HEDGEPIE V2

Smart Contract Security Audit

Prepared by ShellBoxes

July 4th, 2022 - August 13th, 2022

Shellboxes.com

contact@shellboxes.com

Document Properties

Client	HedgePie
Version	1.0
Classification	Public

Scope

The HEDGEPIE V2 Contract in the HEDGEPIE V2 Repository

Repo	Commit Hash
https://github.com/innovation-upstream/hedgepie-dev	73027fb4d1c98d6cc86bf47529ccf68ea4f32033
https://github.com/innovation-upstream/hedgepie-dev	b0e10dada92fa4a9e1bad6a14b17609c9f089e67

Files	MD5 Hash
HedgepieAdapterManager.sol	78ab3954d42eb0cebe51e80ce7e1ea3c
HedgepieInvestor.sol	c85cb74b75745f24579c862472c9594a
contracts/adapters/autofarm/autofarm-vault-adapter.sol	b5487ed8517c29e6f08d70c5c6186362
contracts/adapters/apeswap/apeswap-farm-adapter.sol	42d895214d25ac452666d65c78f49566

Re-Audit Files

Files	MD5 Hash
contracts/HedgepieAdapterManager.sol	8115071012f1c068e5fd809cfe814084
contracts/HedgepieInvestor.sol	a138552561acffa1b60b32e64ceb16e6
contracts/adapters/bnb/autofarm/auto-vault-adapter.sol	881384add400f965d79c4f37cc17048c
contracts/adapters/bnb/apeswap/apeswap-farm-lp-adapter.sol	44b6ad7dae43ffdc594a81c09534712d

Contacts

COMPANY	EMAIL
ShellBoxes	contact@shellboxes.com

Contents

1 Introduction

HedgePie engaged ShellBoxes to conduct a security assessment on the HEDGEPIE V2 beginning on July 4th, 2022 and ending August 13th, 2022. In this report, we detail our methodical approach to evaluate potential security issues associated with the implementation of smart contracts, by exposing possible semantic discrepancies between the smart contract code and design document, and by recommending additional ideas to optimize the existing code. Our findings indicate that the current version of smart contracts can still be enhanced further due to the presence of many security and performance concerns.

This document summarizes the findings of our audit.

1.1 About HedgePie

HedgePie is a hedge-funds based layer on top of decentralized finance that allows skilled investors and traders to design investment strategies that others can then invest into. Like the stock market, people can diversify their investment by buying into an index-style fund composed of various assets. In addition, subject matter experts can design investment strategies and publish them so others can buy into them, allowing non-experts to access well-derived strategies.

Issuer	HedgePie
Website	`https://hedgepie.finance/`
Type	Solidity Smart Contract
Audit Method	Whitebox

1.2 Approach & Methodology

ShellBoxes used a combination of manual and automated security testing to achieve a balance between efficiency, timeliness, practicability, and correctness within the audit's scope. While manual testing is advised for identifying problems in logic, procedure, and implementation, automated testing techniques help to expand the coverage of smart contracts and can quickly detect code that does not comply with security best practices.

1.2.1 Risk Methodology

Vulnerabilities or bugs identified by ShellBoxes are ranked using a risk assessment technique that considers both the LIKELIHOOD and IMPACT of a security incident. This framework is effective at conveying the features and consequences of technological vulnerabilities.

Its quantitative paradigm enables repeatable and precise measurement, while also revealing the underlying susceptibility characteristics that were used to calculate the Risk scores. A risk level will be assigned to each vulnerability on a scale of 5 to 1, with 5 indicating the greatest possibility or impact.

- Likelihood quantifies the probability of a certain vulnerability being discovered and exploited in the untamed.
- Impact quantifies the technical and economic costs of a successful attack.
- Severity indicates the risk's overall criticality.

Probability and impact are classified into three categories: H, M, and L, which correspond to high, medium, and low, respectively. Severity is determined by probability and impact and is categorized into four levels, namely Critical, High, Medium, and Low.

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

2 Findings Overview

2.1 Summary

The following is a synopsis of our conclusions from our analysis of the HEDGEPIE V2 implementation. During the first part of our audit, we examine the smart contract source code and run the codebase via a static code analyzer. The objective here is to find known coding problems statically and then manually check (reject or confirm) issues highlighted by the tool. Additionally, we check business logics, system processes, and DeFi-related components manually to identify potential hazards and/or defects.

2.2 Key Findings

In general, these smart contracts are well-designed and constructed, but their implementation might be improved by addressing the discovered flaws, which include , 1 medium-severity, 11 low-severity vulnerabilities.

Vulnerabilities	Severity	Status
Tax can be bypassed	MEDIUM	Acknowledged
Missing Value Verification	LOW	Fixed
Unnecessary arguments	LOW	Acknowledged
Avoid using .transfer() to transfer Ether	LOW	Fixed
Owner Can Renounce Ownership	LOW	Acknowledged
Floating Pragma	LOW	Acknowledged
Owner Can Renounce Ownership	LOW	Acknowledged
Floating Pragma	LOW	Acknowledged
Owner Can Renounce Ownership	LOW	Acknowledged
Floating Pragma	LOW	Acknowledged
Owner Can Renounce Ownership	LOW	Acknowledged
Floating Pragma	LOW	Acknowledged

3 Finding Details

A HedgepieInvestor.sol

A.1 Tax can be bypassed [MEDIUM]

Description:

At every call to withdrawBNB in the contract, a percentage is taken from the rewards and the withdrawal amount as a tax, then the contract sends these taxes to the treasury. In the case where the withdrawal amount and the rewards variable are lower than 10000/taxPercent, the taxAmount variable will be equal to 0 due to the type conversion, therefore bypassing the tax.

Code:

Listing 1: HedgepieInvestor.sol

```
358 if (
359     _vAmount >
360     IAdapter(adapter.addr).getWithdrawalAmount(
361         msg.sender,
362         _tokenId
363     )
364 ) {
365     taxAmount =
366         ((_vAmount -
367             IAdapter(adapter.addr).getWithdrawalAmount(
368                 _user,
369                 _tokenId
370             )) * taxPercent) /
371         1e4;

373     if (taxAmount != 0) {
374         IBEP20(adapter.token).transfer(
```

```
375             treasuryAddr,
376             taxAmount
377         );
378     }
379 }
```

Listing 2: HedgepieInvestor.sol

```
394 uint256 rewards = _getRewards(
395     _tokenId,
396     msg.sender,
397     adapter.addr
398 );

400 taxAmount = (rewards * taxPercent) / 1e4;

402 if (taxAmount != 0) {
403     IBEP20(IAdapter(adapter.addr).rewardToken()).transfer(
404         treasuryAddr,
405         taxAmount
406     );
407 }
```

Risk Level:

Likelihood – 3
Impact - 4

Recommendation:

Consider adding a require statement that verifies the withdrawal amount and the rewards variable cannot be lower than 10000/taxPercent.

Status - Acknowledged

The Hedgepie team has acknowledged the risk, stating that the tax amount is minor and can be ignored.

A.2 Missing Value Verification [LOW]

Description:

Certain functions lack a value safety check, the values of the arguments should be verified to allow only the ones that comply with the contract's logic. In the setTreasury function, the contract must ensure that _percent is less than 10000.

Code:

Listing 3: HedgepieInvestor.sol

```
114 function setTreasury(address _treasury, uint256 _percent)
115     external
116     onlyOwner
117 {
118     require(_treasury != address(0), "Invalid address");

120     treasuryAddr = _treasury;
121     taxPercent = _percent;
122 }
```

Risk Level:

Likelihood – 1
Impact - 3

Recommendation:

We recommend that you verify the values provided in the arguments. The issue can be addressed by utilizing a require statement.

Status - Fixed

The Hedgepie team has fixed the issue by verifying the _percent to be less than 1000.

A.3 Unnecessary arguments [LOW]

Description:

The deposit, depositBNB, withdraw and withdrawBNB functions have an argument called _user which is then verified by the modifier shouldMatchCaller to match msg.sender. This argument is unnecessary as msg.sender can be user directly to get the address of the caller. The same issue is present in the depositBNB function contains an argument called _amount that is verified to be equal to msg.value.

