithUnregisteredAsset() (gas: 804498)
[PASS] test_setCollateralAssetStatus_failsWithoutManager() (gas: 19675)
[PASS] test_setFeeReceiver() (gas: 30541)
[PASS] test_setFeeReceiver_failsWithZeroAddress() (gas: 19026)
[PASS] test_setFeeReceiver_failsWithoutAdmin() (gas: 19467)
[PASS] test_setMintRate_failsWithoutManager() (gas: 19875)
[PASS] test_setPOLFeeCollector() (gas: 30611)
[PASS] test_setPOLFeeCollectorFeeRate() (gas: 27859)
[PASS] test_setPOLFeeCollectorFeeRate_failsWithOverOneHundredPercentRate(uint256) (runs: 1024, μ: 19961,
~: 19938)
[PASS] test_setPOLFeeCollectorFeeRate_failsWithoutManager() (gas: 17221)
[PASS] test_setPOLFeeCollector_failsWithZeroAddress() (gas: 19004)
[PASS] test_setPOLFeeCollector_failsWithoutAdmin() (gas: 19464)
[PASS] test_setRedeemRate_failsWithoutManager() (gas: 19831)
[PASS] test_unpauseVault() (gas: 53842)
[PASS] test_unpauseVault_failsWithoutManager() (gas: 19370)
[PASS] test_withdrawFee_WithZeroCollectedFee() (gas: 29527)
Suite result: ok. 73 passed; 0 failed; 0 skipped; finished in 5.38s (13.76s CPU time)

Ran 14 tests for test/base/Utils.t.sol:UtilsTest
[PASS] testFuzz_Allowance(address,address,uint256) (runs: 1024, μ: 40565, ~: 40915)
[PASS] testFuzz_SafeIncreaseAllowance(address,uint256) (runs: 1024, μ: 45204, ~: 45164)
[PASS] testFuzz_SafeIncreaseAllowance_Overflow(uint256,uint256) (runs: 1024, μ: 45551, ~: 45675)
[PASS] testFuzz_TrySafeTransferDoesNotExeedGasLimit(address) (runs: 1024, μ: 405893, ~: 405893)
[PASS] testFuzz_TrySafeTransferERC20(address,uint256) (runs: 1024, μ: 70713, ~: 71024)
[PASS] testFuzz_TrySafeTransferExeedGasLimit(address) (runs: 1024, μ: 2141731, ~: 2141731)
[PASS] testFuzz_TrySafeTransferMissingReturnToken(address,uint256) (runs: 1024, μ: 45918, ~: 46133)
[PASS] testFuzz_TrySafeTransferReturnFalseToken(address,uint256) (runs: 1024, μ: 20707, ~: 20707)
[PASS] testFuzz_TrySafeTransferReturnTwoToken(address,uint256) (runs: 1024, μ: 20644, ~: 20644)
[PASS] testFuzz_TrySafeTransferRevertingToken(address,uint256) (runs: 1024, μ: 20509, ~: 20509)
[PASS] test_Allowance() (gas: 42491)
[PASS] test_Allowance_WhenTokenNotExists() (gas: 5975)
[PASS] test_SafeIncreaseAllowance() (gas: 46906)
[PASS] test_SafeIncreaseAllowance_IfTokenNotExists() (gas: 12538)
Suite result: ok. 14 passed; 0 failed; 0 skipped; finished in 15.43s (9.99s CPU time)

Ran 75 tests for test/pol/BGT.t.sol:BGTTest
[PASS] testFuzz_ActivateBoost(address,address,uint256) (runs: 1024, μ: 383781, ~: 383304)
[PASS] testFuzz_ActivateBoost_ReturnsFalseIfNotEnoughTimePassed(uint256) (runs: 1024, μ: 239377, ~:
239604)
[PASS] testFuzz_Approve(address,address,uint256) (runs: 1024, μ: 71793, ~: 72143)
[PASS] testFuzz_CancelBoost(address,uint256,uint256) (runs: 1024, μ: 251211, ~: 252450)
[PASS] testFuzz_CancelBoost_FailsIfCancelledMoreThanQueuedBoost(uint256,uint256) (runs: 1024, μ: 221270,
~: 221778)
[PASS] testFuzz_CancelDropBoost(address,uint256,uint256) (runs: 1024, μ: 458138, ~: 458323)
[PASS] testFuzz_CancelDropBoost_FailsIfCancelledMoreThanQueuedDropBoost(uint256,uint256) (runs: 1024, μ:
444757, ~: 444736)
[PASS] testFuzz_DropBoost(address,uint256,uint256) (runs: 1024, μ: 466906, ~: 469056)
[PASS] testFuzz_DropBoost_ReturnsFalseIfNotEnoughTimePassed(uint256) (runs: 1024, μ: 441760, ~: 442070)
[PASS] testFuzz_Mint(address,uint256) (runs: 1024, μ: 117942, ~: 118068)
[PASS] testFuzz_MultiCall_CancelBoostInBatch(address,uint256,uint256) (runs: 1024, μ: 333641, ~: 333395)
[PASS] testFuzz_MultiCall_QueueBoostInBatch(address,uint256,uint256) (runs: 1024, μ: 307566, ~: 307256)
[PASS] testFuzz_MultiCall_QueueDropBoostInBatch(uint256,uint256,uint256) (runs: 1024, μ: 432441, ~:
433081)
[PASS] testFuzz_QueueBoost(address,uint256) (runs: 1024, μ: 212492, ~: 212519)
[PASS] testFuzz_QueueDropBoost(address,uint256,uint256) (runs: 1024, μ: 436971, ~: 437153)
[PASS] testFuzz_QueueDropBoost_FailsIfDroppedMoreThanBoost(uint256,uint256) (runs: 1024, μ: 394403, ~:
394382)
[PASS] testFuzz_Redeem(uint256) (runs: 1024, μ: 179744, ~: 179862)
[PASS] testFuzz_SetActivateBoostDelay(uint256) (runs: 1024, μ: 23366, ~: 23695)
[PASS] testFuzz_SetDropBoostDelay(uint256) (runs: 1024, μ: 23206, ~: 23535)
[PASS] testFuzz_SetMinter(address) (runs: 1024, μ: 27591, ~: 27591)
[PASS] testFuzz_Transfer(address,address,uint256) (runs: 1024, μ: 165485, ~: 166050)
[PASS] testFuzz_TransferFrom(address,address,uint256,uint256) (runs: 1024, μ: 169594, ~: 169884)
```

