HALBORN

DRAFT

Vesper Finance - Synth

Smart Contract Security Audit

Prepared by: **Halborn**

Date of Engagement: **June 6th, 2022 - June 20th, 2022**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	06/15/2022	Omar Alshaeb
0.2	Draft Review	06/20/2022	Gabi Urrutia

CONTACTS

CONTACT	COMPANY	EMAIL
Rob Behnke	Halborn	Rob.Behnke@halborn.com
Steven Walbroehl	Halborn	Steven.Walbroehl@halborn.com
Gabi Urrutia	Halborn	Gabi.Urrutia@halborn.com
Omar Alshaeb	Halborn	Omar.Alshaeb@halborn.com

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

Vesper Finance engaged Halborn to conduct a security audit on their smart contracts beginning on June 6th, 2022 and ending on June 20th, 2022 . The security assessment was scoped to the smart contracts provided to the Halborn team.

1.2 AUDIT SUMMARY

The team at Halborn was provided two weeks for the engagement and assigned a full-time security engineer to audit the security of the smart contract. The security engineer is a blockchain and smart-contract security expert with advanced penetration testing, smart-contract hacking, and deep knowledge of multiple blockchain protocols.

The purpose of this audit is to:

- Ensure that smart contract functions operate as intended
- Identify potential security issues with the smart contracts

In summary, Halborn identified some security risks that need to be addressed by the Vesper Finance team.

1.3 TEST APPROACH & METHODOLOGY

Halborn performed a combination of manual and automated security testing to balance efficiency, timeliness, practicality, and accuracy in regard to the scope of this audit. While manual testing is recommended to uncover flaws in logic, process, and implementation; automated testing techniques help enhance coverage of the bridge code and can quickly identify items that do not follow security best practices. The following phases and associated tools were used throughout the term of the audit:

- Research into architecture and purpose
- Smart contract manual code review and walkthrough
- Graphing out functionality and contract logic/connectivity/functions (solgraph)
- Manual assessment of use and safety for the critical Solidity variables and functions in scope to identify any arithmetic related vulnerability classes
- Manual testing by custom scripts
- Scanning of solidity files for vulnerabilities, security hotspots or bugs. (MythX)
- Static Analysis of security for scoped contract, and imported functions. (Slither)
- Testnet deployment (Brownie, Remix IDE)

RISK METHODOLOGY:

Vulnerabilities or issues observed by Halborn are ranked based on the risk assessment methodology by measuring the **LIKELIHOOD** of a security incident and the **IMPACT** should an incident occur. This framework works for communicating the characteristics and impacts of technology vulnerabilities. The quantitative model ensures repeatable and accurate measurement while enabling users to see the underlying vulnerability characteristics that were used to generate the Risk scores. For every vulnerability, a risk level will be calculated on a scale of 5 to 1 with 5 being the highest likelihood or impact.

RISK SCALE - LIKELIHOOD

5 - Almost certain an incident will occur.
4 - High probability of an incident occurring.
3 - Potential of a security incident in the long term.
2 - Low probability of an incident occurring.
1 - Very unlikely issue will cause an incident.

RISK SCALE - IMPACT

5 - May cause devastating and unrecoverable impact or loss.
4 - May cause a significant level of impact or loss.

3 - May cause a partial impact or loss to many.
2 - May cause temporary impact or loss.
1 - May cause minimal or un-noticeable impact.

The risk level is then calculated using a sum of these two values, creating a value of 10 to 1 with 10 being the highest level of security risk.

10 - CRITICAL
9 - 8 - HIGH
7 - 6 - MEDIUM
5 - 4 - LOW
3 - 1 - VERY LOW AND INFORMATIONAL

1.4 SCOPE

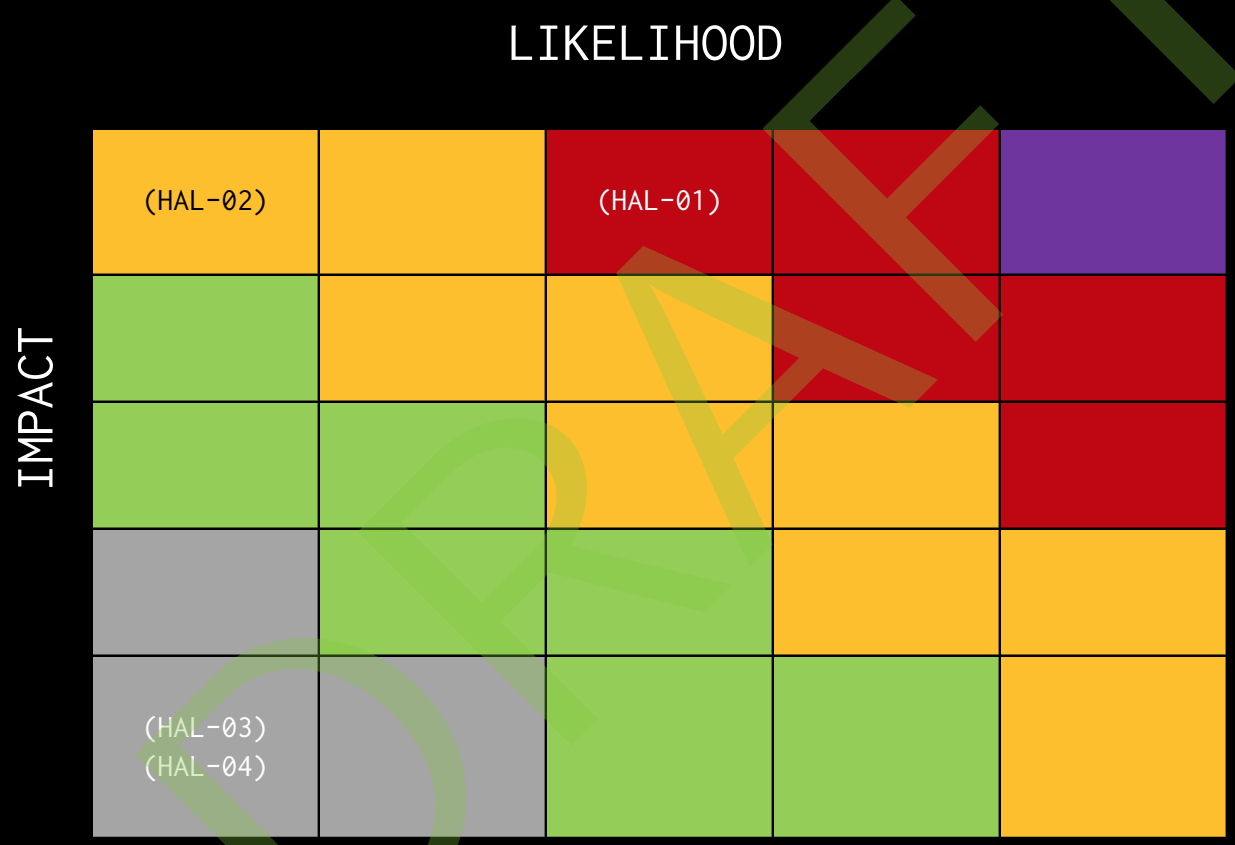
IN-SCOPE:
The security assessment was scoped to the following smart contracts:

- Controller.sol
- DebtToken.sol
- DepositToken.sol
- NativeTokenGateway.sol
- Pausable.sol
- RewardsDistributor.sol
- SyntheticToken.sol
- Treasury.sol
- TokenHolder.sol
- CTokenOracle.sol
- ChainlinkPriceProvider.sol
- DefaultOracle.sol
- MasterOracle.sol
- UniswapV3PriceProvider.sol
- Governable.sol
- Manageable.sol

Commit ID: 8a6ae4b942a3d0f3fcbd734ae1a0cdacc1efea34

2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	1	1	0	2



SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
HAL01 - FREEZING OF USER FUNDS DURING A PERIOD OF TIME	High	-
HAL02 - USERS ARE UNABLE TO CLAIM THEIR REWARDS	Medium	-
HAL03 - ZERO ADDRESS NOT CHECKED	Informational	-
HAL04 - USE ++I INSTEAD OF I++ IN LOOPS FOR GAS OPTIMIZATION	Informational	-

FINDINGS & TECH DETAILS

3.1 (HAL-01) FREEZING OF USER FUNDS DURING A PERIOD OF TIME - HIGH

Description:

Freezing of user funds during a period of time due to not ensuring there are enough funds for users to be able to withdraw their funds when the treasury address has been updated.

Although the protocol has the option to migrate the funds to the new treasury address, that is not enforced. So, it is possible to set a treasury address without the needed funds and users will not be able to withdraw their funds because they will be frozen within the old treasury.

Code Location:

Listing 1: Controller.sol (Line 583)

```
578 function updateTreasury(ITreasury _newTreasury, bool
 ↳ _withMigration) external override onlyGovernor {
579     require(address(_newTreasury) != address(0), "address-is-null"
 ↳ );
580     ITreasury _currentTreasury = treasury;
581     require(_newTreasury != _currentTreasury, "new-same-as-current
 ↳ ");
582
583     if (_withMigration) _currentTreasury.migrateTo(address(
 ↳ _newTreasury));
584
585     emit TreasuryUpdated(_currentTreasury, _newTreasury);
586     treasury = _newTreasury;
587 }
```

Listing 2: DepositToken.sol (Line 291)

```
266 function withdraw(uint256 _amount, address _to)
267     external
268     override
269     whenNotShutdown
```

```
270     nonReentrant
271     onlyIfDepositTokenExists
272 {
273     require(_amount > 0, "amount-is-zero");
274 
275     address _account = _msgSender();
276 
277     require(_amount <= unlockedBalanceOf(_account), "amount-gt-
 ↳ unlocked");
278 
279     ITreasury _treasury = controller.treasury();
280 
281     uint256 _withdrawFee = controller.withdrawFee();
282     uint256 _amountToWithdraw = _amount;
283     uint256 _feeAmount;
284     if (_withdrawFee > 0) {
285         _feeAmount = _amount.wadMul(_withdrawFee);
286         _transfer(_account, address(_treasury), _feeAmount);
287         _amountToWithdraw -= _feeAmount;
288     }
289 
290     _burnForWithdraw(_account, _amountToWithdraw);
291     _treasury.pull(_to, _amountToWithdraw);
292 
293     emit CollateralWithdrawn(_account, _to, _amount, _feeAmount);
294 }
```

Listing 3: Treasury.sol (Line 40)

```
38 function pull(address _to, uint256 _amount) external override
 ↳ nonReentrant onlyIfDepositToken {
39     require(_amount > 0, "amount-is-zero");
40     IDepositToken(_msgSender()).underlying().safeTransfer(_to,
 ↳ _amount);
41 }
```


Proof of Concept:

1. User deposits tokens to the protocol
2. Protocol's owner updates the treasury address without migration
3. Users cannot withdraw their funds

