Bank Of Chain

Smart Contract Security Audit

Prepared by ShellBoxes

Oct 12th, 2022 - Dec 4th, 2022

Shellboxes.com

contact@shellboxes.com

Document Properties

Client	Bank Of Chain
Version	1.0
Classification	Public

Scope

Repository	Commit Hash
https://github.com/Bank-Of-Chain/boc-contract-core	a5bf42dc860297b63bed2fcac57d2d8cabe372ae
https://github.com/Bank-Of-Chain/boc-contract-periphery-eth	582bf01a8719b19e4826ba79e9292bbb5b153306

Re-Audit

Repository	Commit Hash
https://github.com/Bank-Of-Chain/boc-contract-core	b709f778e45b9d5bc37f932af9df5a203ecc14e7
https://github.com/Bank-Of-Chain/boc-contract-periphery-eth	030580fc69f81569c0c28dc04430296457719903

Contacts

COMPANY	EMAIL
ShellBoxes	contact@shellboxes.com

Contents

1 Introduction

Bank Of Chain engaged ShellBoxes to conduct a security assessment on the Bank Of Chain beginning on Oct 12th, 2022 and ending Dec 4th, 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 Bank Of Chain

BoC (Bank Of Chain) is a new and innovative platform in the decentralized finance (DeFi) ecosystem. It helps ordinary users to obtain a near "risk-free" wealth management tool on the blockchain. The BoC platform connects carefully selected protocols within the crypto ecosystem, including Automatic Market Makers (AMMs), lending protocols, yield aggregators, among others.

Issuer	Bank Of Chain
Website	`bankofchain.io`
Type	Solidity Smart Contract
Documentation	`https://docs.bankofchain.io/docs/boc/readme`
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				
	High	Critical	High	Medium
	Medium	High	Medium	Low
	Low	Medium	Low	Low
		High	Medium	Low

Likelihood

2 Findings Overview

2.1 Summary

The following is a synopsis of our conclusions from our analysis of the Bank Of Chain 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 critical-severity, 5 high-severity, 4 medium-severity, 6 low-severity, 1 undetermined-severity vulnerabilities.

Vulnerabilities	Severity	Status
SHB.1. Certain Strategies Allow Anyone To Withdraw Funds And Rewards Of All The Investors	CRITICAL	Fixed
SHB.2. The Investor's Funds May Get Locked In The Vault	HIGH	Fixed
SHB.3. forceRemoveStrategy Can Lock The Investor's Funds	HIGH	Acknowledged
SHB.4. The Swap Caller's Funds Can Get Locked	HIGH	Fixed
SHB.5. The Vault Manager Can Desynchronize The Vesting By Changing The Token	HIGH	Fixed
SHB.6. The Governor Can Take The Harvested Rewards	HIGH	Acknowledged
SHB.7. The Exchange Adapter Can Be Spoofed By The Governor Or The Delegate	MEDIUM	Acknowledged
SHB.8. Centralization Risk	MEDIUM	Acknowledged

SHB.9. Transaction Order Dependency	MEDIUM	Fixed
SHB.10. Front-run In The Contract's Initialization	MEDIUM	Acknowledged
SHB.11. Missing Address Verification	LOW	Fixed
SHB.12. The Prices Can Be Manipulated By The Owner	LOW	Acknowledged
SHB.13. Avoid Using .transfer() To Transfer Ether	LOW	Acknowledged
SHB.14. Approve Race Condition	LOW	Fixed
SHB.15. The Length And Address Of The _exchangeAdapters Argument Are Not Validated	LOW	Fixed
SHB.16. Floating Pragma	LOW	Fixed
SHB.17. Mismatch between the Code and the Documentation	UNDETERMINED	Acknowledged

3 Finding Details

SHB.1 Certain Strategies Allow Anyone To Withdraw Funds And Rewards Of All The Investors

- Severity : CRITICAL
- Status : Fixed
- Likelihood : 3
- Impact : 3

Description:

Rather than a constructor, multiple contracts initialize their state with the initialize method. However, many convex strategy contracts lack the initializer modifier, exposing them to re-initialization attacks from anyone. Due to the fact that the contract can be re-initialized by anyone, the vault and harvester addresses are vulnerable to manipulation by an attacker, allowing him to harvest and withdraw all strategy rewards in addition to the capital invested in this strategy.

Exploit Scenario:

1. The attacker calls the initialize function to overwrite the vault and harvester addresses with his own addresses.
2. The attacker calls the harvest function to withdraw all the rewards generated by the strategy.
3. The attacker calls the repay function to withdraw all the capital invested in the strategy.

Files Affected:

SHB.1.1: ConvexAaveStrategy.sol

```
17 function initialize(
18     address _vault,
```

```
    address _harvester,
    string memory _name
) public {
    address[] memory _wants = new address[](3);
    // the oder is same with underlying coins
    // DAI
    _wants[0] = address(0x6B175474E89094C44Da98b954EedeAC495271d0F);
    // USDC
    _wants[1] = address(0xA0b86991c6218b36c1d19D4a2e9Eb0cE3606eB48);
    // USDT
    _wants[2] = address(0xdAC17F958D2ee523a2206206994597C13D831ec7);
    super._initialize(
        _vault,
        _harvester,
        _name,
        _wants,
        0xDeBF20617708857ebe4F679508E7b7863a8A8EeE,
        0xE82c1eB4BC6F92f85BF7EB6421ab3b882C3F5a7B
    );
}
```

SHB.1.2: Convex3CrvStrategy.sol

```
function initialize(
    address _vault,
    address _harvester,
    string memory _name
) public {
    address[] memory _wants = new address[](3);
    // the oder is same with coins
    // DAI
    _wants[0] = address(0x6B175474E89094C44Da98b954EedeAC495271d0F);
    // USDC
    _wants[1] = address(0xA0b86991c6218b36c1d19D4a2e9Eb0cE3606eB48);
    // USDT
```

```
26      _wants[2] = address(0xdAC17F958D2ee523a2206206994597C13D831ec7);
27      super._initialize(
28          _vault,
29          _harvester,
30          _name,
31          _wants,
32          0xbEbc44782C7dB0a1A60Cb6fe97d0b483032FF1C7,
33          0x689440f2Ff927E1f24c72F1087E1FAF471eCe1c8
34      );
35  }
```

SHB.1.3: ConvexCompoundStrategy.sol

```
18  function initialize(address _vault, address _harvester,string memory
        ↪ _name) public {
19      address[] memory _wants = new address[](2);
20      _wants[0] = DAI;
21      _wants[1] = USDC;
22      super._initialize(
23          _vault,
24          _harvester,
25          _name,
26          _wants,
27          0xA2B47E3D5c44877cca798226B7B8118F9BFb7A56,
28          0xf34DFF761145FF0B05e917811d488B441F33a968
29      );
30  }
```

SHB.1.4: ConvexSaaveStrategy.sol

```
16  function initialize(
17      address _vault,
18      address _harvester,
19      string memory _name
20  ) public {
21      address[] memory _wants = new address[](2);
```

```
22      // the oder is same with underlying coins
23      // DAI
24      _wants[0] = address(0x6B175474E89094C44Da98b954EedeAC495271d0F);
25      // sUSD
26      _wants[1] = address(0x57Ab1ec28D129707052df4dF418D58a2D46d5f51);
27      super._initialize(
28          _vault,
29          _harvester,
30          _name,
31          _wants,
32          0xEB16Ae0052ed37f479f7fe63849198Df1765a733,
33          0xF86AE6790654b70727dbE58BF1a863B270317fD0
34      );
35  }
```

SHB.1.5: ConvexSusdStrategy.sol

```
16  function initialize(address _vault, address _harvester,string memory
        ↪ _name) public {
17      address[] memory _wants = new address[](4);
18      // the oder is same with underlying coins
19      // DAI
20      _wants[0] = address(0x6B175474E89094C44Da98b954EedeAC495271d0F);
21      // USDC
22      _wants[1] = address(0xA0b86991c6218b36c1d19D4a2e9Eb0cE3606eB48);
23      // USDT
24      _wants[2] = address(0xdAC17F958D2ee523a2206206994597C13D831ec7);
25      // sUSD
26      _wants[3] = address(0x57Ab1ec28D129707052df4dF418D58a2D46d5f51);
27      super._initialize(
28          _vault,
29          _harvester,
30          _name,
31          _wants,
32          0xA5407eAE9Ba41422680e2e00537571bcC53efBfD,
```

```
        0x22eE18aca7F3Ee920D01F25dA85840D12d98E8Ca
    );
}
```

Recommendation:

Consider adding the initializer modifier to protect the initialize function, so it can only be called once.

Updates

The team has fixed the issue by implementing the recommended solution of adding the initializer modifier to protect the initialize function.

SHB.1.6: ConvexAaveStrategy.sol

```
    function initialize(
        address _vault,
        address _harvester,
        string memory _name
    ) public initializer{
        address[] memory _wants = new address[](3);
```

SHB.1.7: Convex3CrvStrategy.sol

```
    function initialize(
        address _vault,
        address _harvester,
        string memory _name
    ) public initializer {
        address[] memory _wants = new address[](3);
```

SHB.1.8: ConvexCompoundStrategy.sol

```
    function initialize(address _vault, address _harvester,string memory
        ↪  _name) public initializer{
        address[] memory _wants = new address[](2);
```

SHB.1.9: ConvexSaaveStrategy.sol

```
24      function initialize(
25          address _vault,
26          address _harvester,
27          string memory _name
28      ) public initializer{
29          address[] memory _wants = new address[](2);
```

SHB.1.10: ConvexSusdStrategy.sol

```
24      function initialize(address _vault, address _harvester,string memory
            ↪  _name) public initializer{
25          address[] memory _wants = new address[](4);
```

SHB.2 The Investor's Funds May Get Locked In The Vault

- Severity : HIGH
- Status : Fixed
- Likelihood : 2
- Impact : 3

Description:

Any change in the vaultBufferAddress or the pegTokenAddress might result in locking the investor's funds in the vault as the previously minted USDi tickets, USDi tokens, ETHi tickets and ETHi tokens will not be valid to the new token address.

Exploit Scenario:

1. The governor changes the vaultBufferAddress, the investor's USDi tickets will not be available in the new vault buffer contract, therefore, the investor will not get any minted USDi tokens in the token distribution.
2. The governor changes the pegTokenAddress, the investor's USDi tokens will not be available in the new peg token contract, therefore, the investor will not be able to exchange his USDi tokens for the supported stable coins.

Files Affected:

SHB.2.1: VaultAdmin.sol

```
96  function setVaultBufferAddress(address _address) external onlyRole(
        ↪ BocRoles.GOV_ROLE) {
97      require(_address != address(0), "vaultBuffer ad is 0");
98      vaultBufferAddress = _address;
99  }
```

SHB.2.2: VaultAdmin.sol

```
103 function setPegTokenAddress(address _address) external onlyRole(BocRoles
        ↪ .GOV_ROLE) {
104     require(_address != address(0), "PegTokenAddress ad is 0");
105     pegTokenAddress = _address;
106 }
```

SHB.2.3: ETHVaultAdmin.sol

```
104 function setVaultBufferAddress(address _address) external onlyRole(
        ↪ BocRoles.GOV_ROLE) {
105     require(_address != address(0), "vaultBuffer ad is 0");
106     vaultBufferAddress = _address;
107 }
```

SHB.2.4: ETHVaultAdmin.sol

```
109 function setPegTokenAddress(address _address) external onlyRole(BocRoles
        ↪ .GOV_ROLE) {
110     require(_address != address(0), "PegTokenAddress ad is 0");
111     pegTokenAddress = _address;
112 }
```

Recommendation:

It is recommended that these setters be removed to avoid exposing the investor's funds to this risk.

Updates

The team has fixed this issue by adding a require statement to validate that the new address can only be set once.This ensures that any changes to the addresses are prevented.

SHB.2.5: VaultAdmin.sol

```
98  function setVaultBufferAddress(address _address) external onlyRole(
        ↪ BocRoles.GOV_ROLE) {
99          require(_address != address(0), "vaultBuffer ad is 0");
100         require(vaultBufferAddress == address(0), "VaultBuffer ad has
                ↪ been set");
101         vaultBufferAddress = _address;
102     }
```

SHB.2.6: VaultAdmin.sol

```
106 function setPegTokenAddress(address _address) external onlyRole(BocRoles
        ↪ .GOV_ROLE) {
107         require(_address != address(0), "PegToken ad is 0");
108         require(pegTokenAddress == address(0), "PegToken ad has been set
                ↪ ");
109         pegTokenAddress = _address;
110 }
```

SHB.2.7: ETHVaultAdmin.sol

```
99  function setVaultBufferAddress(address _address) external onlyRole(
        ↪ BocRoles.GOV_ROLE) {
100         require(_address != address(0), "VaultBuffer ad is 0");
101         require(vaultBufferAddress == address(0), "VaultBuffer ad has
                ↪ been set");
102         vaultBufferAddress = _address;
103     }
```

SHB.2.8: ETHVaultAdmin.sol

```
107 function setPegTokenAddress(address _address) external onlyRole(BocRoles
        ↪ .GOV_ROLE) {
```

```
108         require(_address != address(0), "PegToken ad is 0");
109         require(pegTokenAddress == address(0), "PegToken ad has been set
                ↪ ");
110         pegTokenAddress = _address;
111     }
```

SHB.3 forceRemoveStrategy Can Lock The Investor's Funds

- Severity : HIGH
- Status : Acknowledged
- Likelihood : 2
- Impact : 3

Description:

The forceRemoveStrategy function allows the governor/delegate to remove a strategy even if it has funds and the repay call will fail; therefore, it puts the user's funds into a risky position.

Exploit Scenario:

A number of users are investing in a specific strategy, then the governor calls the forceRemoveStrategy to remove the strategy and the repay call will fail to repay the invested funds to the vault.Hence, locking the investors' funds in the contract.

Files Affected:

SHB.3.1: VaultAdmin.sol

```
260 function forceRemoveStrategy(address _strategy) external
        ↪ onlyGovOrDelegate {
261     _removeStrategy(_strategy, true);
262     emit RemoveStrategyByForce(_strategy);
263 }
```

SHB.3.2: VaultAdmin.sol

```
286 function _removeStrategy(address _addr, bool _force) internal {
287     if(strategies[_addr].totalDebt > 0){
288         // Withdraw all assets
289         try IStrategy(_addr).repay(MAX_BPS, MAX_BPS, 0) {} catch {
290             if (!_force) {
291                 revert();
292             }
293         }
294     }
295 
296     address[] memory _wants = IStrategy(_addr).getWants();
297     for (uint256 i = 0; i < _wants.length; i++) {
298         address _wantToken = _wants[i];
299         trackedAssetsMap.minus(_wantToken, 1);
300         if (
301             trackedAssetsMap.get(_wantToken) <= 0 &&
302             IERC20Upgradeable(_wantToken).balanceOf(address(this)) == 0
303         ) {
304             trackedAssetsMap.remove(_wantToken);
305         }
306     }
307     if(strategies[_addr].totalDebt > 0){
308         totalDebt -= strategies[_addr].totalDebt;
309     }
310     delete strategies[_addr];
311     strategySet.remove(_addr);
312     _removeStrategyFromQueue(_addr);
313 }
```

SHB.3.3: ETHVaultAdmin.sol

```
249 function forceRemoveStrategy(address _strategy) external
    ↪ onlyGovOrDelegate {
250     _removeStrategy(_strategy, true);
```

```
251     emit RemoveStrategyByForce(_strategy);
252 }
```

SHB.3.4: ETHVaultAdmin.sol

```
258 function _removeStrategy(address _addr, bool _force) internal {
259     if (strategies[_addr].totalDebt > 0) {
260         // Withdraw all assets
261         try IETHStrategy(_addr).repay(MAX_BPS, MAX_BPS, 0) {} catch {
262             if (!_force) {
263                 revert();
264             }
265         }
266     }
267 
268     address[] memory _wants = IETHStrategy(_addr).getWants();
269     for (uint256 i = 0; i < _wants.length; i++) {
270         address _wantToken = _wants[i];
271         trackedAssetsMap.minus(_wantToken, 1);
272         if (trackedAssetsMap.get(_wantToken) <= 0) {
273             uint256 _balance;
274             if (_wantToken == NativeToken.NATIVE_TOKEN) {
275                 _balance = address(this).balance;
276             } else {
277                 _balance = IERC20Upgradeable(_wantToken).balanceOf(address
                        ↪ (this));
278             }
279             if (_balance == 0) {
280                 trackedAssetsMap.remove(_wantToken);
281             }
282         }
283     }
284     if (strategies[_addr].totalDebt > 0) {
285         totalDebt -= strategies[_addr].totalDebt;
286     }
```

```
287         delete strategies[_addr];
288         strategySet.remove(_addr);
289         _removeStrategyFromQueue(_addr);
290     }
```

Recommendation:

Consider removing this functionality, as the strategy removal should only occur when the strategy has no funds.

Updates

The team has acknowledged the issue, stating that the forceRemoveStrategy function will only be used in exceptional circumstances, such as when a third-party strategy experiences major problems that are nearly impossible to be resolved and its funds are no longer redeemable.In normal cases, the removeStrategy function will be utilized.

SHB.4 The Swap Caller's Funds Can Get Locked

- Severity : HIGH
- Status : Fixed
- Likelihood : 2
- Impact : 3

Description:

In the swap function, there exists a scenario in which the user's funds are locked in the contract without being spent for any purpose.