```
[PASS] testFuzz_WhitelistSender(address,bool) (runs: 1024, μ: 32731, ~: 22937)
[PASS] testMetadata() (gas: 14981)
[PASS] test_ActivateBoost() (gas: 384828)
[PASS] test_ActivateBoost_ReturnsFalseIfNotEnoughTimePassed() (gas: 238986)
[PASS] test_ActivateBoost_ReturnsFalseIfNotQueued() (gas: 20075)
[PASS] test_Approve() (gas: 44805)
[PASS] test_CancelBoost() (gas: 232630)
[PASS] test_CancelBoost_FailsIfCancelledMoreThanQueuedBoost() (gas: 219867)
[PASS] test_CancelDropBoost() (gas: 458405)
[PASS] test_CancelDropBoost_FailsIfCancelledMoreThanQueuedDropBoost() (gas: 442954)
[PASS] test_DropBoost() (gas: 466978)
[PASS] test_DropBoost_FailsIfQueueAmountIsZero() (gas: 447440)
[PASS] test_DropBoost_ReturnsFalseIfNotEnoughTimePassed() (gas: 440442)
[PASS] test_FailIfMinterIsZero() (gas: 14126)
[PASS] test_FailIfNotApprovedSender() (gas: 25562)
[PASS] test_FailIfNotOwner() (gas: 22024)
[PASS] test_FailIndirectTransferFromInsufficientAllowance() (gas: 153733)
[PASS] test_FailMintIfNotMinter() (gas: 13672)
[PASS] test_FailMintOverMaxUint() (gas: 116524)
[PASS] test_FailTransferFromInsufficientAllowance() (gas: 151575)
[PASS] test_FailTransferFromInsufficientBalance() (gas: 47065)
[PASS] test_FailTransferInsufficientBalance() (gas: 19688)
[PASS] test_Mint() (gas: 119761)
[PASS] test_Mint_FailsIfInvariantCheckFails() (gas: 114698)
[PASS] test_MinterIsBlockRewardController() (gas: 13350)
[PASS] test_MultiCall_ActivateBoostInBatch() (gas: 488928)
[PASS] test_MultiCall_ActivateBoostInBatch_DoesNotFailIfNotEnoughTimePassed() (gas: 401063)
[PASS] test_MultiCall_CancelBoostInBatch() (gas: 334252)
[PASS] test_MultiCall_QueueBoostInBatch() (gas: 308194)
[PASS] test_MultiCall_QueueBoostInBatch_FailsIfNotEnoughUnboostedBalance() (gas: 27998)
[PASS] test_MultiCall_QueueDropBoostInBatch() (gas: 429537)
[PASS] test_MultiCall_SetPramsInBatch() (gas: 37889)
[PASS] test_MultiCall_SetPramsInBatch_FailsIfNotOwner() (gas: 17456)
[PASS] test_OwnerIsGovernance() (gas: 13517)
[PASS] test_QueueBoost() (gas: 214232)
[PASS] test_QueueDropBoost() (gas: 437210)
[PASS] test_QueueDropBoost_FailsIfDroppedMoreThanBoost() (gas: 392537)
[PASS] test_Redeem() (gas: 179122)
[PASS] test_Redeem_FailsIfInvariantCheckFails() (gas: 169734)
[PASS] test_Redeem_FailsIfNotEnoughUnboostedBalance() (gas: 218334)
[PASS] test_Redeem_FailsWithETHTransferFail() (gas: 107317)
[PASS] test_SetActivateBoostDelay() (gas: 21992)
[PASS] test_SetActivateBoostDelay_FailsIfGreaterThanHistoryBufferLength() (gas: 14655)
[PASS] test_SetActivateBoostDelay_FailsIfNotOwner() (gas: 12094)
[PASS] test_SetActivateBoostDelay_FailsIfZero() (gas: 14471)
[PASS] test_SetDropBoostDelay() (gas: 21902)
[PASS] test_SetDropBoostDelay_FailsIfGreaterThanHistoryBufferLength() (gas: 14564)
[PASS] test_SetDropBoostDelay_FailsIfNotOwner() (gas: 11984)
[PASS] test_SetDropBoostDelay_FailsIfZero() (gas: 14449)
[PASS] test_SetMinter() (gas: 26731)
[PASS] test_Transfer() (gas: 140462)
[PASS] test_TransferFrom() (gas: 146758)
[PASS] test_WhitelistSender() (gas: 42351)
Suite result: ok. 75 passed; 0 failed; 0 skipped; finished in 17.49s (22.16s CPU time)

Ran 44 tests for test/pol/BGTStaker.t.sol:BGTStakerTest
[PASS] testFuzz_NotifyRewardsFailsIfNotFeeCollector(address) (runs: 1024, μ: 19701, ~: 19701)
[PASS] testFuzz_NotifyRewardsFailsIfRewardInsolvent(uint256,uint256) (runs: 1024, μ: 258823, ~: 259058)
[PASS] testFuzz_PartialWithdraw(address,uint256,uint256) (runs: 1024, μ: 106167, ~: 106198)
[PASS] testFuzz_RecoverERC20(uint256) (runs: 1024, μ: 1415186, ~: 1414501)
[PASS] testFuzz_RecoverERC20FailsIfNotOwner(address) (runs: 1024, μ: 20937, ~: 20937)
[PASS] testFuzz_SetRewardDuration(uint256) (runs: 1024, μ: 30745, ~: 30738)
[PASS] testFuzz_SetRewardDurationFailsIfNotOwner(address) (runs: 1024, μ: 20569, ~: 20569)
[PASS] testFuzz_SetRewardsDurationDuringCycleEarned(uint256,uint256) (runs: 1024, μ: 495993, ~: 496144)
[PASS] testFuzz_Stake(address,uint256) (runs: 1024, μ: 89987, ~: 89993)
[PASS] testFuzz_StakeFailsIfNotBGT(address) (runs: 1024, μ: 20045, ~: 20045)
[PASS] testFuzz_Withdraw(address,uint256) (runs: 1024, μ: 80616, ~: 80619)
[PASS] testFuzz_WithdrawFailsIfInsufficientStake(address,uint256,uint256) (runs: 1024, μ: 92748, ~:
93125)
[PASS] testFuzz_WithdrawFailsIfNotBGT(address) (runs: 1024, μ: 19977, ~: 19977)
[PASS] testFuzz_getRewardWithLowTotalSupply(uint256) (runs: 1024, μ: 646014, ~: 645472)
[PASS] test_GetReward_AfterDuration(uint256,uint256) (runs: 1024, μ: 469533, ~: 469085)
```