Code:

Listing 4: HedgepieInvestor.sol

```
189 function depositBNB(
190     address _user,
191     uint256 _tokenId,
192     uint256 _amount
193 ) external payable shouldMatchCaller(_user) nonReentrant {
194     require(_amount != 0, "Error: Amount can not be 0");
195     require(msg.value == _amount, "Error: Insufficient BNB");
196     require(
197         IYBNFT(ybnft).exists(_tokenId),
198         "Error: nft tokenId is invalid"
199     );

201     IYBNFT.Adapter[] memory adapterInfo = IYBNFT(ybnft).getAdapterInfo(
202         _tokenId
203     );
```

Risk Level:

Likelihood – 1
Impact - 2

Recommendation:

Consider using directly msg.sender and msg.value to get the address of the caller and the amount of native tokens passed to the function.

Status - Acknowledged

The Hedgepie team has acknowledged the issue, stating that they are planning to expand the investor contract to support cross-chain where the _user argument might differ from the msg.sender.

A.4 Avoid using .transfer() to transfer Ether [LOW]

Description:

Although transfer() and send() are recommended as a security best-practice to prevent reentrancy attacks because they only forward 2300 gas, the gas repricing of opcodes may break deployed contracts.

Code:

Listing 5: HedgepieInvestor.sol

```
240 uint256 afterBalance = address(this).balance;
241 if (afterBalance > beforeBalance)
242     payable(_user).transfer(afterBalance - beforeBalance);
```

Listing 6: HedgepieInvestor.sol

```
428 if (amountOut != 0) payable(_user).transfer(amountOut);
```

Risk Level:

Likelihood – 1
Impact - 2

Recommendation:

Consider using .call{ value: ... }("") instead, without hardcoded gas limits along with checks-effects-interactions pattern or reentrancy guards for reentrancy protection.

Status - Fixed

The Hedgepie team has fixed the issue by implementing the use of .call{ value: ... }("") instead of .transfer().

A.5 Owner Can Renounce Ownership [LOW]

Description:

Typically, the account that deploys the contract is also its owner. Consequently, the owner is able to engage in certain privileged activities in his own name. In smart contracts, the renounceOwnership function is used to renounce the ownership, therefore, the contract will never have an Owner, rendering some owner-exclusive functionality unavailable.

Code:

Listing 7: HedgepieInvestor.sol

```
114 contract HedgepieInvestor is Ownable, ReentrancyGuard {
```

Risk Level:

Likelihood – 1
Impact - 2

Recommendation:

We recommend that you prevent the owner from calling renounceOwnership. Additionally, if you decide to use a multi-signature wallet, then the execution of the renounceOwnership will require for at least two or more users to be confirmed. Alternatively, you can disable Renounce Ownership functionality by overriding it.

Status - Acknowledged

The Hedgepie team has acknowledged the risk, stating that they will transfer the ownership to a multi-sig wallet after deployment.

A.6 Floating Pragma [LOW]

Description:

The contract makes use of the floating-point pragma 0.8.4. Contracts should be deployed using the same compiler version.
Locking the pragma helps ensure that contracts are not unintentionally deployed using another pragma, such as an obsolete version, that may introduce issues in the contract system.

Code:

Listing 8: HedgepieInvestor.sol

```
1 // SPDX-License-Identifier: AGPL-3.0-or-later
2 pragma solidity ^0.8.4;
```

Risk Level:

Likelihood – 1
Impact - 2

Recommendation:

Consider locking the pragma version. It is advised that the floating pragma should not be used in production. Both truffle-config.js and hardhat.config.js support locking the pragma version.

Status - Acknowledged

The Hedgepie team has acknowledged the risk, stating that the pragma version is defined in the hardhat configuration.

B HedgepieAdapterManager.sol

B.1 Owner Can Renounce Ownership [LOW]

Description:

Typically, the account that deploys the contract is also its owner. Consequently, the owner is able to engage in certain privileged activities in his own name. In smart contracts, the renounceOwnership function is used to renounce the ownership, therefore, the contract will never have an Owner, rendering some owner-exclusive functionality unavailable.

Code:

Listing 9: HedgepieAdapterManager.sol

```
7 contract HedgepieAdapterManager is Ownable {
```

Risk Level:

Likelihood – 1
Impact - 2

Recommendation:

We recommend that you prevent the owner from calling renounceOwnership. Additionally, if you decide to use a multi-signature wallet, then the execution of the renounceOwnership will require for at least two or more users to be confirmed. Alternatively, you can disable Renounce Ownership functionality by overriding it.

Status - Acknowledged

The Hedgepie team has acknowledged the risk, stating that they will transfer the ownership to a multi-sig wallet after deployment.

B.2 Floating Pragma [LOW]

Description:

The contract makes use of the floating-point pragma 0.8.4. Contracts should be deployed using the same compiler version. Locking the pragma helps ensure that contracts are not unintentionally deployed using another pragma, such as an obsolete version, that may introduce issues in the contract system.

Code:

Listing 10: HedgepieAdapterManager.sol

```
1 // SPDX-License-Identifier: AGPL-3.0-or-later
2 pragma solidity ^0.8.4;
```

Risk Level:

Likelihood – 1
Impact - 2

Recommendation:

Consider locking the pragma version. It is advised that the floating pragma should not be used in production. Both truffle-config.js and hardhat.config.js support locking the pragma

version.

Status - Acknowledged

The Hedgepie team has acknowledged the risk, stating that the pragma version is defined in the hardhat configuration.

C autofarm-vault-adapter.sol

C.1 Owner Can Renounce Ownership [LOW]

Description:

Typically, the account that deploys the contract is also its owner. Consequently, the owner is able to engage in certain privileged activities in his own name. In smart contracts, the renounceOwnership function is used to renounce the ownership, therefore, the contract will never have an Owner, rendering some owner-exclusive functionality unavailable.

Code:

Listing 11: autofarm-vault-adapter.sol

```
18 contract AutoVaultAdapter is Ownable {
```

Risk Level:

Likelihood – 1
Impact - 2

Recommendation:

We recommend that you prevent the owner from calling renounceOwnership. Additionally, if you decide to use a multi-signature wallet, then the execution of the renounceOwnership will require for at least two or more users to be confirmed. Alternatively, you can disable Renounce Ownership functionality by overriding it.

Status - Acknowledged

The Hedgepie team has acknowledged the risk, stating that they will transfer the ownership to a multi-sig wallet after deployment.

C.2 Floating Pragma [LOW]

Description:

The contract makes use of the floating-point pragma 0.8.4. Contracts should be deployed using the same compiler version. Locking the pragma helps ensure that contracts are not unintentionally deployed using another pragma, such as an obsolete version, that may introduce issues in the contract system.

Code:

Listing 12: autofarm-vault-adapter.sol

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.4;
```

Risk Level:

Likelihood – 1
Impact - 2

Recommendation:

Consider locking the pragma version. It is advised that the floating pragma should not be used in production. Both truffle-config.js and hardhat.config.js support locking the pragma version.

Status - Acknowledged

The Hedgepie team has acknowledged the risk, stating that the pragma version is defined in the hardhat configuration.

D apeswap-farm-adapter.sol

D.1 Owner Can Renounce Ownership [LOW]

Description:

Typically, the account that deploys the contract is also its owner. Consequently, the owner is able to engage in certain privileged activities in his own name. In smart contracts, the renounceOwnership function is used to renounce the ownership, therefore, the contract will never have an Owner, rendering some owner-exclusive functionality unavailable.

Code:

Listing 13: apeswap-farm-adapter.sol

```
13 contract ApeswapFarmLPAdapter is Ownable {
```

Risk Level:

Likelihood – 1
Impact - 2

Recommendation:

We recommend that you prevent the owner from calling renounceOwnership. Additionally, if you decide to use a multi-signature wallet, then the execution of the renounceOwnership will require for at least two or more users to be confirmed. Alternatively, you can disable Renounce Ownership functionality by overriding it.

Status - Acknowledged

The Hedgepie team has acknowledged the risk, stating that they will transfer the ownership to a multi-sig wallet after deployment.

D.2 Floating Pragma [LOW]

Description:

The contract makes use of the floating-point pragma 0.8.4. Contracts should be deployed using the same compiler version. Locking the pragma helps ensure that contracts are not unintentionally deployed using another pragma, such as an obsolete version, that may introduce issues in the contract system.

Code:

Listing 14: apeswap-farm-adapter.sol

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.4;
```

Risk Level:

Likelihood – 1
Impact - 2

Recommendation:

Consider locking the pragma version. It is advised that the floating pragma should not be used in production. Both truffle-config.js and hardhat.config.js support locking the pragma version.