Exploit Scenario:

The caller will send a value of the native asset, and sd.srcToken is distinct from NativeToken.NATIVE TOKEN; thus, the native token funds are gone. Moreover, the contract ensures that msg.value is greater than _ethValue, which means that msg.value - _ethValue Wei will be lost.

Files Affected:

SHB.4.1: ExchangeAggregator.sol

```
59 function swap(
60     address _platform,
61     uint8 _method,
62     bytes calldata _data,
63     IExchangeAdapter.SwapDescription calldata _sd
64 ) public payable override returns (uint256) {
65     require(exchangeAdapters.contains(_platform), "error swap platform")
           ↪ ;
66     require(_sd.receiver != address(0), "error receiver");
67     uint256 _exchangeAmount = 0;
68     if (_sd.srcToken == NativeToken.NATIVE_TOKEN) {
69         uint256 _ethValue = _sd.amount;
70         require(_ethValue <= msg.value, "ETH not enough");
71         _exchangeAmount = IExchangeAdapter(_platform).swap{value:
               ↪ _ethValue}(_method, _data, _sd);
72     } else {
73         IERC20(_sd.srcToken).safeTransferFrom(msg.sender, _platform, _sd.
               ↪ amount);
74         _exchangeAmount = IExchangeAdapter(_platform).swap(_method, _data
               ↪ , _sd);
75     }
```

Recommendation:

Consider requiring the msg.value to be equal to zero when the _sd.srcToken is different from the NativeToken.NATIVE_TOKEN, also we recommend that you verify the _ethValue to be equal to msg.value or to transfer back the msg.value - _ethValue at the end of the swap.

Updates

The team has resolved the issue by requiring the msg.value to be equal to zero when the _sd.srcToken is different from the NativeToken.NATIVE_TOKEN.

SHB.4.2: ExchangeAggregator.sol

```
66  function swap(
67          address _platform,
68          uint8 _method,
69          bytes calldata _data,
70          IExchangeAdapter.SwapDescription calldata _sd
71      ) public payable override returns (uint256) {
72          if (_sd.srcToken == NativeToken.NATIVE_TOKEN) {
73              require(_sd.amount == msg.value, "amount invalid");
74          }else{
75              require(0 == msg.value, "msg.value invalid");
76          }
77          return _swap(_platform, _method, _data, _sd);
78      }
```

SHB.5 The Vault Manager Can Desynchronize The Vesting By Changing The Token

- Severity : HIGH
- Likelihood : 2
- Status : Fixed
- Impact : 3

Description:

The vault manager has the ability to change the token. However, the drip variable is not reinitialized after changing the token, which will generate a desynchronization .So, the new token will use the old token's parameters.

Exploit Scenario:

The vault manager changes the address of the token variable, then in the next collect call, the function will use the perBlock attribute of the old token which will generate unexpected

outputs.

Files Affected:

SHB.5.1: Dripper.sol

```
123 function setToken(address _token) external isVaultManager {
124     require(_token != address(0), "Must be a non-zero address");
125     token = _token;
126     emit TokenChanged(_token);
127 }
```

SHB.5.2: Dripper.sol

```
150 function _collect() internal {
151     // Calculate send
152     uint256 _balance = IERC20Upgradeable(token).balanceOf(address(this))
        ↪ ;
153     uint256 _amountToSend = _availableFunds(_balance, drip);
154     uint256 _remaining = _balance - _amountToSend;
155     // Calculate new drip perBlock
156     // Gas savings by setting entire struct at one time
157     drip = Drip({perBlock: uint192(_remaining / dripDuration),
        ↪ lastCollect: uint64(block.timestamp)});
158     // Send funds
159     IERC20Upgradeable(token).safeTransfer(vault, _amountToSend);
160     emit Collection(token, _amountToSend);
161 }
```

Recommendation:

Consider re-initializing the drip variable after changing the token address.

Updates

The team has fixed the issue by eliminating the Dripper.sol contract.

SHB.6 The Governor Can Take The Harvested Rewards

- Severity : HIGH
- Status : Acknowledged
- Likelihood : 2
- Impact : 3

Description:

The setProfitReceiver function in the Harvester contract enables the governor to set the profitReceiver address. This enables the governor to receive all the profit generated by the investments of various users in multiple strategies.

Exploit Scenario:

1. The governor calls the setProfitReceiver function and setS the profitReceiver address to his wallet.
2. When the keeper invokes the exchangeAndSend function in the Harvester contract, the governor begins getting all the revenue.

Files Affected:

SHB.6.1: Harvester.sol

```
54 function setProfitReceiver(address _receiver) external override onlyRole
       ↪ (BocRoles.GOV_ROLE) {
55     require(_receiver != address(0), "Must be a non-zero address");
56     profitReceiver = _receiver;
57
58     emit ReceiverChanged(profitReceiver);
59 }
```

Recommendation:

Consider removing the setProfitReceiver function and returning the exchanged rewards to the vault.

Updates

The team has acknowledged the issue, stating that the authority will be transferred to the governance contract.

SHB.7 The Exchange Adapter Can Be Spoofed By The Governor Or The Delegate

- Severity : MEDIUM
- Status : Acknowledged
- Likelihood : 1
- Impact : 3

Description:

The governor/delegate is able to modify the addresses of the exchange adapters, allowing them to enter a malicious contract that only takes the caller's funds instead of swapping.

Exploit Scenario:

The governor/delegate creates a contract containing a swap function that receives the user's funds and transfers them to an external wallet. He then specifies the address of this contract as an exchange adapter in the ExchangeAggregator contract, allowing him to receive the caller's funds.

Files Affected:

SHB.7.1: ExchangeAggregator.sol

```
33 function addExchangeAdapters(address[] calldata _exchangeAdapters)
34     external
35     override
36     onlyGovOrDelegate
37 {
38     __addExchangeAdapters(_exchangeAdapters);
```

```
39 }
```

Recommendation:

Consider using a multisig wallet as the governor and the delegate to avoid centralization risks and allow multiple parties to contribute to the protocol's safety.

Updates

The team has acknowledged the issue, stating that The effect can be achieved when new strategies are deployed.

SHB.8 Centralization Risk

- Severity : MEDIUM
- Status : Acknowledged
- Likelihood : 1
- Impact : 3

Description:

The transferToken function provides the governor with complete authority over the Dripper contract, allowing him to transfer any amount of any asset to the treasury, which can result in unanticipated behavior and will violate the vesting structure. The same issue has been identified in the Harvester contract.

Exploit Scenario:

When the governor drains the contract from the token assets, the contract will have no funds to transfer to the vault; the perBlock attribute will be set to 0 in the next collect call; and the vault will never be able to take the available funds, even if the contract is funded later on.

Files Affected:

SHB.8.1: Dripper.sol

```
132 function transferToken(address _asset, uint256 _amount) external
        ↪ onlyRole(BocRoles.GOV_ROLE) {
133     IERC20Upgradeable(_asset).safeTransfer(IVault(vault).treasury(),
            ↪ _amount);
134 }
```

SHB.8.2: Harvester.sol

```
70      function transferToken(address _asset, uint256 _amount)
71          external
72          override
73          onlyRole(BocRoles.GOV_ROLE)
74      {
75          IERC20Upgradeable(_asset).safeTransfer(IVault(vaultAddress).
                ↪ treasury(), _amount);
76      }
```

Recommendation:

To avoid the centralization risk, it is recommended to delete this feature and utilize a multi-sig wallet as the governor.

Updates

The team has acknowledged the issue, stating that the authority will be transferred to the governance contract.

SHB.9 Transaction Order Dependency

- Severity : MEDIUM
- Status : Fixed
- Likelihood : 1
- Impact : 3

Description:

A race condition vulnerability exists when code is dependent on the order of transactions submitted to it. There are some changeable variables within the project that may be affected by the transaction's execution order.

Exploit Scenario:

1. The investor calls the burn function from the Vault contract using a specific value of the redeemFeeBps, then the vault manager changes the redeemFeeBps. If the vault manager's transaction gets mined first, the investor's transaction will be executed using the new value of redeemFeeBps generating an unexpected output.
2. A caller executes the rebase function from the Vault contract using a specific value of the trusteeFeeBps, then the vault manager changes the trusteeFeeBps. If the vault manager's transaction gets mined first, the transaction of the rebase's caller will be executed using the new value of trusteeFeeBps generating an unexpected output.
3. The same scenario can be applied to the ETHVaultAdmin contract.

Files Affected:

SHB.9.1: VaultAdmin.sol

```
45 function setRedeemFeeBps(uint256 _redeemFeeBps) external isVaultManager
       ↪ {
46     require(_redeemFeeBps <= 1000, "Redeem fee should not be over 10%");
47     redeemFeeBps = _redeemFeeBps;
48     emit RedeemFeeUpdated(_redeemFeeBps);
49 }
```

SHB.9.2: VaultAdmin.sol

```
111 function setTrusteeFeeBps(uint256 _basis) external isVaultManager {
112     require(_basis <= 5000, "basis cannot exceed 50%");
113     trusteeFeeBps = _basis;
114     emit TrusteeFeeBpsChanged(_basis);
115 }
```

SHB.9.3: ETHVaultAdmin.sol

```
47 function setRedeemFeeBps(uint256 _redeemFeeBps) external isVaultManager
    ↪ {
48     require(_redeemFeeBps <= 1000, "Redeem fee should not be over 10%");
49     redeemFeeBps = _redeemFeeBps;
50     emit RedeemFeeUpdated(_redeemFeeBps);
51 }
```

SHB.9.4: ETHVaultAdmin.sol

```
118 function setTrusteeFeeBps(uint256 _basis) external isVaultManager {
119     require(_basis <= 5000, "basis cannot exceed 50%");
120     trusteeFeeBps = _basis;
121     emit TrusteeFeeBpsChanged(_basis);
122 }
```

Recommendation:

Consider adding redeemFeeBps and trusteeFeeBps as arguments then adding a require statement to ensure that the fee values provided in the arguments match those that are stored in the smart contract, or consider notifying the community with any change in terms of the fees to mitigate the risk.

Updates

The team resolved the issue by adding the redeemFeeBps and trusteeFeeBps as arguments to the burn and the rebase functions as recommended, avoiding transaction order dependency.

SHB.9.5: Vault.sol

```
214     function burn(uint256 _amount, uint256 _minimumAmount, uint256
            ↪ _redeemFeeBps, uint256 _trusteeFeeBps)
215         external
216         whenNotEmergency
217         whenNotAdjustPosition
218         nonReentrant
219         returns (address[] memory _assets, uint256[] memory _amounts)
220     {
221         uint256 _accountBalance = IPegToken(pegTokenAddress).balanceOf(
                ↪ msg.sender);
222         require(_amount > 0 && _amount <= _accountBalance, "AI");//USDi
                ↪ not enough,amount is invalid
223         require(_redeemFeeBps == redeemFeeBps, "RI");//redeemFeeBps
                ↪ invalid
224         require(_trusteeFeeBps == trusteeFeeBps, "TI");//trusteeFeeBps
                ↪ invalid
```

SHB.9.6: Vault.sol

```
374     function rebase(uint256 _trusteeFeeBps) external whenNotEmergency
            ↪ whenNotAdjustPosition whenNotRebasePaused nonReentrant {
375         require(_trusteeFeeBps == trusteeFeeBps, "TI");//trusteeFeeBps
                ↪ invalid
376         uint256 _totalAssets = _totalValueInVault() + totalDebt;
377         _rebase(_totalAssets, _trusteeFeeBps);
378     }
```

SHB.9.7: ETHVault.sol

```
239     function burn(uint256 _amount, uint256 _minimumAmount, uint256
            ↪ _redeemFeeBps, uint256 _trusteeFeeBps)
240         external
241         whenNotEmergency
242         whenNotAdjustPosition
243         nonReentrant
```

```
244         returns (address[] memory _assets, uint256[] memory _amounts)
245     {
246         uint256 _accountBalance = IPegToken(pegTokenAddress).balanceOf(
                ↪ msg.sender);
247         require(_amount > 0 && _amount <= _accountBalance, "AI");//ETHi
                ↪ not enough,amount is invalid
248         require(_redeemFeeBps == redeemFeeBps, "RI");//redeemFeeBps
                ↪ invalid
249         require(_trusteeFeeBps == trusteeFeeBps, "TI");//trusteeFeeBps
                ↪ invalid
```

SHB.9.8: ETHVault.sol

```
411     function rebase(uint256 _trusteeFeeBps)
412         external
413         whenNotEmergency
414         whenNotAdjustPosition
415         whenNotRebasePaused
416         nonReentrant
417     {
418         require(_trusteeFeeBps == trusteeFeeBps, "TI");//trusteeFeeBps
                ↪ invalid
419         uint256 _totalAssets = _totalAssetInVault() + totalDebt;
420         _rebase(_totalAssets, _trusteeFeeBps);
421     }
```

SHB.10 Front-run In The Contract's Initialization

- Severity : MEDIUM
- Status : Acknowledged
- Likelihood : 1
- Impact : 3

Description:

Multiple contracts initialize their state with an initialize function instead of a constructor to implement upgradability, leaving the initialization vulnerable to being front-run by an attacker.

Exploit Scenario:

The owner deploys the contract and performs the initialize function, then the attacker front-runs the transaction by paying a higher gas price and inputting malicious values into the contract.

Files Affected:

SHB.10.1: Dripper.sol

```
81 function initialize(
82     address _accessControlProxy,
83     address _vault,
84     address _token
85 ) external initializer {
86     require(_vault != address(0), "Must be a non-zero address");
87     require(_token != address(0), "Must be a non-zero address");
88
89     vault = _vault;
90     token = _token;
91     _initAccessControl(_accessControlProxy);
92 }
```

SHB.10.2: Harvester.sol

```
36 function initialize(
37     address _accessControlProxy,
38     address _receiver,
39     address _sellTo,
40     address _vault
41 ) external initializer {
```

```
42      require(_receiver != address(0), "Must be a non-zero address");
43      require(_vault != address(0), "Must be a non-zero address");
44      require(_sellTo != address(0), "Must be a non-zero address");
45      profitReceiver = _receiver;
46      sellTo = _sellTo;
47      vaultAddress = _vault;
48      exchangeManager = IVault(_vault).exchangeManager();
49      _initAccessControl(_accessControlProxy);
50  }
```

SHB.10.3: Treasury.sol

```
22  function initialize(address _accessControlProxy) public initializer {
23      _initAccessControl(_accessControlProxy);
24  }
```

SHB.10.4: Vault.sol

```
21  function initialize(
22      address _accessControlProxy,
23      address _treasury,
24      address _exchangeManager,
25      address _valueInterpreter
26  ) public initializer {
27      _initAccessControl(_accessControlProxy);
28
29      treasury = _treasury;
30      exchangeManager = _exchangeManager;
31      valueInterpreter = _valueInterpreter;
32
33      rebasePaused = false;
34      // Initial redeem fee of 0 basis points
35      redeemFeeBps = 0;
36      // 1 / 1000e4
37      rebaseThreshold = 1;
38      // one week
```

```
39      maxTimestampBetweenTwoReported = 604800;
40      underlyingUnitsPerShare = 1e18;
41      minCheckedStrategyTotalDebt = 1000e18;
42  }
```

SHB.10.5: VaultBuffer.sol

```
71  function initialize(
72      string memory _name,
73      string memory _symbol,
74      address _vault,
75      address _pegTokenAddr,
76      address _accessControlProxy
77  ) external initializer {
78      mName = _name;
79      mSymbol = _symbol;
80      vault = _vault;
81      pegTokenAddr = _pegTokenAddr;
82      _initAccessControl(_accessControlProxy);
83
84      mDistributeLimit = 50;
85  }
```

SHB.10.6: AuraREthWEthStrategy.sol

```
51  function initialize(address _vault,string memory _name) external
        ↪ initializer {
52
53      address[] memory _wants = new address[](2);
54      _wants[0] = RETH; //rETH
55      _wants[1] = WETH; //wETH
```

SHB.10.7: AuraWstETHWETHStrategy.sol

```
52  function initialize(address _vault, string memory _name) external
        ↪ initializer {
53      address[] memory _wants = new address[](2);
```

```
54      _wants[0] = WSTETH; //wstETH
55      _wants[1] = WETH; //wETH
```

SHB.10.8: ConvexrETHwstETHStrategy.sol

```
15  function initialize(address _vault, string memory _name) external
        ↪ initializer {
16      super._initialize(_vault, _name);
17      //set up sell reward path
18      address[] memory _rewardCRVPath = new address[](2);
19      _rewardCRVPath[0] = CRV;
20      _rewardCRVPath[1] = W_ETH;
21      uniswapRewardRoutes[CRV] = _rewardCRVPath;
22      address[] memory _rewardCVXPath = new address[](2);
23      _rewardCVXPath[0] = CVX;
24      _rewardCVXPath[1] = W_ETH;
25      uniswapRewardRoutes[CVX] = _rewardCVXPath;
26  }
```

SHB.10.9: ConvexSETHStrategy.sol

```
14  function initialize(address _vault, string memory _name) external
        ↪ initializer {
15      super._initialize(_vault, _name);
16      //set up sell reward path
17      address[] memory _rewardCRVPath = new address[](2);
18      _rewardCRVPath[0] = CRV;
19      _rewardCRVPath[1] = W_ETH;
20      uniswapRewardRoutes[CRV] = _rewardCRVPath;
21      address[] memory _rewardCVXPath = new address[](2);
22      _rewardCVXPath[0] = CVX;
23      _rewardCVXPath[1] = W_ETH;
24      uniswapRewardRoutes[CVX] = _rewardCVXPath;
25  }
```