Listing 4: Proof of Concept using Brownie (Lines 42,49,54)

```
weth = interface.IWETH("0xC02aaA39b223FE8D0A0e5C4F27eAD9083C756Cc2")

nativetokengateway = NativeTokenGateway.deploy(weth, {'from': owner})

controller = Controller.deploy({'from': owner})
masteroracle = MasterOracle.deploy({'from': owner})
controller.initialize(masteroracle, {'from': owner})

vsdToken = DepositToken.deploy({'from': owner})
vsdToken.initialize(weth, controller, "VSDT", 18, 1_00000000000000000, 1000000000, {'from': owner})

synthetictoken = SyntheticToken.deploy({'from': owner})
debttoken = DebtToken.deploy({'from': owner})
synthetictoken.initialize("synttoken", "SYNT", 18, controller, debttoken, 10000000000000000, 1000000000, {'from': owner})
debttoken.initialize("debttoken", "DEBTT", 18, controller, {'from': owner})
debttoken.setSyntheticToken(synthetictoken, {'from': owner})
controller.addSyntheticToken(synthetictoken, {'from': owner})

rewardtoken = ERC20Mock.deploy("rewardToken", "REWT", 18, {'from': owner})
rewardsdistributor = RewardsDistributor.deploy({'from': owner})
rewardsdistributor.initialize(controller, rewardtoken, {'from': owner})
controller.addRewardsDistributor(rewardsdistributor, {'from': owner})
rewardtoken.mint(rewardsdistributor, 1000_000000000000000000, {'from': owner})

tx = controller.addDepositToken(vsdToken, {'from': owner})
controller.depositTokenOf(weth)

treasury = Treasury.deploy({'from': owner})
treasury.initialize(controller, {'from': owner})

controller.updateTreasury(treasury, False, {'from': owner})

defaultoraclemock = DefaultOracleMock.deploy({'from': owner})
defaultoraclemock.updatePrice(vsdToken, 1, {'from': owner})
```

```
defaultoraclemock.updatePrice(syntheticToken, 1, {'from': owner})

masteroracle.setDefaultOracle(defaultoraclemock, {'from': owner})
#PoC
output.greenn("user1.balance() --> " + str(user1.balance()))
output.greenn("vsdToken.balanceOf(user1) --> " + str(vsdToken.balanceOf(user1)))
output.greenn("weth.balanceOf(treasury) --> " + str(weth.balanceOf(treasury)))
tx = nativeTokenGateway.deposit(controller, {'from': user1, 'value': 5_000000000000000000})
output.greenn("user1.balance() --> " + str(user1.balance()))
output.greenn("vsdToken.balanceOf(user1) --> " + str(vsdToken.balanceOf(user1)))
output.greenn("weth.balanceOf(treasury) --> " + str(weth.balanceOf(treasury)))

treasury2 = Treasury.deploy({'from': owner})
treasury2.initialize(controller, {'from': owner})
controller.updateTreasury(treasury2, False, {'from': owner})

output.greenn("weth.balanceOf(user1) --> " + str(weth.balanceOf(user1)))
output.greenn("controller.debtPositionOf(user1) --> " + str(controller.debtPositionOf(user1)))
output.redd("executing --> vsdToken.withdraw(5_000000000000000000, user1, {'from': user1}) <-- ")
vsdToken.withdraw(5_000000000000000000, user1, {'from': user1})
output.greenn("weth.balanceOf(user1) --> " + str(weth.balanceOf(user1)))
output.greenn("controller.debtPositionOf(user1) --> " + str(controller.debtPositionOf(user1)))
```

Risk Level:

Likelihood - 3
Impact - 5

Recommendation:

The protocol should ensure the new treasury address has enough balance to allow users to withdraw their funds.

3.2 (HAL-02) USERS ARE UNABLE TO CLAIM THEIR REWARDS - MEDIUM

Description:

If the protocol's owner's private key gets stolen or is a malicious owner, a DoS attack is possible against the protocol. Many tokens can be added as rewards from the protocol. Users can only claim their rewards by using the `claimRewards` function within `RewardsDistributor.sol` contract, which uses a loop to check every different reward token added previously.

If this token array gets too big, users will not be able to claim their rewards, and they will be frozen within the contract as the transaction will reach the gas limit.

Code Location:

Listing 5: RewardsDistributor.sol (Line 169)

```
168 function claimRewards(address _account) external {
169     claimRewards(_account, tokens);
170 }
```

Listing 6: RewardsDistributor.sol (Line 178)

```
175 function claimRewards(address _account, IERC20[] memory _tokens)
 ↳ public {
176     address[] memory accounts = new address[](1);
177     accounts[0] = _account;
178     claimRewards(accounts, _tokens);
179 }
```

Listing 7: RewardsDistributor.sol (Line 187)

```
184 function claimRewards(address[] memory _accounts, IERC20[] memory
 ↳ _tokens) public nonReentrant {
185     uint256 _accountsLength = _accounts.length;
186     uint256 _tokensLength = _tokens.length;
```

```
187     for (uint256 i; i < _tokensLength; i++) {
188         IERC20 _token = _tokens[i];
189 
190         if (tokenStates[_token].index > 0) {
191             _updateTokenIndex(_token);
192             for (uint256 j; j < _accountsLength; j++) {
193                 _updateTokensAccruedOf(_token, _accounts[j]);
194             }
195         }
196     }
197 
198     for (uint256 j; j < _accountsLength; j++) {
199         address _account = _accounts[j];
200         _transferRewardIfEnoughTokens(_account, tokensAccruedOf[
 ↳ _account]);
201     }
202 }
```

Proof of Concept:

1. The protocol's owner adds many reward tokens
2. Users cannot claim their rewards

Listing 8: Proof of Concept using Brownie (Lines 39,45)

```
 1 weth = interface.IWETH("0xC02aaA39b223FE8D0A0e5C4F27eAD9083C756Cc2
 ↳ ")
 2 
 3 nativetokengateway = NativeTokenGateway.deploy(weth, {'from':
 ↳ owner})
 4 
 5 controller = Controller.deploy({'from': owner})
 6 masteroracle = MasterOracle.deploy({'from': owner})
 7 controller.initialize(masteroracle, {'from': owner})
 8 
 9 vsdToken = DepositToken.deploy({'from': owner})
10 vsdToken.initialize(weth, controller, "VSDT", 18, 1
 ↳ _000000000000000000, 1000000000, {'from': owner})
11 
12 synthetictoken = SyntheticToken.deploy({'from': owner})
13 debttoken = DebtToken.deploy({'from': owner})
14 synthetictoken.initialize("synttoken", "SYNT", 18, controller,
```

```
↳ debttoken, 10000000000000000, 1000000000, {'from': owner})
15 debttoken.initialize("debttoken", "DEBTT", 18, controller, {'from'
↳ : owner})
16 debttoken.setSyntheticToken(synthetictoken, {'from': owner})
17 controller.addSyntheticToken(synthetictoken, {'from': owner})
18
19 rewardtoken = ERC20Mock.deploy("rewardToken", "REWT", 18, {'from':
↳  owner})
20 rewardsdistributor = RewardsDistributor.deploy({'from': owner})
21 rewardsdistributor.initialize(controller, rewardtoken, {'from':
↳ owner})
22 controller.addRewardsDistributor(rewardsdistributor, {'from':
↳ owner})
23 rewardtoken.mint(rewardsdistributor, 1000_000000000000000, {'
↳ from': owner})
24
25 tx = controller.addDepositToken(vsdToken, {'from': owner})
26 controller.depositTokenOf(weth)
27
28 treasury = Treasury.deploy({'from': owner})
29 treasury.initialize(controller, {'from': owner})
30
31 controller.updateTreasury(treasury, False, {'from': owner})
32
33 defaultoraclemock = DefaultOracleMock.deploy({'from': owner})
34 defaultoraclemock.updatePrice(vsdToken, 1, {'from': owner})
35 defaultoraclemock.updatePrice(synthetictoken, 1, {'from': owner})
36
37 masteroracle.setDefaultOracle(defaultoraclemock, {'from': owner})
38 #DoS
39 for x in range(20000):
40     vsdToken = DepositToken.deploy({'from': owner})
41     vsdToken.initialize(weth, controller, "VSDT", 18, 1
↳ _000000000000000000, 1000000000, {'from': owner})
42     tx = controller.addDepositToken(vsdToken, {'from': owner})
43     rewardsdistributor.updateTokenSpeed(vsdToken, 1, {'from':
↳ owner})
44
45 rewardsdistributor.claimRewards(user1, {'from': user1})
```

Risk Level:

Likelihood - 1
Impact - 5

Recommendation:

The protocol should set a limit for the number of different reward tokens that can be added to the protocol.

3.3 (HAL-03) ZERO ADDRESS NOT CHECKED - INFORMATIONAL

Description:

In the constructor of CTokenOracle.sol contract, the address variables are not being checked to avoid pointing to the zero address.

Code Location:

Listing 9: CTokenOracle.sol (Lines 28,29)

```
27 constructor(IOracle _underlyingOracle, address _wethLike) {
28     underlyingOracle = _underlyingOracle;
29     wethLike = _wethLike;
30 }
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

When setting an address variable, always make sure the value is not zero.

3.4 (HAL-04) USE ++I INSTEAD OF I++ IN LOOPS FOR GAS OPTIMIZATION - INFORMATIONAL

Description:

In some loops, the variable `i` is incremented using `i++`. It is known that, in loops, using `++i` costs less gas per iteration than `i++`. This also affects variables incremented inside the loop code block.

Code Location:

Listing 10: Controller.sol (Line 595)

```
592 function addRewardsDistributor(IRewardsDistributor _distributor)
 ↳ external override onlyGovernor {
593     require(address(_distributor) != address(0), "address-is-null"
 ↳ );
594
595     for (uint256 i; i < rewardsDistributors.length; i++)
596         require(_distributor != rewardsDistributors[i], "contract-
 ↳ already-added");
597
598     rewardsDistributors.push(_distributor);
599     emit RewardsDistributorAdded(_distributor);
600 }
601
```

Listing 11: DebtToken.sol (Line 40)

```
37 modifier updateRewardsBeforeMintOrBurn(address _account) {
38     IRewardsDistributor[] memory _rewardsDistributors = controller
 ↳ .getRewardsDistributors();
39     uint256 _length = _rewardsDistributors.length;
40     for (uint256 i; i < _length; i++) {
41         _rewardsDistributors[i].updateBeforeMintOrBurn(
 ↳ syntheticToken, _account);
42     }
43     _;
```

```
44 }
45
```

Listing 12: DepositToken.sol (Line 69)

```
66 modifier updateRewardsBeforeMintOrBurn(address _account) {
67     IRewardsDistributor[] memory _rewardsDistributors = controller
↳ .getRewardsDistributors();
68     uint256 _length = _rewardsDistributors.length;
69     for (uint256 i; i < _length; i++) {
70         _rewardsDistributors[i].updateBeforeMintOrBurn(this,
↳ _account);
71     }
72     _;
73 }
74
```

Listing 13: DepositToken.sol (Line 82)

```
79 modifier updateRewardsBeforeTransfer(address _sender, address
↳ _recipient) {
80     IRewardsDistributor[] memory _rewardsDistributors = controller
↳ .getRewardsDistributors();
81     uint256 _length = _rewardsDistributors.length;
82     for (uint256 i; i < _length; i++) {
83         _rewardsDistributors[i].updateBeforeTransfer(this, _sender
↳ , _recipient);
84     }
85     _;
86 }
87
```

Listing 14: RewardsDistributor.sol (Line 38)

```
34 modifier onlyIfDistributorExists() {
35     bool _distributorAdded = false;
36     IRewardsDistributor[] memory _rewardsDistributors = controller
↳ .getRewardsDistributors();
37     uint256 _length = _rewardsDistributors.length;
38     for (uint256 i; i < _length; i++) {
39         if (_rewardsDistributors[i] == this) {
40             _distributorAdded = true;
```

```
41                 break;
42             }
43         }
44         require(_distributorAdded, "distributor-not-added");
45         _;
46 }
47
```

Listing 15: RewardsDistributor.sol (Lines 187,192,198)