Status - Acknowledged

The Hedgepie team has acknowledged the risk, stating that the pragma version is defined in the hardhat configuration.

4 Tests

Results:

```
AlpacaAUSDPoolAdapter Integration Test
Owner: 0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
Investor: 0xB02B7F872ff9C316F4b734de0D5f2476b2C62682
Strategy: 0x158Da805682BdC8ee32d52833aD41E74bb951E59
AlpacaAUSDPoolAdapter: 0xc538D528a9eE0A8137C99a15d1DE873e902C1115
   depositBNB function test
      (1)should be reverted when nft tokenId is invalid (97ms)
      (2)should be reverted when amount is 0
      (3) deposit should success for Alice (4188ms)
      (4) deposit should success for Bob (416ms)
      (5) deposit should success for tom (412ms)
   withdrawBNB() function test
      (1)should be reverted when nft tokenId is invalid
      (2)should receive the BNB successfully after withdraw function for
         ↪  Alice (543ms)
      (3)should receive the BNB successfully after withdraw function for
         ↪  Bob (340ms)
      (4)should receive the BNB successfully after withdraw function for
         ↪  tom (312ms)


  AlpacaLendAdapter Integration Test
Owner: 0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
Investor: 0x1Da65b8874ed1290899A9Aa626A351296d4B274A
Strategy: 0xd7D069493685A581d27824Fc46EdA46B7EfC0063
AlpacaLendAdapter: 0xFb88e91a878fE2733561b0d81358D08b20Ac4328
   depositBNB function test
      (1)should be reverted when nft tokenId is invalid
      (2)should be reverted when amount is 0
      (3) deposit should success for Alice (2249ms)
      (4) deposit should success for Bob (277ms)
```

```
      (5) deposit should success for tom (278ms)
    withdrawBNB() function test
      (1)should be reverted when nft tokenId is invalid
      (2)should receive the BNB successfully after withdraw function for
         ↪  Alice (976ms)
      (3)should receive the BNB successfully after withdraw function for
         ↪  Bob (238ms)
      (4)should receive the BNB successfully after withdraw function for
         ↪  tom (221ms)

  AlpacaStakeAdapter Integration Test
Owner: 0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
Investor: 0xFD02c2291fb4F832831666Df5960A590d5e231cF
Strategy: 0xA625AB01B08ce023B2a342Dbb12a16f2C8489A8F
AlpacaStakeAdapter: 0x290d5b2F55866d2357cbf0a31724850091dF5dd5
    depositBNB function test
      (1)should be reverted when nft tokenId is invalid
      (2)should be reverted when amount is 0
      (3) deposit should success for Alice (6284ms)
      (4) deposit should success for Bob (1162ms)
      (5) deposit should success for tom (445ms)
    withdrawBNB() function test
      (1)should be reverted when nft tokenId is invalid
      (2)should receive the BNB successfully after withdraw function for
         ↪  Alice (639ms)
      (3)should receive the BNB successfully after withdraw function for
         ↪  Bob (408ms)
      (4)should receive the BNB successfully after withdraw function for
         ↪  tom (380ms)

  ApeswapBananaAdapter Integration Test
Owner: 0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
Investor: 0x670e44Ccd6a351b0403b3E889B263f874f4e1026
Strategy: 0x5c8D727b265DBAfaba67E050f2f739cAeEB4A6F9
```

```
ApeswapBananaAdapter: 0x66de75651060d9EC7218abCc7a2e4400525a1B6E
    deposit function test
      (1)should be reverted when nft tokenId is invalid
      (2)should be reverted when amount is 0
      (3) deposit should success for alice (4017ms)
      (4) deposit should success for bob (444ms)
      (5) deposit should success for tom (292ms)
    withdrawBNB() function test
      (1)should be reverted when nft tokenId is invalid
      (2)should receive the BNB successfully after withdraw function for
        ↪  alice (463ms)
      (3)should receive the BNB successfully after withdraw function for
        ↪  bob (284ms)
      (4)should receive the BNB successfully after withdraw function for
        ↪  tom (278ms)

  ApeswapFarmLPAdapter Integration Test
Owner: 0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
Investor: 0x02184Db0Ca92Ff24b9dF33AEb22bC1d978E3B032
Strategy: 0x5c8D727b265DBAfaba67E050f2f739cAeEB4A6F9
ApeswapFarmLPAdapter: 0xf96C190E181b38c840B7832BbA9E8D527250a5FB
    depositBNB function test
      (1)should be reverted when nft tokenId is invalid
      (2)should be reverted when amount is 0
      (3) deposit should success for Alice (3558ms)
      (4) deposit should success for Bob (413ms)
    withdrawBNB() function test
      (1) revert when nft tokenId is invalid
      (2) should receive the BNB successfully after withdraw function
        ↪ for Alice (1074ms)
      (3) should receive the BNB successfully after withdraw function
        ↪ for Bob (525ms)

  ApeswapJungleAdapter Integration Test
```

```
Owner: 0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
Investor: 0x9f3F78951bBf68fc3cBA976f1370a87B0Fc13cd4
Strategy: 0xc81af2222ac6ec0f3ec08b875df25326b40e7a76
ApeswapJungleAdapter: 0x4c07ce6454D5340591f62fD7d3978B6f42Ef953e
    deposit function test
      (1)should be reverted when nft tokenId is invalid
      (2)should be reverted when amount is 0
      (3) deposit should success for alice (4036ms)
      (4) deposit should success for bob (902ms)
      (5) deposit should success for tom (552ms)
    withdrawBNB() function test
      (1)should be reverted when nft tokenId is invalid
      (2)should receive the BNB successfully after withdraw function for
         ↪  alice (2857ms)
      (3)should receive the BNB successfully after withdraw function for
         ↪  bob (689ms)
      (4)should receive the BNB successfully after withdraw function for
         ↪  tom (730ms)

  ApeswapPoolAdapter Integration Test
Owner: 0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
Investor: 0x50f9CA3275E26cC1a89DaF0ce7f4942F3dDA9f61
Strategy: 0xd0378c1b37D530a00E91764A7a41EfEB3d6A5fbC
ApeswapPoolAdapter: 0x6CC14037395F7B84ba4eDaF239a6D97f0DcE3CEf
    depositBNB function test
      (1)should be reverted when nft tokenId is invalid
      (2)should be reverted when amount is 0
      (3) deposit should success for alice (2167ms)
      (4) deposit should success for bob (272ms)
      (5) deposit should success for tom (258ms)
    withdrawBNB() function test
      (1)should be reverted when nft tokenId is invalid
      (2)should receive BNB successfully after withdrawBNB function (373
         ↪ ms)
```

```
      (3)should receive BNB successfully after withdrawBNB function for
         ↪ bob (247ms)
      (4)should receive BNB successfully after withdrawBNB function for
         ↪ tom (232ms)

  ApeswapVaultAdapter Integration Test
Owner: 0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
Investor: 0x62153519C210d21f1B67dE11Cf60d6F467190707
Strategy: 0x5711a833C943AD1e8312A9c7E5403d48c717e1aa
ApeswapVaultAdapter: 0x9Bda88dA960e08Cc166D3e824109b5af3E376278
    deposit function test
      (1)should be reverted when nft tokenId is invalid
      (2)should be reverted when amount is 0
      (3) deposit should success for alice (5853ms)
      (4) deposit should success for bob (601ms)
      (5) deposit should success for tom (584ms)
    withdrawBNB() function test
      (1)should be reverted when nft tokenId is invalid
      (2)should receive the BNB successfully after withdraw function for
         ↪  alice (940ms)
      (3)should receive the BNB successfully after withdraw function for
         ↪  bob (574ms)
      (4)should receive the BNB successfully after withdraw function for
         ↪  tom (542ms)

  AutoVaultAdapter Integration Test
Owner: 0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
AutoFarm Adapter: 0xe2A04F5d91D1AD137f854C5820C76e5b711158c5
YBNFT: 0xD9d684546ED4c727B136503016E137822DD9b4D7
Investor: 0x79b3f34855c3478Fa45f65344a3f5E3c7b94405c
AdapterManager: 0x7CA40e352B402C9800b3fedBFC63b6FE79B8Fc0B
    depositBNB function test
      (1) should be reverted when nft tokenId is invalid (42ms)
      (2) should be reverted when amount is 0
```

```
      (3) deposit should success (6335ms)
   withdrawBNB() function test
      (1) should be reverted when nft tokenId is invalid
      (2) should receive the BNB successfully after withdraw function
         ↪ (1118ms)

  BeefyLPVaultAdapter Integration Test
Owner: 0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
Investor: 0x4f07450Ef721147D38f29739eEe8079bC147f1f6
Strategy: 0x164fb78cAf2730eFD63380c2a645c32eBa1C52bc
BeefyLPVaultAdapter: 0x0991d3831Ee86D349497039bB604FA1FB2aE0571
   depositBNB function test
      (1)should be reverted when nft tokenId is invalid
      (2)should be reverted when amount is 0
      (3) deposit should success for Alice (14644ms)
      (4) deposit should success for Bob (821ms)
      (5) deposit should success for tom (734ms)