```
[PASS] test_GetReward_BeforeDuration(uint256,uint256) (runs: 1024, μ: 469155, ~: 469281)
[PASS] test_GetReward_Notified_Twice(uint256,uint256) (runs: 1024, μ: 548295, ~: 547980)
[PASS] test_GetRewards() (gas: 545685)
[PASS] test_NotifyRewardsDoesNotSetRewardRate() (gas: 359528)
[PASS] test_NotifyRewardsFailsIfNotFeeCollector() (gas: 19402)
[PASS] test_NotifyRewardsFailsIfRewardInsolvent() (gas: 258205)
[PASS] test_NotifyRewardsSetRewardRate() (gas: 464988)
[PASS] test_OwnerIsGovernance() (gas: 18380)
[PASS] test_RecoverERC20() (gas: 1414250)
[PASS] test_RecoverERC20FailsIfNotOwner() (gas: 20703)
[PASS] test_RecoverERC20FailsIfRewardToken() (gas: 229119)
[PASS] test_RewardRateRemains_Zero_UntilFirstStake() (gas: 269804)
[PASS] test_RewardRateWithMultipleDistributeRewards() (gas: 795916)
[PASS] test_SetRewardDuration() (gas: 30508)
[PASS] test_SetRewardDurationDuringCycleEarned() (gas: 494767)
[PASS] test_SetRewardDurationFailsIfNotOwner() (gas: 20336)
[PASS] test_SetRewardDuration_FailsIfZero() (gas: 18890)
[PASS] test_SetRewardRateAfterFirstStake() (gas: 390288)
[PASS] test_SetRewardsDurationDuringCycle() (gas: 508455)
[PASS] test_SetRewardsDurationDuringCycleMultipleUsers() (gas: 544980)
[PASS] test_Stake() (gas: 89527)
[PASS] test_StakeFailsIfNotBGT() (gas: 19745)
[PASS] test_Stake_FailsIfOverflow(address) (runs: 1024, μ: 96884, ~: 96884)
[PASS] test_Stake_FailsIfZeroAmount() (gas: 19115)
[PASS] test_UpgradeFailsIfNewImplNotUUPS() (gas: 24949)
[PASS] test_UpgradeFailsIfNotOwner() (gas: 2162006)
[PASS] test_UpgradeTo() (gas: 2170561)
[PASS] test_WithdrawFailsIfNotBGT() (gas: 19700)
[PASS] test_Withdraw_FailsIfZeroAmount() (gas: 19113)
Suite result: ok. 44 passed; 0 failed; 0 skipped; finished in 17.49s (16.85s CPU time)

Ran 120 tests for test/pol/RewardVault.t.sol:RewardVaultTest
[PASS] testFuzz_AddIncentive_FailsIfAmountLessThanMinRate(uint256) (runs: 1024, μ: 316708, ~: 316045)
[PASS] testFuzz_AddIncentive_FailsIfRateMoreThanMaxIncentiveRate(uint256) (runs: 1024, μ: 315652, ~:
316172)
[PASS] testFuzz_AddIncentive_IncentiveRateNotChanged(uint256) (runs: 1024, μ: 600697, ~: 600740)
[PASS] testFuzz_AddIncentive_IncreaseIncentiveRate(uint256) (runs: 1024, μ: 601302, ~: 601519)
[PASS] testFuzz_AddIncentive_UpdatesIncentiveRate(uint256,uint256) (runs: 1024, μ: 586656, ~: 587208)
[PASS] testFuzz_ChangeInFactoryOwner(address) (runs: 1024, μ: 74110, ~: 74110)
[PASS] testFuzz_DelegateStake(address,address,uint256) (runs: 1024, μ: 225810, ~: 225675)
[PASS] testFuzz_DelegateWithdraw(address,address,uint256,uint256) (runs: 1024, μ: 273327, ~: 273300)
[PASS] testFuzz_DelegateWithdrawFailsIfNotEnoughStakedByDelegate(uint256,uint256,uint256) (runs: 1024, μ:
457294, ~: 457080)
[PASS] testFuzz_DistributeDoesNotLeaveDust(uint256) (runs: 1024, μ: 2382034, ~: 2382393)
[PASS] testFuzz_DistributeDoesNotLeaveDust(uint256,uint256,uint256,uint256,uint256) (runs: 1024, μ:
2452352, ~: 2457351)
[PASS] testFuzz_Exit(uint256,uint256) (runs: 1024, μ: 1220225, ~: 1184911)
[PASS] testFuzz_GetRewardToRecipient(address) (runs: 1024, μ: 1115251, ~: 1115251)
[PASS] testFuzz_NotifyRewardsFailsIfRewardInsolvent(uint256,uint256) (runs: 1024, μ: 265825, ~: 266016)
[PASS] testFuzz_PartialWithdraw(address,uint256,uint256) (runs: 1024, μ: 376398, ~: 376374)
[PASS] testFuzz_ProcessIncentives(uint256) (runs: 1024, μ: 902787, ~: 940933)
[PASS] testFuzz_RemoveIncentiveToken(address) (runs: 1024, μ: 145480, ~: 145466)
[PASS] testFuzz_RemoveIncentiveToken_Multiple(uint8,uint256) (runs: 1024, μ: 5760290, ~: 5025062)
[PASS] testFuzz_SetRewardDuration(uint256) (runs: 1024, μ: 41288, ~: 41281)
[PASS] testFuzz_SetRewardsDurationDuringCycleEarned(uint256,uint256) (runs: 1024, μ: 821727, ~: 821866)
[PASS] testFuzz_Stake(address,uint256) (runs: 1024, μ: 341986, ~: 342005)
[PASS] testFuzz_UpdateIncentiveManager(address,address) (runs: 1024, μ: 178826, ~: 178826)
[PASS] testFuzz_WhitelistIncentiveToken(address,address) (runs: 1024, μ: 162344, ~: 162344)
[PASS] testFuzz_WhitelistIncentiveToken_FailsIfMinIncentiveRateMoreThanMax(uint256) (runs: 1024, μ:
39625, ~: 40145)
[PASS] testFuzz_Withdraw(address,uint256) (runs: 1024, μ: 375893, ~: 375914)
[PASS] testFuzz_Withdraw_FailsIfInsufficientSelfStake(uint256,uint256,uint256) (runs: 1024, μ: 460893, ~:
460767)
[PASS] test_AddIncentive_FailIfNotManager() (gas: 311920)
[PASS] test_AddIncentive_FailsIfAmountLessThanMinIncentiveRate() (gas: 312267)
[PASS] test_AddIncentive_FailsIfNotWhitelisted() (gas: 39121)
[PASS] test_AddIncentive_FailsIfNotWhitelistedToken() (gas: 39080)
[PASS] test_AddIncentive_FailsIfRateIsMoreThanMaxIncentiveRate() (gas: 315920)
Logs:
  Bound result 115792089237316195423570985008687907853269084665640564039457584007913129639935

