PERIMETER

Berachain Honey #1

Defensive Fuzzing Report

Jan. 30, 2025

Draft

Prepared By:
0xScourgedev | Lead Fuzzing Specialist
0xscourgedev@perimetersec.io

Rappie | Lead Fuzzing Specialist
rappie@perimetersec.io

Table of Contents

Draft

About Perimeter

Perimeter's mission is to deliver the highest quality fuzzing services to protocols by uniting the world's foremost fuzzing specialists. We possess extensive expertise in fuzzing a diverse range of protocols, from smaller, niche protocols to some of the largest and most complex in DeFi.

In order to deliver on our mission, we have developed the most advanced scaffolding and libraries, enabling us to create highly sophisticated fuzzing suites tailored to meet the unique challenges of each protocol.

Learn more about us at perimetersec.io.

Draft

Risk Classification

The severity of security issues identified during the security review is classified according to the table below.

A. Critical findings are highly likely to be exploited with severe impact on the protocol and require immediate attention.

B. High findings are very likely to occur, easy to exploit, or difficult but highly incentivized, and should be resolved as quickly as possible.

C. Medium findings are possible in certain circumstances or when incentivized, with a moderate likelihood of occurring, and should be addressed.

D. Low findings involve rare circumstances to exploit or offer little to no incentives, though addressing them is still recommended.

E. Informational issues represent improvements that do not impact the project's overall security but are worth considering.

Severity Level	High Impact	Medium Impact	Low Impact
High Likelihood	Critical	High	Medium
Medium Likelihood	High	Medium	Low
Low Likelihood	Medium	Low	Low

Draft

Services Provided

Perimeter has successfully delivered a comprehensive suite of services that include:

- **Fuzzing Suite Development:** Design and implement a stateful fuzzing suite using Echidna and Medusa. This suite will be tailor-made for the protocol and contracts in scope. The completed fuzzing suite can later be integrated into the testing suite to serve long-term security needs.
- **Findings Reporting:** We provided thorough documentation and reporting of all findings identified throughout the engagement.
- **Invariant Testing Assurance:** Guarantee that each invariant implemented will be tested no fewer than 50,000,000 instances, ensuring thorough validation and reliability.
- **Proof-of-Concept Development:** Develop a corresponding Proof-of-Concept (PoC) for each finding and assertion/property counterexample identified, to demonstrate potential vulnerabilities and their implications.
- **Comprehensive Final Report:** Create a detailed final report that will include all findings, along with their corresponding PoCs. This report will also detail the invariants tested, their run status, and the number of runs, providing a comprehensive overview of the engagement's outcomes.

Draft

Files in Scope

The engagement will be focused on the files listed below, acquired from commit 90ddc6b835d1b9eb2c09088cdfdb1cda2fba910f.

```
src
├── honey
│   ├── CollateralVault.sol
│   ├── HoneyFactoryReader.sol
│   ├── HoneyFactory.sol
│   ├── Honey.sol
│   ├── IHoneyErrors.sol
│   ├── IHoneyFactory.sol
│   └── VaultAdmin.sol
└── libraries
    └── Utils.sol
```

Files Out of Scope

Files outside the scope were not directly considered in achieving the target. However, since many of these files are utilized by those within the scope, a significant portion was indirectly covered.

Draft

Methodology

The primary goal of this engagement was to achieve comprehensive coverage of the Honey codebase with a robust defensive fuzzing suite. During the process, numerous invariants were identified and implemented. However, many of these invariants were found to be broken during the build phase. This led to a significant shift in focus, as additional time was allocated to identifying and addressing the root causes of these breaks.

This shift in priorities limited the implementation of new invariants. The recurring root cause of these breaks made testing the correctness of newly implemented invariants unfeasible for the frozen commit hash. As a result and after consulting the Berachain team, efforts were concentrated on identifying the underlying issues rather than expanding the invariant set.

The engagement was conducted concurrently with a review by Spearbit. Issues highlighted in this report were identified independently and through the detection of broken invariants. To streamline the debugging process and uncover deeper issues, collaboration with the Spearbit team was initiated.

As part of this collaboration, some issue write-ups in this report reference findings from the Spearbit report. This approach helped save time on debugging and provided deeper insights into the underlying problems in the codebase.

Draft

Issues

At Perimeter, our objective is to thoroughly investigate and uncover critical vulnerabilities through fuzzing and reveal issues often overlooked in manual reviews. Any lower-severity findings are incidental to this core focus.

Severity	Count	Fixed	Acknowledged
Critical	1	1	0
High	0	0	0
Medium	0	0	0
Low	3	-	-
Informational	1	-	-
Total	**5**	**-**	**-**

Draft

CRIT-01: `InsufficientAssets` revert when redeeming, liquidating, or recapitalizing

Description

The current implementation of minting and redeeming relies on the default ERC-4626 logic, which does not enforce a strict 1:1 ratio between Honey tokens and the collateral asset. This behavior can result in fewer shares being minted than the collateral deposited, leading to the `InsufficientAssets` revert in the `checkInvariants` function. Consequently, this issue can disrupt key functionalities such as redemption, liquidation, and recapitalization.