SHB.10.10: ConvexStETHStrategy.sol

```
14 function initialize(address _vault, string memory _name) external
       ↪ initializer {
15     super._initialize(_vault, _name);
16     //set up sell reward path
17     address[] memory _rewardCRVPath = new address[](2);
18     _rewardCRVPath[0] = CRV;
19     _rewardCRVPath[1] = W_ETH;
20     uniswapRewardRoutes[CRV] = _rewardCRVPath;
21     address[] memory _rewardCVXPath = new address[](2);
22     _rewardCVXPath[0] = CVX;
23     _rewardCVXPath[1] = W_ETH;
24     uniswapRewardRoutes[CVX] = _rewardCVXPath;
25     address[] memory _rewardLDOPath = new address[](2);
26     _rewardLDOPath[0] = LDO;
27     _rewardLDOPath[1] = W_ETH;
28     uniswapRewardRoutes[LDO] = _rewardLDOPath;
29 }
```

SHB.10.11: StakeWiseEthSeth23000Strategy.sol

```
18 function initialize(address _vault, string memory _name) public
       ↪ initializer {
19     uniswapV3Initialize(0x7379e81228514a1D2a6Cf7559203998E20598346, 60,
           ↪ 60, 41400, 0, 100, 60, 60);
20     address[] memory _wants = new address[](2);
21     _wants[0] = token0;
22     _wants[1] = token1;
23     super._initialize(_vault, uint16(ProtocolEnum.StakeWise), _name,
           ↪ _wants);
24 }
```

SHB.10.12: StakeWiseReth2Seth2500Strategy.sol

```
18 function initialize(address _vault, string memory _name) public
       ↪ initializer {
19     uniswapV3Initialize(0xa9ffb27d36901F87f1D0F20773f7072e38C5bfbA, 10,
```

```
        ↪ 10, 41400, 0, 100, 60, 10);
    address[] memory _wants = new address[](2);
    _wants[0] = token0;
    _wants[1] = token1;
    super._initialize(_vault, uint16(ProtocolEnum.StakeWise), _name,
        ↪ _wants);
}
```

SHB.10.13: ETHUniswapV2Strategy.sol

```
function initialize(address _vault,string memory _name,address _pair)
    ↪ external initializer {
    uniswapV2Pair = IUniswapV2Pair(_pair);
    address[] memory _wants = new address[](2);
    _wants[0] = uniswapV2Pair.token0();
    _wants[1] = uniswapV2Pair.token1();
    _initialize(_vault, uint16(ProtocolEnum.UniswapV2), _name,_wants);
    _initializeUniswapV2(0x7a250d5630B4cF539739dF2C5dAcb4c659F2488D);
}
```

SHB.10.14: ETHUniswapV3Strategy.sol

```
function initialize(
    address _vault,
    string memory _name,
    address _pool,
    int24 _baseThreshold,
    int24 _limitThreshold,
    uint256 _period,
    int24 _minTickMove,
    int24 _maxTwapDeviation,
    uint32 _twapDuration,
    int24 _tickSpacing
) public initializer {
    super._initialize(
        _vault,
```

```
        _name,
        _pool,
        _baseThreshold,
        _limitThreshold,
        _period,
        _minTickMove,
        _maxTwapDeviation,
        _twapDuration,
        _tickSpacing
    );
}
```

SHB.10.15: YearnV2Strategy.sol

```
function initialize(
    address _vault,
    string memory _name,
    address _yVault,
    address _token
) external initializer {
    yVault = IYearnVaultV2(_yVault);
    address[] memory _wants = new address[](1);
    _wants[0] = _token;
    super._initialize(_vault, uint16(ProtocolEnum.YearnV2), _name,
        ↪ _wants);
}
```

SHB.10.16: ETHVault.sol

```
function initialize(
    address _accessControlProxy,
    address _treasury,
    address _exchangeManager,
    address _priceProvider
) public initializer {
    _initAccessControl(_accessControlProxy);
```

```
27
28      treasury = _treasury;
29      exchangeManager = _exchangeManager;
30      priceProvider = _priceProvider;
31      // 1 / 1000e4
32      rebaseThreshold = 1;
33      // one week
34      maxTimestampBetweenTwoReported = 604800;
35      underlyingUnitsPerShare = 1e18;
36      minCheckedStrategyTotalDebt = 1e17;
37  }
```

SHB.10.17: Aura3PoolStrategy.sol

```
43  function initialize(address _vault, address _harvester,string memory
        ↪ _name) external initializer {
44      address[] memory _wants = new address[](3);
45      _wants[0] = DAI; //DAI
46      _wants[1] = USDC; //USDC
47      _wants[2] = USDT; //USDT
```

SHB.10.18: Convex3CrvStrategy.sol

```
14  function initialize(
15      address _vault,
16      address _harvester,
17      string memory _name
18  ) public {
19      address[] memory _wants = new address[](3);
20      // the oder is same with coins
21      // DAI
22      _wants[0] = address(0x6B175474E89094C44Da98b954EedeAC495271d0F);
23      // USDC
24      _wants[1] = address(0xA0b86991c6218b36c1d19D4a2e9Eb0cE3606eB48);
25      // USDT
26      _wants[2] = address(0xdAC17F958D2ee523a2206206994597C13D831ec7);
```

SHB.10.19: ConvexAaveStrategy.sol

```
function initialize(
    address _vault,
    address _harvester,
    string memory _name
) public {
    address[] memory _wants = new address[](3);
    // the oder is same with underlying coins
    // DAI
    _wants[0] = address(0x6B175474E89094C44Da98b954EedeAC495271d0F);
    // USDC
    _wants[1] = address(0xA0b86991c6218b36c1d19D4a2e9Eb0cE3606eB48);
    // USDT
    _wants[2] = address(0xdAC17F958D2ee523a2206206994597C13D831ec7);
```

SHB.10.20: ConvexCompoundStrategy.sol

```
function initialize(address _vault, address _harvester,string memory
    ↪ _name) public {
    address[] memory _wants = new address[](2);
    _wants[0] = DAI;
    _wants[1] = USDC;
    super._initialize(
        _vault,
        _harvester,
        _name,
        _wants,
        0xA2B47E3D5c44877cca798226B7B8118F9BFb7A56,
        0xf34DFF761145FF0B05e917811d488B441F33a968
    );
}
```

SHB.10.21: ConvexPaxStrategy.sol

```
function initialize(address _vault, address _harvester,string memory
    ↪ _name) public initializer {
```

```
21      address[] memory _wants = new address[](4);
22      _wants[0] = DAI;
23      _wants[1] = USDC;
24      _wants[2] = USDT;
25      _wants[3] = PAX;
```

SHB.10.22: ConvexSaaveStrategy.sol

```
16  function initialize(
17      address _vault,
18      address _harvester,
19      string memory _name
20  ) public {
21      address[] memory _wants = new address[](2);
22      // the oder is same with underlying coins
23      // DAI
24      _wants[0] = address(0x6B175474E89094C44Da98b954EedeAC495271d0F);
25      // sUSD
26      _wants[1] = address(0x57Ab1ec28D129707052df4dF418D58a2D46d5f51);
```

SHB.10.23: ConvexSusdStrategy.sol

```
16  function initialize(address _vault, address _harvester,string memory
        ↪ _name) public {
17      address[] memory _wants = new address[](4);
18      // the oder is same with underlying coins
19      // DAI
20      _wants[0] = address(0x6B175474E89094C44Da98b954EedeAC495271d0F);
21      // USDC
22      _wants[1] = address(0xA0b86991c6218b36c1d19D4a2e9Eb0cE3606eB48);
23      // USDT
24      _wants[2] = address(0xdAC17F958D2ee523a2206206994597C13D831ec7);
25      // sUSD
26      _wants[3] = address(0x57Ab1ec28D129707052df4dF418D58a2D46d5f51);
```

SHB.10.24: ConvexUsdtStrategy.sol

```
function initialize(address _vault, address _harvester,string memory
    ↪ _name) public initializer {
    address[] memory _wants = new address[](3);
    _wants[0] = address(0x6B175474E89094C44Da98b954EedeAC495271d0F);
    _wants[1] = address(0xA0b86991c6218b36c1d19D4a2e9Eb0cE3606eB48);
    _wants[2] = address(0xdAC17F958D2ee523a2206206994597C13D831ec7);
```

SHB.10.25: DForceLendStrategy.sol

```
function initialize(
    address _vault,
    address _harvester,
    string memory _name,
    address _underlyingToken,
    address _iToken
) external initializer {
    address[] memory _wants = new address[](1);
    _wants[0] = _underlyingToken;
    iToken = _iToken;
    super._initialize(_vault, _harvester, _name,uint16(ProtocolEnum.
        ↪ DForce), _wants);
}
```

SHB.10.26: DodoStrategy.sol

```
function initialize(
    address _vault,
    address _harvester,
    string memory _name,
    address _lpTokenPool,
    address _stakePool
) external initializer {
    require(_vault != address(0), "vault cannot be 0.");
    require(_stakePool != address(0), "stakePool cannot be 0.");
    require(_lpTokenPool != address(0), "lpTokenPool cannot be 0.");
    lpTokenPool = _lpTokenPool;
```

```
32      STAKE_POOL_ADDRESS = _stakePool;
```

SHB.10.27: DodoV1Strategy.sol

```
23  function initialize(
24      address _vault,
25      address _harvester,
26      string memory _name,
27      address _lpTokenPool,
28      address _stakePool
29  ) external initializer {
30      require(_vault != address(0), "vault cannot be 0.");
31      require(_stakePool != address(0), "stakePool cannot be 0.");
32      require(_lpTokenPool != address(0), "lpTokenPool cannot be 0.");
33      lpTokenPool = _lpTokenPool;
34      STAKE_POOL_V1_ADDRESS = _stakePool;
```

SHB.10.28: StargateSingleStrategy.sol

```
22  function initialize(
23      address _vault,
24      address _harvester,
25      string memory _name,
26      address _underlying,
27      address _router,
28      address _lpToken,
29      uint256 _poolId,
30      uint256 _stakePoolId
31  ) external initializer {
32      address[] memory _wants = new address[](1);
33      _wants[0] = _underlying;
34      stargatePool = IStargatePool(_lpToken);
35      stargateRouterPool = IStargateRouterPool(_router);
36      poolId = _poolId;
37      stakePoolId = _stakePoolId;
38      super._initialize(_vault, _harvester, _name, uint16(ProtocolEnum.
```

```
        ↪ Stargate), _wants);
}
```

SHB.10.29: SushiKashiStakeStrategy.sol

```
function initialize(
    address _vault,
    address _harvester,
    string memory _name,
    address _underlyingToken,
    address _pair,
    uint256 _poolId
) external initializer {
    address[] memory _wants = new address[](1);
    _wants[0] = _underlyingToken;
    kashiPari = IKashiPair(_pair);
    poolId = _poolId;
    bentoBox = kashiPari.bentoBox();
    super._initialize(_vault, _harvester, _name, uint16(ProtocolEnum.
        ↪ Sushi_Kashi), _wants);
}
```

SHB.10.30: UniswapV3Strategy.sol

```
function initialize(
    address _vault,
    address _harvester,
    string memory _name,
    address _pool,
    int24 _baseThreshold,
    int24 _limitThreshold,
    uint256 _period,
    int24 _minTickMove,
    int24 _maxTwapDeviation,
    uint32 _twapDuration,
    int24 _tickSpacing
```

```
60 ) external initializer {
61     _initializeUniswapV3Liquidity(_pool);
62     address[] memory _wants = new address[](2);
63     _wants[0] = token0;
64     _wants[1] = token1;
```

SHB.10.31: YearnEarnStrategy.sol

```
17 function initialize(
18     address _vault,
19     address _harvester,
20     string memory _name,
21     address _yVault,
22     address _underlyingToken
23 ) external initializer {
24     yVault = IYearnVault(_yVault);
25     underlyingToken = _underlyingToken;
26     address[] memory _wants = new address[](1);
27     _wants[0] = underlyingToken;
28     _initialize(_vault, _harvester, _name, uint16(ProtocolEnum.YearnEarn
           ↪ ), _wants);
29
30     isWantRatioIgnorable = true;
31 }
```

Recommendation:

Consider calling the initialize and the deployment of the contract in the same transaction, this can be done by using another contract, it can be either a proxy or a new contract.

Updates

The team acknowledged the risk, stating that the initialization and deployment will be both executed simultaneously during deployment.

SHB.11 Missing Address Verification

- Severity : LOW
- Status : Fixed
- Likelihood : 1
- Impact : 2

Description:

Certain functions lack a safety check in the address, the address-type arguments should include a zero-address test, otherwise, the contract's functionality may become inaccessible.

Exploit Scenario:

1. If a contract uses the AccessControlMixin contract for access control, the accessControlProxy argument can be set to address(0), which may deny access to all access control features.
2. The _vault argument can be set to address(0), which may deny access to all the functionalities that makes use of the vaultAddr variable.
3. The _pegToken argument can be set to address(0), which may deny access to all the functionalities that makes use of the pegToken variable.
4. The _address argument can be set to address(0), which will burn all the funds sent to the treasury.
5. The _exchangeManagerAddress argument can be set to address(0), which may deny access to all the exchange functionalities in the vault that makes use of the exchangeManager variable.
6. The _treasury, _exchangeManager and the _priceProvider arguments can be set to address(0), which may deny access to all the functionalities that make use of the treasury, exchangeManager and the priceProvider variables.
7. The _sd.receiver argument can be set to address(0), which will burn the output of the swap.

Files Affected:

SHB.11.1: AccessControlMixin.sol

```
13 function _initAccessControl(address _accessControlProxy) internal {
14     accessControlProxy = IAccessControlProxy(_accessControlProxy);
15 }
```

SHB.11.2: PegToken.sol

```
73 function initialize(
74     string calldata _nameArg,
75     string calldata _symbolArg,
76     uint8 _decimalsArg,
77     address _vault,
78     address _accessControlProxy
79 ) external initializer {
80     mName = _nameArg;
81     mSymbol = _symbolArg;
82     mDecimals = _decimalsArg;
83     vaultAddr = _vault;
84     _initAccessControl(_accessControlProxy);
85 }
```

SHB.11.3: WrappedPegToken.sol

```
18 constructor(
19     IPegToken _pegToken,
20     string memory _name,
21     string memory _symbol
22 ) ERC20Permit(_name) ERC20(_name, _symbol) {
23     pegToken = _pegToken;
24 }
```

SHB.11.4: VaultAdmin.sol

```
81 function setTreasuryAddress(address _address) external onlyRole(BocRoles
       ↪ .GOV_ROLE) {
```

```
82      treasury = _address;
83      emit TreasuryAddressChanged(_address);
84  }
```

SHB.11.5: ETHVault.sol

```
20  function initialize(
21      address _accessControlProxy,
22      address _treasury,
23      address _exchangeManager,
24      address _priceProvider
25  ) public initializer {
26      _initAccessControl(_accessControlProxy);
27
28      treasury = _treasury;
29      exchangeManager = _exchangeManager;
30      priceProvider = _priceProvider;
31      // 1 / 1000e4
32      rebaseThreshold = 1;
33      // one week
34      maxTimestampBetweenTwoReported = 604800;
35      underlyingUnitsPerShare = 1e18;
36      minCheckedStrategyTotalDebt = 1e17;
37  }
```

SHB.11.6: ParaSwapV5Adapter.sol

```
42  function swap(uint8 _method, bytes calldata _encodedCallArgs,
        ↪ IExchangeAdapter.SwapDescription calldata _sd) external payable
        ↪ override returns (uint256){
43      require(_method < swapMethodSelector.length, "ParaswapAdapter method
            ↪  out of range");
44      bytes4 _selector = swapMethodSelector[_method];
45      bytes memory _data = abi.encodeWithSelector(_selector,
            ↪ _encodedCallArgs, _sd);
46      bool _success;
```

```
47      bytes memory _result;
48      uint256 _toTokenBefore = getTokenBalance(_sd.dstToken, address(_sd.
            ↪ receiver));
49      (_success, _result) = address(this).delegatecall(_data);
50
51      if (_success) {
52          return getTokenBalance(_sd.dstToken, address(_sd.receiver)) -
                ↪ _toTokenBefore;
53      } else {
54          revert(RevertReasonParser.parse(_result, "paraswap callBytes
                ↪ failed: "));
55      }
56  }
```

Recommendation:

We recommend that you make sure the addresses provided in the arguments are different from the address(0).

Updates

The team resolved the issue by verifying the address arguments to be different from the address(0).

SHB.12 The Prices Can Be Manipulated By The Owner

- Severity : LOW
- Status : Acknowledged
- Likelihood : 1
- Impact : 2

Description:

The valueInterpreter and priceProvider variables record the contract address used to obtain asset prices. However, this variable can point to any contract, allowing the owner to

manipulate the prices by setting a malicious contract.

Exploit Scenario:

The owner constructs a malicious contract that returns custom pricing and sets its address in the valueInterpreter or priceProvider variable; therefore, manipulating the prices and producing unexpected outcomes in contracts that utilize this value interpreter.