[PASS] test_ChangeInFactoryOwner() (gas: 75970)
```

```
[PASS] test_DelegateStake() (gas: 228832)
[PASS] test_DelegateStakeFailsIfPused() (gas: 81700)
[PASS] test_DelegateStakeWithSelfStake() (gas: 539576)
[PASS] test_DelegateWithdraw() (gas: 243854)
[PASS] test_DelegateWithdrawFailsIfNotDelegate() (gas: 34338)
[PASS] test_DelegateWithdrawFailsIfNotEnoughStakedByDelegate() (gas: 222113)
[PASS] test_Distribute() (gas: 667368)
[PASS] test_DistributeAndActivateQueuedRewardAllocation() (gas: 764859)
[PASS] test_DistributeDuringGenesisNoBgtWaste() (gas: 341253)
[PASS] test_DistributeForNonReentrant() (gas: 4109769)
[PASS] test_DistributeMulticall() (gas: 899033)
[PASS] test_DoNotUpdateIncentiveRateWhenAmountIsInsufficient() (gas: 596232)
[PASS] test_Exit() (gas: 1257719)
Logs:
  Bound result 100000000000000000000
  Bound result 100000000000000000000

[PASS] test_ExitWithZeroBalance() (gas: 44350)
[PASS] test_FactoryOwner() (gas: 16061)
[PASS] test_FailIfInitializeAgain() (gas: 23828)
[PASS] test_FailIfNotManager() (gas: 34308)
[PASS] test_FailIfNotOwner() (gas: 101421)
[PASS] test_FailNotifyRewardNoSufficientAllowance(int256) (runs: 1024, μ: 1055684, ~: 1058694)
[PASS] test_GetRewardNotOperatorOrUser() (gas: 41694)
[PASS] test_GetRewardWithDelegateStake() (gas: 1021228)
[PASS] test_GetReward_AfterDuration(uint256,uint256) (runs: 1024, μ: 765213, ~: 764880)
[PASS] test_GetReward_BeforeDuration(uint256,uint256) (runs: 1024, μ: 764865, ~: 764940)
[PASS] test_GetReward_Notified_Twice(uint256,uint256) (runs: 1024, μ: 873645, ~: 873312)
[PASS] test_GetRewards() (gas: 970359)
[PASS] test_GetWhitelistedTokens() (gas: 366189)
[PASS] test_IncreaseIncentiveRate() (gas: 596297)
[PASS] test_InitialState() (gas: 37461)
[PASS] test_IsTimestampActionable_OutOfBuffer() (gas: 20634)
[PASS] test_IsTimestampActionable_Processing() (gas: 296016)
[PASS] test_NotifyRewardsDoesNotSetRewardRate() (gas: 510633)
[PASS] test_NotifyRewardsFailsIfRewardInsolvent() (gas: 265124)
[PASS] test_NotifyRewardsSetRewardRate() (gas: 880268)
[PASS] test_OperatorWorks() (gas: 1143643)
[PASS] test_OwnerIsGovernance() (gas: 26503)
[PASS] test_PanprogL1() (gas: 690603)
[PASS] test_Pause() (gas: 54396)
[PASS] test_Pause_FailIfNotVaultManager() (gas: 27172)
[PASS] test_ProcessIncentives() (gas: 733435)
[PASS] test_ProcessIncentives_NotFailWithMaliciusIncentive() (gas: 1053925)
[PASS] test_ProcessTimestamp_OutOfBuffer() (gas: 194596)
[PASS] test_ProcessTimestamp_Processing() (gas: 439068)
[PASS] test_RecoverERC20() (gas: 111774)
[PASS] test_RecoverERC20_FailIfNotOwner() (gas: 30148)
[PASS] test_RecoverERC20_FailIfStakingToken() (gas: 34373)
[PASS] test_RecoverERC20_FailsIfIncentiveToken() (gas: 175941)
[PASS] test_RemoveIncentiveToken() (gas: 260898)
[PASS] test_RemoveIncentiveToken_FailsIfNotVaultManager() (gas: 294533)
[PASS] test_RemoveIncentiveToken_FailsIfNotWhitelisted() (gas: 34029)
[PASS] test_RewardRateRemains_Zero_UntilFirstStake() (gas: 311862)
[PASS] test_RewardRateWithMultipleDistributeRewards() (gas: 1300475)
[PASS] test_SelfStake() (gas: 351969)
[PASS] test_SetDistributor() (gas: 41383)
[PASS] test_SetDistributor_FailIfNotOwner() (gas: 27569)
[PASS] test_SetDistributor_FailWithZeroAddress() (gas: 29786)
[PASS] test_SetMaxIncentiveTokensCount() (gas: 41348)
[PASS] test_SetMaxIncentiveTokensCount_FailsIfLessThanCurrentWhitelistedCount() (gas: 292730)
[PASS] test_SetMaxIncentiveTokensCount_FailsIfNotOwner() (gas: 27612)
[PASS] test_SetOperator() (gas: 45608)
[PASS] test_SetRewardDuration() (gas: 40986)
[PASS] test_SetRewardDurationDuringCycleEarned() (gas: 820488)
[PASS] test_SetRewardDuration_FailIfNotOwner() (gas: 27665)
[PASS] test_SetRewardDuration_FailsIfZero() (gas: 29749)
[PASS] test_SetRewardRateAfterFirstStake() (gas: 675773)
[PASS] test_SetRewardsDurationDuringCycle() (gas: 833201)
[PASS] test_SetRewardsDurationDuringCycleMultipleUsers() (gas: 1097422)
[PASS] test_Stake() (gas: 341797)
[PASS] test_StakeFailsIfPaused() (gas: 79194)
```