This issue serves as a root cause for potential downstream problems, with the following identified as resulting from broken invariants:

- Liquidation reverting with `FullMulDivFailed`
- Liquidation failing due to insufficient assets in the collateral vault
- Recapitalization reverting with `FullMulDivFailed`
- Total assets of the asset vault do not strictly increase after a successful recapitalization

Recommendation

Ensure the minting and redeeming logic enforces a strict 1:1 ratio between Honey tokens and collateral.

Response

Fixed.

Draft

LOW-01: Loss of Collected Fees Due to Rounding Errors with Lower-Decimal Tokens

Description

The current fee collection mechanism represents fees in vault shares using 18 decimals. However, this discrepancy can lead to significant rounding errors when the underlying asset has fewer than 18 decimals. If the calculated fee amount falls below the precision of the underlying asset, it is rounded down to zero, effectively causing a loss of collected fees.

Example

For a fee of **1e3** (in 18 decimals) applied to a **6-decimal** underlying asset, the calculation produces the following result:

Fee in assets = $(1e3 \times 10^{6}) / 10^{18}$ = **0**

This calculation results in zero assets collected.

Recommendation

Update the fee collection logic to account for the decimal discrepancy between vault shares and the underlying asset, ensuring that fees are accurately represented and collected regardless of the asset's precision.

Response

Awaiting Response

Draft

LOW-02: Unable to mint in basket mode if there is a bad asset

Description

Refer to Spearbit report: spearbit-audits/review-berachain-honey-1025#14

LOW-03: Initial values for Mint & Redeem rate are out of bounds

Description

Refer to Spearbit report: spearbit-audits/review-berachain-honey-1025#24

Draft

INFO-01: Missing checkInvariants modifier for Liquidate and Recapitalize

Description

The `liquidate` and `recapitalize` functions currently lack the `checkInvariants` modifier. As a result, situations that should trigger a revert, similar to `mint` and `redeem`, may not do so.

Recommendation

While `checkInvariants` should never break in production, it is advisable to apply it consistently across the codebase to maintain uniform behavior and safeguard against unforeseen edge cases.

Response

Awaiting Response

Draft

Invariants

We created many tests to verify the correctness of **48** invariants described in the table below. During the execution phase, these invariants were assessed for a total of **3,100,000,000+** calls.

The table below lists all invariants that are part of this engagement.

Invariant	Description	Tested	Passed	# Runs
MINT-01	Minting does not unexpectedly revert	✅	❌	-
MINT-02	When basket mode is not enabled, minting decreases the user's asset balance by the inputted amount	✅	✅	3.1B+
MINT-03	If the inputted amount is not 0, the honey balance of the receiver strictly increases after a successful mint	✅	❌	-
MINT-04	The receiver's honey balance increases by the returned honeyToMint amount after a successful mint	✅	✅	3.1B+
MINT-05	When basket mode is disabled, the difference in the user's honey balance is less than or equal to the difference in the selected vault's total supply	✅	✅	3.1B+
MINT-06	When basket mode is disabled and the inputted amount is not 0, the difference in the collected fees is less than the difference in the receiver's honey balance	✅	❌	-
MINT-07	When basket mode is disabled, if the mint rates for the asset is 0, then the the collected fees for the fee receiver does not change after a successful mint	✅	✅	3.1B+
MINT-08	When basket mode is disabled, if the mint rates for the asset is 0, then the collected fees for the POL fee collector does not change after a successful mint	✅	❌	-

Invariant	Description	Tested	Passed	# Runs
MINT-09	When basket mode is disabled, if the mint rates for the asset is greater than 0 and the inputted amount is not 0, then the sum of the collected fees will strictly increase after a successful mint	✅	❌	-
REDEEM-01	Redeeming does not unexpectedly revert	✅	❌	-
REDEEM-02	The honey balance of the caller decreases by exactly the inputted amount after a successful redemption	✅	❌	-
REDEEM-03	If the inputted amount is not 0, the balance of asset for the receiver strictly increases after a successful redemption	✅	❌	-
REDEEM-04	When basket mode is disabled, the difference in the user's honey balance is less than or equal to the difference in the selected vault's total supply	✅	❌	-
REDEEM-05	When basket mode is disabled and the inputted amount is not 0, the difference in the collected fees is less than the difference in the caller's honey balance	✅	❌	-
REDEEM-06	When basket mode is disabled, if the redeem rates for the asset is 0, then the the collected fees for the fee receiver does not change after a successful redeem	✅	✅	3.1B+
REDEEM-07	When basket mode is disabled, if the redeem rates for the asset is 0, then the collected fees for the POL fee collector does not change after a successful redeem	✅	✅	3.1B+
REDEEM-08	When basket mode is disabled, if the redeem rates for the asset is greater than 0 and the inputted amount is not 0, then the sum of the collected fees will strictly increase after a successful redeem	✅	✅	3.1B+
LIQ-01	Liquidation does not unexpectedly revert	✅	❌	-