Files Affected:

SHB.12.1: Vault.sol

```
function initialize(
    address _accessControlProxy,
    address _treasury,
    address _exchangeManager,
    address _valueInterpreter
) public initializer {
    _initAccessControl(_accessControlProxy);

    treasury = _treasury;
    exchangeManager = _exchangeManager;
    valueInterpreter = _valueInterpreter;

    rebasePaused = false;
    // Initial redeem fee of 0 basis points
    redeemFeeBps = 0;
    // 1 / 1000e4
    rebaseThreshold = 1;
    // one week
    maxTimestampBetweenTwoReported = 604800;
    underlyingUnitsPerShare = 1e18;
    minCheckedStrategyTotalDebt = 1000e18;
}
```

SHB.12.2: ETHVault.sol

```
function initialize(
    address _accessControlProxy,
    address _treasury,
    address _exchangeManager,
    address _priceProvider
) public initializer {
    _initAccessControl(_accessControlProxy);

    treasury = _treasury;
    exchangeManager = _exchangeManager;
    priceProvider = _priceProvider;
    // 1 / 1000e4
    rebaseThreshold = 1;
    // one week
    maxTimestampBetweenTwoReported = 604800;
    underlyingUnitsPerShare = 1e18;
    minCheckedStrategyTotalDebt = 1e17;
}
```

Recommendation:

Given the immutability of the valueInterpreter variable, consider hard-coding its address and employing a multisig wallet to avoid the centralization issue.

Updates

The team has acknowledged the issue, stating that the authority will be transferred to the governance contract.

SHB.13 Avoid Using .transfer() To Transfer Ether

- Severity : LOW
- Status : Acknowledged
- Likelihood : 1
- Impact : 2

Description:

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

Files Affected:

SHB.13.1: Treasury.sol

```
59 function withdrawETH(address payable _destination, uint256 _amount)
60     external
61     payable
62     nonReentrant
63     onlyRole(BocRoles.GOV_ROLE)
64 {
65     _destination.transfer(_amount);
66 }
```

SHB.13.2: VaultBuffer.sol

```
110 function transferCashToVault(address[] memory _assets, uint256[] memory
    ↪ _amounts)
111     external
112     override
113     onlyVault
114 {
115     uint256 _len = _assets.length;
116     for (uint256 i = 0; i < _len; i++) {
```

```
117         uint256 amount = _amounts[i];
118         if (amount > 0) {
119             address asset = _assets[i];
120             if (asset == NativeToken.NATIVE_TOKEN) {
121                 payable(vault).transfer(amount);
122             } else {
123                 IERC20Upgradeable(asset).safeTransfer(vault, amount);
124             }
125         }
126     }
127 }
```

SHB.13.3: ETHBaseStrategy.sol

```
235 function transferTokensToTarget(
236     address _target,
237     address[] memory _assets,
238     uint256[] memory _amounts
239 ) internal {
240     for (uint256 i = 0; i < _assets.length; i++) {
241         uint256 _amount = _amounts[i];
242         if (_amount > 0) {
243             if (_assets[i] == NativeToken.NATIVE_TOKEN) {
244                 payable(_target).transfer(_amount);
245             } else {
246                 IERC20Upgradeable(_assets[i]).safeTransfer(address(_target
                        ↪ ), _amount);
247             }
248         }
249     }
250 }
```

SHB.13.4: OneInchV4Adapter.sol

```
52 uint256 _exchangeAmount = getTokenBalance(_sd.dstToken, address(this)) -
       ↪  _toTokenBefore;
```

```
53 if (_sd.dstToken != NativeToken.NATIVE_TOKEN) {
54     IERC20(_sd.dstToken).safeTransfer(_sd.receiver, _exchangeAmount);
55 } else {
56     payable(_sd.receiver).transfer(_exchangeAmount);
57 }
```

SHB.13.5: ETHVault.sol

```
870 if (_trackedAsset == NativeToken.NATIVE_TOKEN) {
871     _actualAmount = _actualAmount + _amount;
872     payable(msg.sender).transfer(_amount);
873 } else {
```

SHB.13.6: ParaSwapV5Adapter.sol

```
176 uint256 _amount = getTokenBalance(_sd.dstToken, address(this)).sub(
      ↪ _toTokenBefore);
177 _toToken == NativeToken.NATIVE_TOKEN?payable(_sd.receiver).transfer(
      ↪ _amount):IERC20(_toToken).safeTransfer(_sd.receiver, _amount);
```

SHB.13.7: ParaSwapV5Adapter.sol

```
260 uint256 _amount = getTokenBalance(_sd.dstToken, address(this)) -
      ↪ _toTokenBefore;
261 _toToken == NativeToken.NATIVE_TOKEN?payable(_sd.receiver).transfer(
      ↪ _amount):IERC20(_toToken).safeTransfer(_sd.receiver, _amount);
```

Recommendation:

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

Updates

The team has acknowledged the issue, stating that they have considered the recommended method for addressing the issue. However, they have decided that further updates will be made in the near future only if necessary, such as in case of any changes in gas fees.

SHB.14 Approve Race Condition

- Severity : LOW
- Status : Fixed
- Likelihood : 1
- Impact : 2

Description:

The standard ERC20 implementation contains a widely known racing condition in its approve function.

Exploit Scenario:

A spender can witness the token owner broadcast a transaction altering their approval and quickly sign and broadcast a transaction using transferFrom to move the current approved amount from the owner's balance to the spender. If the spender's transaction is validated before the owner's, the spender will be able to get both approval amounts of both transactions.

Files Affected:

SHB.14.1: VaultBuffer.sol

```
254 function approve(address _spender, uint256 _amount) public virtual
        ↪ override returns (bool) {
255     address _owner = _msgSender();
256     _approve(_owner, _spender, _amount);
257     return true;
258 }
```

SHB.14.2: PegToken.sol

```
155 function approve(address _spender, uint256 _amount)
156     public
157     override
158     returns (bool)
```

```
{
    _approve(msg.sender, _spender, _amount);
    return true;
}
```

Recommendation:

We recommend using increaseAllowance and decreaseAllowance functions to modify the approval amount instead of using the approve function to modify it.

Updates

The team resolved the issue by adding a safety check that makes sure the allowance can only change from zero to a value ,or from a value to zero. This prevents overriding the amount directly which can result in the spender taking both approval amounts.

SHB.14.3: PegToken.sol

```
function approve(address _spender, uint256 _amount)
    public
    override
    returns (bool)
{
    require(
        (_amount == 0)  (allowance(msg.sender, _spender) == 0),
        "approve from non-zero to non-zero allowance"
    );
    _approve(msg.sender, _spender, _amount);
    return true;
}
```

SHB.14.4: VaultBuffer.sol

```
function approve(address _spender, uint256 _amount)
    public
    override
    returns (bool)
```

```
{
    require(
        (_amount == 0)  (allowance(msg.sender, _spender) == 0),
        "approve from non-zero to non-zero allowance"
    );
    _approve(msg.sender, _spender, _amount);
    return true;
}
```

SHB.15 The Length And Address Of The _exchangeAdapters Argument Are Not Validated

- Severity : LOW
- Status : Fixed
- Likelihood : 1
- Impact : 2

Description:

Certain functions lack a value safety check, the values of the arguments should be verified to allow only those that comply with the contract's logic.

Exploit Scenario:

The contract's deployer sets the _exchangeAdapters argument to an empty array or the elements of the array are equal to the address(0), implying that the contract will not have any exchange adapters. As a result, the exchange functionality will be unavailable until the governor or delegate adds new exchange adapters using the addExchangeAdapters function.

Files Affected:

SHB.15.1: ExchangeAggregator.sol

```
23 constructor(address[] memory _exchangeAdapters, address
       ↪ _accessControlProxy) {
24     _initAccessControl(_accessControlProxy);
25     __addExchangeAdapters(_exchangeAdapters);
26 }
```

Recommendation:

Consider verifying the _exchangeAdapters argument's length to be different from zero and the addresses to be different from the address(0).

Updates

The team resolved the issue by implementing the recommended solution.

SHB.15.2: ExchangeAggregator.sol

```
23     constructor(address[] memory _exchangeAdapters, address
           ↪ _accessControlProxy) {
24         require(_exchangeAdapters.length > 0, "The length must GT 0");
25         for (uint256 i = 0; i < _exchangeAdapters.length; i++) {
26             //The error message "NNA" represents "The input address need
                   ↪ be non-zero address"
27             require(_exchangeAdapters[i] != address(0), "NNA");
28         }
29
30         // '_accessControlProxy' will be verified in function
               ↪ _initAccessControl
31         _initAccessControl(_accessControlProxy);
32         __addExchangeAdapters(_exchangeAdapters);
33     }
```

SHB.16 Floating Pragma

- Severity : LOW
- Status : Fixed
- Likelihood : 1
- Impact : 1

Description:

The contract makes use of the floating-point pragma. Contracts should be deployed using the same compiler version. Locking the pragma helps ensure that contracts will not be unintentionally deployed using another pragma, which in some cases may be an obsolete version, that may introduce issues to the contract system.

Files Affected:

All Contracts

Recommendation:

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

Updates

The team resolved the issue by locking the pragma version to 0.8.17.

SHB.17 Mismatch between the Code and the Documentation

- Severity : UNDETERMINED
- Status : Acknowledged
- Likelihood : 1
- Impact : -

Description:

The documentation states that the Treasury will benefit users by using buyback to repurchase the BoC governance token. However, there is no functionality in the Treasury contract that guarantees the BoC governance token buyback operation.

Files Affected:

SHB.17.1: Treasury.sol

Recommendation:

Consider adding a mechanism to guarantee the correct use of the Treasury funds.

Updates

The team acknowledged the risk, stating that the BoC Gitbook documentation is currently undergoing a new round of updates, including renewing the latest contract addresses for reference.

4 Best Practices

BP.1 Unused Functions

Description:

The AssetHelpers contract has unused functions which should be removed to reduce the contract's size. The functions are: __getAssetBalances(), __pullPartialAssetBalances, __pullFullAssetBalances and __pushFullAssetBalances.

Files Affected:

BP.1.1: AssetHelpers.sol

```
28 function __getAssetBalances(address _target, address[] memory _assets)
       ↪ internal view returns (uint256[] memory _balances) {
29     _balances = new uint256[](_assets.length);
30     for (uint256 i=0; i < _assets.length; i++) {
31         _balances[i] = IERC20Upgradeable(_assets[i]).balanceOf(_target);
32     }
33
34     return _balances;
35 }
```

BP.1.2: AssetHelpers.sol

```
39 function __pullFullAssetBalances(address _target, address[] memory
       ↪ _assets) internal returns (uint256[] memory _amountsTransferred)
       ↪ {
40     _amountsTransferred = new uint256[](_assets.length);
41     for (uint256 i=0; i < _assets.length; i++) {
42         IERC20Upgradeable assetContract = IERC20Upgradeable(_assets[i]);
43         _amountsTransferred[i] = assetContract.balanceOf(_target);
44         if (_amountsTransferred[i] > 0) {
45             assetContract.safeTransferFrom(_target, address(this),
                   ↪ _amountsTransferred[i]);
```

```
        }
    }

    return _amountsTransferred;
}
```

BP.1.3: AssetHelpers.sol

```
function __pullPartialAssetBalances(
        address _target,
        address[] memory _assets,
        uint256[] memory _amountsToExclude
    ) internal returns (uint256[] memory _amountsTransferred) {
        _amountsTransferred = new uint256[](_assets.length);
        for (uint256 i=0; i < _assets.length; i++) {
            IERC20Upgradeable assetContract = IERC20Upgradeable(_assets[i
                ↪ ]);
            _amountsTransferred[i] = assetContract.balanceOf(_target) -
                ↪ _amountsToExclude[i];
            if (_amountsTransferred[i] > 0) {
                assetContract.safeTransferFrom(_target, address(this),
                    ↪ _amountsTransferred[i]);
            }
        }

        return _amountsTransferred;
    }
```

BP.1.4: AssetHelpers.sol

```
function __pushFullAssetBalances(address _target, address[] memory
    ↪ _assets) internal returns (uint256[] memory _amountsTransferred)
    ↪ {
        _amountsTransferred = new uint256[](_assets.length);
        for (uint256 i=0; i < _assets.length; i++) {
            IERC20Upgradeable assetContract = IERC20Upgradeable(_assets[i
```

```
                ↪ ]);
76              _amountsTransferred[i] = assetContract.balanceOf(address(this
                ↪ ));
77              if (_amountsTransferred[i] > 0) {
78                  assetContract.safeTransfer(_target, _amountsTransferred[i
                        ↪ ]);
79              }
80          }
81
82          return _amountsTransferred;
83      }
```

Status - Fixed

BP.2 Remove Zero Initialization

Description:

In solidity, there is no need to initialize a variable with its default value, this is done automatically after the variable declaration.

Files Affected:

BP.2.1: ExchangeAggregator.sol

```
67  uint256 _exchangeAmount = 0;
```

BP.2.2: ExchangeAggregator.sol

```
98  uint256 _ethValue = 0;
```

BP.2.3: BaseStrategy.sol

```
118 uint256 _totalUsdValue = 0;
```

BP.2.4: VaultAdmin.sol

```
329 uint256 _offset = 0;
```

BP.2.5: Vault.sol

```
35 redeemFeeBps = 0;
```

BP.2.6: Vault.sol

```
138 uint256 _totalValueInVault = 0;
139 uint256 _totalTransferValue = 0;
```

BP.2.7: Vault.sol

```
234 uint256 _actuallyReceivedAmount = 0;
```

BP.2.8: Vault.sol

```
304 uint256 _minProductIndex = 0;
```

BP.2.9: Vault.sol

```
388 uint256 _totalValueInVault = 0;
```

BP.2.10: Vault.sol

```
425 uint256 _transferValue = 0;
426 uint256 _redeemValue = 0;
427 uint256 _vaultValueOfNow = 0;
428 uint256 _vaultValueOfBefore = 0;
```

BP.2.11: Vault.sol

```
470 uint256 _transferAssets = 0;
471 uint256 _old2LendAssets = 0;
```

BP.2.12: Vault.sol

```
512 totalDebtOfBeforeAdjustPosition = 0;
```

BP.2.13: Vault.sol

```
515 redeemAssetsMap[_trackedAsset] = 0;
516 beforeAdjustPositionAssetsMap[_trackedAsset] = 0;
517 transferFromVaultBufferAssetsMap[_trackedAsset] = 0;
```

BP.2.14: Vault.sol

```
548 uint256 _balance = 0;
```

BP.2.15: Vault.sol

```
603 uint256 _totalAssetInVaultAndVaultBuffer = 0;
```

BP.2.16: Vault.sol

```
626 uint256 _mintAmount = 0;
```

BP.2.17: Vault.sol

```
754 uint256 _shareAmount = 0;
```

BP.2.18: Vault.sol

```
1097 uint256 _gain = 0;
1098 uint256 _loss = 0;
```

BP.2.19: Vault.sol

```
1133 uint256 _type = 0;
```

Status - Fixed

BP.3 Rename removeStrategy Function

Description:

The removeStrategy() function removes many strategies on its call, it is recommended to rename it as removeStrategies so that the name is more explicit of the implementation.

Files Affected:

BP.3.1: VaultAdmin.sol

```
239 function removeStrategy(address[] memory _strategies) external
        ↪ isVaultManager {
```

```
240     for (uint256 i = 0; i < _strategies.length; i++) {
241         require(strategySet.contains(_strategies[i]), "Strategy not exist
                ↪ ");
242         _removeStrategy(_strategies[i], false);
243     }
244     emit RemoveStrategies(_strategies);
245 }
```

BP.3.2: ETHVaultAdmin.sol

```
239 function removeStrategy(address[] memory _strategies) external
        ↪ isVaultManager {
240     for (uint256 i = 0; i < _strategies.length; i++) {
241         require(strategySet.contains(_strategies[i]), "Strategy not exist
                ↪ ");
242         _removeStrategy(_strategies[i], false);
243     }
244     emit RemoveStrategies(_strategies);
245 }
```

Status - Fixed

BP.4 Wrong isKeeper Modifier Name

Description:

The isKeeper() modifier checks that msg.sender has a KEEPER_ROLE, VAULT_ROLE, DEFAULT_ADMIN_ROLE, or DELEGATE_ROLE.It is recommended to rename it to reflect the implementation. ex.isKeeperOrVaultOrGovOrDelegate.

Files Affected:

BP.4.1: AccessControlMixin.sol

```
45  modifier isKeeper() {
46         accessControlProxy.checkKeeperOrVaultOrGov(msg.sender);
```

```
47          _;
48      }
```

Status - Fixed

BP.5 Wrong Function Name isVaultOrGov()

Description:

The isVaultOrGov() function checks that the _account argument has a VAULT_ROLE, DEFAULT_ADMIN_ROLE or DELEGATE_ROLE. It's advised to change the function's name to accurately reflect the function's logic, ex. isVaultGovOrDelegate.

Files Affected:

BP.5.1: AccessControlProxy.sol

```
65      function isVaultOrGov(address _account) public view returns (bool) {
66          return hasRole(VAULT_ROLE, _account) || isGovOrDelegate(_account)
                ↪ ;
67      }
```

New function name is isVaultManager.

Status - Fixed

5 Tests

5.1 boc-contract-core

-> Whitelist

✓ Verify: Whitelist lenth is eq 0

✓ Verify: Whitelist can batch add and remove (1170ms)

× Verify: Whitelist can check permission

-> ExchangeAggregator test.