```
[PASS] test_Stake_FailsIfOverflow(address) (runs: 1024, μ: 349286, ~: 349291)
[PASS] test_Stake_FailsIfZeroAmount() (gas: 36637)
[PASS] test_TotalSupply() (gas: 600115)
[PASS] test_Unpause() (gas: 45013)
[PASS] test_Unpause_FailIfNotVaultManager() (gas: 62782)
[PASS] test_UpdateIncentiveManager() (gas: 182583)
[PASS] test_UpdateIncentiveManager_FailsIfNotFactoryOwner() (gas: 30118)
[PASS] test_UpdateIncentiveManager_FailsIfTokenNotWhitelisted() (gas: 57110)
[PASS] test_UpdateIncentiveManager_FailsIfZeroAddress() (gas: 293303)
[PASS] test_UpgradeTo() (gas: 2639127)
[PASS] test_WhitelistIncentiveToken() (gas: 285555)
[PASS] test_WhitelistIncentiveToken_FailsIfAlreadyWhitelisted() (gas: 294177)
[PASS] test_WhitelistIncentiveToken_FailsIfCountEqualToMax() (gas: 1090961)
[PASS] test_WhitelistIncentiveToken_FailsIfMinIncentiveRateIsZero() (gas: 34820)
[PASS] test_WhitelistIncentiveToken_FailsIfMinIncentiveRateMoreThanMax() (gas: 38419)
Logs:
  Bound result 1000000000000000000000000000000000000

[PASS] test_WhitelistIncentiveToken_FailsIfNotOwner() (gas: 32652)
[PASS] test_WhitelistIncentiveToken_FailsIfZeroAddress() (gas: 55496)
[PASS] test_Withdraw_FailsIfInsufficientSelfStake() (gas: 459937)
Logs:
  Bound result 10000000000000000000
  Bound result 100000000000000000000
  Bound result 100000000000000000000

[PASS] test_Withdraw_FailsIfZeroAmount() (gas: 38861)
[PASS] test_ZeroRewards() (gas: 314543)
Suite result: ok. 120 passed; 0 failed; 0 skipped; finished in 23.54s (60.12s CPU time)

Ran 2 tests for test/pol/e2e/POLE2EFuzz.t.sol:POLE2EFuzz
[PASS] testFuzzDistribution(uint96[],uint256,uint256) (runs: 1024, μ: 21047796, ~: 21574187)
[PASS] testGasPOLDistribution() (gas: 275219)
Suite result: ok. 2 passed; 0 failed; 0 skipped; finished in 29.35s (28.99s CPU time)

Ran 29 test suites in 29.43s (137.50s CPU time): 610 tests passed, 0 failed, 0 skipped (610 total tests)
```

Code Coverage

Test coverage was checked in the commit `97608d2`, with near 100% coverage for the smart contracts within the scope of this audit.