```
184 function claimRewards(address[] memory _accounts, IERC20[] memory
 ↳ _tokens) public nonReentrant {
185     uint256 _accountsLength = _accounts.length;
186     uint256 _tokensLength = _tokens.length;
187     for (uint256 i; i < _tokensLength; i++) {
188         IERC20 _token = _tokens[i];
189
190         if (tokenStates[_token].index > 0) {
191             _updateTokenIndex(_token);
192             for (uint256 j; j < _accountsLength; j++) {
193                 _updateTokensAccruedOf(_token, _accounts[j]);
194             }
195         }
196     }
197
198     for (uint256 j; j < _accountsLength; j++) {
199         address _account = _accounts[j];
200         _transferRewardIfEnoughTokens(_account, tokensAccruedOf[
 ↳ _account]);
201     }
202 }
203
```

Listing 16: RewardsDistributor.sol (Line 231)

```
227 function updateTokenSpeeds(IERC20[] calldata _tokens, uint256[]
 ↳ calldata _speeds) external onlyGovernor {
228     uint256 _tokensLength = _tokens.length;
229     require(_tokensLength == _speeds.length, "invalid-input");
230
231     for (uint256 i; i < _tokensLength; ++i) {
232         _updateTokenSpeed(_tokens[i], _speeds[i]);
233     }
```

```
234 }
235
```

Listing 17: MasterOracle.sol (Line 55)

```
51 function _updateOracles(address[] calldata _assets, IOracle[]
↳ calldata _oracles) private {
52     uint256 _assetsLength = _assets.length;
53     require(_assetsLength == _oracles.length, "invalid-arrays-
↳ length");
54
55     for (uint256 i; i < _assetsLength; i++) {
56         address _asset = _assets[i];
57         require(_asset != address(0), "an-asset-has-null-address")
↳ ;
58         IOracle _currentOracle = oracles[_asset];
59         IOracle _newOracle = _oracles[i];
60         require(_newOracle != _currentOracle, "a-new-oracle-same-
↳ as-current");
61         emit OracleUpdated(_asset, _currentOracle, _newOracle);
62         oracles[_asset] = _newOracle;
63     }
64 }
65
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

It is recommended to use `++i` instead of `i++` to increment the value of a `uint` variable inside a loop. This also applies to variables declared inside the `for` loop, but does not apply outside of loops.

AUTOMATED TESTING

4.1 STATIC ANALYSIS REPORT

Description:

Halborn used automated testing techniques to enhance the coverage of certain areas of the scoped contracts. Among the tools used was Slither, a Solidity static analysis framework. After Halborn verified all the contracts in the repository and was able to compile them correctly into their ABI and binary formats, Slither was run on the all-scoped contracts. This tool can statically verify mathematical relationships between Solidity variables to detect invalid or inconsistent usage of the contracts' APIs across the entire code-base.

Slither results:

Controller.sol

```
Controller.liquidate(ISyntheticToken,address,uint256,IDepositToken) (contracts/Controller.sol#274-320) uses a dangerous strict equality:
        - require(bool,string)(_newDebtInUsd == 0 || _newDebtInUsd >= debtFloorInUsd,remaining-debt-lt-floor) (contracts/Controller.sol#298)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities

Reentrancy in Controller.updateTreasury(ITreasury,bool) (contracts/Controller.sol#578-587):
        External calls:
        - _currentTreasury.migrateTo(address(_newTreasury)) (contracts/Controller.sol#583)
        State variables written after the call(s):
        - treasury = _newTreasury (contracts/Controller.sol#586)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1

Controller.swap(ISyntheticToken,ISyntheticToken,uint256)._feeAmount (contracts/Controller.sol#354) is a local variable never initialized
Controller.depositOf(address).i (contracts/Controller.sol#232) is a local variable never initialized
Controller.debtOf(address).i (contracts/Controller.sol#212) is a local variable never initialized
Controller.addRewardsDistributor(IRewardsDistributor).i (contracts/Controller.sol#595) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables

Controller.debtOf(address) (contracts/Controller.sol#210-217) has external calls inside a loop: _syntheticToken = _debtToken.syntheticToken() (contracts/Controller.sol#214)
Controller.debtOf(address) (contracts/Controller.sol#210-217) has external calls inside a loop: _debtInUsd += masterOracle.quoteTokenToUsd(_syntheticToken,_debtToken.balanceOf(_account)) (contracts/Controller.sol#2
Controller.depositOf(address) (contracts/Controller.sol#225-238) has external calls inside a loop: _amountInUsd = masterOracle.quoteTokenToUsd(_depositToken,_depositToken.balanceOf(_account)) (contracts/Controller.
Controller.depositOf(address) (contracts/Controller.sol#225-238) has external calls inside a loop: _issuableLimitInUsd += _amountInUsd.wadMul(_depositToken.collateralizationRatio()) (contracts/Controller.sol#236)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-a-loop

Reentrancy in Controller.liquidate(ISyntheticToken,address,uint256,IDepositToken) (contracts/Controller.sol#274-320):
        External calls:
        - _syntheticToken.accrueInterest() (contracts/Controller.sol#285)
        - _syntheticToken.burn(_liquidator,_amountToRepay) (contracts/Controller.sol#311)
        - _debtToken.burn(_account,_amountToRepay) (contracts/Controller.sol#312)
        - _depositToken.seize(_account,_liquidator,_toLiquidator) (contracts/Controller.sol#313)
        - _depositToken.seize(_account,address(treasury),_toProtocol) (contracts/Controller.sol#316)
        Event emitted after the call(s):
        - PositionLiquidated(_liquidator,_account,_syntheticToken,_amountToRepay,_depositToSeize,_toProtocol) (contracts/Controller.sol#319)
Reentrancy in Controller.swap(ISyntheticToken,ISyntheticToken,uint256) (contracts/Controller.sol#328-364):
        External calls:
        - _syntheticTokenIn.accrueInterest() (contracts/Controller.sol#342)
        - _syntheticTokenOut.accrueInterest() (contracts/Controller.sol#343)
        - _syntheticTokenIn.burn(_account,_amountIn) (contracts/Controller.sol#350)
        - _syntheticTokenOut.mint(address(treasury),_feeAmount) (contracts/Controller.sol#357)
        - _syntheticTokenOut.mint(_account,_amountOut) (contracts/Controller.sol#361)
        Event emitted after the call(s):
        - SyntheticTokenSwapped(_account,_syntheticTokenIn,_syntheticTokenOut,_amountIn,_amountOut,_feeAmount) (contracts/Controller.sol#363)
Reentrancy in Controller.updateTreasury(ITreasury,bool) (contracts/Controller.sol#578-587):
        External calls:
        - _currentTreasury.migrateTo(address(_newTreasury)) (contracts/Controller.sol#583)
        Event emitted after the call(s):
        - TreasuryUpdated(_currentTreasury,_newTreasury) (contracts/Controller.sol#585)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Address.isContract(address) (contracts/dependencies/openzeppelin/utils/Address.sol#26-36) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#32-34)
Address.verifyCallResult(bool,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#195-215) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#207-210)
EnumerableSet.values(EnumerableSet.AddressSet) (contracts/dependencies/openzeppelin/utils/structs/EnumerableSet.sol#273-282) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/structs/EnumerableSet.sol#277-279)
EnumerableSet.values(EnumerableSet.UintSet) (contracts/dependencies/openzeppelin/utils/structs/EnumerableSet.sol#346-355) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/structs/EnumerableSet.sol#350-352)
MappedEnumerableSet.values(MappedEnumerableSet.AddressSet,address) (contracts/lib/MappedEnumerableSet.sol#136-145) uses assembly
        - INLINE ASM (contracts/lib/MappedEnumerableSet.sol#140-142)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['0.8.9', '^0.8.0']
        - 0.8.9 (contracts/Controller.sol#3)
        - 0.8.9 (contracts/Pausable.sol#3)
        - 0.8.9 (contracts/access/Governable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3)
```

```
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/math/Math.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/structs/EnumerableSet.sol#3)
        - 0.8.9 (contracts/interface/IController.sol#3)
        - 0.8.9 (contracts/interface/IDebtToken.sol#3)
        - 0.8.9 (contracts/interface/IDepositToken.sol#3)
        - 0.8.9 (contracts/interface/IGovernable.sol#3)
        - 0.8.9 (contracts/interface/IPausable.sol#3)
        - 0.8.9 (contracts/interface/IRewardsDistributor.sol#3)
        - 0.8.9 (contracts/interface/ISyntheticToken.sol#3)
        - 0.8.9 (contracts/interface/ITreasury.sol#3)
        - 0.8.9 (contracts/interface/oracle/IMasterOracle.sol#3)
        - ^0.8.0 (contracts/lib/MappedEnumerableSet.sol#3)
        - 0.8.9 (contracts/lib/WadRayMath.sol#3)
        - 0.8.9 (contracts/storage/ControllerStorage.sol#3)
        - 0.8.9 (contracts/utils/TokenHolder.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#108-114) is never used and should be removed
Address.functionDelegateCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#168-170) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187) is never used and should be removed
Address.functionStaticCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#141-143) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160) is never used and should be removed
Context._msgData() (contracts/dependencies/openzeppelin/utils/Context.sol#20-22) is never used and should be removed
EnumerableSet._at(EnumerableSet.Set,uint256) (contracts/dependencies/openzeppelin/utils/structs/EnumerableSet.sol#129-131) is never used and should be removed
EnumerableSet._length(EnumerableSet.Set) (contracts/dependencies/openzeppelin/utils/structs/EnumerableSet.sol#115-117) is never used and should be removed
EnumerableSet.add(EnumerableSet.Bytes32Set,bytes32) (contracts/dependencies/openzeppelin/utils/structs/EnumerableSet.sol#157-159) is never used and should be removed
EnumerableSet.add(EnumerableSet.UintSet,uint256) (contracts/dependencies/openzeppelin/utils/structs/EnumerableSet.sol#296-298) is never used and should be removed
EnumerableSet.at(EnumerableSet.AddressSet,uint256) (contracts/dependencies/openzeppelin/utils/structs/EnumerableSet.sol#261-263) is never used and should be removed
EnumerableSet.at(EnumerableSet.Bytes32Set,uint256) (contracts/dependencies/openzeppelin/utils/structs/EnumerableSet.sol#195-197) is never used and should be removed
EnumerableSet.at(EnumerableSet.UintSet,uint256) (contracts/dependencies/openzeppelin/utils/structs/EnumerableSet.sol#334-336) is never used and should be removed
EnumerableSet.contains(EnumerableSet.Bytes32Set,bytes32) (contracts/dependencies/openzeppelin/utils/structs/EnumerableSet.sol#174-176) is never used and should be removed
EnumerableSet.contains(EnumerableSet.UintSet,uint256) (contracts/dependencies/openzeppelin/utils/structs/EnumerableSet.sol#313-315) is never used and should be removed
EnumerableSet.length(EnumerableSet.AddressSet) (contracts/dependencies/openzeppelin/utils/structs/EnumerableSet.sol#247-249) is never used and should be removed
EnumerableSet.length(EnumerableSet.Bytes32Set) (contracts/dependencies/openzeppelin/utils/structs/EnumerableSet.sol#181-183) is never used and should be removed
EnumerableSet.length(EnumerableSet.UintSet) (contracts/dependencies/openzeppelin/utils/structs/EnumerableSet.sol#320-322) is never used and should be removed
EnumerableSet.remove(EnumerableSet.Bytes32Set,bytes32) (contracts/dependencies/openzeppelin/utils/structs/EnumerableSet.sol#167-169) is never used and should be removed
EnumerableSet.remove(EnumerableSet.UintSet,uint256) (contracts/dependencies/openzeppelin/utils/structs/EnumerableSet.sol#306-308) is never used and should be removed
EnumerableSet.values(EnumerableSet.Bytes32Set) (contracts/dependencies/openzeppelin/utils/structs/EnumerableSet.sol#207-209) is never used and should be removed
EnumerableSet.values(EnumerableSet.UintSet) (contracts/dependencies/openzeppelin/utils/structs/EnumerableSet.sol#346-355) is never used and should be removed
MappedEnumerableSet.contains(MappedEnumerableSet.AddressSet,address,address) (contracts/lib/MappedEnumerableSet.sol#116-122) is never used and should be removed
Math.average(uint256,uint256) (contracts/dependencies/openzeppelin/utils/math/Math.sol#27-30) is never used and should be removed
Math.ceilDiv(uint256,uint256) (contracts/dependencies/openzeppelin/utils/math/Math.sol#38-41) is never used and should be removed
Math.max(uint256,uint256) (contracts/dependencies/openzeppelin/utils/math/Math.sol#12-14) is never used and should be removed
Math.min(uint256,uint256) (contracts/dependencies/openzeppelin/utils/math/Math.sol#19-21) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#44-57) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#68-79) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#59-66) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#28-35) is never used and should be removed
TokenHolder._requireCanSweep() (contracts/utils/TokenHolder.sol#17) is never used and should be removed
WadRayMath.rayDiv(uint256,uint256) (contracts/lib/WadRayMath.sol#63-65) is never used and should be removed
WadRayMath.rayMul(uint256,uint256) (contracts/lib/WadRayMath.sol#49-55) is never used and should be removed
WadRayMath.rayToWad(uint256) (contracts/lib/WadRayMath.sol#72-74) is never used and should be removed
WadRayMath.wadToRay(uint256) (contracts/lib/WadRayMath.sol#81-83) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version0.8.9 (contracts/Controller.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/Pausable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/access/Governable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/math/Math.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/structs/EnumerableSet.sol#3) allows old versions
Pragma version0.8.9 (contracts/interface/IController.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IDebtToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IDepositToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IGovernable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IPausable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IRewardsDistributor.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/ISyntheticToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/ITreasury.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/oracle/IMasterOracle.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version^0.8.0 (contracts/lib/MappedEnumerableSet.sol#3) allows old versions
Pragma version0.8.9 (contracts/lib/WadRayMath.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/storage/ControllerStorage.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/utils/TokenHolder.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.9 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#54-59):
        - (success) = recipient.call{value: amount}() (contracts/dependencies/openzeppelin/utils/Address.sol#57)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (contracts/dependencies/openzeppelin/utils/Address.sol#122-133):
        - (success,returndata) = target.call{value: value}(data) (contracts/dependencies/openzeppelin/utils/Address.sol#131)
Low level call in Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160):
        - (success,returndata) = target.staticcall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#158)
Low level call in Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187):
        - (success,returndata) = target.delegatecall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#185)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter Controller.initialize(IMasterOracle)._masterOracle (contracts/Controller.sol#129) is not in mixedCase
Parameter Controller.getDepositTokensOfAccount(address)._account (contracts/Controller.sol#167) is not in mixedCase
Parameter Controller.getDebtTokensOfAccount(address)._account (contracts/Controller.sol#176) is not in mixedCase
Parameter Controller.isSyntheticTokenExists(ISyntheticToken)._syntheticToken (contracts/Controller.sol#192) is not in mixedCase
Parameter Controller.isDepositTokenExists(IDepositToken)._depositToken (contracts/Controller.sol#201) is not in mixedCase
Parameter Controller.debtOf(address)._account (contracts/Controller.sol#210) is not in mixedCase
Parameter Controller.depositOf(address)._account (contracts/Controller.sol#226) is not in mixedCase
Parameter Controller.debtPositionOf(address)._account (contracts/Controller.sol#249) is not in mixedCase
Parameter Controller.liquidate(ISyntheticToken,address,uint256,IDepositToken)._syntheticToken (contracts/Controller.sol#275) is not in mixedCase
Parameter Controller.liquidate(ISyntheticToken,address,uint256,IDepositToken)._account (contracts/Controller.sol#276) is not in mixedCase
Parameter Controller.liquidate(ISyntheticToken,address,uint256,IDepositToken)._amountToRepay (contracts/Controller.sol#277) is not in mixedCase
Parameter Controller.liquidate(ISyntheticToken,address,uint256,IDepositToken)._depositToken (contracts/Controller.sol#278) is not in mixedCase
Parameter Controller.swap(ISyntheticToken,ISyntheticToken,uint256)._syntheticTokenIn (contracts/Controller.sol#329) is not in mixedCase
Parameter Controller.swap(ISyntheticToken,ISyntheticToken,uint256)._syntheticTokenOut (contracts/Controller.sol#330) is not in mixedCase
Parameter Controller.swap(ISyntheticToken,ISyntheticToken,uint256)._amountIn (contracts/Controller.sol#331) is not in mixedCase
Parameter Controller.addSyntheticToken(address)._syntheticToken (contracts/Controller.sol#369) is not in mixedCase
Parameter Controller.removeSyntheticToken(ISyntheticToken)._syntheticToken (contracts/Controller.sol#380) is not in mixedCase
Parameter Controller.addDepositToken(address)._depositToken (contracts/Controller.sol#397) is not in mixedCase
Parameter Controller.removeDepositToken(IDepositToken)._depositToken (contracts/Controller.sol#409) is not in mixedCase
Parameter Controller.addToDepositTokensOfAccount(address)._account (contracts/Controller.sol#429) is not in mixedCase
Parameter Controller.removeFromDepositTokensOfAccount(address)._account (contracts/Controller.sol#440) is not in mixedCase
Parameter Controller.addToDebtTokensOfAccount(address)._account (contracts/Controller.sol#451) is not in mixedCase
Parameter Controller.removeFromDebtTokensOfAccount(address)._account (contracts/Controller.sol#461) is not in mixedCase
Parameter Controller.updateMasterOracle(IMasterOracle)._newMasterOracle (contracts/Controller.sol#648) is not in mixedCase
Parameter Controller.updateDepositFee(uint256)._newDepositFee (contracts/Controller.sol#480) is not in mixedCase
Parameter Controller.updateIssueFee(uint256)._newIssueFee (contracts/Controller.sol#491) is not in mixedCase
Parameter Controller.updateWithdrawFee(uint256)._newWithdrawFee (contracts/Controller.sol#502) is not in mixedCase
Parameter Controller.updateRepayFee(uint256)._newRepayFee (contracts/Controller.sol#513) is not in mixedCase
Parameter Controller.updateSwapFee(uint256)._newSwapFee (contracts/Controller.sol#524) is not in mixedCase
Parameter Controller.updateLiquidatorLiquidationFee(uint256)._newLiquidatorLiquidationFee (contracts/Controller.sol#535) is not in mixedCase
Parameter Controller.updateProtocolLiquidationFee(uint256)._newProtocolLiquidationFee (contracts/Controller.sol#546) is not in mixedCase
Parameter Controller.updateMaxLiquidable(uint256)._newMaxLiquidable (contracts/Controller.sol#557) is not in mixedCase
Parameter Controller.updateDebtFloor(uint256)._newDebtFloorUsd (contracts/Controller.sol#568) is not in mixedCase
Parameter Controller.updateTreasury(ITreasury,bool)._newTreasury (contracts/Controller.sol#578) is not in mixedCase
Parameter Controller.updateTreasury(ITreasury,bool)._withMigration (contracts/Controller.sol#578) is not in mixedCase
Parameter Controller.addRewardsDistributor(IRewardsDistributor)._distributor (contracts/Controller.sol#592) is not in mixedCase
Function Governable.__Governable_init() (contracts/access/Governable.sol#39-43) is not in mixedCase
Parameter Governable.transferGovernorship(address)._proposedGovernor (contracts/access/Governable.sol#59) is not in mixedCase
Function ReentrancyGuard.__ReentrancyGuard_init() (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#40-42) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._token (contracts/utils/TokenHolder.sol#26) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._to (contracts/utils/TokenHolder.sol#27) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._amount (contracts/utils/TokenHolder.sol#28) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Variable IController.liquidate(ISyntheticToken,address,uint256,IDepositToken)._depositToken (contracts/interface/IController.sol#65) is too similar to ControllerStorageV1.depositTokens (contracts/storage/Controller
Variable Controller.liquidate(ISyntheticToken,address,uint256,IDepositToken)._depositToken (contracts/Controller.sol#278) is too similar to ControllerStorageV1.depositTokens (contracts/storage/ControllerStorage.sol
Variable IController.removeDepositToken(IDepositToken)._depositToken (contracts/interface/IController.sol#59) is too similar to ControllerStorageV1.depositTokens (contracts/storage/ControllerStorage.sol#79)
Variable Controller.removeDepositToken(IDepositToken)._depositToken (contracts/Controller.sol#409) is too similar to ControllerStorageV1.depositTokens (contracts/storage/ControllerStorage.sol#79)
Variable IController.depositTokenExists(IDepositToken)._depositToken (contracts/interface/IController.sol#28) is too similar to ControllerStorageV1.depositTokens (contracts/storage/ControllerStorage.sol#79)
Variable Controller.depositOf(address)._depositToken (contracts/Controller.sol#233) is too similar to ControllerStorageV1.depositTokens (contracts/storage/ControllerStorage.sol#79)
Variable Controller.addDepositToken(address)._depositToken (contracts/Controller.sol#397) is too similar to ControllerStorageV1.depositTokens (contracts/storage/ControllerStorage.sol#79)
Variable IController.addDepositToken(address)._depositToken (contracts/interface/IController.sol#57) is too similar to ControllerStorageV1.depositTokens (contracts/storage/ControllerStorage.sol#79)
Variable Controller.debtOf(address)._syntheticToken (contracts/Controller.sol#214) is too similar to ControllerStorageV1.syntheticTokens (contracts/storage/ControllerStorage.sol#89)
Variable Controller.addSyntheticToken(address)._syntheticToken (contracts/Controller.sol#369) is too similar to ControllerStorageV1.syntheticTokens (contracts/storage/ControllerStorage.sol#89)
Variable Controller.liquidate(ISyntheticToken,address,uint256,IDepositToken)._syntheticToken (contracts/Controller.sol#275) is too similar to ControllerStorageV1.syntheticTokens (contracts/storage/ControllerStorage.sol#89)
Variable IController.liquidate(ISyntheticToken,address,uint256,IDepositToken)._syntheticToken (contracts/interface/IController.sol#62) is too similar to ControllerStorageV1.syntheticTokens (contracts/storage/ControllerStorage.so
Variable Controller.removeSyntheticToken(ISyntheticToken)._syntheticToken (contracts/Controller.sol#388) is too similar to ControllerStorageV1.syntheticTokens (contracts/storage/ControllerStorage.sol#89)
Variable IController.isSyntheticTokenExists(ISyntheticToken)._syntheticToken (contracts/interface/IController.sol#26) is too similar to ControllerStorageV1.syntheticTokens (contracts/storage/ControllerStorage.sol#89)
Variable Controller.isDepositTokenExists(IDepositToken)._depositToken (contracts/Controller.sol#201) is too similar to ControllerStorageV1.depositTokens (contracts/storage/ControllerStorage.sol#79)
Variable Controller.isSyntheticTokenExists(ISyntheticToken)._syntheticToken (contracts/Controller.sol#192) is too similar to ControllerStorageV1.syntheticTokens (contracts/storage/ControllerStorage.sol#89)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar

initialize(IMasterOracle) should be declared external:
        - Controller.initialize(IMasterOracle) (contracts/Controller.sol#129-142)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

AUTOMATED TESTING

DebtToken.sol