Invariant	Description	Tested	Passed	# Runs
LIQ-02	If bad collateral shares of honey factory is not 0 after a successful liquidation, then the caller's good collateral balance decreases by exactly the inputted goodAmount	✅	✅	3.1B+
LIQ-03	If the liquidation rate for the bad collateral is not 0, the caller's bad collateral balance increases by an amount greater than or equal to the inputted goodAmount multiplied by the price of the good collateral divided by the price of the bad collateral after a successful liquidation	✅	❌	-
LIQ-04	The caller's bad collateral balance increases by an amount less than or equal to the inputted goodAmount multiplied by the price of the good collateral divided by the price of the bad collateral multiplied by (1 + liquidation rate) after a successful liquidation	✅	✅	3.1B+
LIQ-05	The total assets of the good collateral vault increases by exactly the difference in the caller's good collateral balance after a successful liquidation	✅	❌	-
LIQ-06	The total assets of the bad collateral vault decreases by exactly the difference in the caller's bad collateral balance after a successful liquidation	✅	❌	-
LIQ-07	The total supply of honey does not change after a successful liquidation	✅	✅	3.1B+
RECAP-01	Recapitalizing does not unexpectedly revert	✅	❌	-
RECAP-02	The caller's asset balance decreases by exactly the inputted amount after a successful recapitalization	✅	✅	3.1B+
RECAP-03	If the inputted amount is not 0, the total assets of the asset vault strictly increases after a successful recapitalization	✅	❌	-
RECAP-04	The total supply of honey does not change after a successful recapitalization	✅	✅	3.1B+

Invariant	Description	Tested	Passed	# Runs
FEE-01	withdrawAllFees does not unexpectedly revert	✅	❌	-
FEE-02	withdrawFee does not unexpectedly revert	✅	❌	-
FEE-03	The collected fees for the fee receiver with the selected asset is 0 after a withdrawFee call for the fee receiver	✅	✅	3.1B+
FEE-04	The collected fees for the POL fee collector with the selected asset is 0 after a withdrawFee call for the POL fee collector	✅	✅	3.1B+
FEE-05	If the collected fees for the fee receiver is not 0 for the selected asset, then the asset balance for the fee receiver strictly increases after a successful withdrawFee call	✅	❌	-
FEE-06	If the collected fees for the POL fee collector is not 0 for the selected asset, then the asset balance for the POL fee collector strictly increases after a successful withdrawFee call	✅	❌	-
FEE-07	The collected fees for the fee receiver for all assets is 0 after a withdrawAllFees call for the fee receiver	✅	✅	3.1B+
FEE-08	The collected fees for the POL fee collector for all assets is 0 after a withdrawAllFees call for the POL fee collector	✅	✅	3.1B+
FEE-09	For each asset, if the collected fees for the fee receiver is not 0, then the asset balance for the fee receiver strictly increases after a successful withdrawAllFees call	✅	❌	-
FEE-10	For each asset, if the collected fees for the POL fee collector is not 0, then the asset balance for the POL fee collector strictly increases after a successful withdrawAllFees call	✅	❌	-
ADMIN-01	setCollateralAssetStatus does not unexpectedly revert	✅	✅	3.1B+
ADMIN-02	setMintRate does not unexpectedly revert	✅	✅	3.1B+

Invariant	Description	Tested	Passed	# Runs
ADMIN-03	setDepegOffsets does not unexpectedly revert	✅	✅	3.1B+
ADMIN-04	setForcedBasketMode does not unexpectedly revert	✅	✅	3.1B+
ADMIN-05	setGlobalCap does not unexpectedly revert	✅	✅	3.1B+
ADMIN-06	setLiquidationEnabled does not unexpectedly revert	✅	✅	3.1B+
ADMIN-07	setLiquidationRate does not unexpectedly revert	✅	✅	3.1B+
ADMIN-08	setPOLFeeCollectorFeeRate does not unexpectedly revert	✅	✅	3.1B+
ADMIN-09	setRedeemRate does not unexpectedly revert	✅	✅	3.1B+
ADMIN-10	setReferenceCollateral does not unexpectedly revert	✅	✅	3.1B+

Draft

Disclaimer

All activities conducted by Perimeter in connection with this project were carried out in accordance with the terms outlined in a Statement of Work and an agreed-upon project plan, as set forth in a proposal document delivered prior to the commencement of the project.

Security assessment projects are subject to time limitations, and as such, the findings presented in this report should not be interpreted as an exhaustive or comprehensive identification of all security issues, vulnerabilities, or defects within the target codebase. Perimeter makes no representations or warranties that the target codebase is free from defects.

Furthermore, this report is not intended to be, and should not be construed as, investment advice or a recommendation to participate in any financial transactions. The content herein does not constitute endorsements or recommendations for any financial decisions, securities, or investment strategies.

Draft