```
╭--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------╮
| File                                                   | % Lines            | % Statements       | %
Branches       | % Funcs            |
+========================================================================================================
================================+
| script/base/Base.s.sol                                 | 0.00% (0/24)       | 0.00% (0/19)       |
0.00% (0/10)     | 0.00% (0/6)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| script/base/BasePredict.s.sol                          | 0.00% (0/6)        | 0.00% (0/6)        |
100.00% (0/0)    | 0.00% (0/2)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| script/base/actions/ValidateUpgrade.s.sol              | 0.00% (0/6)        | 0.00% (0/5)        |
100.00% (0/0)    | 0.00% (0/1)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| script/berps/Deploy.s.sol                              | 0.00% (0/77)       | 0.00% (0/67)       |
100.00% (0/0)    | 0.00% (0/11)     |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| script/berps/Markets.s.sol                             | 0.00% (0/70)       | 0.00% (0/83)       |
100.00% (0/0)    | 0.00% (0/5)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| script/berps/UpdateParams.s.sol                        | 0.00% (0/52)       | 0.00% (0/52)       |
100.00% (0/0)     | 0.00% (0/6)       |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+-----------------|
| script/gov/GovernancePredictAddresses.s.sol            | 0.00% (0/3)        | 0.00% (0/2)        |
100.00% (0/0)     | 0.00% (0/1)       |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+-----------------|
| script/gov/deployment/1_DeployGovernance.s.sol         | 0.00% (0/19)       | 0.00% (0/22)       |
0.00% (0/22)      | 0.00% (0/1)       |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+-----------------|
| script/honey/HoneyPredictAddresses.s.sol               | 0.00% (0/3)        | 0.00% (0/2)        |
100.00% (0/0)     | 0.00% (0/1)       |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+-----------------|
| script/honey/actions/AddCollateral.s.sol               | 0.00% (0/15)       | 0.00% (0/14)       |
0.00% (0/6)       | 0.00% (0/2)       |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+-----------------|
| script/honey/actions/MintHoney.s.sol                   | 0.00% (0/16)       | 0.00% (0/15)       |
0.00% (0/6)       | 0.00% (0/2)       |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+-----------------|
| script/honey/deployment/1_DeployHoney.s.sol            | 0.00% (0/22)       | 0.00% (0/20)       |
0.00% (0/10)      | 0.00% (0/2)       |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+-----------------|
| script/honey/deployment/2_TransferHoneyOwnership.s.sol | 0.00% (0/41)       | 0.00% (0/37)       |
0.00% (0/16)      | 0.00% (0/4)       |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+-----------------|
| script/misc/testnet/DeployToken.s.sol                  | 0.00% (0/11)       | 0.00% (0/7)        |
100.00% (0/0)     | 0.00% (0/4)       |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+-----------------|
| script/misc/testnet/tokens/DAI.sol                     | 0.00% (0/12)       | 0.00% (0/6)        |
100.00% (0/0)     | 0.00% (0/6)       |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+-----------------|
| script/misc/testnet/tokens/USDC.sol                    | 0.00% (0/12)       | 0.00% (0/6)        |
100.00% (0/0)     | 0.00% (0/6)       |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+-----------------|
| script/misc/testnet/tokens/USDT.sol                    | 0.00% (0/8)        | 0.00% (0/4)        |
100.00% (0/0)     | 0.00% (0/4)       |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+-----------------|
| script/pol/0_DeployBeaconDeposit.s.sol                 | 0.00% (0/5)        | 0.00% (0/4)        |
100.00% (0/0)     | 0.00% (0/2)       |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+-----------------|
| script/pol/POLPredictAddresses.s.sol                   | 0.00% (0/10)       | 0.00% (0/9)        |
100.00% (0/0)     | 0.00% (0/1)       |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+-----------------|
| script/pol/actions/DeployRewardVault.s.sol             | 0.00% (0/18)       | 0.00% (0/18)       |
0.00% (0/5)       | 0.00% (0/3)       |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+-----------------|
| script/pol/actions/SetDefaultRewardAllocation.s.sol    | 0.00% (0/12)       | 0.00% (0/13)       |
0.00% (0/2)       | 0.00% (0/2)       |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+-----------------|
| script/pol/actions/TransferBera.s.sol                  | 0.00% (0/4)        | 0.00% (0/3)        |
100.00% (0/0)     | 0.00% (0/1)       |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+-----------------|
| script/pol/actions/TriggerFeeCollector.s.sol           | 0.00% (0/29)       | 0.00% (0/32)       |
0.00% (0/5)       | 0.00% (0/4)       |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+-----------------|
| script/pol/actions/WhitelistIncentiveToken.s.sol       | 0.00% (0/11)       | 0.00% (0/10)       |
0.00% (0/2)      | 0.00% (0/3)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| script/pol/actions/WhitelistRewardVault.s.sol          | 0.00% (0/16)       | 0.00% (0/15)       |
0.00% (0/5)      | 0.00% (0/3)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| script/pol/deployment/1_DeployWBERA.s.sol              | 0.00% (0/3)        | 0.00% (0/3)        |
100.00% (0/0)    | 0.00% (0/1)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| script/pol/deployment/2_DeployBGT.s.sol                | 0.00% (0/6)        | 0.00% (0/5)        |
0.00% (0/3)      | 0.00% (0/1)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| script/pol/deployment/3_DeployPoL.s.sol                | 0.00% (0/35)       | 0.00% (0/32)       |
100.00% (0/0)    | 0.00% (0/3)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| script/pol/deployment/4_TransferPOLOwnership.s.sol     | 0.00% (0/70)       | 0.00% (0/65)       |
0.00% (0/26)     | 0.00% (0/5)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| script/pol/logic/BGTDeployer.sol                       | 0.00% (0/5)        | 0.00% (0/4)        |
0.00% (0/2)      | 0.00% (0/1)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| script/pol/logic/ConfigPOL.sol                         | 0.00% (0/31)       | 0.00% (0/30)       |
0.00% (0/20)     | 0.00% (0/1)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| script/pol/logic/ForceTransferBera.sol                 | 0.00% (0/7)        | 0.00% (0/7)        |
0.00% (0/3)      | 0.00% (0/1)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| script/pol/logic/WBERADeployer.sol                     | 0.00% (0/3)        | 0.00% (0/2)        |
100.00% (0/0)    | 0.00% (0/1)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| script/pol/proposals/SetDefaultRewardAllocation.s.sol  | 0.00% (0/28)       | 0.00% (0/33)       |
0.00% (0/4)      | 0.00% (0/2)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| script/pol/proposals/WhitelistRewardVaults.s.sol       | 0.00% (0/24)       | 0.00% (0/28)       |
0.00% (0/4)      | 0.00% (0/2)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| script/pol/testnet/DeployProver.s.sol                  | 0.00% (0/17)       | 0.00% (0/19)       |
100.00% (0/0)    | 0.00% (0/1)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/WBERA.sol                                          | 0.00% (0/4)        | 0.00% (0/2)        |
100.00% (0/0)    | 0.00% (0/2)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/base/Create2Deployer.sol                           | 100.00% (18/18)    | 100.00% (17/17)    |
100.00% (1/1)    | 100.00% (6/6)    |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/base/FactoryOwnable.sol                            | 96.67% (29/30)     | 100.00% (30/30)    |
100.00% (2/2)    | 91.67% (11/12)   |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/base/StakingRewards.sol                            | 100.00% (101/101)  | 100.00% (102/102)  |