```
DebtToken._addToDebtTokensOfRecipientIfNeeded(address,uint256) (contracts/DebtToken.sol#166-170) uses a dangerous strict equality:
        - _recipientBalanceBefore == 0 (contracts/DebtToken.sol#167)
DebtToken._calculateInterestAccrual() (contracts/DebtToken.sol#204-225) uses a dangerous strict equality:
        - lastTimestampAccrued == _currentTimestamp (contracts/DebtToken.sol#215)
DebtToken._removeFromDebtTokensOfSenderIfNeeded(address,uint256) (contracts/DebtToken.sol#172-176) uses a dangerous strict equality:
        - _senderBalanceAfter == 0 (contracts/DebtToken.sol#173)
DebtToken.accrueInterest() (contracts/DebtToken.sol#231-244) uses a dangerous strict equality:
        - _currentTimestamp == lastTimestampAccrued (contracts/DebtToken.sol#237)
DebtToken.balanceOf(address) (contracts/DebtToken.sol#81-90) uses a dangerous strict equality:
        - principalOf[_account] == 0 (contracts/DebtToken.sol#82)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities

Reentrancy in DebtToken._burn(address,uint256) (contracts/DebtToken.sol#148-164):
        External calls:
        - updateRewardsBeforeMintOrBurn(_account) (contracts/DebtToken.sol#148)
                - _rewardsDistributors[i].updateBeforeMintOrBurn(syntheticToken,_account) (contracts/DebtToken.sol#41)
        State variables written after the call(s):
        - debtIndexOf[_account] = debtIndex (contracts/DebtToken.sol#156)
        - principalOf[_account] = accountBalance - _amount (contracts/DebtToken.sol#155)
        - totalSupply_ -= _amount (contracts/DebtToken.sol#158)
Reentrancy in DebtToken._mint(address,uint256) (contracts/DebtToken.sol#135-146):
        External calls:
        - updateRewardsBeforeMintOrBurn(_account) (contracts/DebtToken.sol#135)
                - _rewardsDistributors[i].updateBeforeMintOrBurn(syntheticToken,_account) (contracts/DebtToken.sol#41)
        State variables written after the call(s):
        - debtIndexOf[_account] = debtIndex (contracts/DebtToken.sol#142)
        - principalOf[_account] += _amount (contracts/DebtToken.sol#141)
        - totalSupply_ += _amount (contracts/DebtToken.sol#140)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2

Reentrancy in DebtToken._burn(address,uint256) (contracts/DebtToken.sol#148-164):
        External calls:
        - updateRewardsBeforeMintOrBurn(_account) (contracts/DebtToken.sol#148)
                - _rewardsDistributors[i].updateBeforeMintOrBurn(syntheticToken,_account) (contracts/DebtToken.sol#41)
        Event emitted after the call(s):
        - Transfer(_account,address(0),_amount) (contracts/DebtToken.sol#161)
Reentrancy in DebtToken._mint(address,uint256) (contracts/DebtToken.sol#135-146):
        External calls:
        - updateRewardsBeforeMintOrBurn(_account) (contracts/DebtToken.sol#135)
                - _rewardsDistributors[i].updateBeforeMintOrBurn(syntheticToken,_account) (contracts/DebtToken.sol#41)
        Event emitted after the call(s):
        - Transfer(address(0),_account,_amount) (contracts/DebtToken.sol#143)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

DebtToken.balanceOf(address) (contracts/DebtToken.sol#81-90) uses timestamp for comparisons
        Dangerous comparisons:
        - principalOf[_account] == 0 (contracts/DebtToken.sol#82)
DebtToken._burn(address,uint256) (contracts/DebtToken.sol#148-164) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(accountBalance >= _amount,burn-amount-exceeds-balance) (contracts/DebtToken.sol#152)
DebtToken._addToDebtTokensOfRecipientIfNeeded(address,uint256) (contracts/DebtToken.sol#166-170) uses timestamp for comparisons
        Dangerous comparisons:
        - _recipientBalanceBefore == 0 (contracts/DebtToken.sol#167)
DebtToken._removeFromDebtTokensOfSenderIfNeeded(address,uint256) (contracts/DebtToken.sol#172-176) uses timestamp for comparisons
        Dangerous comparisons:
        - _senderBalanceAfter == 0 (contracts/DebtToken.sol#173)
DebtToken._calculateInterestAccrual() (contracts/DebtToken.sol#204-225) uses timestamp for comparisons
        Dangerous comparisons:
        - lastTimestampAccrued == _currentTimestamp (contracts/DebtToken.sol#215)
DebtToken.accrueInterest() (contracts/DebtToken.sol#231-244) uses timestamp for comparisons
        Dangerous comparisons:
        - _currentTimestamp == lastTimestampAccrued (contracts/DebtToken.sol#237)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp

Address.isContract(address) (contracts/dependencies/openzeppelin/utils/Address.sol#26-36) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#32-34)
Address.verifyCallResult(bool,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#195-215) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#207-210)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['0.8.9', '^0.8.0']
        - 0.8.9 (contracts/DebtToken.sol#3)
        - 0.8.9 (contracts/access/Manageable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3)
        - 0.8.9 (contracts/interface/IController.sol#3)
        - 0.8.9 (contracts/interface/IDebtToken.sol#3)
        - 0.8.9 (contracts/interface/IDepositToken.sol#3)
        - 0.8.9 (contracts/interface/IGovernable.sol#3)
        - 0.8.9 (contracts/interface/IPausable.sol#3)
        - 0.8.9 (contracts/interface/IRewardsDistributor.sol#3)
        - 0.8.9 (contracts/interface/ISyntheticToken.sol#3)
        - 0.8.9 (contracts/interface/ITreasury.sol#3)
        - 0.8.9 (contracts/interface/oracle/IMasterOracle.sol#3)
        - 0.8.9 (contracts/lib/WadRayMath.sol#3)
        - 0.8.9 (contracts/storage/DebtTokenStorage.sol#3)
        - 0.8.9 (contracts/utils/TokenHolder.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#108-114) is never used and should be removed
Address.functionDelegateCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#168-170) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187) is never used and should be removed
Address.functionStaticCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#141-143) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160) is never used and should be removed
Context._msgData() (contracts/dependencies/openzeppelin/utils/Context.sol#20-22) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#44-57) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#68-79) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#59-66) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#28-35) is never used and should be removed
TokenHolder._requireCanSweep() (contracts/utils/TokenHolder.sol#17) is never used and should be removed
WadRayMath.rayDiv(uint256,uint256) (contracts/lib/WadRayMath.sol#63-65) is never used and should be removed
WadRayMath.rayMul(uint256,uint256) (contracts/lib/WadRayMath.sol#49-55) is never used and should be removed
WadRayMath.rayToWad(uint256) (contracts/lib/WadRayMath.sol#72-76) is never used and should be removed
WadRayMath.wadDiv(uint256,uint256) (contracts/lib/WadRayMath.sol#39-41) is never used and should be removed
WadRayMath.wadToRay(uint256) (contracts/lib/WadRayMath.sol#81-83) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version0.8.9 (contracts/DebtToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/access/Manageable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3) allows old versions
Pragma version0.8.9 (contracts/interface/IController.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IDebtToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IDepositToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IGovernable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IPausable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IRewardsDistributor.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/ISyntheticToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/ITreasury.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/oracle/IMasterOracle.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/lib/WadRayMath.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/storage/DebtTokenStorage.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/utils/TokenHolder.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.9 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#54-59):
        - (success) = recipient.call{value: amount}() (contracts/dependencies/openzeppelin/utils/Address.sol#57)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (contracts/dependencies/openzeppelin/utils/Address.sol#122-133):
        - (success,returndata) = target.call{value: value}(data) (contracts/dependencies/openzeppelin/utils/Address.sol#131)
Low level call in Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160):
        - (success,returndata) = target.staticcall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#158)
Low level call in Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187):
        - (success,returndata) = target.delegatecall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#185)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter DebtToken.initialize(string,string,uint8,IController)._name (contracts/DebtToken.sol#47) is not in mixedCase
Parameter DebtToken.initialize(string,string,uint8,IController)._symbol (contracts/DebtToken.sol#48) is not in mixedCase
Parameter DebtToken.initialize(string,string,uint8,IController)._decimals (contracts/DebtToken.sol#49) is not in mixedCase
Parameter DebtToken.initialize(string,string,uint8,IController)._controller (contracts/DebtToken.sol#50) is not in mixedCase
Parameter DebtToken.setSyntheticToken(ISyntheticToken)._syntheticToken (contracts/DebtToken.sol#64) is not in mixedCase
Parameter DebtToken.balanceOf(address)._account (contracts/DebtToken.sol#81) is not in mixedCase
Parameter DebtToken.mint(address,uint256)._to (contracts/DebtToken.sol#183) is not in mixedCase
Parameter DebtToken.mint(address,uint256)._amount (contracts/DebtToken.sol#183) is not in mixedCase
Parameter DebtToken.burn(address,uint256)._from (contracts/DebtToken.sol#192) is not in mixedCase
Parameter DebtToken.burn(address,uint256)._amount (contracts/DebtToken.sol#192) is not in mixedCase
Function Manageable.__Manageable_init() (contracts/access/Manageable.sol#21) is not in mixedCase
Parameter Manageable.setController(IController)._controller (contracts/access/Manageable.sol#59) is not in mixedCase
Variable Manageable.__gap (contracts/access/Manageable.sol#64) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._token (contracts/utils/TokenHolder.sol#26) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._to (contracts/utils/TokenHolder.sol#27) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._amount (contracts/utils/TokenHolder.sol#28) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Manageable.__gap (contracts/access/Manageable.sol#64) is never used in DebtToken (contracts/DebtToken.sol#12-245)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

initialize(string,string,uint8,IController) should be declared external:
        - DebtToken.initialize(string,string,uint8,IController) (contracts/DebtToken.sol#46-62)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