   withdrawBNB() function test
      (1)should be reverted when nft tokenId is invalid
      (2)should receive the BNB successfully after withdraw function for
         ↪  Alice (2478ms)
      (3)should receive the BNB successfully after withdraw function for
         ↪  Bob (707ms)
      (4)should receive the BNB successfully after withdraw function for
         ↪  tom (691ms)

  BeefySingleVaultAdapter Integration Test
Owner: 0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
Investor: 0xC0Bd89573eDD265D3d9E0073f6D1e21e9Df5E1DA
Strategy: 0x725E14C3106EBf4778e01eA974e492f909029aE8
BeefySingleVaultAdapter: 0xa40E009b306B3b4f27374f6e833291DaAeC88cc6
   depositBNB function test
      (1)should be reverted when nft tokenId is invalid
      (2)should be reverted when amount is 0
```

```
      (3) deposit should success for Alice (12246ms)
      (4) deposit should success for Bob (920ms)
      (5) deposit should success for tom (913ms)
    withdrawBNB() function test
      (1)should be reverted when nft tokenId is invalid
      (2)should receive the BNB successfully after withdraw function for
         ↪ Alice (1973ms)
      (3)should receive the BNB successfully after withdraw function for
         ↪ Bob (1563ms)
      (4)should receive the BNB successfully after withdraw function for
         ↪ tom (1549ms)

  BeltVaultStakingAdapter Integration Test
Owner: 0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
Investor: 0x66908bE96DC195d937530b03f89f23EE54d5202d
Strategy: 0x9171Bf7c050aC8B4cf7835e51F7b4841DFB2cCD0
BeltVaultStakeAdapter: 0x409C36E5Cd41DC79b8B6E13B2371aB6dA506Cb01
    depositBNB function test
      (1)should be reverted when nft tokenId is invalid
      (2)should be reverted when amount is 0
      (3) deposit should success for Alice (22170ms)
      (4) deposit should success for Bob (1631ms)
      (5) deposit should success for tom (1747ms)
    withdrawBNB() function test
      (1)should be reverted when nft tokenId is invalid
      (2)should receive the BNB successfully after withdraw function for
         ↪ Alice (1882ms)
      (3)should receive the BNB successfully after withdraw function for
         ↪ Bob (778ms)
      (4)should receive the BNB successfully after withdraw function for
         ↪ tom (849ms)

  BiswapFarmLPAdapter Integration Test
Owner: 0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
```

```
Investor: 0x3D7CD28EfD08FfE9Ce8cA329EC2e67822C756526
Strategy: 0xDbc1A13490deeF9c3C12b44FE77b503c1B061739
BiswapFarmLPAdapter: 0x0c8Cd13ff68D41263E6937224B9e5c7fF54d72f9
    depositBNB function test
      (1) should be reverted when nft tokenId is invalid
      (2) should be reverted when amount is 0
      (3) deposit should success for Alice (2253ms)
      (4) deposit should success for Bob (280ms)
    withdrawBNB() function test
      (1) revert when nft tokenId is invalid (60ms)
      (2) should receive the BNB successfully after withdraw function
         ↪ for Alice (403ms)
      (3) should receive the BNB successfully after withdraw function
         ↪ for Bob (257ms)

  BiswapFarmLPAdapter Integration Test
Owner: 0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
Investor: 0x2639EDb06Fa9aF056AfF91eA0268Bf06Afc7564F
Strategy: 0xDbc1A13490deeF9c3C12b44FE77b503c1B061739
BiswapFarmLPAdapter: 0x340E02C02639522951264df540Fce1A66D2885E9
    depositBNB function test
      (1) should be reverted when nft tokenId is invalid
      (2) should be reverted when amount is 0
      (3) deposit should success for Alice (4271ms)
      (4) deposit should success for Bob (583ms)
    withdrawBNB() function test
      (1) revert when nft tokenId is invalid
      (2) should receive the BNB successfully after withdraw function
         ↪ for Alice (1171ms)
      (3) should receive the BNB successfully after withdraw function
         ↪ for Bob (786ms)

  PancakeSwapFarmLPAdapter Integration Test
Owner: 0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
```

```
Investor: 0xbf5d778c6E2040bDf3a0985ac2f386b624Ff5c7a
Strategy: 0xa5f8C5Dbd5F286960b9d90548680aE5ebFf07652
PancakeSwapFarmLPAdapter: 0x7914a8b73E11432953d9cCda060018EA1d9DCde9
    depositBNB function test
      (1) should be reverted when nft tokenId is invalid
      (2) should be reverted when amount is 0
      (3) deposit should success for Alice (1119ms)
      (4) deposit should success for Bob (431ms)
    withdrawBNB() function test
      (1) revert when nft tokenId is invalid
      (2) should receive the BNB successfully after withdraw function
        ↪ for Alice (716ms)
      (3) should receive the BNB successfully after withdraw function
        ↪ for Bob (513ms)

  Pancakeswap Stake Adapter Integration Test
YBNFT: 0x10FcD639d832188e94c50eD1C4C3A9F6403a0a64
Investor: 0x6706EB14Aa62f96F605A8492063c810C2a411e9d
PKSStakeAdapter: 0x1b53D58fFC6e69e6589a1A42eDf363584c0760f7
AdapterManager: 0x71498Ed008E4C968551e26eB4B803631241B5D26
Strategy: 0xa5D57C5dca083a7051797920c78fb2b19564176B
Owner: 0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
    should set correct state variable
      (1)Check strategy address
      (2)Check owner wallet
      (3)Check AdapterManager address in Investor contract
      (4)Check Investor address in AdapterManager contract
      (5)Check owner wallet
      (6)Check AdapterInfo of YBNFT
    depositBNB function test
      (1)should be reverted when nft tokenId is invalid
      (2)should be reverted when amount is 0
      (3)deposit should success for Alice (2371ms)
      (4)deposit should success for Bob (274ms)
```

```
      (5)deposit should success for Tom (261ms)
    withdrawBNB() function test
      (1)should be reverted when nft tokenId is invalid
      (2)should receive BNB successfully after withdraw function for
        ↪ Alice (399ms)
      (3)should receive BNB successfully after withdraw function for Bob
        ↪  (249ms)
      (4)should receive BNB successfully after withdraw function for Tom
        ↪  (244ms)

  VenusLendAdapter Integration Test
YBNFT: 0xf9fe9360A5849437Dda072652c4dA0f7ac73f8E3
Investor: 0x641c1a78EBAe28B8AC165f7F9B0E41f4DD3C4f5B
VenusAdapter: 0xE35C265ECE9fdda7c99708dEc45E67Ddb7804193
AdapterManager: 0x9E1Aa060Ad2Af934a9aD876705E320063Dae1492
Strategy: 0x95c78222B3D6e262426483D42CfA53685A67Ab9D
Owner: 0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
    should set correct state variable
      (1) Check strategy address
      (2) Check owner wallet
      (3) Check AdapterManager address in Investor contract
      (4) Check Investor address in AdapterManager contract
      (5) Check owner wallet
      (6) Check AdapterInfo of YBNFT
    deposit() function test
      (1)should be reverted when nft tokenId is invalid
      (2)should be reverted when amount is 0
      (3)should success 1 time and receive the vToken successfully after
        ↪  deposit function (3575ms)
      (4)should success multiple times (676ms)
    withdraw() function test
      (1)should be reverted when nft tokenId is invalid
      (2)should be reverted when amount is 0
      (3)should receive the WBNB successfully after withdraw function
```

```
        ↪ (957ms)

  VenusLongLevAdapter Integration Test
Owner: 0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
Investor: 0x2C007C7c20f06036DfBe8D7A05c24775a8370199
Strategy: 0x95c78222B3D6e262426483D42CfA53685A67Ab9D
VenusLongLevAdapter: 0x621420020d4C2345b72a47A5324c74462A440263
    depositBNB function test
      (1) should be reverted when nft tokenId is invalid
      (2) should be reverted when amount is 0
      (3) deposit should success for Alice (4887ms)
      (4) deposit should success for Bob (1425ms)
    withdrawBNB() function test
      (1) revert when nft tokenId is invalid
      (2) should receive the BNB successfully after withdraw function
        ↪ for Alice (1558ms)
      (3) should receive the BNB successfully after withdraw function
        ↪ for Bob (1386ms)

  VenusShortLevAdapter Integration Test
Owner: 0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
Investor: 0xfFD89CD4cA0e083f25E1bFFc2fb480e251Cc2208
Strategy: 0x95c78222B3D6e262426483D42CfA53685A67Ab9D
VenusShortLevAdapter: 0x2F7e3763CAb88d161eA199d591db36aBA7536474
    depositBNB function test
      (1) should be reverted when nft tokenId is invalid
      (2) should be reverted when amount is 0
      (3) deposit should success for Alice (2221ms)
      (4) deposit should success for Bob (709ms)
    withdrawBNB() function test
      (1) revert when nft tokenId is invalid
      (2) should receive the BNB successfully after withdraw function
        ↪ for Alice (916ms)
      (3) should receive the BNB successfully after withdraw function
```

```
        ↪ for Bob (764ms)


    165 passing (3m)
```