100.00% (12/12)  | 100.00% (21/21)  |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/berps/core/v0/Entrypoint.sol                       | 61.21% (101/165)   | 60.73% (133/219)   |
14.29% (6/42)    | 59.26% (16/27)   |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/berps/core/v0/FeesAccrued.sol                      | 39.45% (101/256)   | 40.44% (110/272)   |
12.50% (4/32)     | 31.71% (13/41)     |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/berps/core/v0/FeesMarkets.sol                      | 38.20% (68/178)    | 44.57% (82/184)    |
14.81% (4/27)     | 22.73% (10/44)     |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/berps/core/v0/Markets.sol                          | 49.47% (47/95)     | 42.86% (33/77)     |
34.48% (10/29)    | 54.84% (17/31)     |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/berps/core/v0/Orders.sol                           | 73.33% (121/165)   | 75.58% (130/172)   |
12.50% (2/16)     | 72.41% (21/29)     |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/berps/core/v0/Referrals.sol                        | 67.50% (54/80)     | 68.42% (65/95)     |
41.67% (5/12)     | 52.94% (9/17)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/berps/core/v0/Settlement.sol                       | 70.97% (176/248)   | 74.20% (210/283)   |
38.46% (15/39)    | 60.61% (20/33)     |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/berps/core/v0/Vault.sol                            | 59.41% (161/271)   | 63.01% (201/319)   |
15.00% (6/40)     | 39.62% (21/53)     |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/berps/core/v0/VaultSafetyModule.sol                | 20.00% (9/45)      | 26.09% (12/46)     |
0.00% (0/10)      | 16.67% (2/12)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/berps/deploy/v0/BerpsDeployer.sol                  | 0.00% (0/76)       | 0.00% (0/71)       |
0.00% (0/11)      | 0.00% (0/20)       |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/berps/utils/Delegatable.sol                        | 72.22% (13/18)     | 75.00% (12/16)     |
50.00% (4/8)      | 40.00% (2/5)       |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/berps/utils/PriceUtils.sol                         | 71.64% (48/67)     | 69.79% (67/96)     |
47.37% (9/19)     | 90.91% (10/11)     |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/berps/utils/StorageUtils.sol                       | 100.00% (6/6)      | 100.00% (4/4)      |
100.00% (0/0)     | 100.00% (3/3)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/berps/utils/TradeUtils.sol                         | 48.98% (24/49)     | 60.00% (36/60)     |
0.00% (0/19)      | 77.78% (7/9)       |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/gov/BerachainGovernance.sol                        | 100.00% (46/46)    | 100.00% (49/49)    |
100.00% (6/6)     | 100.00% (13/13)    |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/gov/GovDeployer.sol                                | 0.00% (0/22)       | 0.00% (0/21)       |
0.00% (0/4)       | 0.00% (0/3)        |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/gov/TimeLock.sol                                   | 100.00% (3/3)      | 100.00% (1/1)      |
100.00% (0/0)     | 100.00% (2/2)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/honey/CollateralVault.sol                          | 96.77% (60/62)     | 98.31% (58/59)     |
100.00% (4/4)     | 95.24% (20/21)     |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/honey/Honey.sol                                    | 100.00% (18/18)    | 100.00% (13/13)    |
100.00% (3/3)     | 100.00% (8/8)      |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/honey/HoneyDeployer.sol                            | 100.00% (7/7)      | 100.00% (8/8)      |
100.00% (0/0)    | 100.00% (1/1)    |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/honey/HoneyFactory.sol                             | 98.94% (93/94)     | 98.97% (96/97)     |
85.71% (6/7)     | 100.00% (18/18)  |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/honey/VaultAdmin.sol                               | 98.57% (69/70)     | 100.00% (60/60)    |
100.00% (9/9)    | 94.12% (16/17)   |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/libraries/BeaconRoots.sol                          | 100.00% (10/10)    | 100.00% (8/8)      |
100.00% (1/1)    | 100.00% (2/2)    |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/libraries/SSZ.sol                                  | 85.29% (29/34)     | 78.12% (25/32)     |
28.57% (2/7)     | 100.00% (4/4)    |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/libraries/Utils.sol                                | 70.00% (28/40)     | 70.59% (24/34)     |
0.00% (0/1)      | 66.67% (6/9)     |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/pol/BGTFeeDeployer.sol                             | 100.00% (7/7)      | 100.00% (8/8)      |
100.00% (0/0)    | 100.00% (1/1)    |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/pol/BGTStaker.sol                                  | 100.00% (27/27)    | 100.00% (18/18)    |
100.00% (3/3)    | 100.00% (13/13)  |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/pol/BeaconDeposit.sol                              | 93.33% (56/60)     | 93.22% (55/59)     |
81.25% (13/16)   | 100.00% (8/8)    |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/pol/BeaconRootsHelper.sol                          | 88.89% (24/27)     | 89.29% (25/28)     |
25.00% (1/4)     | 100.00% (6/6)    |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/pol/FeeCollector.sol                               | 97.50% (39/40)     | 100.00% (41/41)    |
100.00% (5/5)    | 88.89% (8/9)     |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/pol/POLDeployer.sol                                | 100.00% (14/14)    | 100.00% (18/18)    |
100.00% (0/0)    | 100.00% (1/1)    |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/pol/rewards/BeraChef.sol                           | 98.13% (105/107)   | 97.98% (97/99)     |
90.00% (18/20)   | 100.00% (20/20)  |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/pol/rewards/BlockRewardController.sol              | 92.31% (60/65)     | 92.31% (60/65)     |
66.67% (10/15)   | 100.00% (12/12)  |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/pol/rewards/Distributor.sol                        | 95.00% (38/40)     | 97.37% (37/38)     |
100.00% (3/3)    | 85.71% (6/7)     |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/pol/rewards/RewardVault.sol                        | 100.00% (149/149)  | 99.34% (150/151)   |
96.55% (28/29)   | 100.00% (34/34)  |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| src/pol/rewards/RewardVaultFactory.sol                 | 100.00% (34/34)    | 100.00% (31/31)    |
100.00% (2/2)    | 100.00% (8/8)    |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| test/base/BeaconRoots.t.sol                            | 100.00% (2/2)      | 100.00% (2/2)      |
100.00% (0/0)    | 100.00% (1/1)    |
|--------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| test/base/Create2Deployer.t.sol                        | 100.00% (2/2)      | 100.00% (2/2)      |
100.00% (0/0)     | 100.00% (1/1)     |
|---------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| test/berps/integration/BaseTradingTest.t.sol            | 97.78% (44/45)     | 100.00% (44/44)    |
100.00% (0/0)     | 50.00% (1/2)      |
|---------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| test/gov/GovernanceBase.t.sol                           | 98.51% (66/67)     | 98.95% (94/95)     |
100.00% (5/5)     | 100.00% (6/6)     |
|---------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| test/honey/HoneyBase.t.sol                              | 100.00% (20/20)    | 100.00% (21/21)    |
100.00% (0/0)     | 100.00% (1/1)     |
|---------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| test/mock/berps/MockFeeCollector.sol                    | 40.00% (2/5)       | 50.00% (1/2)       |
100.00% (0/0)     | 33.33% (1/3)      |