DepositToken.sol

```
DepositToken._addToDepositTokensOfRecipientIfNeeded(address,uint256) (contracts/DepositToken.sol#211-215) uses a dangerous strict equality:
        - _recipientBalanceBefore == 0 (contracts/DepositToken.sol#212)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities

Reentrancy in DepositToken.deposit(uint256,address) (contracts/DepositToken.sol#228-259):
        External calls:
        - underlying.safeTransferFrom(_sender,address(_treasury),_amount) (contracts/DepositToken.sol#243)
        - _depositFee = controller.depositFee() (contracts/DepositToken.sol#247)
        - _mint(address(_treasury),_feeAmount) (contracts/DepositToken.sol#252)
                - controller.addToDepositTokensOfAccount(_recipient) (contracts/DepositToken.sol#213)
                - _rewardsDistributors[i].updateBeforeMintOrBurn(this,_account) (contracts/DepositToken.sol#70)
        - _mint(_onBehalfOf,_amountToDeposit) (contracts/DepositToken.sol#256)
                - controller.addToDepositTokensOfAccount(_recipient) (contracts/DepositToken.sol#213)
                - _rewardsDistributors[i].updateBeforeMintOrBurn(this,_account) (contracts/DepositToken.sol#70)
        - whenNotPaused() (contracts/DepositToken.sol#231)
                - require(bool,string)(! controller.paused(),paused) (contracts/access/Manageable.sol#40)
        State variables written after the call(s):
        - _mint(_onBehalfOf,_amountToDeposit) (contracts/DepositToken.sol#256)
                - balanceOf[_account] += _amount (contracts/DepositToken.sol#176)
        - _mint(_onBehalfOf,_amountToDeposit) (contracts/DepositToken.sol#256)
                - totalSupply += _amount (contracts/DepositToken.sol#175)
Reentrancy in DepositToken.withdraw(uint256,address) (contracts/DepositToken.sol#266-294):
        External calls:
        - _withdrawFee = controller.withdrawFee() (contracts/DepositToken.sol#281)
        - _transfer(_account,address(_treasury),_feeAmount) (contracts/DepositToken.sol#286)
                - controller.removeFromDepositTokensOfAccount(_sender) (contracts/DepositToken.sol#219)
                - controller.addToDepositTokensOfAccount(_recipient) (contracts/DepositToken.sol#213)
                - _rewardsDistributors[i].updateBeforeTransfer(this,_sender,_recipient) (contracts/DepositToken.sol#83)
        - whenNotShutdown() (contracts/DepositToken.sol#269)
                - require(bool,string)(! controller.everythingStopped(),not-shutdown) (contracts/access/Manageable.sol#45)
        State variables written after the call(s):
        - _transfer(_account,address(_treasury),_feeAmount) (contracts/DepositToken.sol#286)
                - balanceOf[_sender] = _senderBalance - _amount (contracts/DepositToken.sol#152)
                - balanceOf[_recipient] += _amount (contracts/DepositToken.sol#154)
Reentrancy in DepositToken.withdraw(uint256,address) (contracts/DepositToken.sol#266-294):
        External calls:
        - _withdrawFee = controller.withdrawFee() (contracts/DepositToken.sol#281)
        - _transfer(_account,address(_treasury),_feeAmount) (contracts/DepositToken.sol#286)
                - controller.removeFromDepositTokensOfAccount(_sender) (contracts/DepositToken.sol#219)
                - controller.addToDepositTokensOfAccount(_recipient) (contracts/DepositToken.sol#213)
                - _rewardsDistributors[i].updateBeforeTransfer(this,_sender,_recipient) (contracts/DepositToken.sol#83)
        - _burnForWithdraw(_account,_amountToWithdraw) (contracts/DepositToken.sol#290)
                - controller.removeFromDepositTokensOfAccount(_sender) (contracts/DepositToken.sol#219)
                - _rewardsDistributors[i].updateBeforeMintOrBurn(this,_account) (contracts/DepositToken.sol#70)
        - whenNotShutdown() (contracts/DepositToken.sol#269)
                - require(bool,string)(! controller.everythingStopped(),not-shutdown) (contracts/access/Manageable.sol#45)
        State variables written after the call(s):
        - _burnForWithdraw(_account,_amountToWithdraw) (contracts/DepositToken.sol#290)
                - balanceOf[_account] = _accountBalance - _amount (contracts/DepositToken.sol#190)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1

DepositToken.deposit(uint256,address)._feeAmount (contracts/DepositToken.sol#249) is a local variable never initialized
DepositToken.withdraw(uint256,address)._feeAmount (contracts/DepositToken.sol#283) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables

Reentrancy in DepositToken._burn(address,uint256) (contracts/DepositToken.sol#184-197):
        External calls:
        - updateRewardsBeforeMintOrBurn(_account) (contracts/DepositToken.sol#184)
                - _rewardsDistributors[i].updateBeforeMintOrBurn(this,_account) (contracts/DepositToken.sol#70)
        State variables written after the call(s):
        - balanceOf[_account] = _accountBalance - _amount (contracts/DepositToken.sol#190)
        - totalSupply -= _amount (contracts/DepositToken.sol#192)
Reentrancy in DepositToken._mint(address,uint256) (contracts/DepositToken.sol#164-182):
        External calls:
        - updateRewardsBeforeMintOrBurn(_account) (contracts/DepositToken.sol#167)
                - _rewardsDistributors[i].updateBeforeMintOrBurn(this,_account) (contracts/DepositToken.sol#70)
        State variables written after the call(s):
        - balanceOf[_account] += _amount (contracts/DepositToken.sol#176)
        - lastDepositOf[_account] = block.timestamp (contracts/DepositToken.sol#173)
        - totalSupply += _amount (contracts/DepositToken.sol#175)
Reentrancy in DepositToken._transfer(address,address,uint256) (contracts/DepositToken.sol#141-162):
```

```
        External calls:
        - updateRewardsBeforeTransfer(_sender,_recipient) (contracts/DepositToken.sol#145)
                - _rewardsDistributors[i].updateBeforeTransfer(this,_sender,_recipient) (contracts/DepositToken.sol#83)
        State variables written after the call(s):
        - balanceOf[_sender] = _senderBalance - _amount (contracts/DepositToken.sol#152)
        - balanceOf[_recipient] += _amount (contracts/DepositToken.sol#154)
Reentrancy in DepositToken.deposit(uint256,address) (contracts/DepositToken.sol#228-259):
        External calls:
        - underlying.safeTransferFrom(_sender,address(_treasury),_amount) (contracts/DepositToken.sol#243)
        - _depositFee = controller.depositFee() (contracts/DepositToken.sol#247)
        - _mint(address(_treasury),_feeAmount) (contracts/DepositToken.sol#252)
                - controller.addToDepositTokensOfAccount(_recipient) (contracts/DepositToken.sol#213)
                - _rewardsDistributors[i].updateBeforeMintOrBurn(this,_account) (contracts/DepositToken.sol#70)
        - whenNotPaused() (contracts/DepositToken.sol#231)
                - require(bool,string)(! controller.paused(),paused) (contracts/access/Manageable.sol#40)
        State variables written after the call(s):
        - _mint(address(_treasury),_feeAmount) (contracts/DepositToken.sol#252)
                - lastDepositOf[_account] = block.timestamp (contracts/DepositToken.sol#173)
Reentrancy in DepositToken.deposit(uint256,address) (contracts/DepositToken.sol#228-259):
        External calls:
        - underlying.safeTransferFrom(_sender,address(_treasury),_amount) (contracts/DepositToken.sol#243)
        - _depositFee = controller.depositFee() (contracts/DepositToken.sol#247)
        - _mint(address(_treasury),_feeAmount) (contracts/DepositToken.sol#252)
                - controller.addToDepositTokensOfAccount(_recipient) (contracts/DepositToken.sol#213)
                - _rewardsDistributors[i].updateBeforeMintOrBurn(this,_account) (contracts/DepositToken.sol#70)
        - _mint(_onBehalfOf,_amountToDeposit) (contracts/DepositToken.sol#256)
                - controller.addToDepositTokensOfAccount(_recipient) (contracts/DepositToken.sol#213)
                - _rewardsDistributors[i].updateBeforeMintOrBurn(this,_account) (contracts/DepositToken.sol#70)
        - whenNotPaused() (contracts/DepositToken.sol#231)
                - require(bool,string)(! controller.paused(),paused) (contracts/access/Manageable.sol#40)
        State variables written after the call(s):
        - _mint(_onBehalfOf,_amountToDeposit) (contracts/DepositToken.sol#256)
                - lastDepositOf[_account] = block.timestamp (contracts/DepositToken.sol#173)
Reentrancy in DepositToken.deposit(uint256,address) (contracts/DepositToken.sol#228-259):
        External calls:
        - whenNotPaused() (contracts/DepositToken.sol#231)
                - require(bool,string)(! controller.paused(),paused) (contracts/access/Manageable.sol#40)
        State variables written after the call(s):
        - nonReentrant() (contracts/DepositToken.sol#232)
                - _status = _ENTERED (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#56)
                - _status = _NOT_ENTERED (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#62)
Reentrancy in DepositToken.transferFrom(address,address,uint256) (contracts/DepositToken.sol#342-358):
        External calls:
        - _transferWithChecks(_sender,_recipient,_amount) (contracts/DepositToken.sol#347)
                - controller.removeFromDepositTokensOfAccount(_sender) (contracts/DepositToken.sol#219)
                - controller.addToDepositTokensOfAccount(_recipient) (contracts/DepositToken.sol#213)
                - _rewardsDistributors[i].updateBeforeTransfer(this,_sender,_recipient) (contracts/DepositToken.sol#83)
        State variables written after the call(s):
        - _approve(_sender,_msgSender(),currentAllowance - _amount) (contracts/DepositToken.sol#353)
                - allowance[_owner][_spender] = _amount (contracts/DepositToken.sol#207)
Reentrancy in DepositToken.withdraw(uint256,address) (contracts/DepositToken.sol#266-294):
        External calls:
        - whenNotShutdown() (contracts/DepositToken.sol#269)
                - require(bool,string)(! controller.everythingStopped(),not-shutdown) (contracts/access/Manageable.sol#45)
        State variables written after the call(s):
        - nonReentrant() (contracts/DepositToken.sol#270)
                - _status = _ENTERED (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#56)
                - _status = _NOT_ENTERED (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#62)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2

Reentrancy in DepositToken._burn(address,uint256) (contracts/DepositToken.sol#184-197):
        External calls:
        - updateRewardsBeforeMintOrBurn(_account) (contracts/DepositToken.sol#184)
                - _rewardsDistributors[i].updateBeforeMintOrBurn(this,_account) (contracts/DepositToken.sol#70)
        Event emitted after the call(s):
        - Transfer(_account,address(0),_amount) (contracts/DepositToken.sol#194)
Reentrancy in DepositToken._mint(address,uint256) (contracts/DepositToken.sol#164-182):
        External calls:
        - updateRewardsBeforeMintOrBurn(_account) (contracts/DepositToken.sol#167)
                - _rewardsDistributors[i].updateBeforeMintOrBurn(this,_account) (contracts/DepositToken.sol#70)
        Event emitted after the call(s):
        - Transfer(address(0),_account,_amount) (contracts/DepositToken.sol#177)
Reentrancy in DepositToken._transfer(address,address,uint256) (contracts/DepositToken.sol#141-162):
        External calls:
        - updateRewardsBeforeTransfer(_sender,_recipient) (contracts/DepositToken.sol#145)
                - _rewardsDistributors[i].updateBeforeTransfer(this,_sender,_recipient) (contracts/DepositToken.sol#83)
        Event emitted after the call(s):
        - Transfer(_sender,_recipient,_amount) (contracts/DepositToken.sol#156)
Reentrancy in DepositToken.deposit(uint256,address) (contracts/DepositToken.sol#228-259):
        External calls:
        - underlying.safeTransferFrom(_sender,address(_treasury),_amount) (contracts/DepositToken.sol#243)
        - _depositFee = controller.depositFee() (contracts/DepositToken.sol#247)
        - _mint(address(_treasury),_feeAmount) (contracts/DepositToken.sol#252)
                - controller.addToDepositTokensOfAccount(_recipient) (contracts/DepositToken.sol#213)
                - _rewardsDistributors[i].updateBeforeMintOrBurn(this,_account) (contracts/DepositToken.sol#70)
        - whenNotPaused() (contracts/DepositToken.sol#231)
                - require(bool,string)(! controller.paused(),paused) (contracts/access/Manageable.sol#40)
        Event emitted after the call(s):
        - Transfer(address(0),_account,_amount) (contracts/DepositToken.sol#177)
                - _mint(address(_treasury),_feeAmount) (contracts/DepositToken.sol#252)
Reentrancy in DepositToken.deposit(uint256,address) (contracts/DepositToken.sol#228-259):
        External calls:
        - underlying.safeTransferFrom(_sender,address(_treasury),_amount) (contracts/DepositToken.sol#243)
        - _depositFee = controller.depositFee() (contracts/DepositToken.sol#247)
        - _mint(address(_treasury),_feeAmount) (contracts/DepositToken.sol#252)
                - controller.addToDepositTokensOfAccount(_recipient) (contracts/DepositToken.sol#213)
                - _rewardsDistributors[i].updateBeforeMintOrBurn(this,_account) (contracts/DepositToken.sol#70)
        - _mint(_onBehalfOf,_amountToDeposit) (contracts/DepositToken.sol#256)
                - controller.addToDepositTokensOfAccount(_recipient) (contracts/DepositToken.sol#213)
                - _rewardsDistributors[i].updateBeforeMintOrBurn(this,_account) (contracts/DepositToken.sol#70)
        - whenNotPaused() (contracts/DepositToken.sol#231)
                - require(bool,string)(! controller.paused(),paused) (contracts/access/Manageable.sol#40)
        Event emitted after the call(s):
        - CollateralDeposited(_sender,_onBehalfOf,_amount,_feeAmount) (contracts/DepositToken.sol#258)
        - Transfer(address(0),_account,_amount) (contracts/DepositToken.sol#177)
                - _mint(_onBehalfOf,_amountToDeposit) (contracts/DepositToken.sol#256)
Reentrancy in DepositToken.transferFrom(address,address,uint256) (contracts/DepositToken.sol#342-358):
        External calls:
        - _transferWithChecks(_sender,_recipient,_amount) (contracts/DepositToken.sol#347)
                - controller.removeFromDepositTokensOfAccount(_sender) (contracts/DepositToken.sol#219)
                - controller.addToDepositTokensOfAccount(_recipient) (contracts/DepositToken.sol#213)
                - _rewardsDistributors[i].updateBeforeTransfer(this,_sender,_recipient) (contracts/DepositToken.sol#83)
        Event emitted after the call(s):
        - Approval(_owner,_spender,_amount) (contracts/DepositToken.sol#208)
                - _approve(_sender,_msgSender(),currentAllowance - _amount) (contracts/DepositToken.sol#353)
Reentrancy in DepositToken.withdraw(uint256,address) (contracts/DepositToken.sol#266-294):
        External calls:
        - _withdrawFee = controller.withdrawFee() (contracts/DepositToken.sol#281)
        - _transfer(_account,address(_treasury),_feeAmount) (contracts/DepositToken.sol#286)
                - controller.removeFromDepositTokensOfAccount(_sender) (contracts/DepositToken.sol#219)
                - controller.addToDepositTokensOfAccount(_recipient) (contracts/DepositToken.sol#213)
                - _rewardsDistributors[i].updateBeforeTransfer(this,_sender,_recipient) (contracts/DepositToken.sol#83)
        - whenNotShutdown() (contracts/DepositToken.sol#269)
                - require(bool,string)(! controller.everythingStopped(),not-shutdown) (contracts/access/Manageable.sol#45)
        Event emitted after the call(s):
        - Transfer(_sender,_recipient,_amount) (contracts/DepositToken.sol#156)
                - _transfer(_account,address(_treasury),_feeAmount) (contracts/DepositToken.sol#286)
Reentrancy in DepositToken.withdraw(uint256,address) (contracts/DepositToken.sol#266-294):
        External calls:
        - _withdrawFee = controller.withdrawFee() (contracts/DepositToken.sol#281)
        - _transfer(_account,address(_treasury),_feeAmount) (contracts/DepositToken.sol#286)
                - controller.removeFromDepositTokensOfAccount(_sender) (contracts/DepositToken.sol#219)
                - controller.addToDepositTokensOfAccount(_recipient) (contracts/DepositToken.sol#213)
                - _rewardsDistributors[i].updateBeforeTransfer(this,_sender,_recipient) (contracts/DepositToken.sol#83)
        - _burnForWithdraw(_account,_amountToWithdraw) (contracts/DepositToken.sol#290)
                - controller.removeFromDepositTokensOfAccount(_sender) (contracts/DepositToken.sol#219)
                - _rewardsDistributors[i].updateBeforeMintOrBurn(this,_account) (contracts/DepositToken.sol#70)
        - whenNotShutdown() (contracts/DepositToken.sol#269)
                - require(bool,string)(! controller.everythingStopped(),not-shutdown) (contracts/access/Manageable.sol#45)
        Event emitted after the call(s):
        - Transfer(_account,address(0),_amount) (contracts/DepositToken.sol#194)
                - _burnForWithdraw(_account,_amountToWithdraw) (contracts/DepositToken.sol#290)
Reentrancy in DepositToken.withdraw(uint256,address) (contracts/DepositToken.sol#266-294):
        External calls:
```

```
        - _withdrawFee = controller.withdrawFee() (contracts/DepositToken.sol#281)
        - _transfer(_account,address(_treasury),_feeAmount) (contracts/DepositToken.sol#286)
            - controller.removeFromDepositTokensOfAccount(_sender) (contracts/DepositToken.sol#219)
            - controller.addToDepositTokensOfAccount(_recipient) (contracts/DepositToken.sol#213)
            - _rewardsDistributors[i].updateBeforeTransfer(this,_sender,_recipient) (contracts/DepositToken.sol#83)
        - _burnForWithdraw(_account,_amountToWithdraw) (contracts/DepositToken.sol#290)
            - controller.removeFromDepositTokensOfAccount(_sender) (contracts/DepositToken.sol#219)
            - _rewardsDistributors[i].updateBeforeMintOrBurn(this,_account) (contracts/DepositToken.sol#70)
        - _treasury.pull(_to,_amountToWithdraw) (contracts/DepositToken.sol#291)
        - whenNotShutdown() (contracts/DepositToken.sol#269)
            - require(bool,string)(! controller.everythingStopped(),not-shutdown) (contracts/access/Manageable.sol#45)
    Event emitted after the call(s):
        - CollateralWithdrawn(_account,_to,_amount,_feeAmount) (contracts/DepositToken.sol#293)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Address.isContract(address) (contracts/dependencies/openzeppelin/utils/Address.sol#26-36) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#32-34)
Address.verifyCallResult(bool,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#195-215) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#207-210)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['0.8.9', '^0.8.0']
        - 0.8.9 (contracts/DepositToken.sol#3)
        - 0.8.9 (contracts/access/Manageable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/math/Math.sol#3)
        - 0.8.9 (contracts/interface/IController.sol#3)
        - 0.8.9 (contracts/interface/IDebtToken.sol#3)
        - 0.8.9 (contracts/interface/IDepositToken.sol#3)
        - 0.8.9 (contracts/interface/IGovernable.sol#3)
        - 0.8.9 (contracts/interface/IPausable.sol#3)
        - 0.8.9 (contracts/interface/IRewardsDistributor.sol#3)
        - 0.8.9 (contracts/interface/ISyntheticToken.sol#3)
        - 0.8.9 (contracts/interface/ITreasury.sol#3)
        - 0.8.9 (contracts/interface/oracle/IMasterOracle.sol#3)
        - 0.8.9 (contracts/lib/WadRayMath.sol#3)
        - 0.8.9 (contracts/storage/DepositTokenStorage.sol#3)
        - 0.8.9 (contracts/utils/TokenHolder.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#108-114) is never used and should be removed
Address.functionDelegateCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#168-170) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187) is never used and should be removed
Address.functionStaticCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#141-143) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160) is never used and should be removed
Context._msgData() (contracts/dependencies/openzeppelin/utils/Context.sol#20-22) is never used and should be removed
Math.average(uint256,uint256) (contracts/dependencies/openzeppelin/utils/math/Math.sol#27-30) is never used and should be removed
Math.ceilDiv(uint256,uint256) (contracts/dependencies/openzeppelin/utils/math/Math.sol#38-41) is never used and should be removed
Math.max(uint256,uint256) (contracts/dependencies/openzeppelin/utils/math/Math.sol#12-14) is never used and should be removed
ReentrancyGuard.__ReentrancyGuard_init() (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#40-42) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#46-57) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#68-70) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#59-66) is never used and should be removed
TokenHolder._requireCanSweep() (contracts/utils/TokenHolder.sol#17) is never used and should be removed
WadRayMath.rayDiv(uint256,uint256) (contracts/lib/WadRayMath.sol#63-65) is never used and should be removed
WadRayMath.rayMul(uint256,uint256) (contracts/lib/WadRayMath.sol#49-55) is never used and should be removed
WadRayMath.rayToWad(uint256) (contracts/lib/WadRayMath.sol#72-74) is never used and should be removed
WadRayMath.wadToRay(uint256) (contracts/lib/WadRayMath.sol#81-83) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version0.8.9 (contracts/DepositToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/access/Manageable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/math/Math.sol#3) allows old versions
Pragma version0.8.9 (contracts/interface/IController.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IDebtToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IDepositToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IGovernable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IPausable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IRewardsDistributor.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/ISyntheticToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/ITreasury.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/oracle/IMasterOracle.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/lib/WadRayMath.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/storage/DepositTokenStorage.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/utils/TokenHolder.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.9 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#54-59):
        - (success) = recipient.call{value: amount}() (contracts/dependencies/openzeppelin/utils/Address.sol#57)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (contracts/dependencies/openzeppelin/utils/Address.sol#122-133):
        - (success,returndata) = target.call{value: value}(data) (contracts/dependencies/openzeppelin/utils/Address.sol#131)
Low level call in Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160):
        - (success,returndata) = target.staticcall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#158)
Low level call in Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187):
        - (success,returndata) = target.delegatecall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#185)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter DepositToken.initialize(IERC20,IController,string,uint8,uint128,uint256)._underlying (contracts/DepositToken.sol#97) is not in mixedCase
Parameter DepositToken.initialize(IERC20,IController,string,uint8,uint128,uint256)._controller (contracts/DepositToken.sol#98) is not in mixedCase
Parameter DepositToken.initialize(IERC20,IController,string,uint8,uint128,uint256)._symbol (contracts/DepositToken.sol#99) is not in mixedCase
Parameter DepositToken.initialize(IERC20,IController,string,uint8,uint128,uint256)._decimals (contracts/DepositToken.sol#100) is not in mixedCase
Parameter DepositToken.initialize(IERC20,IController,string,uint8,uint128,uint256)._collateralizationRatio (contracts/DepositToken.sol#101) is not in mixedCase
Parameter DepositToken.initialize(IERC20,IController,string,uint8,uint128,uint256)._maxTotalSupplyInUsd (contracts/DepositToken.sol#102) is not in mixedCase
Parameter DepositToken.approve(address,uint256)._amount (contracts/DepositToken.sol#121) is not in mixedCase
Parameter DepositToken.deposit(uint256,address)._amount (contracts/DepositToken.sol#228) is not in mixedCase
Parameter DepositToken.deposit(uint256,address)._onBehalfOf (contracts/DepositToken.sol#228) is not in mixedCase
Parameter DepositToken.withdraw(uint256,address)._amount (contracts/DepositToken.sol#266) is not in mixedCase
Parameter DepositToken.withdraw(uint256,address)._to (contracts/DepositToken.sol#266) is not in mixedCase
Parameter DepositToken.mint(address,uint256)._to (contracts/DepositToken.sol#301) is not in mixedCase
Parameter DepositToken.mint(address,uint256)._amount (contracts/DepositToken.sol#301) is not in mixedCase
Parameter DepositToken.burn(address,uint256)._from (contracts/DepositToken.sol#319) is not in mixedCase
Parameter DepositToken.burn(address,uint256)._amount (contracts/DepositToken.sol#319) is not in mixedCase
Parameter DepositToken.transfer(address,uint256)._to (contracts/DepositToken.sol#337) is not in mixedCase
Parameter DepositToken.transfer(address,uint256)._amount (contracts/DepositToken.sol#337) is not in mixedCase
Parameter DepositToken.transferFrom(address,address,uint256)._sender (contracts/DepositToken.sol#343) is not in mixedCase
Parameter DepositToken.transferFrom(address,address,uint256)._recipient (contracts/DepositToken.sol#344) is not in mixedCase
Parameter DepositToken.transferFrom(address,address,uint256)._amount (contracts/DepositToken.sol#345) is not in mixedCase
Parameter DepositToken.unlockedBalanceOf(address)._account (contracts/DepositToken.sol#365) is not in mixedCase
Parameter DepositToken.lockedBalanceOf(address)._account (contracts/DepositToken.sol#382) is not in mixedCase
Parameter DepositToken.seize(address,address,uint256)._from (contracts/DepositToken.sol#396) is not in mixedCase
Parameter DepositToken.seize(address,address,uint256)._to (contracts/DepositToken.sol#397) is not in mixedCase
Parameter DepositToken.seize(address,address,uint256)._amount (contracts/DepositToken.sol#398) is not in mixedCase
Parameter DepositToken.updateCollateralizationRatio(uint128)._newCollateralizationRatio (contracts/DepositToken.sol#407) is not in mixedCase
Parameter DepositToken.updateMinDepositTime(uint256)._newMinDepositTime (contracts/DepositToken.sol#418) is not in mixedCase
Parameter DepositToken.updateMaxTotalSupplyInUsd(uint256)._newMaxTotalSupplyInUsd (contracts/DepositToken.sol#429) is not in mixedCase
Function Manageable.__Manageable_init() (contracts/access/Manageable.sol#21) is not in mixedCase
Parameter Manageable.setController(IController)._controller (contracts/access/Manageable.sol#59) is not in mixedCase
Variable Manageable.__gap (contracts/access/Manageable.sol#64) is not in mixedCase
Function ReentrancyGuard.__ReentrancyGuard_init() (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#40-42) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._token (contracts/utils/TokenHolder.sol#26) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._to (contracts/utils/TokenHolder.sol#27) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._amount (contracts/utils/TokenHolder.sol#28) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Manageable.__gap (contracts/access/Manageable.sol#64) is never used in DepositToken (contracts/DepositToken.sol#14-444)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

initialize(IERC20,IController,string,uint8,uint128,uint256) should be declared external:


        - DepositToken.initialize(IERC20,IController,string,uint8,uint128,uint256) (contracts/DepositToken.sol#96-119)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

NativeTokenGateway.sol