Conclusion:

The tests are passed successfully, and they cover the majority of tests cases, which guarantees the functionality of the contracts

5 Static Analysis (Slither)

Description:

ShellBoxes expanded the coverage of the specific contract areas using automated testing methodologies. Slither, a Solidity static analysis framework, was one of the tools used. Slither was run on all-scoped contracts in both text and binary formats. This tool can be used to test mathematical relationships between Solidity instances statically and variables that allow for the detection of errors or inconsistent usage of the contracts' APIs throughout the entire codebase.

Results:

```
IAdapter.strategy().strategy (interfaces/IAdapter.sol#20) shadows:
        - IAdapter.strategy() (interfaces/IAdapter.sol#20) (function)
IAdapter.vStrategy().vStrategy (interfaces/IAdapter.sol#22) shadows:
        - IAdapter.vStrategy() (interfaces/IAdapter.sol#22) (function)
IAdapter.getAdapterStrategy(uint256).strategy (interfaces/IAdapter.sol
    ↪ #41) shadows:
        - IAdapter.strategy() (interfaces/IAdapter.sol#20) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #local-variable-shadowing

HedgepieAdapterManager.setInvestor(address) (HedgepieAdapterManager.sol
    ↪ #144-147) should emit an event for:
        - investor = _investor (HedgepieAdapterManager.sol#146)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #missing-events-access-control

Ownable.constructor().msgSender (libraries/Ownable.sol#30) lacks a zero-
    ↪ check on :
                - _owner = msgSender (libraries/Ownable.sol#31)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #missing-zero-address-validation
```

```
Context._msgData() (libraries/Context.sol#23-26) is never used and
    ↪ should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #dead-code

Pragma version^0.8.4 (HedgepieAdapterManager.sol#2) necessitates a
    ↪ version too recent to be trusted. Consider deploying with
    ↪ 0.6.12/0.7.6
Pragma version^0.8.4 (interfaces/IAdapter.sol#2) necessitates a version
    ↪ too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.4 (libraries/Context.sol#2) necessitates a version
    ↪ too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.4 (libraries/Ownable.sol#2) necessitates a version
    ↪ too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.6 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #incorrect-versions-of-solidity

Parameter HedgepieAdapterManager.getAdapterStrat(address)._adapter (
    ↪ HedgepieAdapterManager.sol#60) is not in mixedCase
Parameter HedgepieAdapterManager.getDepositCallData(address,uint256).
    ↪ _adapter (HedgepieAdapterManager.sol#74) is not in mixedCase
Parameter HedgepieAdapterManager.getDepositCallData(address,uint256).
    ↪ _amount (HedgepieAdapterManager.sol#74) is not in mixedCase
Parameter HedgepieAdapterManager.getWithdrawCallData(address,uint256).
    ↪ _adapter (HedgepieAdapterManager.sol#94) is not in mixedCase
Parameter HedgepieAdapterManager.getWithdrawCallData(address,uint256).
    ↪ _amount (HedgepieAdapterManager.sol#94) is not in mixedCase
Parameter HedgepieAdapterManager.addAdapter(address)._adapter (
    ↪ HedgepieAdapterManager.sol#114) is not in mixedCase
Parameter HedgepieAdapterManager.setAdapter(uint256,bool)._adapterId (
    ↪ HedgepieAdapterManager.sol#134) is not in mixedCase
```

```
Parameter HedgepieAdapterManager.setAdapter(uint256,bool)._status (
    ↪ HedgepieAdapterManager.sol#134) is not in mixedCase
Parameter HedgepieAdapterManager.setInvestor(address)._investor (
    ↪ HedgepieAdapterManager.sol#144) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #conformance-to-solidity-naming-conventions

Redundant expression "this (libraries/Context.sol#24)" inContext (
    ↪ libraries/Context.sol#14-27)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #redundant-statements

owner() should be declared external:
        - Ownable.owner() (libraries/Ownable.sol#38-40)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (libraries/Ownable.sol#57-60)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (libraries/Ownable.sol
            ↪ #66-68)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #public-function-that-could-be-declared-external

HedgepieMasterChef.pendingReward(uint256,address) (HedgepieMasterChef.
    ↪ sol#118-141) performs a multiplication on the result of a
    ↪ division:
        -hpieReward = multiplier.mul(rewardPerBlock).mul(pool.allocPoint)
            ↪ .div(totalAllocPoint) (HedgepieMasterChef.sol#132-135)
        -accHpiePerShare = accHpiePerShare.add(hpieReward.mul(1e12).div(
            ↪ lpSupply)) (HedgepieMasterChef.sol#136-138)
HedgepieMasterChef.updatePool(uint256) (HedgepieMasterChef.sol#198-217)
    ↪ performs a multiplication on the result of a division:
        -hpieReward = multiplier.mul(rewardPerBlock).mul(pool.allocPoint)
            ↪ .div(totalAllocPoint) (HedgepieMasterChef.sol#209-212)
```

```
        -pool.accHpiePerShare = pool.accHpiePerShare.add(hpieReward.mul(1
            ↪ e12).div(lpSupply)) (HedgepieMasterChef.sol#213-215)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #divide-before-multiply

Reentrancy in HedgepieMasterChef.deposit(uint256,uint256) (
    ↪ HedgepieMasterChef.sol#234-265):
        External calls:
        - rewardToken.safeTransferFrom(rewardHolder,address(msg.sender),
            ↪ pending) (HedgepieMasterChef.sol#246-250)
        - pool.lpToken.safeTransferFrom(address(msg.sender),address(this)
            ↪ ,_amount) (HedgepieMasterChef.sol#254-258)
        State variables written after the call(s):
        - pool.totalShares += _amount (HedgepieMasterChef.sol#259)
        - user.amount = user.amount.add(_amount) (HedgepieMasterChef.sol
            ↪ #260)
        - user.rewardDebt = user.amount.mul(pool.accHpiePerShare).div(1
            ↪ e12) (HedgepieMasterChef.sol#262)
Reentrancy in HedgepieMasterChef.emergencyWithdraw(uint256) (
    ↪ HedgepieMasterChef.sol#304-314):
        External calls:
        - pool.lpToken.safeTransfer(address(msg.sender),user.amount) (
            ↪ HedgepieMasterChef.sol#308)
        State variables written after the call(s):
        - pool.totalShares -= user.amount (HedgepieMasterChef.sol#309)
        - user.amount = 0 (HedgepieMasterChef.sol#312)
        - user.rewardDebt = 0 (HedgepieMasterChef.sol#313)
Reentrancy in HedgepieMasterChef.withdraw(uint256,uint256) (
    ↪ HedgepieMasterChef.sol#272-298):
        External calls:
        - rewardToken.safeTransferFrom(rewardHolder,address(msg.sender),
            ↪ pending) (HedgepieMasterChef.sol#284-288)
        State variables written after the call(s):
```

```
        - user.amount = user.amount.sub(_amount) (HedgepieMasterChef.sol
            ↪ #291)
Reentrancy in HedgepieMasterChef.withdraw(uint256,uint256) (
    ↪ HedgepieMasterChef.sol#272-298):
        External calls:
        - rewardToken.safeTransferFrom(rewardHolder,address(msg.sender),
            ↪ pending) (HedgepieMasterChef.sol#284-288)
        - pool.lpToken.safeTransfer(address(msg.sender),_amount) (