|---------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| test/mock/berps/MockOrders.sol                          | 100.00% (8/8)      | 100.00% (5/5)      |
100.00% (0/0)     | 100.00% (3/3)     |
|---------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| test/mock/honey/MockAssets.sol                          | 78.57% (22/28)     | 78.57% (11/14)     |
100.00% (0/0)     | 78.57% (11/14)    |
|---------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| test/mock/honey/MockHoney.sol                           | 80.00% (8/10)      | 80.00% (4/5)       |
100.00% (0/0)     | 80.00% (4/5)      |
|---------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| test/mock/honey/MockVault.sol                           | 33.33% (8/24)      | 33.33% (4/12)      |
100.00% (0/0)     | 33.33% (4/12)     |
|---------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| test/mock/pol/BeaconDepositMock.sol                     | 100.00% (4/4)      | 100.00% (2/2)      |
100.00% (0/0)     | 100.00% (2/2)     |
|---------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| test/mock/pol/Mock4788BeaconRoots.sol                   | 100.00% (12/12)    | 100.00% (9/9)      |
100.00% (1/1)     | 100.00% (3/3)     |
|---------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| test/mock/pol/MockRewardVault.sol                       | 66.67% (4/6)       | 66.67% (2/3)       |
100.00% (0/0)     | 66.67% (2/3)      |
|---------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| test/mock/pol/NoopBeraChef.sol                          | 0.00% (0/19)       | 0.00% (0/6)        |
100.00% (0/0)     | 0.00% (0/13)      |
|---------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| test/mock/pol/NoopBlockRewardController.sol             | 0.00% (0/20)       | 0.00% (0/7)        |
100.00% (0/0)     | 0.00% (0/13)      |
|---------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| test/mock/token/MaxGasConsumeERC20.sol                  | 100.00% (5/5)      | 100.00% (6/6)      |
100.00% (0/0)     | 100.00% (2/2)     |
|---------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| test/mock/token/MockBGT.sol                             | 100.00% (5/5)      | 100.00% (3/3)      |
100.00% (0/0)     | 100.00% (2/2)     |
|---------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| test/mock/token/MockERC20.sol                           | 66.67% (4/6)       | 66.67% (2/3)       |
100.00% (0/0)     | 66.67% (2/3)      |
|---------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| test/mock/token/MockERC4626.sol                         | 0.00% (0/11)       | 0.00% (0/8)        |
100.00% (0/0)     | 0.00% (0/5)       |
|---------------------------------------------------------+--------------------+--------------------+-----
-------------+------------------|
| test/mock/token/PausableERC20.sol                       | 50.00% (3/6)       | 25.00% (1/4)       |
100.00% (0/0)    | 66.67% (2/3)     |
|--------------------------------------------------------+---------------------+---------------------+-----
-------------+------------------|
| test/mock/token/ReentrantERC20.sol                    | 92.31% (12/13)     | 80.00% (8/10)       |
100.00% (1/1)    | 100.00% (3/3)    |
|--------------------------------------------------------+---------------------+---------------------+-----
-------------+------------------|
| test/pol/POL.t.sol                                    | 100.00% (28/28)    | 100.00% (24/24)     |
100.00% (0/0)    | 100.00% (4/4)    |
|--------------------------------------------------------+---------------------+---------------------+-----
-------------+------------------|
| test/pol/e2e/POLGasSimulationAdvance.t.sol            | 100.00% (9/9)      | 100.00% (10/10)     |
100.00% (0/0)    | 100.00% (1/1)    |
|--------------------------------------------------------+---------------------+---------------------+-----
-------------+------------------|
| Total                                                 | 57.47% (2261/3934) | 59.54% (2381/3999) |
32.63% (201/616) | 56.50% (452/800) |
╰--------------------------------------------------------+---------------------+---------------------+-----
-------------+------------------╯
```

Changelog

- 2024-12-13 - Initial Report
- 2024-12-31 - Updated Report
- 2025-01-14 - Final Report

About Quantstamp

Quantstamp is a global leader in blockchain security. Founded in 2017, Quantstamp's mission is to securely onboard the next billion users to Web3 through its best-in-class Web3 security products and services.

Quantstamp's team consists of cybersecurity experts hailing from globally recognized organizations including Microsoft, AWS, BMW, Meta, and the Ethereum Foundation. Quantstamp engineers hold PhDs or advanced computer science degrees, with decades of combined experience in formal verification, static analysis, blockchain audits, penetration testing, and original leading-edge research.

To date, Quantstamp has performed more than 500 audits and secured over $200 billion in digital asset risk from hackers. Quantstamp has worked with a diverse range of customers, including startups, category leaders and financial institutions. Brands that Quantstamp has worked with include Ethereum 2.0, Binance, Visa, PayPal, Polygon, Avalanche, Curve, Solana, Compound, Lido, MakerDAO, Arbitrum, OpenSea and the World Economic Forum.

Quantstamp's collaborations and partnerships showcase our commitment to world-class research, development and security. We're honored to work with some of the top names in the industry and proud to secure the future of web3.

Notable Collaborations & Customers:

- Blockchains: Ethereum 2.0, Near, Flow, Avalanche, Solana, Cardano, Binance Smart Chain, Hedera Hashgraph, Tezos
- DeFi: Curve, Compound, Maker, Lido, Polygon, Arbitrum, SushiSwap
- NFT: OpenSea, Parallel, Dapper Labs, Decentraland, Sandbox, Axie Infinity, Illuvium, NBA Top Shot, Zora
- Academic institutions: National University of Singapore, MIT

responsibility for the use of third-party software on any website and shall have no liability whatsoever to any person or entity for the accuracy or completeness of any output generated by such software.

Disclaimer

The review and this report are provided on an as-is, where-is, and as-available basis. To the fullest extent permitted by law, Quantstamp disclaims all warranties, expressed implied, in connection with this report, its content, and the related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement. You agree that access and/or use of the report and other results of the review, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE. This report is based on the scope of materials and documentation provided for a limited review at the time provided. You acknowledge that Blockchain technology remains under development and is subject to unknown risks and flaws and, as such, the report may not be complete or inclusive of all vulnerabilities. The review is limited to the materials identified in the report and does not extend to the compiler layer, or any other areas beyond the programming language, or programming aspects that could present security risks. The report does not indicate the endorsement by Quantstamp of any particular project or team, nor guarantee its security, and and may not be represented as such. No third party is entitled to rely on the report in any any way, including for the purpose of making any decisions to buy or sell a product, product, service or any other asset. Quantstamp does not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party, or or any open source or third-party software, code, libraries, materials, or information to, to, called by, referenced by or accessible through the report, its content, or any related related services and products, any hyperlinked websites, or any other websites or mobile applications, and we will not be a party to or in any way be responsible for monitoring any any transaction between you and any third party. As with the purchase or use of a product or service through any medium or in any environment, you should use your best judgment and exercise caution where appropriate.

Quantstamp