```
Address.isContract(address) (contracts/dependencies/openzeppelin/utils/Address.sol#26-36) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#32-34)
Address.verifyCallResult(bool,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#195-215) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#207-210)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['0.8.9', '^0.8.0']
        - 0.8.9 (contracts/NativeTokenGateway.sol#3)
        - 0.8.9 (contracts/access/Governable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3)
        - 0.8.9 (contracts/interface/IController.sol#3)
        - 0.8.9 (contracts/interface/IDebtToken.sol#3)
        - 0.8.9 (contracts/interface/IDepositToken.sol#3)
        - 0.8.9 (contracts/interface/IGovernable.sol#3)
        - 0.8.9 (contracts/interface/INativeTokenGateway.sol#3)
        - 0.8.9 (contracts/interface/IPausable.sol#3)
        - 0.8.9 (contracts/interface/IRewardsDistributor.sol#3)
        - 0.8.9 (contracts/interface/ISyntheticToken.sol#3)
        - 0.8.9 (contracts/interface/ITreasury.sol#3)
        - 0.8.9 (contracts/interface/external/IWETH.sol#3)
        - 0.8.9 (contracts/interface/oracle/IMasterOracle.sol#3)
        - 0.8.9 (contracts/utils/TokenHolder.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#108-114) is never used and should be removed
Address.functionDelegateCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#168-170) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187) is never used and should be removed
Address.functionStaticCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#141-143) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160) is never used and should be removed
Context._msgData() (contracts/dependencies/openzeppelin/utils/Context.sol#20-22) is never used and should be removed
Governable.__Governable_init() (contracts/access/Governable.sol#39-43) is never used and should be removed
ReentrancyGuard.__ReentrancyGuard_init() (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#40-42) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#68-79) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#59-66) is never used and should be removed
TokenHolder._requireCanSweep() (contracts/utils/TokenHolder.sol#17) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version0.8.9 (contracts/NativeTokenGateway.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/access/Governable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3) allows old versions
Pragma version0.8.9 (contracts/interface/IController.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IDebtToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IDepositToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IGovernable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/INativeTokenGateway.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IPausable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IRewardsDistributor.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/ISyntheticToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/ITreasury.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/external/IWETH.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/oracle/IMasterOracle.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/utils/TokenHolder.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.9 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#54-59):
        - (success) = recipient.call{value: amount}() (contracts/dependencies/openzeppelin/utils/Address.sol#57)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (contracts/dependencies/openzeppelin/utils/Address.sol#122-133):
        - (success,returndata) = target.call{value: value}(data) (contracts/dependencies/openzeppelin/utils/Address.sol#131)
Low level call in Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160):
        - (success,returndata) = target.staticcall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#158)
Low level call in Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187):
        - (success,returndata) = target.delegatecall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#185)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter NativeTokenGateway.deposit(IController)._controller (contracts/NativeTokenGateway.sol#29) is not in mixedCase
Parameter NativeTokenGateway.withdraw(IController,uint256)._controller (contracts/NativeTokenGateway.sol#41) is not in mixedCase
Parameter NativeTokenGateway.withdraw(IController,uint256)._amount (contracts/NativeTokenGateway.sol#41) is not in mixedCase
Function Governable.__Governable_init() (contracts/access/Governable.sol#39-43) is not in mixedCase
Parameter Governable.transferGovernorship(address)._proposedGovernor (contracts/access/Governable.sol#59) is not in mixedCase
Function ReentrancyGuard.__ReentrancyGuard_init() (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#40-42) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._token (contracts/utils/TokenHolder.sol#26) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._to (contracts/utils/TokenHolder.sol#27) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._amount (contracts/utils/TokenHolder.sol#28) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