            ↪ HedgepieMasterChef.sol#292)
        State variables written after the call(s):
        - pool.totalShares -= _amount (HedgepieMasterChef.sol#293)
        - user.rewardDebt = user.amount.mul(pool.accHpiePerShare).div(1
            ↪ e12) (HedgepieMasterChef.sol#295)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #reentrancy-vulnerabilities-1

HedgepieMasterChef.add(uint256,IBEP20) (HedgepieMasterChef.sol#149-167)
    ↪ should emit an event for:
        - totalAllocPoint = totalAllocPoint.add(_allocPoint) (
            ↪ HedgepieMasterChef.sol#157)
HedgepieMasterChef.set(uint256,uint256) (HedgepieMasterChef.sol#174-183)
    ↪  should emit an event for:
        - totalAllocPoint = totalAllocPoint.sub(prevAllocPoint).add(
            ↪ _allocPoint) (HedgepieMasterChef.sol#179-181)
HedgepieMasterChef.updateMultiplier(uint256) (HedgepieMasterChef.sol
    ↪ #189-192) should emit an event for:
        - BONUS_MULTIPLIER = _multiplierNumber (HedgepieMasterChef.sol
            ↪ #191)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #missing-events-arithmetic

Ownable.constructor().msgSender (libraries/Ownable.sol#30) lacks a zero-
    ↪ check on :
                - _owner = msgSender (libraries/Ownable.sol#31)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #missing-zero-address-validation

Reentrancy in HedgepieMasterChef.deposit(uint256,uint256) (
    ↪ HedgepieMasterChef.sol#234-265):
        External calls:
        - rewardToken.safeTransferFrom(rewardHolder,address(msg.sender),
            ↪ pending) (HedgepieMasterChef.sol#246-250)
        - pool.lpToken.safeTransferFrom(address(msg.sender),address(this)
            ↪ ,_amount) (HedgepieMasterChef.sol#254-258)
        Event emitted after the call(s):
        - Deposit(msg.sender,_pid,_amount) (HedgepieMasterChef.sol#264)
Reentrancy in HedgepieMasterChef.emergencyWithdraw(uint256) (
    ↪ HedgepieMasterChef.sol#304-314):
        External calls:
        - pool.lpToken.safeTransfer(address(msg.sender),user.amount) (
            ↪ HedgepieMasterChef.sol#308)
        Event emitted after the call(s):
        - EmergencyWithdraw(msg.sender,_pid,user.amount) (
            ↪ HedgepieMasterChef.sol#310)
Reentrancy in HedgepieMasterChef.withdraw(uint256,uint256) (
    ↪ HedgepieMasterChef.sol#272-298):
        External calls:
        - rewardToken.safeTransferFrom(rewardHolder,address(msg.sender),
            ↪ pending) (HedgepieMasterChef.sol#284-288)
        - pool.lpToken.safeTransfer(address(msg.sender),_amount) (
            ↪ HedgepieMasterChef.sol#292)
        Event emitted after the call(s):
        - Withdraw(msg.sender,_pid,_amount) (HedgepieMasterChef.sol#297)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #reentrancy-vulnerabilities-3

Address.isContract(address) (libraries/Address.sol#25-36) uses assembly
        - INLINE ASM (libraries/Address.sol#32-34)
```

```
Address._functionCallWithValue(address,bytes,uint256,string) (libraries/
    ↪ Address.sol#151-179) uses assembly
        - INLINE ASM (libraries/Address.sol#171-174)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #assembly-usage

Address.functionCall(address,bytes) (libraries/Address.sol#86-91) is
    ↪ never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (libraries/Address.
    ↪ sol#118-130) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (libraries/
    ↪ Address.sol#138-149) is never used and should be removed
Address.sendValue(address,uint256) (libraries/Address.sol#54-66) is
    ↪ never used and should be removed
Context._msgData() (libraries/Context.sol#23-26) is never used and
    ↪ should be removed
SafeBEP20.safeApprove(IBEP20,address,uint256) (libraries/SafeBEP20.sol
    ↪ #42-59) is never used and should be removed
SafeBEP20.safeDecreaseAllowance(IBEP20,address,uint256) (libraries/
    ↪ SafeBEP20.sol#79-96) is never used and should be removed
SafeBEP20.safeIncreaseAllowance(IBEP20,address,uint256) (libraries/
    ↪ SafeBEP20.sol#61-77) is never used and should be removed
SafeMath.mod(uint256,uint256) (libraries/SafeMath.sol#55-57) is never
    ↪ used and should be removed
SafeMath.mod(uint256,uint256,string) (libraries/SafeMath.sol#59-66) is
    ↪ never used and should be removed
SafeMath.sqrrt(uint256) (libraries/SafeMath.sol#69-80) is never used and
    ↪  should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #dead-code

Pragma version^0.8.4 (HedgepieMasterChef.sol#2) necessitates a version
    ↪ too recent to be trusted. Consider deploying with 0.6.12/0.7.6
```

```
Pragma version^0.8.4 (interfaces/IBEP20.sol#2) necessitates a version
   ↪ too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.4 (libraries/Address.sol#2) necessitates a version
   ↪ too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.4 (libraries/Context.sol#2) necessitates a version
   ↪ too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.4 (libraries/Ownable.sol#2) necessitates a version
   ↪ too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.4 (libraries/SafeBEP20.sol#2) necessitates a version
   ↪ too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.4 (libraries/SafeMath.sol#2) necessitates a version
   ↪ too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.6 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
   ↪ #incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (libraries/Address.
   ↪ sol#54-66):
       - (success) = recipient.call{value: amount}() (libraries/Address.
          ↪ sol#61)
Low level call in Address._functionCallWithValue(address,bytes,uint256,
   ↪ string) (libraries/Address.sol#151-179):
       - (success,returndata) = target.call{value: weiValue}(data) (
          ↪ libraries/Address.sol#160-162)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
   ↪ #low-level-calls

Parameter HedgepieMasterChef.getMultiplier(uint256,uint256)._from (
   ↪ HedgepieMasterChef.sol#105) is not in mixedCase
Parameter HedgepieMasterChef.getMultiplier(uint256,uint256)._to (
   ↪ HedgepieMasterChef.sol#105) is not in mixedCase
Parameter HedgepieMasterChef.pendingReward(uint256,address)._pid (
   ↪ HedgepieMasterChef.sol#118) is not in mixedCase
```

```
Parameter HedgepieMasterChef.pendingReward(uint256,address)._user (
    ↪ HedgepieMasterChef.sol#118) is not in mixedCase
Parameter HedgepieMasterChef.add(uint256,IBEP20)._allocPoint (
    ↪ HedgepieMasterChef.sol#149) is not in mixedCase
Parameter HedgepieMasterChef.add(uint256,IBEP20)._lpToken (
    ↪ HedgepieMasterChef.sol#149) is not in mixedCase
Parameter HedgepieMasterChef.set(uint256,uint256)._pid (
    ↪ HedgepieMasterChef.sol#174) is not in mixedCase
Parameter HedgepieMasterChef.set(uint256,uint256)._allocPoint (
    ↪ HedgepieMasterChef.sol#174) is not in mixedCase
Parameter HedgepieMasterChef.updateMultiplier(uint256)._multiplierNumber
    ↪  (HedgepieMasterChef.sol#189) is not in mixedCase
Parameter HedgepieMasterChef.updatePool(uint256)._pid (
    ↪ HedgepieMasterChef.sol#198) is not in mixedCase
Parameter HedgepieMasterChef.deposit(uint256,uint256)._pid (
    ↪ HedgepieMasterChef.sol#234) is not in mixedCase