✓ verify□ExchangeAggregator removeExchangeAdapters (15ms)

✓ verify□ExchangeAggregator addExchangeAdapters (39ms)

✓ verify□ExchangeAggregator swap(USDT=>USDC) (1644ms)

✓ verify□ExchangeAggregator batchSwap(USDT,USDC=>DAI) (831ms)

✓ verify□ExchangeAggregator swap(DAI=>ETH) (389ms)

✓ verify□ExchangeAggregator swap(ETH=>USDC) (144ms)

✓ verify□ExchangeAggregator batchSwap(USDC,DAI=>ETH) (271ms)

✓ verify□ExchangeAggregator batchSwap(ETH=>USDC,DAI) (223ms)

-> Harvester Test

✓ Harvester the initialization function cannot be executed twice

✓ Harvester base info check

✓ Harvester should can set profit receiver (181ms)

✓ Harvester should can change sellTo token (101ms)

✓ Harvester call collect should call strategy's harvest (950ms)

-> PegToken Test

✓ PegToken the initialization function cannot be executed twice

✓ PegToken base info check

✓ PegToken all functions should available when unpaused (420ms)

✓ PegToken all functions should unavailable when paused (157ms)

-> Vault

✓ Verify: Vault can add and remove Assets normally (1113ms)

✓ Verify: Vault can add and remove all policies normally (1287ms)

✓ Verify□Vault can be invested normally (1948ms)

✓ Verify□Vault can be invested in other assets normally (4977ms)

✓ Verify□Vault can be lend normally (6895ms)

✓ Verify□new funds deposit to vault (7440ms)

✓ Verify□report by strategy and keeper (902ms)

✓ Verify□burn from strategy (1899ms)

27 passing (1m) , 1 failing

5.2 boc-contract-periphery-eth

-> AaveLendActionMixin test

✓ add collaternal

✓ remove collaternal

✓ borrow

✓ repay

-> ExchangeAggregator test

× Case 0,0: swap 0xEeeeeEeeeEeEeeEeEeEeeEEEeeeeEeeeeeeeEEeE to 0xae78736Cd615f374D3085123A210448E74Fc6393 should be success.

× Case 0,1: swap 0xEeeeeEeeeEeEeeEeEeEeeEEEeeeeEeeeeeeeEEeE to 0xae7ab96520DE3A18E5e111B5EaAb095312D7fE84 should be success.

✓ Case 0,2: swap 0xEeeeeEeeeEeEeeEeEeEeeEEEeeeeEeeeeeeeEEeE to 0xC02aaA39b223FE8D0A0e5C4F27eAD9083C756Cc2 should be success.

× Case 0,3: swap 0xEeeeeEeeeEeEeeEeEeEeeEEEeeeeEeeeeeeeEEeE to 0x7f39C581F595B53c5cb19bD0b3f8dA6c935E2Ca0 should be success.

× Case 0,4: swap 0xEeeeeEeeeEeEeeEeEeEeeEEEeeeeEeeeeeeeEEeE to 0xFe2e637202056d30016725477c5da089Ab0A043A should be success

× Case 1,0: swap 0xae78736Cd615f374D3085123A210448E74Fc6393 to 0xae7ab96520DE3A18E5e111B5EaAb095312D7fE84 should be success.

× Case 1,1: swap 0xae78736Cd615f374D3085123A210448E74Fc6393 to 0xC02aaA39b223FE8D0A0e5C4F27eAD9083C756Cc2 should be success.

× Case 1,2: swap 0xae78736Cd615f374D3085123A210448E74Fc6393 to 0x7f39C581F595B53c5cb19bD0b3f8dA6c935E2Ca0 should be success.

× Case 1,3: swap 0xae78736Cd615f374D3085123A210448E74Fc6393 to 0xFe2e637202056d30016725477c5da089Ab0A043A should be success

× Case 1,4: swap 0xae78736Cd615f374D3085123A210448E74Fc6393 to 0xEeeeeEeeeEeEeeEeEeEeeEEEeeeeEeeeeeeeEEeE should be success.

× Case 2,0: swap 0xae7ab96520DE3A18E5e111B5EaAb095312D7fE84 to 0xC02aaA39b223FE8D0A0e5C4F27eAD9083C756Cc2 should be success.

× Case 2,1: swap 0xae7ab96520DE3A18E5e111B5EaAb095312D7fE84 to 0x7f39C581F595B53c5cb19bD0b3f8dA6c935E2Ca0 should be success.

× Case 2,2: swap 0xae7ab96520DE3A18E5e111B5EaAb095312D7fE84 to 0xFe2e637202056d30016725477c5da089Ab0A043A should be success.

× Case 2,3: swap 0xae7ab96520DE3A18E5e111B5EaAb095312D7fE84 to 0xEeeeeEeeeEeEeeEeEeEeeEEEeeeeEeeeeeeeEEeE should be success.

× Case 2,4: swap 0xae7ab96520DE3A18E5e111B5EaAb095312D7fE84 to 0xae78736Cd615f374D3085123A210448E74Fc6393 should be success

× Case 3,0: swap 0xC02aaA39b223FE8D0A0e5C4F27eAD9083C756Cc2 to 0x7f39C581F595B53c5cb19bD0b3f8dA6c935E2Ca0 should be success.

× Case 3,1: swap 0xC02aaA39b223FE8D0A0e5C4F27eAD9083C756Cc2 to 0xFe2e637202056d30016725477c5da089Ab0A043A should be success

× Case 3,2: swap 0xC02aaA39b223FE8D0A0e5C4F27eAD9083C756Cc2 to 0xEeeeeEeeeEeEeeEeEeEeeEEEeeeeEeeeeeeeEEeE should be success.

× Case 3,3: swap 0xC02aaA39b223FE8D0A0e5C4F27eAD9083C756Cc2 to 0xae78736Cd615f374D3085123A210448E74Fc6393 should be success.

× Case 3,4: swap 0xC02aaA39b223FE8D0A0e5C4F27eAD9083C756Cc2 to 0xae7ab96520DE3A18E5e111B5EaAb095312D7fE84 should be success.

× Case 4,0: swap 0x7f39C581F595B53c5cb19bD0b3f8dA6c935E2Ca0 to 0xFe2e637202056d30016725477c5da089Ab0A043A should be success.

× Case 4,1: swap 0x7f39C581F595B53c5cb19bD0b3f8dA6c935E2Ca0 to 0xEeeeeEeeeEeEeeEeEeEeeEEEeeeeEeeeeeeeEEeE should be success.

× Case 4,2: swap 0x7f39C581F595B53c5cb19bD0b3f8dA6c935E2Ca0 to 0xae78736Cd615f374D3085123A210448E74Fc6393 should be succes

× Case 4,3: swap 0x7f39C581F595B53c5cb19bD0b3f8dA6c935E2Ca0 to 0xae7ab96520DE3A18E5e111B5EaAb095312D7fE84 should be success.

× Case 4,4: swap 0x7f39C581F595B53c5cb19bD0b3f8dA6c935E2Ca0 to 0xC02aaA39b223FE8D0A0e5C4F27eAD9083C756Cc2 should be success.

× Case 5,0: swap 0xFe2e637202056d30016725477c5da089Ab0A043A to 0xEeeeeEeeeEeEeeEeEeEeeEEEeeeeEeeeeeeeEEeE should be success

× Case 5,1: swap 0xFe2e637202056d30016725477c5da089Ab0A043A to 0xae78736Cd615f374D3085123A210448E74Fc6393 should be success.

× Case 5,2: swap 0xFe2e637202056d30016725477c5da089Ab0A043A to 0xae7ab96520DE3A18E5e111B5EaAb095312D7fE84 should be success

× Case 5,4: swap 0xFe2e637202056d30016725477c5da089Ab0A043A to 0x7f39C581F595B53c5cb19bD0b3f8dA6c935E2Ca0 should be success.

-> AaveWETHstETHStrategy Strategy Checker

× "before all" hook for "[strategy name should match the file name]"

-> AuraREthWEthStrategy Strategy Checker

× "before all" hook for "[strategy name should match the file name]"

-> AuraWstETHWETHStrategy Strategy Checker

× "before all" hook for "[strategy name should match the file name]"

-> BalancerREthWEthStrategy Strategy Checker

× "before all" hook for "[strategy name should match the file name]"

-> BalancerWstEthWEthStrategy Strategy Checker

× "before all" hook for "[strategy name should match the file name]"

-> ConvexrETHwstETHStrategy Strategy Checker

× "before all" hook for "[strategy name should match the file name]"

-> ConvexSETHStrategy Strategy Checker

× "before all" hook for "[strategy name should match the file name]"

-> ConvexStETHStrategy Strategy Checker

× "before all" hook for "[strategy name should match the file name]"

-> DForceRevolvingLoanETHStrategy Strategy Checker

× "before all" hook for "[strategy name should match the file name]"

-> EulerRevolvingLoanWETHStrategy Strategy Checker

✓ [strategy name should match the file name]

✓ [strategy version should not be empty]

✓ [strategy outputsInfo should not be empty]

✓ [wants info should be same with wants]

✓ [50ETH < Third Pool Assets < 10,000,000ETH]

✓ [estimatedTotalAssets = transferred tokens value]

× [estimatedTotalAssets = transferred tokens value

12 passing (2m) , 38 failing

5.3 Coverage

The code coverage results were obtained by running npx hardhat coverage in the core-contract-core project. We found the following results :

- Statements Coverage : 60.75%
- Branches Coverage : 37.38%
- Functions Coverage : 54.99%
- Lines Coverage : 60.80%

5.4 Conclusion

The project offers a testing mechanism to improve the correctness of smart contracts; nonetheless, a number of tests have failed; therefore, we advise on resolving this issue. In addition, the test coverage percentage is low; it must be increased to cover all functionalities and test cases in order to guarantee the integrity of the code and the functionality of the protocol.

6 Conclusion

In this audit, we examined the design and implementation of Bank Of Chain contract and discovered several issues of varying severity. Bank Of Chain team addressed 9 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 Bank Of Chain Team to maintain a high level of vigilance and to keep those findings in mind in order to avoid any future complications.

7 Scope Files

7.1 Audit

Files	MD5 Hash
boc-contract-core/contracts/Verification.sol	6d6954770123bfaf055d32926aa29307
boc-contract-core/contracts/treasury/Treasury.sol	9381b94ea0cba81ab1a7ee9d3963694e
boc-contract-core/contracts/exchanges/IExchangeAdapter.sol	6313dc1f0fe6e08c70f499a582e2b2c3
boc-contract-core/contracts/exchanges/IExchangeAggregator.sol	e6ff1baa244d671f7216a0c3ee3a75f6
boc-contract-core/contracts/exchanges/ExchangeAggregator.sol	10e0e025b456fea2c6df61ac2e5fc5a9
boc-contract-core/contracts/exchanges/adapters/TestAdapter.sol	77d35631ea4f08d25cda97f02b861fb3
boc-contract-core/contracts/library/NativeToken.sol	9faf317d68d39b2b4ff74ae8cfb01b6d
boc-contract-core/contracts/library/LibLinkedList.sol	5cae001c87b8a8398e5a02d3efc68027
boc-contract-core/contracts/library/RevertReasonParser.sol	436cf7364bdbd5f770966067994fa4dd
boc-contract-core/contracts/library/BocRoles.sol	5a04595f1124997895dd9ad6df8735ec
boc-contract-core/contracts/library/SafeUint128.sol	e69574438dc99c79216af4e71eed0857
boc-contract-core/contracts/library/WadRayMath.sol	45d618a29ff989b5589f79460ac83485
boc-contract-core/contracts/library/IterableUintMap.sol	0415e823fe25606a6f9b52fdaedaf8d3
boc-contract-core/contracts/library/StableMath.sol	62e976e70b9b031975d418d85e716338
boc-contract-core/contracts/library/LibRankedList.sol	3089bb621672454eb8d61e1e0c6bbd21

boc-contract-core/contracts/library/IterableIntMap.sol	268dbcbd47374bc99acfbe94b6f65449
boc-contract-core/contracts/price-feeds/ValueInterpreter.sol	b9d991ec5dbaf82c478bed6b6b851ec8
boc-contract-core/contracts/price-feeds/IValueInterpreter.sol	783dd2fdfa8af3030c29657cbaaaa0f3
boc-contract-core/contracts/price-feeds/derivatives/AggregatedDerivativePriceFeed.sol	9572ccbd7411a31941f16efa6b767956
boc-contract-core/contracts/price-feeds/derivatives/IDerivativePriceFeed.sol	622b788c30d002eafa514deba3c456b1
boc-contract-core/contracts/price-feeds/derivatives/IAggregatedDerivativePriceFeed.sol	6337b7cc74366c2bb318832d1b45e901
boc-contract-core/contracts/price-feeds/primitives/IPrimitivePriceFeed.sol	2b73d61b81e5c07f7ac885e1cd80f216
boc-contract-core/contracts/price-feeds/primitives/ChainlinkPriceFeed.sol	79732404bcbd67d3a11fa2cd2abbd4c7
boc-contract-core/contracts/mock/MockStrategy.sol	c0e77807c13d644eb587259f08b376a2
boc-contract-core/contracts/mock/IEREC20Mint.sol	4a721f3274d4cde6a4b00f22227f35f7
boc-contract-core/contracts/mock/MockVault.sol	aee788a881e8db769b6cada6014668cb
boc-contract-core/contracts/mock/Mock3rdPool.sol	56cda336ac53cf344a5d32b03783e7bd
boc-contract-core/contracts/mock/Mock3CoinStrategy.sol	b14b2a9d41a2df8194629d7ec97d5ddd
boc-contract-core/contracts/vault/VaultStorage.sol	73bfff2cfdf24106accf46f27cbd73e8
boc-contract-core/contracts/vault/IVaultBuffer.sol	84b3f9d129c3b7db8d27ad77214223eb
boc-contract-core/contracts/vault/Vault.sol	988a4a1c95fd72f67a8e03cfdac9c6cc
boc-contract-core/contracts/vault/IVault.sol	e92fc1fdb8aff3d8975fa74c0e939b49

boc-contract-core/contracts/vault/VaultBuffer.sol	c75603de68818a1a91677487c1ec67fa
boc-contract-core/contracts/vault/VaultAdmin.sol	7dd71dd9f0275767414f4c995cc8afdd
boc-contract-core/contracts/interface/IBasicToken.sol	c618ebba778fed0b0e2ef15926d88c5a
boc-contract-core/contracts/util/Helpers.sol	e4ac589aeae1437183b41f19a7d1aab9
boc-contract-core/contracts/token/PegToken.sol	9cfb098e448b31bec3e0000380ebb929
boc-contract-core/contracts/token/IPegToken.sol	4dfcd2fa45d0a21d3e69ff4aee071052
boc-contract-core/contracts/token/WrappedPegToken.sol	01a6d216ecd1c477db44c78e8adcc447
boc-contract-core/contracts/harvester/Dripper.sol	84ad78bdb9a3f4228b55182b54a4e595
boc-contract-core/contracts/harvester/IHarvester.sol	e6350cce64872316bf3063ef3e910948
boc-contract-core/contracts/harvester/Harvester.sol	90741670c8a52e4b51f8befeab5537ca
boc-contract-core/contracts/strategy/IStrategy.sol	079c3556ebfbfc84ac535ee00684a5bd
boc-contract-core/contracts/strategy/BaseClaimableStrategy.sol	e5b119d0b739c8c660143c1f48055b39
boc-contract-core/contracts/strategy/BaseStrategy.sol	94b74969b8ef52b426d24e6745084247
boc-contract-core/contracts/access-control/AccessControlProxy.sol	1aa15ec3ae597c1440c0aa1bdf890361
boc-contract-core/contracts/access-control/AccessControlMixin.sol	ef0ed16f62806a37ab689bbe09c20570
boc-contract-core/contracts/access-control/IAccessControlProxy.sol	1c010c7b2f654049dcc80d23a8a74c9f
boc-contract-periphery-eth/contracts/DependenciesPlaceholder.sol	25883ba41022a432cc0b8a07631b661d

boc-contract-periphery-eth/contracts/exchanges/utils/ParaSwapV5ActionsMixin.sol	d7830992d57403d25f76ecc7a5806d35
boc-contract-periphery-eth/contracts/exchanges/utils/ExchangeHelpers.sol	e6eb465e1038452741ed09f0ac15685b
boc-contract-periphery-eth/contracts/exchanges/adapters/OneInchV4Adapter.sol	bc752aee2fef9ea400411554eff74496
boc-contract-periphery-eth/contracts/exchanges/adapters/ParaSwapV5Adapter.sol	1c0eab82d01369021d17577356599c84
boc-contract-periphery-eth/contracts/usd/strategies/convex/ConvexCompoundStrategy.sol	a8a68353f17bb0511dbb838220685bff
boc-contract-periphery-eth/contracts/usd/strategies/convex/ConvexUsdtStrategy.sol	3ff80b6cfb81b894695e09e7ee9a9e7d
boc-contract-periphery-eth/contracts/usd/strategies/convex/ConvexAaveStrategy.sol	6389b2c8f16cc5fc26d5f31be86fd9f9
boc-contract-periphery-eth/contracts/usd/strategies/convex/ConvexSaaveStrategy.sol	515c7629ec9f10cd24217de3343b39bc
boc-contract-periphery-eth/contracts/usd/strategies/convex/ConvexPaxStrategy.sol	b9d9f8d4c91db9e67d57e86d7006cb85
boc-contract-periphery-eth/contracts/usd/strategies/convex/ConvexBaseStrategy.sol	d19c4cd818a4b0d0911a70f6e220dbca
boc-contract-periphery-eth/contracts/usd/strategies/convex/ConvexSusdStrategy.sol	349d17222b4b215be3c0032f6225ec7b
boc-contract-periphery-eth/contracts/usd/strategies/convex/Convex3CrvStrategy.sol	2233b7e51cc177ff30b656488a0baff6
boc-contract-periphery-eth/contracts/usd/strategies/convex/ib/ConvexIBUsdtStrategy.sol	c79c5d39e9e727db6d521c017fc8005e
boc-contract-periphery-eth/contracts/usd/strategies/convex/meta/ConvexMetaPoolStrategy.sol	db7572ec7f7ff3151d4a1dc7d52e9e83

