HoneyFun Security Review

Duration: February 01, 2025 - February 02, 2025

Date: February 7, 2025

Conducted by: **KeySecurity**
gkrastenov, Lead Security Researcher

Table of Contents

1 About KeySecurity

KeySecurity is a new, innovative Web3 security company that hires top-talented security researchers for your project. We have conducted over 30 security reviews for various projects, collectively holding over $300,000,000 in TVL. For security audit inquiries, you can reach out to us on Twitter/X or Telegram @gkrastenov or check our previous work here.

2 About HoneyFun

HoneyFun AI is revolutionizing Berachain with Virtuals, NFTs, and AI Agents! The first mover in AI trend on Berachain, merging the power of NFTs, memecoins, and AI Agents into one groundbreaking protocol.

Effortless Launchpad: Launch your AI Agents or memecoins in just a few clicks—we're taking PumpFun experience to an entirely new level with our NFT technology.

Redefining Bonding Curves: Forget traditional token trading! With HoneyFun AI, you trade NFTs embedded with AI Agent or memecoin tokens.

Next-Level Utility: Once the bonding curve ends, NFTs become soulbound and unlock tokens, while LP seamlessly transitions to Kodiak DEX through our official partnership.

3 Disclaimer

Audits are a time, resource, and expertise bound effort where trained experts evaluate smart contracts using a combination of automated and manual techniques to identify as many vulnerabilities as possible. Audits can show the presence of vulnerabilities **but not their absence**.

4 Risk classification

Severity	Impact: High	Impact: Medium	Impact: Low
Likelihood: High	Critical	High	Medium
Likelihood: Medium	High	Medium	Low
Likelihood: Low	Medium	Low	Low

4.1 Impact

- **High** - leads to a significant loss of assets in the protocol or significantly harms a group of users.
- **Medium** - only a small amount of funds can be lost or a functionality of the protocol is affected.
- **Low** - any kind of unexpected behaviour that's not so critical.

4.2 Likelihood

- **High** - direct attack vector; the cost is relatively low to the amount of funds that can be lost.
- **Medium** - only conditionally incentivized attack vector, but still relatively likely.
- **Low** - too many or too unlikely assumptions; provides little or no incentive.

4.3 Actions required by severity level

- **Critical** - client **must** fix the issue.
- **High** - client **must** fix the issue.
- **Medium** - client **should** fix the issue.
- **Low** - client **could** fix the issue.

5 Executive summary

Overview

Project Name	HoneyFun
Repository	https://github.com/honey-fun/honey-fun-contracts
Commit Hash	bb156f84b1bb887a6e262daf7635f9fb7d4b03dd
Review Hash	7274b4fc1da2120578b9aa14b6efc60ec6eef227
Documentation	N/A
Methods	Manual review

Scope

HoneyFunAccessControl.sol
Agent.sol
Badge.sol
BondingCurve.sol
CollectionsRegistry.sol
Factory.sol
FundSplitter.sol
Router.sol
SystemContext.sol
Treasury.sol

Timeline

February 01, 2025	Audit kick-off
February 02, 2025	Preliminary report
February 05, 2025	Mitigation review

Issues Found

Severity	**Count**
High	0
Medium	1
Low	0
Information	4
Total	**5**

6 Findings

6.1 Medium

6.1.1 Users can launch their token after the 3-day period has passed

Severity: *Medium*

Context: Router.sol#L503

Description: The check for whether the DEX launch period has passed is performed after checking if the distributed shares equal the maximum shares. This creates a possibility for users to launch their token even if the 3-day period has already passed.

For example, over 2 days, these buys were made: • Alice bought 200 shares • Bob bought 100 shares • Tom bought 200 share

On the 3rd day, there was no activity—neither buying nor selling.

On the 4th day, Alice bought 500 shares, so all shares were sold, and the Meme token will be launched.

This is possible if the user buys all the remaining shares. If they buy fewer than the remaining shares, or if a sale is made after 3 day, the launch will fail, and the Meme token will not be created.

```
function _dexLaunchStatus(
    IBadge badge,
    uint256 deadline,
    uint256 shares
) internal view returns (DexLaunchStatus) {
    if (shares > badge.maxShares()) {
        return DexLaunchStatus.INVALID;
    } else if (shares == badge.maxShares()) {
        return
            liquidityProvidedFor(
                _collectionsRegistry().getGuidByBadge(badge)
            )
                ? DexLaunchStatus.COMPLETED
                : DexLaunchStatus.READY_TO_ADD_LIQUIDITY;
    } else if (block.timestamp > deadline) {
        return DexLaunchStatus.FAILED;
    }
     return DexLaunchStatus.PENDING;
    }
```

Recommendation: Move check `block.timestamp > deadline` before `shares == badge.maxShares()`.

Resolution and Client comment: Resolved. PR: `#19`

6.2 Information

6.2.1 Wrong event name

Severity: *Information*

Context: FundSplitter.sol#L108

Description: The status of the call when the collection owner receives their fee is no longer checked, so the boolean variable `status` from the `LiquiditySentToOwnerStatus` event has been removed.

```
         address(collectionOwner).call{value: memeCreatorFee}("");

         _transferNative(
            systemContext.getContractByName("TreasuryWallet"),
            address(this).balance
        );

        emit LiquiditySentToOwnerStatus(guid, collectionOwner, memeCreatorFee);
```

The name of the event is not accurate, as it contains `Status` in its name, which is no longer included.

Recommendation: Change the name of the event to `LiquiditySentToOwner`.

Resolution and Client comment: Resolved. PR: #19

6.2.2 Old comments are being used

Severity: *Information*

Context: BondingCurve.sol#L187-L201

Description: In the bonding curve contract, the constants `TOTAL_DOLLARS_TO_COLLECT` and `LAST_TOKEN_PRICE_IN_CENTS` have been changed. This means that now the first and last share prices will be different, so the old comments left in the contract are no longer accurate.

```
    function _lastTokenPrice(
        uint256 totalToCollect_
    ) internal pure returns (uint256) {
        // Last token price is 7.4x the first token price (740/100)
        return
            (_firstTokenPrice(totalToCollect_) * LAST_TOKEN_PRICE_IN_CENTS) /
            100;
    }

    function _firstTokenPrice(
        uint256 totalToCollect_
    ) internal pure returns (uint256) {
        // Calculate first token price for linear distribution
        // For 1 ETH total: 1000000000000000000 / 4200 = 238095238095238 wei
        return totalToCollect_ / TOTAL_DOLLARS_TO_COLLECT;
    }
```

Recommendation: Change the comments.

Resolution and Client comment: Resolved. PR: #19

6.2.3 Typo in event name

Severity: *Information*

Context: Treasury.sol

Description: In the event `CollectedNativeTranferred` has a typo.

Recommendation: Change the event name to `CollectedNativeTransferred`.

Resolution and Client comment: Resolved. PR: `#19`

6.2.4 Duplicated logic for transferring native tokens

Severity: *Information*

Context: Gloal

Description: The `_transferNative` function is used in both the `FundSplitter` and `Treasury` contracts. The logic behind this function is the same in both contracts. This is duplicated code, which can be moved to a library and the same logic to be reused.

Recommendation: Move the logic for transferring native tokens to a library so it can be easily reused from other contracts.

Resolution and Client comment: Resolved. PR: `#19`