Parameter HedgepieMasterChef.deposit(uint256,uint256)._amount (
    ↪ HedgepieMasterChef.sol#234) is not in mixedCase
Parameter HedgepieMasterChef.withdraw(uint256,uint256)._pid (
    ↪ HedgepieMasterChef.sol#272) is not in mixedCase
Parameter HedgepieMasterChef.withdraw(uint256,uint256)._amount (
    ↪ HedgepieMasterChef.sol#272) is not in mixedCase
Parameter HedgepieMasterChef.emergencyWithdraw(uint256)._pid (
    ↪ HedgepieMasterChef.sol#304) is not in mixedCase
Variable HedgepieMasterChef.BONUS_MULTIPLIER (HedgepieMasterChef.sol#39)
    ↪  is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #conformance-to-solidity-naming-conventions

Redundant expression "this (libraries/Context.sol#24)" inContext (
    ↪ libraries/Context.sol#14-27)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #redundant-statements
```

```
add(uint256,IBEP20) should be declared external:
        - HedgepieMasterChef.add(uint256,IBEP20) (HedgepieMasterChef.sol
          ↪ #149-167)
set(uint256,uint256) should be declared external:
        - HedgepieMasterChef.set(uint256,uint256) (HedgepieMasterChef.sol
          ↪ #174-183)
updateMultiplier(uint256) should be declared external:
        - HedgepieMasterChef.updateMultiplier(uint256) (
          ↪ HedgepieMasterChef.sol#189-192)
deposit(uint256,uint256) should be declared external:
        - HedgepieMasterChef.deposit(uint256,uint256) (HedgepieMasterChef
          ↪ .sol#234-265)
withdraw(uint256,uint256) should be declared external:
        - HedgepieMasterChef.withdraw(uint256,uint256) (
          ↪ HedgepieMasterChef.sol#272-298)
emergencyWithdraw(uint256) should be declared external:
        - HedgepieMasterChef.emergencyWithdraw(uint256) (
          ↪ HedgepieMasterChef.sol#304-314)
owner() should be declared external:
        - Ownable.owner() (libraries/Ownable.sol#38-40)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (libraries/Ownable.sol#57-60)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (libraries/Ownable.sol
          ↪ #66-68)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #public-function-that-could-be-declared-external

ApeswapFarmLPAdapter.setInvestor(address) (apeswap-farm-lp-adapter.sol
    ↪ #147-150) should emit an event for:
        - investor = _investor (apeswap-farm-lp-adapter.sol#149)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #missing-events-access-control
```

```
ApeswapFarmLPAdapter.constructor(uint256,address,address,address,address
    ↪ ,string)._stakingToken (apeswap-farm-lp-adapter.sol#45) lacks a
    ↪ zero-check on :
                - stakingToken = _stakingToken (apeswap-farm-lp-adapter.
                    ↪ sol#51)
ApeswapFarmLPAdapter.constructor(uint256,address,address,address,address
    ↪ ,string)._rewardToken (apeswap-farm-lp-adapter.sol#46) lacks a
    ↪ zero-check on :
                - rewardToken = _rewardToken (apeswap-farm-lp-adapter.sol
                    ↪ #52)
ApeswapFarmLPAdapter.constructor(uint256,address,address,address,address
    ↪ ,string)._strategy (apeswap-farm-lp-adapter.sol#44) lacks a zero-
    ↪ check on :
                - strategy = _strategy (apeswap-farm-lp-adapter.sol#53)
ApeswapFarmLPAdapter.constructor(uint256,address,address,address,address
    ↪ ,string)._router (apeswap-farm-lp-adapter.sol#47) lacks a zero-
    ↪ check on :
                - router = _router (apeswap-farm-lp-adapter.sol#54)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #missing-zero-address-validation

Different versions of Solidity is used:
        - Version used: ['^0.8.0', '^0.8.4']
        - ^0.8.0 (../../../../../openzeppelin-contracts/contracts/access/
            ↪ Ownable.sol#3)
        - ^0.8.0 (../../../../../openzeppelin-contracts/contracts/utils/
            ↪ Context.sol#3)
        - ^0.8.4 (apeswap-farm-lp-adapter.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #different-pragma-directives-are-used

Context._msgData() (../../../../../openzeppelin-contracts/contracts/
    ↪ utils/Context.sol#20-22) is never used and should be removed
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #dead-code

Pragma version^0.8.0 (../../../../../openzeppelin-contracts/contracts/
    ↪ access/Ownable.sol#3) necessitates a version too recent to be
    ↪ trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.0 (../../../../../openzeppelin-contracts/contracts/
    ↪ utils/Context.sol#3) necessitates a version too recent to be
    ↪ trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.4 (apeswap-farm-lp-adapter.sol#2) necessitates a
    ↪ version too recent to be trusted. Consider deploying with
    ↪ 0.6.12/0.7.6
solc-0.8.6 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #incorrect-versions-of-solidity

Parameter ApeswapFarmLPAdapter.getWithdrawalAmount(address,uint256).
    ↪ _user (apeswap-farm-lp-adapter.sol#63) is not in mixedCase
Parameter ApeswapFarmLPAdapter.getWithdrawalAmount(address,uint256).
    ↪ _nftId (apeswap-farm-lp-adapter.sol#63) is not in mixedCase
Parameter ApeswapFarmLPAdapter.getInvestCallData(uint256)._amount (
    ↪ apeswap-farm-lp-adapter.sol#75) is not in mixedCase
Parameter ApeswapFarmLPAdapter.getDevestCallData(uint256)._amount (
    ↪ apeswap-farm-lp-adapter.sol#97) is not in mixedCase
Parameter ApeswapFarmLPAdapter.increaseWithdrawalAmount(address,uint256,
    ↪ uint256)._user (apeswap-farm-lp-adapter.sol#122) is not in
    ↪ mixedCase
Parameter ApeswapFarmLPAdapter.increaseWithdrawalAmount(address,uint256,
    ↪ uint256)._nftId (apeswap-farm-lp-adapter.sol#123) is not in
    ↪ mixedCase
Parameter ApeswapFarmLPAdapter.increaseWithdrawalAmount(address,uint256,
    ↪ uint256)._amount (apeswap-farm-lp-adapter.sol#124) is not in
    ↪ mixedCase
```

```
Parameter ApeswapFarmLPAdapter.setWithdrawalAmount(address,uint256,
    ↪ uint256)._user (apeswap-farm-lp-adapter.sol#136) is not in
    ↪ mixedCase
Parameter ApeswapFarmLPAdapter.setWithdrawalAmount(address,uint256,
    ↪ uint256)._nftId (apeswap-farm-lp-adapter.sol#137) is not in
    ↪ mixedCase
Parameter ApeswapFarmLPAdapter.setWithdrawalAmount(address,uint256,
    ↪ uint256)._amount (apeswap-farm-lp-adapter.sol#138) is not in
    ↪ mixedCase
Parameter ApeswapFarmLPAdapter.setInvestor(address)._investor (apeswap-
    ↪ farm-lp-adapter.sol#147) is not in mixedCase
Parameter ApeswapFarmLPAdapter.getPaths(address,address)._inToken (
    ↪ apeswap-farm-lp-adapter.sol#157) is not in mixedCase
Parameter ApeswapFarmLPAdapter.getPaths(address,address)._outToken (
    ↪ apeswap-farm-lp-adapter.sol#157) is not in mixedCase
Parameter ApeswapFarmLPAdapter.setPath(address,address,address[]).
    ↪ _inToken (apeswap-farm-lp-adapter.sol#187) is not in mixedCase
Parameter ApeswapFarmLPAdapter.setPath(address,address,address[]).
    ↪ _outToken (apeswap-farm-lp-adapter.sol#188) is not in mixedCase
Parameter ApeswapFarmLPAdapter.setPath(address,address,address[])._paths
    ↪  (apeswap-farm-lp-adapter.sol#189) is not in mixedCase
Parameter ApeswapFarmLPAdapter.getReward(address)._user (apeswap-farm-lp
    ↪ -adapter.sol#220) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #conformance-to-solidity-naming-conventions