boc-contract-periphery-eth/contracts/usd/strategies/convex/ib-usdc/ConvexIBUsdcStrategy.sol	87ee9edba15d9477387e6cd0bb994c15
boc-contract-periphery-eth/contracts/usd/strategies/uniswapv3/UniswapV3Strategy.sol	361c6712a7373d572557e6ef1c812cf5
boc-contract-periphery-eth/contracts/usd/strategies/yearn/earn/YearnEarnStrategy.sol	1e9c634a27a62b4fd4081c75ab594e89
boc-contract-periphery-eth/contracts/usd/strategies/dodo/DodoV1Strategy.sol	86f8687cb86a9718f6cb24907e2447a9
boc-contract-periphery-eth/contracts/usd/strategies/dodo/DodoStrategy.sol	3b83f8b102066551547861de5f0b8e61
boc-contract-periphery-eth/contracts/usd/strategies/aura/Aura3PoolStrategy.sol	4004536e796b8f5fec316e7428947399
boc-contract-periphery-eth/contracts/usd/strategies/sushi/kashi/stake/SushiKashiStakeStrategy.sol	18aa25f03252b71219b60db3762f6ecd
boc-contract-periphery-eth/contracts/usd/strategies/stargate/StargateSingleStrategy.sol	60025c8e98fd1b832214d161db9d8811
boc-contract-periphery-eth/contracts/usd/strategies/dforce/DForceLendStrategy.sol	7061e51d5350cc90555bb801d6ee0df9
boc-contract-periphery-eth/contracts/usd/enums/ProtocolEnum.sol	be86d6cb3d6ba78f60761688e2bed58d
boc-contract-periphery-eth/contracts/mock/MockUniswapV3Router.sol	13961abed7941a844329477df5247533
boc-contract-periphery-eth/contracts/utils/AssetHelpers.sol	90defde894e1363d955d447b84498644
boc-contract-periphery-eth/contracts/utils/HarvestHelper.sol	610c49ff1b7e7f01164a43b316ef4a13
boc-contract-periphery-eth/contracts/utils/actions/UniswapV3ActionsMixin.sol	80597fe21c251ad2ef1d0e282fdddb53

boc-contract-periphery-eth/contracts/utils/actions/UniswapV2LiquidityActionsMixin.sol	1798a2230075f36cbd0fd135e95ec804
boc-contract-periphery-eth/contracts/utils/actions/DodoPoolActionsMixin.sol	2582598a526765ae9c6f8fa8c037d99f
boc-contract-periphery-eth/contracts/utils/actions/UniswapV3LiquidityActionsMixin.sol	db218626bc7a2b19a7951437fef3fc8d
boc-contract-periphery-eth/contracts/utils/actions/DodoPoolV1ActionsMixin.sol	de0fb09cb74acbe3d52e1bb7b71028e4
boc-contract-periphery-eth/contracts/utils/actions/UniswapV2ActionsMixin.sol	de0a8b386d21d85829e3af7342ddb3a2
boc-contract-periphery-eth/contracts/external/balancer/IBalancerHelper.sol	0f339cf3d7a4b62b2e1a26d54b867744
boc-contract-periphery-eth/contracts/external/balancer/IBalancerMinter.sol	6efb78dd417c434c30e15d2cf2cf369b
boc-contract-periphery-eth/contracts/external/balancer/IBalancerVault.sol	eac0401a56b7eb56babdf3e77b0a7ebd
boc-contract-periphery-eth/contracts/external/balancer/IAsset.sol	99e5944fbcfe01f5ec5531243c74d813
boc-contract-periphery-eth/contracts/external/balancer/IStakingLiquidityGauge.sol	e86cac75cc43dd1635ca20643842ddab
boc-contract-periphery-eth/contracts/external/convex/IConvexStrategy.sol	aac9ee5ead40da77198b9dc136a6e5ed
boc-contract-periphery-eth/contracts/external/convex/IConvexReward.sol	ed8495b017710e24351deb787d3a13db
boc-contract-periphery-eth/contracts/external/convex/IConvexDusdPoolToken.sol	7ecaf0d295410badcb7dfe7f8fd99322
boc-contract-periphery-eth/contracts/external/convex/IConvex.sol	10e0e683dbdc5cdb3f37893d112d0fc2

boc-contract-periphery-eth/contracts/external/synthetix/ISynthetixProxyERC20.sol	b2b80d4d3ff9ef179a70230043ad39b6
boc-contract-periphery-eth/contracts/external/synthetix/IExchanger.sol	cae93ca2cccae8918a2a463771b06d18
boc-contract-periphery-eth/contracts/external/synthetix/ISynthetix.sol	2f45a47e5c4cc980c824697481654775
boc-contract-periphery-eth/contracts/external/synthetix/IAddressResolver.sol	dc47152fdae727f90e46f26ffebb2604
boc-contract-periphery-eth/contracts/external/synthetix/ISynth.sol	8b22cbe59d34c40459972db4335b04f9
boc-contract-periphery-eth/contracts/external/synthetix/ISynthetixAddressResolver.sol	fe9481422bd8f9cba50c9fcc21cbb5b4
boc-contract-periphery-eth/contracts/external/synthetix/IReadProxy.sol	5b7bafc9a1b3a80eb60b05c6a21aa85c
boc-contract-periphery-eth/contracts/external/synthetix/ISystemStatus.sol	87472b6a5073ebac5f9b2a86c4413ce7
boc-contract-periphery-eth/contracts/external/synthetix/ISynthetixExchangeRates.sol	360a38978317fac1bb9cf22948d50b60
boc-contract-periphery-eth/contracts/external/synthetix/IExchangeRates.sol	f9b060bebc483715824ef3ec481b9504
boc-contract-periphery-eth/contracts/external/synthetix/IExchangeState.sol	2ca150c489ac38c797470b84bb6531ae
boc-contract-periphery-eth/contracts/external/synthetix/ISynthetixExchanger.sol	e16cb01ccde605db251d9f9e7ca7dfb0
boc-contract-periphery-eth/contracts/external/synthetix/ISystemSettings.sol	b3aa3a5ed55439fdafc9c04716666996
boc-contract-periphery-eth/contracts/external/synthetix/ISynthetixSynth.sol	b1339899ba541781e811caeef1fa58de

boc-contract-periphery-eth/contracts/external/rari/IRariFundManager.sol	66b6e9d1c5f48765f43d1a73c511a35c
boc-contract-periphery-eth/contracts/external/cream/IPriceOracle.sol	62973ca31991f1e3d8f12b7968eec5e2
boc-contract-periphery-eth/contracts/external/cream/Comptroller.sol	d7524d20c00e043d41ce92e20b9484b7
boc-contract-periphery-eth/contracts/external/cream/CTokenInterface.sol	7cc277cc9255190521d8d02d9e8f963f
boc-contract-periphery-eth/contracts/external/cream/Exponential.sol	995ec2a1dbcf56f44b6d113b3eb04eea
boc-contract-periphery-eth/contracts/external/cream/CarefulMath.sol	10f6170ada167c3bfba41cd4cc0a680a
boc-contract-periphery-eth/contracts/external/cream/IInterestRateModel.sol	7af21cb896b5e30354d57d056331df31
boc-contract-periphery-eth/contracts/external/yearn/IYearnVaultV2.sol	d2150413c3efe4630e94ac430b6b9b85
boc-contract-periphery-eth/contracts/external/yearn/IYearnVault.sol	a7b3185b48472c76ab77220b41cc83ce
boc-contract-periphery-eth/contracts/external/yearn/IYearnVaultV2Registry.sol	c278e46c7bd13ad11315ed9348d97a97
boc-contract-periphery-eth/contracts/external/yearn/IYearnStrategyV2.sol	a2aa28698efb88b62d790bf9b99b2c27
boc-contract-periphery-eth/contracts/external/lido/IWstETH.sol	736d579e619f8f8b8a570fb76128b7cc
boc-contract-periphery-eth/contracts/external/curve/ICurveFi.sol	30e30097e62641fef4409316fcc451a2
boc-contract-periphery-eth/contracts/external/curve/ICurveLiquidityCustomPool.sol	9a18150c2a53c93e4e1aa3cdb1ff4d6d

boc-contract-periphery-eth/contracts/external/curve/ICurveStableSwapMim.sol	44c871f2388c025514936071129f88d6
boc-contract-periphery-eth/contracts/external/curve/ICurveAddressProvider.sol	27706b6142ef7fa4c08159573585f1ba
boc-contract-periphery-eth/contracts/external/curve/ICurveStableSwap3Crv.sol	4894bc99806e5fd4f3db791a23562145
boc-contract-periphery-eth/contracts/external/curve/ICurveLiquidityPoolPayable.sol	4a986306e56397f564aaff50bea84a3a
boc-contract-periphery-eth/contracts/external/curve/ICurveRegistry.sol	d286ad6f323413d7ee3247162e95685f
boc-contract-periphery-eth/contracts/external/curve/ICurveStableSwapDusd.sol	96760ae7baef945c62ef4042433a5591
boc-contract-periphery-eth/contracts/external/curve/ICurveStableSwapUsdn.sol	ad610561fa96d12b522d2749b10f3cc1
boc-contract-periphery-eth/contracts/external/curve/ICurveLiquidityGaugeToken.sol	03ca5fcc47fd88e10377f284c1e2caad
boc-contract-periphery-eth/contracts/external/curve/ICurveStableSwapLusd.sol	4a07865f2efaa512d5d1c5d274fa3418
boc-contract-periphery-eth/contracts/external/curve/ICurveStableSwapEurs.sol	e29b711e9b40338e0d37d9eb903f8430
boc-contract-periphery-eth/contracts/external/curve/ICurveMini.sol	be2914250090508e37d147af40f3b6b3
boc-contract-periphery-eth/contracts/external/curve/ICurveLiquidityPool.sol	64915f99c89ca771d5dfe285fd0ee148
boc-contract-periphery-eth/contracts/external/dodo/DodoStakePool.sol	36f324fae39fc43ea0666b56ad75f6d3
boc-contract-periphery-eth/contracts/external/dodo/DodoVaultV1.sol	167dcbd7f68d91118d8c892326df7d03

boc-contract-periphery-eth/contracts/external/dodo/DodoVault.sol	2f2b1cef4b7f91aa9609bd6baeb6533f
boc-contract-periphery-eth/contracts/external/dodo/DodoStakePoolV1.sol	a7eb462bd7d015581b631163a9707184
boc-contract-periphery-eth/contracts/external/aura/IAuraBooster.sol	d50621ce16b4c6a0f65e210eb4dfdc0d
boc-contract-periphery-eth/contracts/external/aura/IRewardPool.sol	90fb1548332439d8ec901feed54a75cd
boc-contract-periphery-eth/contracts/external/uniswap/IUniswapV2Factory.sol	55493e606c0717555641114da1a8996b
boc-contract-periphery-eth/contracts/external/uniswap/IUniswapV3SwapRouter.sol	73d9b655e478b9ced33bfd8425667790
boc-contract-periphery-eth/contracts/external/uniswap/IUniswapV2Pair.sol	22b7856b8d23db604532447d882225db
boc-contract-periphery-eth/contracts/external/uniswap/IQuoter.sol	fe8b4279fc6b913c2b33c72db1ed3eb1
boc-contract-periphery-eth/contracts/external/uniswap/IUniswapV2Router2.sol	7fd44b1cada5466f5fd2a1cad9d8f0aa
boc-contract-periphery-eth/contracts/external/uniswap/IUniswapV2.sol	f291ba2f427bcd7c4a649cbd96973c22
boc-contract-periphery-eth/contracts/external/uniswap/IUniswapV3.sol	8c10dab07e54d07a9ce18b3310c00452
boc-contract-periphery-eth/contracts/external/oneinch/IOneInchV4.sol	f667bab31ddd2a67eb6e63773a23c12a
boc-contract-periphery-eth/contracts/external/compound/ICToken.sol	4a4cb128d1bee7164330a588d08ec2af
boc-contract-periphery-eth/contracts/external/uniswapV3/IPoolInitializer.sol	09bd571a2bbf2791ae2d8c150bf4a44b

boc-contract-periphery-eth/contracts/external/uniswapV3/IERC721Permit.sol	9d11d193e09a47b24130016712088fe8
boc-contract-periphery-eth/contracts/external/uniswapV3/IPeripheryImmutableState.sol	81067dfc301123e3d9ad3ebec1fca9dc
boc-contract-periphery-eth/contracts/external/uniswapV3/IPeripheryPayments.sol	764ad43b44b1cde24a42935ca05374a3
boc-contract-periphery-eth/contracts/external/uniswapV3/INonfungiblePositionManager.sol	5306acf050494e0900305a876eb8e86f
boc-contract-periphery-eth/contracts/external/uniswapV3/libraries/PoolAddress.sol	4286baec182bca8e07a1044ee4d882d4
boc-contract-periphery-eth/contracts/external/uniswapV3/libraries/PositionValue.sol	2795c5d4263e0ca5b7a4073405cefc07
boc-contract-periphery-eth/contracts/external/uniswapV3/libraries/UniswapV3FullMath.sol	54b4bb06e94f2634d749b23e04c214dd
boc-contract-periphery-eth/contracts/external/uniswapV3/libraries/PositionKey.sol	d60458c4593e0d6e10ffce7f2408eb7e
boc-contract-periphery-eth/contracts/external/uniswapV3/libraries/LiquidityAmounts.sol	45ff58492b4740d8e924388d16069057
boc-contract-periphery-eth/contracts/external/sushi/IMasterChefV2.sol	49b199b25cf9b2b0f99e498522356067
boc-contract-periphery-eth/contracts/external/sushi/Sushi.sol	4ce78b819a5ce32156185baee49f1b25
boc-contract-periphery-eth/contracts/external/sushi/IMasterChef.sol	384c43b74f6dec61170c94bcf32a00db
boc-contract-periphery-eth/contracts/external/sushi/kashi/IBentoBoxMinimal.sol	7eb7f94024dc81c0a2a377c3d0eabbbe
boc-contract-periphery-eth/contracts/external/sushi/kashi/IKashiPair.sol	27f85ecdbc39f1a574e0db38311e14c3

boc-contract-periphery-eth/contracts/external/paraswap/IParaswapV5.sol	e65374fb58950698845ce824f519767a
boc-contract-periphery-eth/contracts/external/paraswap/lib/v5/Utils.sol	ce1d225128f473a748510c4f5098b405
boc-contract-periphery-eth/contracts/external/bancor/IBancorNetwork.sol	fc161f2d8ff84ea4243833faa195b843
boc-contract-periphery-eth/contracts/external/bancor/ILiquidityProtection.sol	a8216d2e6e4d10f3e3e93ef0aa5fc141
boc-contract-periphery-eth/contracts/external/bancor/IBancorContractRegistry.sol	680cd8f706c475973fc7429bff872824
boc-contract-periphery-eth/contracts/external/stakewise/IPoolEscrow.sol	739cf7fb2a0659c32027df34794e9ac2
boc-contract-periphery-eth/contracts/external/stakewise/IPoolValidators.sol	5ab42c837c0f5b4b33c48db451cd8aae
boc-contract-periphery-eth/contracts/external/stakewise/IDepositContract.sol	1fb10358a4fab36cb271827595c4b622
boc-contract-periphery-eth/contracts/external/stakewise/IOracles.sol	c052cd293f9773f5585efd2292faeb69
boc-contract-periphery-eth/contracts/external/stakewise/IRewardEthToken.sol	983eafc7f1e512aa1505e56c76e0c5e0
boc-contract-periphery-eth/contracts/external/stakewise/IMerkleDistributor.sol	f7187e08d82b5e124a9928315f8a7b8e
boc-contract-periphery-eth/contracts/external/stakewise/IPool.sol	9d2d40b7e9192d6616be36e39ad24201
boc-contract-periphery-eth/contracts/external/g-uni/IGUniPool.sol	6e442cddaf318fcc1d8784b469f266eb
boc-contract-periphery-eth/contracts/external/weth/IWeth.sol	e2b31bff4fb62a7e9aeabee0504317aa

