Hidden Hand Audit Report

Introduction

This audit was requested by Redacted team and was conducted by kebabsec members sai, FlameHorizon and okkothejawa. As always, Redacted team delivers with easy to understand, clean and neat code.

Note: This report does not provide any guarantee or warranty of security for the project.

Scope

This audit includes the following contracts as in scope: 1. BribeBase.sol 2. BribeVault.sol 3. HummusBribe.sol

Branch: redacted-cartel/hidden-hand/main

Table of Contents

1. [INFO] BribeBase.sol#L320-L324 Dead code function `setRewardForwarding`

Description:

- This function allows voters to opt in or out of reward forwarding, setting `rewardForwarding[msg.caller]` to any address.
- However the mapping `rewardForwarding` is not used anywhere, which unnecessarily wastes gas.

Suggestion:

- Assuming no other contracts access the `rewardForwarding` mapping, removing redundant function may decrease the gas cost.

2. [GAS] BribeBase.sol#L243 - redundant token zero address check

Description:

- This check is not needed, as `require(isWhitelistedToken(token), "Token is not whitelisted");` already returns false on address(0).
    - Removing the check improves gas usage:
        - **Before change:** *depositBribeERC20 - 173720 gas*
        - **After change:** *depositBribeERC20 - 173685 gas*
    1. It's not possible to add zero address token to whitelist
    2. Removing a token occurs without check for address(0), as address 0 can't be added to list
    3. Thus it can be concluded that `isWhitelistedToken` can't return true for address(0)

To demonstrate that `isWhitelistedToken` already checks against `address(0)`, a function `removeWhitelistTokens` is used as an example:

```
    ├─ [52994] BribeBase::addWhitelistTokens([0x0000000000000000000000000000000000000001])
    │   ├─ emit AddWhitelistTokens(tokens: [0x0000000000000000000000000000000000000001])
    │   └─ ← ()
    ├─ [3918] BribeBase::removeWhitelistTokens([0x0000000000000000000000000000000000000000])
    │   └─ ← "Token not whitelisted"
    └─ ← "Token not whitelisted"
```

The trace shows `isWhitelistedToken` results in revert for address(0), therefore line 243: `require(token != address(0), "Invalid token");` is not necessary.

Suggestion: Remove the `require(token != address(0), "Invalid token")` check.

3. [INFO] BribeVault.sol#L9-L13 - Redundant Interface `IRewardDistributor`

Description: Interface is not inherited and can be removed.

Suggestion: Remove the interface.

4. [INFO] BribeVault.sol#L30 & L80-L81 - Unnecessary assignment of `_feeMax`

Description: We feel that is perhaps unnecessary to have `require(_feeMax < FEE_DIVISOR / 2, "Invalid _feeMax");` and to have a maximum `FEE_MAX` as

you can't really lower it or increase it after deployment, since there's no function to change that variable, and there is no incentive to make `FEE_MAX` lower than the limit that is set by the require check.

Suggestion: This is obviously irrelevant on how the code works, but if `FEE_MAX` is already bound by a constant divided by two, it seemed to us that it would make sense to also make it a constant, or just have a function to change `FEE_MAX` within those bounds.

5. [INFO] BribeVault.sol#L92 & BribeBase.sol #L47 - Deployer addresses have admin privileges

Description: In both `BribeVault` and `BribeBase` deployers are granted admin privileges, thus an EOA may get admin privileges, and this is not ideal.

Suggestion: For more ideal security posture, either pass a multi-sig address as a parameter to be set as `DEFAULT_ADMIN_ROLE` and/or implement a two-step ownership transfer mechanism to transfer the ownership to a multi-sig later.