ApeswapFarmLPAdapter.repayToken (apeswap-farm-lp-adapter.sol#17) should
    ↪ be constant
ApeswapFarmLPAdapter.vStrategy (apeswap-farm-lp-adapter.sol#19) should
    ↪ be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #state-variables-that-could-be-declared-constant

renounceOwnership() should be declared external:
```

```
        - Ownable.renounceOwnership() (../../../../../openzeppelin-
            ↪ contracts/contracts/access/Ownable.sol#53-55)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (../../../../../openzeppelin
            ↪ -contracts/contracts/access/Ownable.sol#61-64)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #public-function-that-could-be-declared-external

AutoVaultAdapter.setInvestor(address) (auto-vault-adapter.sol#168-171)
    ↪ should emit an event for:
        - investor = _investor (auto-vault-adapter.sol#170)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #missing-events-access-control

AutoVaultAdapter.setPoolID(uint256) (auto-vault-adapter.sol#177-179)
    ↪ should emit an event for:
        - poolID = _poolID (auto-vault-adapter.sol#178)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #missing-events-arithmetic

AutoVaultAdapter.constructor(address,address,address,address,address,
    ↪ string)._stakingToken (auto-vault-adapter.sol#52) lacks a zero-
    ↪ check on :
                - stakingToken = _stakingToken (auto-vault-adapter.sol#57)
AutoVaultAdapter.constructor(address,address,address,address,address,
    ↪ string)._rewardToken (auto-vault-adapter.sol#53) lacks a zero-
    ↪ check on :
                - rewardToken = _rewardToken (auto-vault-adapter.sol#58)
AutoVaultAdapter.constructor(address,address,address,address,address,
    ↪ string)._strategy (auto-vault-adapter.sol#50) lacks a zero-check
    ↪ on :
                - strategy = _strategy (auto-vault-adapter.sol#59)
AutoVaultAdapter.constructor(address,address,address,address,address,
    ↪ string)._vStrategy (auto-vault-adapter.sol#51) lacks a zero-check
```

```
    ↪  on :
                - vStrategy = _vStrategy (auto-vault-adapter.sol#60)
AutoVaultAdapter.constructor(address,address,address,address,address,
    ↪ string)._router (auto-vault-adapter.sol#54) lacks a zero-check on
    ↪  :
                - router = _router (auto-vault-adapter.sol#62)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #missing-zero-address-validation

Different versions of Solidity is used:
        - Version used: ['^0.8.0', '^0.8.4']
        - ^0.8.0 (../../../../../openzeppelin-contracts/contracts/access/
            ↪ Ownable.sol#3)
        - ^0.8.0 (../../../../../openzeppelin-contracts/contracts/utils/
            ↪ Context.sol#3)
        - ^0.8.4 (auto-vault-adapter.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #different-pragma-directives-are-used

Context._msgData() (../../../../../openzeppelin-contracts/contracts/
    ↪ utils/Context.sol#20-22) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #dead-code

Pragma version^0.8.0 (../../../../../openzeppelin-contracts/contracts/
    ↪ access/Ownable.sol#3) necessitates a version too recent to be
    ↪ trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.0 (../../../../../openzeppelin-contracts/contracts/
    ↪ utils/Context.sol#3) necessitates a version too recent to be
    ↪ trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.4 (auto-vault-adapter.sol#2) necessitates a version
    ↪ too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.6 is not recommended for deployment
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #incorrect-versions-of-solidity

Parameter AutoVaultAdapter.getWithdrawalAmount(address,uint256)._user (
    ↪ auto-vault-adapter.sol#70) is not in mixedCase
Parameter AutoVaultAdapter.getWithdrawalAmount(address,uint256)._nftId (
    ↪ auto-vault-adapter.sol#70) is not in mixedCase
Parameter AutoVaultAdapter.getInvestCallData(uint256)._amount (auto-
    ↪ vault-adapter.sol#82) is not in mixedCase
Parameter AutoVaultAdapter.getDevestCallData(uint256)._amount (auto-
    ↪ vault-adapter.sol#104) is not in mixedCase
Parameter AutoVaultAdapter.increaseWithdrawalAmount(address,uint256,
    ↪ uint256)._user (auto-vault-adapter.sol#143) is not in mixedCase
Parameter AutoVaultAdapter.increaseWithdrawalAmount(address,uint256,
    ↪ uint256)._nftId (auto-vault-adapter.sol#144) is not in mixedCase
Parameter AutoVaultAdapter.increaseWithdrawalAmount(address,uint256,
    ↪ uint256)._amount (auto-vault-adapter.sol#145) is not in mixedCase
Parameter AutoVaultAdapter.setWithdrawalAmount(address,uint256,uint256).
    ↪ _user (auto-vault-adapter.sol#157) is not in mixedCase
Parameter AutoVaultAdapter.setWithdrawalAmount(address,uint256,uint256).
    ↪ _nftId (auto-vault-adapter.sol#158) is not in mixedCase
Parameter AutoVaultAdapter.setWithdrawalAmount(address,uint256,uint256).
    ↪ _amount (auto-vault-adapter.sol#159) is not in mixedCase
Parameter AutoVaultAdapter.setInvestor(address)._investor (auto-vault-
    ↪ adapter.sol#168) is not in mixedCase
Parameter AutoVaultAdapter.setPoolID(uint256)._poolID (auto-vault-
    ↪ adapter.sol#177) is not in mixedCase
Parameter AutoVaultAdapter.getPaths(address,address)._inToken (auto-
    ↪ vault-adapter.sol#186) is not in mixedCase
Parameter AutoVaultAdapter.getPaths(address,address)._outToken (auto-
    ↪ vault-adapter.sol#186) is not in mixedCase
Parameter AutoVaultAdapter.setPath(address,address,address[])._inToken (
    ↪ auto-vault-adapter.sol#216) is not in mixedCase
```

```
Parameter AutoVaultAdapter.setPath(address,address,address[])._outToken
    ↪ (auto-vault-adapter.sol#217) is not in mixedCase
Parameter AutoVaultAdapter.setPath(address,address,address[])._paths (
    ↪ auto-vault-adapter.sol#218) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #conformance-to-solidity-naming-conventions

AutoVaultAdapter.repayToken (auto-vault-adapter.sol#22) should be
    ↪ constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #state-variables-that-could-be-declared-constant

renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (../../../../../openzeppelin-
            ↪ contracts/contracts/access/Ownable.sol#53-55)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (../../../../../openzeppelin
            ↪ -contracts/contracts/access/Ownable.sol#61-64)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #public-function-that-could-be-declared-external
```

Conclusion:

Most of the vulnerabilities found by the analysis have already been addressed by the smart contract code review.

6 Conclusion

In this audit, we examined the design and implementation of HEDGEPIE V2 contract and discovered several issues of varying severity. HedgePie team addressed 2 issues raised in the initial report and implemented the necessary fixes, while classifying the rest as a risk with low-probability of occurrence. Shellboxes' auditors advised HedgePie Team to maintain a high level of vigilance and to keep those findings in mind in order to avoid any future complications.

For a Contract Audit, contact us at contact@shellboxes.com