boc-contract-periphery-eth/contracts/external/stargate/IStargateStakePool.sol	6c7fa63931c11c59203bd48488d3f08d
boc-contract-periphery-eth/contracts/external/stargate/IStargateLiquidityPool.sol	d1c8a6b0e98f160df83e22288d972945
boc-contract-periphery-eth/contracts/external/stargate/IStargatePool.sol	9eaffb37a6d63bc5ccf45cf5d6be5cad
boc-contract-periphery-eth/contracts/external/rocketpool/RocketDepositPoolInterface.sol	021778ed13d2424abdc7d1aa87d878e5
boc-contract-periphery-eth/contracts/external/rocketpool/RocketTokenRETHInterface.sol	c637d4838d9e79bf51c0a895d13ecb54
boc-contract-periphery-eth/contracts/external/dforce/DFiToken.sol	24136b0dafbe1c326b17960c3245d4ca
boc-contract-periphery-eth/contracts/external/dforce/IRewardDistributorV3.sol	2eb63c80269770d63b10bf2d7ec64e12
boc-contract-periphery-eth/contracts/eth/mock/Mock3rdEthPool.sol	d36211fc352b112daeee88ebd9c76cc6
boc-contract-periphery-eth/contracts/eth/mock/MockEthStrategy.sol	f5becfeaeb2b9ae859c4cc472f227b09
boc-contract-periphery-eth/contracts/eth/mock/Mock3CoinStrategy.sol	e7855c186f4e5276ec9da7bddd518ed9
boc-contract-periphery-eth/contracts/eth/mock/MockETHVault.sol	12d51279f8855f20c62652e2b2f5bebd
boc-contract-periphery-eth/contracts/eth/vault/ETHVaultStorage.sol	254eed6ecaf758656a4aa7c22ddfdfeb
boc-contract-periphery-eth/contracts/eth/vault/ETHVaultAdmin.sol	37da6ecbc403a5e4e32661cfcb74c57c
boc-contract-periphery-eth/contracts/eth/vault/IETHVault.sol	5cf3c8411a1cc8dcdb37da21dae3f337

boc-contract-periphery-eth/contracts/eth/vault/ETHVault.sol	fceeac5ea765526c22756551de2dbca5
boc-contract-periphery-eth/contracts/eth/strategies/ETHBaseStrategy.sol	1faf85b62043e6bef94ad66fd1350e43
boc-contract-periphery-eth/contracts/eth/strategies/ETHBaseClaimableStrategy.sol	78dde7fec0b0852398b16eb9891acab0
boc-contract-periphery-eth/contracts/eth/strategies/IETHStrategy.sol	ee9cacc81ced9ffbd0f10ca51da5a001
boc-contract-periphery-eth/contracts/eth/strategies/convex/ConvexrETHwstETHStrategy.sol	213a77e54eadf7f484b70784d6404faa
boc-contract-periphery-eth/contracts/eth/strategies/convex/ETHConvexBaseStrategy.sol	851762d6cd4d52ab8ec22e04d9df06cd
boc-contract-periphery-eth/contracts/eth/strategies/convex/ConvexStETHStrategy.sol	97759a2e55c4831817e4f777f31e88bf
boc-contract-periphery-eth/contracts/eth/strategies/convex/ConvexSETHStrategy.sol	2d1f7097045f3cdb76080c622720080e
boc-contract-periphery-eth/contracts/eth/strategies/uniswapv3/ETHUniswapV3Strategy.sol	18ee76dc1115844f300b19f4fd11e50b
boc-contract-periphery-eth/contracts/eth/strategies/uniswapv3/ETHUniswapV3BaseStrategy.sol	bb8f19de9f2ac96433902383413de2c6
boc-contract-periphery-eth/contracts/eth/strategies/yearn/v2/YearnV2Strategy.sol	c7b49f50b9580284c6fddd1863111d97
boc-contract-periphery-eth/contracts/eth/strategies/aura/AuraWstETHWETHStrategy.sol	28239ee7179d8515c34681d04ea2cc52
boc-contract-periphery-eth/contracts/eth/strategies/aura/AuraREthWEthStrategy.sol	3d42f1c1884ed0257458cb75a6fb2b8e
boc-contract-periphery-eth/contracts/eth/strategies/uniswap-v2/ETHUniswapV2Strategy.sol	6fcd4ff2bffaf65fde855278dc46286a

boc-contract-periphery-eth/contracts/eth/strategies/stakewise/StakeWiseEthSeth23000Strategy.sol	a7521cd39e56376b74d3f1893272d47a
boc-contract-periphery-eth/contracts/eth/strategies/stakewise/StakeWiseReth2Seth2500Strategy.sol	d428d876974b7baf8215ec1e766065d7
boc-contract-periphery-eth/contracts/eth/oracle/IPriceOracleConsumer.sol	67918994a588c3d9a02cfd5e3e3fc1cc
boc-contract-periphery-eth/contracts/eth/oracle/PriceOracleConsumer.sol	ee5ef1ce64360d740787f25bd3026af8
boc-contract-periphery-eth/contracts/eth/enums/ProtocolEnum.sol	36a060265683804af496d6319074e279

7.2 Re-Audit

Files	MD5 Hash
boc-contract-core/contracts/Verification.sol	bb016f534ab12b7fe8a99bf2f6a18ac3
boc-contract-core/contracts/treasury/Treasury.sol	d02b923152efe5e0b66494e94c629544
boc-contract-core/contracts/exchanges/IExchangeAdapter.sol	6313dc1f0fe6e08c70f499a582e2b2c3
boc-contract-core/contracts/exchanges/IExchangeAggregator.sol	35084506d3c972db4bdbfd6341e90e11
boc-contract-core/contracts/exchanges/ExchangeAggregator.sol	946d74539a55508fbb6de0d4872c2bf4
boc-contract-core/contracts/exchanges/adapters/TestAdapter.sol	a6b2c89ae50518cf38c5dc5c0e3a2246
boc-contract-core/contracts/library/NativeToken.sol	9faf317d68d39b2b4ff74ae8cfb01b6d
boc-contract-core/contracts/library/LibLinkedList.sol	1c471153a0ab385d2319b820ad70468a

boc-contract-core/contracts/library/RevertReasonParser.sol	2f97cd444d169389291d4f7ecc2cdd11
boc-contract-core/contracts/library/BocRoles.sol	f3ac82341629fc23aa39432f66c66c94
boc-contract-core/contracts/library/SafeUint128.sol	2dfa9de8506dd60740114ce73b55e16f
boc-contract-core/contracts/library/WadRayMath.sol	d1e7dd2811972773ef26182725c10b86
boc-contract-core/contracts/library/IterableUintMap.sol	3f97c1b9a3914430adcf284ffdcef386
boc-contract-core/contracts/library/StableMath.sol	24d88448b620c1746e1d6a6082cdbd24
boc-contract-core/contracts/library/LibRankedList.sol	59ca7f109330e868a710b5e7018f0256
boc-contract-core/contracts/library/IterableIntMap.sol	37562c1656e0da31f33be81fa0766c9d
boc-contract-core/contracts/price-feeds/ValueInterpreter.sol	b61313253f838360540443bd80f25fdb
boc-contract-core/contracts/price-feeds/IValueInterpreter.sol	3be65e6bad506cc706b44032f217fc87
boc-contract-core/contracts/price-feeds/derivatives/AggregatedDerivativePriceFeed.sol	d3577df80004715d57563b04380019f1
boc-contract-core/contracts/price-feeds/derivatives/IDerivativePriceFeed.sol	5c48f228946fba3eb00ba02df458a328
boc-contract-core/contracts/price-feeds/derivatives/IAggregatedDerivativePriceFeed.sol	79c617b60eba5d1df47f5276f5e1be50
boc-contract-core/contracts/price-feeds/primitives/IPrimitivePriceFeed.sol	c1b029c591a95e5225ea26d83b5cd16e
boc-contract-core/contracts/price-feeds/primitives/ChainlinkPriceFeed.sol	a2475e75a6cf218f01bd0614dd286adc
boc-contract-core/contracts/mock/MockStrategy.sol	407177a4203bea67cec5ec2816db5b40
boc-contract-core/contracts/mock/IEREC20Mint.sol	3cfafedca889ea308387eac3d2dade84

boc-contract-core/contracts/mock/MockVault.sol	c7a29a72bc65d2168de4c48180a01a8a
boc-contract-core/contracts/mock/Mock3rdPool.sol	bd7680a9f365b81cd8c914b214423ad9
boc-contract-core/contracts/mock/Mock3CoinStrategy.sol	b403ecf318ba84ea8be86413b8f6f2c6
boc-contract-core/contracts/vault/VaultStorage.sol	00aca70621db1704e571c98edf07b917
boc-contract-core/contracts/vault/IVaultBuffer.sol	360a8406e0f297a5291a6e4bad27f9f7
boc-contract-core/contracts/vault/Vault.sol	95437dc1e8fd51d43bd6d4c068affe19
boc-contract-core/contracts/vault/IVault.sol	ffbc805dd12f07360a310ab1dda02de4
boc-contract-core/contracts/vault/VaultBuffer.sol	644e296670ea6dd8ddf5e2bb5d8441c4
boc-contract-core/contracts/vault/VaultAdmin.sol	2832f39bfedd57262d09e9755aeaa7b2
boc-contract-core/contracts/interface/IBasicToken.sol	193eca93db1a7a490eb4578ce8a9d260
boc-contract-core/contracts/util/Helpers.sol	28741b7180a3205f1add085c0840e746
boc-contract-core/contracts/token/PegToken.sol	2d8f9d6ebde5fa020a14217cc0e3053b
boc-contract-core/contracts/token/IPegToken.sol	f10344d3dafa5adb2140109510192976
boc-contract-core/contracts/token/WrappedPegToken.sol	cabaf633cd547e552eda5a19790c66c1
boc-contract-core/contracts/harvester/IHarvester.sol	66295cbae0fe4fc085f65bc1b19238f3
boc-contract-core/contracts/harvester/Harvester.sol	0cdc1911228fc3b21d322158bcb9a91e
boc-contract-core/contracts/strategy/IStrategy.sol	f1be9bd1e82351abacb5038256e98c74
boc-contract-core/contracts/strategy/BaseClaimableStrategy.sol	b0bc0e55da55552f3200a839a26b2de3
boc-contract-core/contracts/strategy/BaseStrategy.sol	327d157d5f1a31a668a382dda8c42577

boc-contract-core/contracts/access-control/AccessControlProxy.sol	bacb6e37db39a15af19a15b28aa4096f
boc-contract-core/contracts/access-control/AccessControlMixin.sol	a7d61fba6f17d1b8ee6c41ecd8f48647
boc-contract-core/contracts/access-control/IAccessControlProxy.sol	c96ebbd21292b669167997e9b5f22aaf
boc-contract-periphery-eth/contracts/DependenciesPlaceholder.sol	7e76feda2c4c6163fda2860822e32e60
boc-contract-periphery-eth/contracts/exchanges/utils/ParaSwapV5ActionsMixin.sol	be8a7062ff91918484f7fca98c01072b
boc-contract-periphery-eth/contracts/exchanges/utils/ExchangeHelpers.sol	f4310767a87ca257dc994f4767c42ba6
boc-contract-periphery-eth/contracts/exchanges/adapters/OneInchV4Adapter.sol	c5a2a02f0a6954884cd53c57e964ff93
boc-contract-periphery-eth/contracts/exchanges/adapters/ParaSwapV5Adapter.sol	0cdb9ab615eaf212218a91ae8cb5bbd2
boc-contract-periphery-eth/contracts/usd/strategies/convex/ConvexCompoundStrategy.sol	9f9d7cfa1b8f45eb7de1138ca4e7dbbf
boc-contract-periphery-eth/contracts/usd/strategies/convex/ConvexUsdtStrategy.sol	8e5f562cd4a8395becda461ba3f43165
boc-contract-periphery-eth/contracts/usd/strategies/convex/ConvexAaveStrategy.sol	896a63ee02318633f982125b81cad5a0
boc-contract-periphery-eth/contracts/usd/strategies/convex/ConvexSaaveStrategy.sol	ef4b7b2a43ffbb7aeb67096ad7158f17
boc-contract-periphery-eth/contracts/usd/strategies/convex/ConvexPaxStrategy.sol	6287b2b4a3025ea5b68450ead5edddf1
boc-contract-periphery-eth/contracts/usd/strategies/convex/ConvexBaseStrategy.sol	f11bdae45f85f2b80c45485f919bcd42

boc-contract-periphery-eth/contracts/usd/strategies/convex/ConvexSusdStrategy.sol	ede7edde933d90568df1ea0a06e4743f
boc-contract-periphery-eth/contracts/usd/strategies/convex/Convex3CrvStrategy.sol	d843db5bf30d0a6c2614d9033fd8b654
boc-contract-periphery-eth/contracts/usd/strategies/convex/ib/ConvexIBUsdtStrategy.sol	ee84c9f8057d1aceb4323940a1d997a7
boc-contract-periphery-eth/contracts/usd/strategies/convex/meta/ConvexMetaPoolStrategy.sol	cfe2cbe7e9cc9e9261ed1e85d07859b8
boc-contract-periphery-eth/contracts/usd/strategies/convex/ib-usdc/ConvexIBUsdcStrategy.sol	767b2a5436a20fb571cfd6af521d431b
boc-contract-periphery-eth/contracts/usd/strategies/uniswapv3/UniswapV3Strategy.sol	a5d7cb73664d3be2eba930214d580de5
boc-contract-periphery-eth/contracts/usd/strategies/yearn/earn/YearnEarnStrategy.sol	d53805c429f35536cc9a51007b657ade
boc-contract-periphery-eth/contracts/usd/strategies/dodo/DodoV1Strategy.sol	1574577dbf0bc0f731159ca46e37eaf3
boc-contract-periphery-eth/contracts/usd/strategies/dodo/DodoStrategy.sol	b893254ac11096bfee447dabf29bc1a5
boc-contract-periphery-eth/contracts/usd/strategies/aura/Aura3PoolStrategy.sol	5f076d4552831ee8044a62c133285fed
boc-contract-periphery-eth/contracts/usd/strategies/sushi/kashi/stake/SushiKashiStakeStrategy.sol	3236145f1a54764973f4e72ea9b2ad86
boc-contract-periphery-eth/contracts/usd/strategies/euler/EulerRevolvingLoanStrategy.sol	bdd3f50bc553c376cb08fc190b7c0575
boc-contract-periphery-eth/contracts/usd/strategies/stargate/StargateSingleStrategy.sol	c43e20c12b8ee1d588c7fc02a727ed6c
boc-contract-periphery-eth/contracts/usd/strategies/dforce/DForceRevolvingLoanStrategy.sol	31bf4fd6c8df78ddbe4d62885952 1b32

boc-contract-periphery-eth/contracts/usd/strategies/dforce/DForceLendStrategy.sol	e7c1912446b86f61f3fd7f56b561306b
boc-contract-periphery-eth/contracts/usd/enums/ProtocolEnum.sol	ea37f2eeb1e68481eb0ef8a9e7970977
boc-contract-periphery-eth/contracts/test/TestAaveLendAction.sol	f80666b7ede1b28990e869b9d91e9d67
boc-contract-periphery-eth/contracts/utils/AssetHelpers.sol	c8b88ae14d729e636b09d2438327bcf1
boc-contract-periphery-eth/contracts/utils/HarvestHelper.sol	067f09d7c332db709d37191a9212601a
boc-contract-periphery-eth/contracts/utils/actions/UniswapV3ActionsMixin.sol	8ffed7d5e33344257148fde268026cd7
boc-contract-periphery-eth/contracts/utils/actions/UniswapV2LiquidityActionsMixin.sol	c614305f724afec5fad86ebf3013f4fc
boc-contract-periphery-eth/contracts/utils/actions/DodoPoolActionsMixin.sol	ddec3407f877ea453784240714f9e47d
boc-contract-periphery-eth/contracts/utils/actions/UniswapV3LiquidityActionsMixin.sol	7be55fa809d6aa001abcc394787737d8
boc-contract-periphery-eth/contracts/utils/actions/DodoPoolV1ActionsMixin.sol	5a386003a713eb7d5621c0208e47682a
boc-contract-periphery-eth/contracts/utils/actions/UniswapV2ActionsMixin.sol	0eac701c16c2f1f8a4dfe678bae208c6
boc-contract-periphery-eth/contracts/external/balancer/IBalancerHelper.sol	0f339cf3d7a4b62b2e1a26d54b867744
boc-contract-periphery-eth/contracts/external/balancer/IBalancerMinter.sol	6efb78dd417c434c30e15d2cf2cf369b
boc-contract-periphery-eth/contracts/external/balancer/IBalancerVault.sol	eac0401a56b7eb56babdf3e77b0a7ebd

boc-contract-periphery-eth/contracts/external/balancer/IAsset.sol	99e5944fbcfe01f5ec5531243c74d813
boc-contract-periphery-eth/contracts/external/balancer/IStakingLiquidityGauge.sol	e86cac75cc43dd1635ca20643842ddab
boc-contract-periphery-eth/contracts/external/convex/IConvexStrategy.sol	aac9ee5ead40da77198b9dc136a6e5ed
boc-contract-periphery-eth/contracts/external/convex/IConvexReward.sol	ed8495b017710e24351deb787d3a13db
boc-contract-periphery-eth/contracts/external/convex/IConvexDusdPoolToken.sol	7ecaf0d295410badcb7dfe7f8fd99322
boc-contract-periphery-eth/contracts/external/convex/IConvex.sol	10e0e683dbdc5cdb3f37893d112d0fc2
boc-contract-periphery-eth/contracts/external/synthetix/ISynthetixProxyERC20.sol	b2b80d4d3ff9ef179a70230043ad39b6
boc-contract-periphery-eth/contracts/external/synthetix/IExchanger.sol	cae93ca2cccae8918a2a463771b06d18
boc-contract-periphery-eth/contracts/external/synthetix/ISynthetix.sol	2f45a47e5c4cc980c824697481654775
boc-contract-periphery-eth/contracts/external/synthetix/IAddressResolver.sol	dc47152fdae727f90e46f26ffebb2604
boc-contract-periphery-eth/contracts/external/synthetix/ISynth.sol	8b22cbe59d34c40459972db4335b04f9
boc-contract-periphery-eth/contracts/external/synthetix/ISynthetixAddressResolver.sol	fe9481422bd8f9cba50c9fcc21cbb5b4
boc-contract-periphery-eth/contracts/external/synthetix/IReadProxy.sol	5b7bafc9a1b3a80eb60b05c6a21aa85c
boc-contract-periphery-eth/contracts/external/synthetix/ISystemStatus.sol	87472b6a5073ebac5f9b2a86c4413ce7

boc-contract-periphery-eth/contracts/external/synthetix/ISynthetixExchangeRates.sol	360a38978317fac1bb9cf22948d50b60
boc-contract-periphery-eth/contracts/external/synthetix/IExchangeRates.sol	f9b060bebc483715824ef3ec481b9504
boc-contract-periphery-eth/contracts/external/synthetix/IExchangeState.sol	2ca150c489ac38c797470b84bb6531ae
boc-contract-periphery-eth/contracts/external/synthetix/ISynthetixExchanger.sol	e16cb01ccde605db251d9f9e7ca7dfb0
boc-contract-periphery-eth/contracts/external/synthetix/ISystemSettings.sol	b3aa3a5ed55439fdafc9c04716666996
boc-contract-periphery-eth/contracts/external/synthetix/ISynthetixSynth.sol	b1339899ba541781e811caeef1fa58de
boc-contract-periphery-eth/contracts/external/rari/IRariFundManager.sol	66b6e9d1c5f48765f43d1a73c511a35c
boc-contract-periphery-eth/contracts/external/cream/IPriceOracle.sol	62973ca31991f1e3d8f12b7968eec5e2
boc-contract-periphery-eth/contracts/external/cream/Comptroller.sol	d7524d20c00e043d41ce92e20b9484b7
boc-contract-periphery-eth/contracts/external/cream/CTokenInterface.sol	7cc277cc9255190521d8d02d9e8f963f
boc-contract-periphery-eth/contracts/external/cream/Exponential.sol	995ec2a1dbcf56f44b6d113b3eb04eea
boc-contract-periphery-eth/contracts/external/cream/CarefulMath.sol	10f6170ada167c3bfba41cd4cc0a680a
boc-contract-periphery-eth/contracts/external/cream/IInterestRateModel.sol	7af21cb896b5e30354d57d056331df31
boc-contract-periphery-eth/contracts/external/yearn/IYearnVaultV2.sol	d2150413c3efe4630e94ac430b6b9b85

boc-contract-periphery-eth/contracts/external/yearn/IYearnVault.sol	a7b3185b48472c76ab77220b41cc83ce
boc-contract-periphery-eth/contracts/external/yearn/IYearnVaultV2Registry.sol	c278e46c7bd13ad11315ed9348d97a97
boc-contract-periphery-eth/contracts/external/yearn/IYearnStrategyV2.sol	a2aa28698efb88b62d790bf9b99b2c27
boc-contract-periphery-eth/contracts/external/lido/IWstETH.sol	736d579e619f8f8b8a570fb76128b7cc
boc-contract-periphery-eth/contracts/external/curve/ICurveFi.sol	30e30097e62641fef4409316fcc451a2
boc-contract-periphery-eth/contracts/external/curve/ICurveLiquidityCustomPool.sol	9a18150c2a53c93e4e1aa3cdb1ff4d6d
boc-contract-periphery-eth/contracts/external/curve/ICurveStableSwapMim.sol	44c871f2388c025514936071129f88d6
boc-contract-periphery-eth/contracts/external/curve/ICurveLiquidityFarmingPool.sol	a09f449ea2348ead2f1c778f97e250a5
boc-contract-periphery-eth/contracts/external/curve/ICurveAddressProvider.sol	27706b6142ef7fa4c08159573585f1ba
boc-contract-periphery-eth/contracts/external/curve/ICurveStableSwap3Crv.sol	4894bc99806e5fd4f3db791a23562145
boc-contract-periphery-eth/contracts/external/curve/ICurveLiquidityPoolPayable.sol	4a986306e56397f564aaff50bea84a3a
boc-contract-periphery-eth/contracts/external/curve/ICurveRegistry.sol	d286ad6f323413d7ee3247162e95685f
boc-contract-periphery-eth/contracts/external/curve/ICurveStableSwapDusd.sol	96760ae7baef945c62ef4042433a5591
boc-contract-periphery-eth/contracts/external/curve/ICurveStableSwapUsdn.sol	ad610561fa96d12b522d2749b10f3cc1

boc-contract-periphery-eth/contracts/external/curve/ICurveLiquidityGaugeToken.sol	03ca5fcc47fd88e10377f284c1e2caad
boc-contract-periphery-eth/contracts/external/curve/ICurveStableSwapLusd.sol	4a07865f2efaa512d5d1c5d274fa3418
boc-contract-periphery-eth/contracts/external/curve/ICurveStableSwapEurs.sol	e29b711e9b40338e0d37d9eb903f8430
boc-contract-periphery-eth/contracts/external/curve/ICurveMini.sol	be2914250090508e37d147af40f3b6b3
boc-contract-periphery-eth/contracts/external/curve/ICurveLiquidityPool.sol	64915f99c89ca771d5dfe285fd0ee148
boc-contract-periphery-eth/contracts/external/dodo/DodoStakePool.sol	36f324fae39fc43ea0666b56ad75f6d3
boc-contract-periphery-eth/contracts/external/dodo/DodoVaultV1.sol	167dcbd7f68d91118d8c892326df7d03
boc-contract-periphery-eth/contracts/external/dodo/DodoVault.sol	2f2b1cef4b7f91aa9609bd6baeb6533f
boc-contract-periphery-eth/contracts/external/dodo/DodoStakePoolV1.sol	a7eb462bd7d015581b631163a9707184
boc-contract-periphery-eth/contracts/external/aura/IAuraBooster.sol	d50621ce16b4c6a0f65e210eb4dfdc0d
boc-contract-periphery-eth/contracts/external/aura/IRewardPool.sol	90fb1548332439d8ec901feed54a75cd
boc-contract-periphery-eth/contracts/external/uniswap/IUniswapV2Factory.sol	55493e606c0717555641114da1a8996b
boc-contract-periphery-eth/contracts/external/uniswap/IUniswapV3SwapRouter.sol	73d9b655e478b9ced33bfd8425667790
boc-contract-periphery-eth/contracts/external/uniswap/IUniswapV2Pair.sol	22b7856b8d23db604532447d882225db

boc-contract-periphery-eth/contracts/external/uniswap/IQuoter.sol	fe8b4279fc6b913c2b33c72db1ed3eb1
boc-contract-periphery-eth/contracts/external/uniswap/IUniswapV2Router2.sol	7fd44b1cada5466f5fd2a1cad9d8f0aa
boc-contract-periphery-eth/contracts/external/uniswap/IUniswapV2.sol	f291ba2f427bcd7c4a649cbd96973c22
boc-contract-periphery-eth/contracts/external/uniswap/IUniswapV3.sol	8c10dab07e54d07a9ce18b3310c00452
boc-contract-periphery-eth/contracts/external/oneinch/IOneInchV4.sol	3ac1231c40603fe48de5d92ffea8d940
boc-contract-periphery-eth/contracts/external/compound/ICToken.sol	4a4cb128d1bee7164330a588d08ec2af
boc-contract-periphery-eth/contracts/external/uniswapV3/IPoolInitializer.sol	09bd571a2bbf2791ae2d8c150bf4a44b
boc-contract-periphery-eth/contracts/external/uniswapV3/IERC721Permit.sol	9d11d193e09a47b24130016712088fe8
boc-contract-periphery-eth/contracts/external/uniswapV3/IPeripheryImmutableState.sol	81067dfc301123e3d9ad3ebec1fca9dc
boc-contract-periphery-eth/contracts/external/uniswapV3/IPeripheryPayments.sol	764ad43b44b1cde24a42935ca05374a3
boc-contract-periphery-eth/contracts/external/uniswapV3/INonfungiblePositionManager.sol	5306acf050494e0900305a876eb8e86f
boc-contract-periphery-eth/contracts/external/uniswapV3/libraries/PoolAddress.sol	4286baec182bca8e07a1044ee4d882d4
boc-contract-periphery-eth/contracts/external/uniswapV3/libraries/PositionValue.sol	2795c5d4263e0ca5b7a4073405cefc07
boc-contract-periphery-eth/contracts/external/uniswapV3/libraries/UniswapV3FullMath.sol	54b4bb06e94f2634d749b23e04c214dd

boc-contract-periphery-eth/contracts/external/uniswapV3/libraries/PositionKey.sol	d60458c4593e0d6e10ffce7f2408eb7e
boc-contract-periphery-eth/contracts/external/uniswapV3/libraries/LiquidityAmounts.sol	45ff58492b4740d8e924388d16069057
boc-contract-periphery-eth/contracts/external/sushi/IMasterChefV2.sol	49b199b25cf9b2b0f99e498522356067
boc-contract-periphery-eth/contracts/external/sushi/Sushi.sol	4ce78b819a5ce32156185baee49f1b25
boc-contract-periphery-eth/contracts/external/sushi/IMasterChef.sol	384c43b74f6dec61170c94bcf32a00db
boc-contract-periphery-eth/contracts/external/sushi/kashi/IBentoBoxMinimal.sol	7eb7f94024dc81c0a2a377c3d0eabbbe
boc-contract-periphery-eth/contracts/external/sushi/kashi/IKashiPair.sol	27f85ecdbc39f1a574e0db38311e14c3
boc-contract-periphery-eth/contracts/external/paraswap/IParaswapV5.sol	dfa22f5bd5a830c1014078bbb5d82352
boc-contract-periphery-eth/contracts/external/paraswap/lib/v5/Utils.sol	19bc6c6e966f5ca63c7feb7ac2d4c449
boc-contract-periphery-eth/contracts/external/bancor/IBancorNetwork.sol	fc161f2d8ff84ea4243833faa195b843
boc-contract-periphery-eth/contracts/external/bancor/ILiquidityProtection.sol	a8216d2e6e4d10f3e3e93ef0aa5fc141
boc-contract-periphery-eth/contracts/external/bancor/IBancorContractRegistry.sol	680cd8f706c475973fc7429bff872824
boc-contract-periphery-eth/contracts/external/stakewise/IPoolEscrow.sol	739cf7fb2a0659c32027df34794e9ac2
boc-contract-periphery-eth/contracts/external/stakewise/IPoolValidators.sol	5ab42c837c0f5b4b33c48db451cd8aae

boc-contract-periphery-eth/contracts/external/stakewise/IDepositContract.sol	1fb10358a4fab36cb271827595c4b622
boc-contract-periphery-eth/contracts/external/stakewise/IOracles.sol	c052cd293f9773f5585efd2292faeb69
boc-contract-periphery-eth/contracts/external/stakewise/IRewardEthToken.sol	983eafc7f1e512aa1505e56c76e0c5e0
boc-contract-periphery-eth/contracts/external/stakewise/IMerkleDistributor.sol	f7187e08d82b5e124a9928315f8a7b8e
boc-contract-periphery-eth/contracts/external/stakewise/IPool.sol	9d2d40b7e9192d6616be36e39ad24201
boc-contract-periphery-eth/contracts/external/g-uni/IGUniPool.sol	6e442cddaf318fcc1d8784b469f266eb
boc-contract-periphery-eth/contracts/external/weth/IWeth.sol	e2b31bff4fb62a7e9aeabee0504317aa
boc-contract-periphery-eth/contracts/external/stargate/IStargateStakePool.sol	6c7fa63931c11c59203bd48488d3f08d
boc-contract-periphery-eth/contracts/external/stargate/IStargateLiquidityPool.sol	d1c8a6b0e98f160df83e22288d972945
boc-contract-periphery-eth/contracts/external/stargate/IStargatePool.sol	9eaffb37a6d63bc5ccf45cf5d6be5cad
boc-contract-periphery-eth/contracts/external/rocketpool/RocketDepositPoolInterface.sol	021778ed13d2424abdc7d1aa87d878e5
boc-contract-periphery-eth/contracts/external/rocketpool/RocketTokenRETHInterface.sol	c637d4838d9e79bf51c0a895d13ecb54
boc-contract-periphery-eth/contracts/external/dforce/DFiToken.sol	f88a7daa763924a3971c2b032e8f6fa1
boc-contract-periphery-eth/contracts/external/dforce/IRewardDistributorV3.sol	2eb63c80269770d63b10bf2d7ec64e12

boc-contract-periphery-eth/contracts/eth/mock/Mock3rdEthPool.sol	43ba463cda91dc82762e43787e511e22
boc-contract-periphery-eth/contracts/eth/mock/Mock3CoinETHStrategy.sol	5c2b95a261c33a87b9b98b0130af69dc
boc-contract-periphery-eth/contracts/eth/mock/MockEthStrategy.sol	ee30a216f7c1d8ed8548683c3161c0aa
boc-contract-periphery-eth/contracts/eth/mock/MockETHVault.sol	dd8ce075302ab8ca9d484f4027f8402e
boc-contract-periphery-eth/contracts/eth/vault/ETHVaultStorage.sol	71f97fdda2642ac6709dc2134b916359
boc-contract-periphery-eth/contracts/eth/vault/ETHVaultAdmin.sol	39a59ea404b0c2d4bee47c94b216e0ad
boc-contract-periphery-eth/contracts/eth/vault/IETHVault.sol	d1e8c7d8a8bd0d560c221ffbc9ed2b23
boc-contract-periphery-eth/contracts/eth/vault/ETHVault.sol	8c1c6ab0fb6a4c67a658bca1f1f6e748
boc-contract-periphery-eth/contracts/eth/strategies/ETHBaseStrategy.sol	7d7eeba9c0cad64cee11c18190e744de
boc-contract-periphery-eth/contracts/eth/strategies/ETHBaseClaimableStrategy.sol	5dcc9c8e20bdcc606f1cb8987e4595f5
boc-contract-periphery-eth/contracts/eth/strategies/IETHStrategy.sol	3767d6c07ee1590f596511faf7ffd3a0
boc-contract-periphery-eth/contracts/eth/strategies/convex/ConvexrETHwstETHStrategy.sol	cd24da3b1ad60b30a88cd39457898982
boc-contract-periphery-eth/contracts/eth/strategies/convex/ETHConvexBaseStrategy.sol	a6af429880ea6a5228d3a2a7e71510d2
boc-contract-periphery-eth/contracts/eth/strategies/convex/ConvexStETHStrategy.sol	a265e541cea4566284503bcb8914ea55

boc-contract-periphery-eth/contracts/eth/strategies/convex/ConvexSETHStrategy.sol	71755fa552ba5091161bf3a8192c0248
boc-contract-periphery-eth/contracts/eth/strategies/uniswapv3/ETHUniswapV3Strategy.sol	946d9f58c75cc3d43b35623dcd5a688c
boc-contract-periphery-eth/contracts/eth/strategies/uniswapv3/ETHUniswapV3BaseStrategy.sol	4172bc22b37c4cd0f41d2a6409e7807b
boc-contract-periphery-eth/contracts/eth/strategies/yearn/v2/YearnV2Strategy.sol	90f13d3886810c510f7f0037424fe548
boc-contract-periphery-eth/contracts/eth/strategies/aura/AuraWstETHWETHStrategy.sol	f3500e41d370d3dfea46bbe016bee035
boc-contract-periphery-eth/contracts/eth/strategies/aura/AuraREthWEthStrategy.sol	6472a58eb5df4d023be9d511bf04255a
boc-contract-periphery-eth/contracts/eth/strategies/uniswap-v2/ETHUniswapV2Strategy.sol	4ab9cbe093d750c921e4373d115c334f
boc-contract-periphery-eth/contracts/eth/strategies/stakewise/StakeWiseEthSeth23000Strategy.sol	66f652722a42075f9e7840001e7804ca
boc-contract-periphery-eth/contracts/eth/strategies/stakewise/StakeWiseReth2Seth2500Strategy.sol	edbf44d026ec4ffc828df035554f746f
boc-contract-periphery-eth/contracts/eth/oracle/IPriceOracleConsumer.sol	3a8f16a947f46be2ebb87b7678b24869
boc-contract-periphery-eth/contracts/eth/oracle/PriceOracleConsumer.sol	a28ae5440a4b3b08cefe73ad6ce80273
boc-contract-periphery-eth/contracts/eth/enums/ProtocolEnum.sol	6bd85aef6d10e8b90b4667be57876f0e

8 Disclaimer

Shellboxes reports should not be construed as "endorsements" or "disapprovals" of particular teams or projects. These reports do not reflect the economics or value of any "product" or "asset" produced by any team or project that engages Shellboxes to do a security evaluation, nor should they be regarded as such. Shellboxes Reports do not provide any warranty or guarantee regarding the absolute bug-free nature of the examined technology, nor do they provide any indication of the technology's proprietors, business model, business or legal compliance. Shellboxes Reports should not be used in any way to decide whether to invest in or take part in a certain project. These reports don't offer any kind of investing advice and shouldn't be used that way. Shellboxes Reports are the result of a thorough auditing process designed to assist our clients in improving the quality of their code while lowering the significant risk posed by blockchain technology. According to Shellboxes, each business and person is in charge of their own due diligence and ongoing security. Shellboxes does not guarantee the security or functionality of the technology we agree to research; instead, our purpose is to assist in limiting the attack vectors and the high degree of variation associated with using new and evolving technologies.

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