Pausable.sol

```
Address.isContract(address) (contracts/dependencies/openzeppelin/utils/Address.sol#26-36) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#32-34)
Address.verifyCallResult(bool,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#195-215) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#207-210)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['0.8.9', '^0.8.0']
        - 0.8.9 (contracts/Pausable.sol#3)
        - 0.8.9 (contracts/access/Governable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3)
        - 0.8.9 (contracts/interface/IGovernable.sol#3)
        - 0.8.9 (contracts/interface/IPausable.sol#3)
        - 0.8.9 (contracts/utils/TokenHolder.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#108-114) is never used and should be removed
Address.functionDelegateCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#168-170) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187) is never used and should be removed
Address.functionStaticCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#141-143) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160) is never used and should be removed
Context._msgData() (contracts/dependencies/openzeppelin/utils/Context.sol#20-22) is never used and should be removed
Governable.__Governable_init() (contracts/access/Governable.sol#39-43) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#44-57) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#68-79) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#59-66) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#28-35) is never used and should be removed
TokenHolder._requireCanSweep() (contracts/utils/TokenHolder.sol#17) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version0.8.9 (contracts/Pausable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/access/Governable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3) allows old versions
Pragma version0.8.9 (contracts/interface/IGovernable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IPausable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/utils/TokenHolder.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.9 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#54-59):
        - (success) = recipient.call{value: amount}() (contracts/dependencies/openzeppelin/utils/Address.sol#57)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (contracts/dependencies/openzeppelin/utils/Address.sol#122-133):
        - (success,returndata) = target.call{value: value}(data) (contracts/dependencies/openzeppelin/utils/Address.sol#131)
Low level call in Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160):
        - (success,returndata) = target.staticcall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#158)
Low level call in Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187):
        - (success,returndata) = target.delegatecall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#185)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function Governable.__Governable_init() (contracts/access/Governable.sol#39-43) is not in mixedCase
Parameter Governable.transferGovernorship(address)._proposedGovernor (contracts/access/Governable.sol#59) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._token (contracts/utils/TokenHolder.sol#26) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._to (contracts/utils/TokenHolder.sol#27) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._amount (contracts/utils/TokenHolder.sol#28) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

SyntheticToken.sol

```
SyntheticToken.repay(address,uint256) (contracts/SyntheticToken.sol#229-260) uses a dangerous strict equality:
        - require(bool,string)(_newDebtInUsd == 0 || _newDebtInUsd >= _debtFloorInUsd,debt-lt-floor) (contracts/SyntheticToken.sol#253)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities

Reentrancy in SyntheticToken.issue(uint256,address) (contracts/SyntheticToken.sol#179-221):
        External calls:
        - accrueInterest() (contracts/SyntheticToken.sol#191)
                - _interestAmountAccrued = debtToken.accrueInterest() (contracts/SyntheticToken.sol#315)
        - _debtFloorInUsd = controller.debtFloorInUsd() (contracts/SyntheticToken.sol#199)
        - _issueFee = controller.issueFee() (contracts/SyntheticToken.sol#208)
        - whenNotShutdown() (contracts/SyntheticToken.sol#182)
                - require(bool,string)(! controller.everythingStopped(),not-shutdown) (contracts/access/Manageable.sol#45)
        State variables written after the call(s):
        - _mint(address(controller.treasury()),_feeAmount) (contracts/SyntheticToken.sol#213)
                - balanceOf[account] += amount (contracts/SyntheticToken.sol#146)
        - _mint(_to,_amountToIssue) (contracts/SyntheticToken.sol#217)
                - balanceOf[account] += amount (contracts/SyntheticToken.sol#146)
        - _mint(address(controller.treasury()),_feeAmount) (contracts/SyntheticToken.sol#213)
                - totalSupply += amount (contracts/SyntheticToken.sol#145)
        - _mint(_to,_amountToIssue) (contracts/SyntheticToken.sol#217)
                - totalSupply += amount (contracts/SyntheticToken.sol#145)
Reentrancy in SyntheticToken.repay(address,uint256) (contracts/SyntheticToken.sol#229-260):
        External calls:
        - accrueInterest() (contracts/SyntheticToken.sol#232)
                - _interestAmountAccrued = debtToken.accrueInterest() (contracts/SyntheticToken.sol#315)
        - _repayFee = controller.repayFee() (contracts/SyntheticToken.sol#236)
        - whenNotShutdown() (contracts/SyntheticToken.sol#229)
                - require(bool,string)(! controller.everythingStopped(),not-shutdown) (contracts/access/Manageable.sol#45)
        State variables written after the call(s):
        - _transfer(_payer,address(controller.treasury()),_feeAmount) (contracts/SyntheticToken.sol#243)
                - balanceOf[sender] = senderBalance - amount (contracts/SyntheticToken.sol#133)
                - balanceOf[recipient] += amount (contracts/SyntheticToken.sol#135)
Reentrancy in SyntheticToken.repay(address,uint256) (contracts/SyntheticToken.sol#229-260):
        External calls:
        - accrueInterest() (contracts/SyntheticToken.sol#232)
                - _interestAmountAccrued = debtToken.accrueInterest() (contracts/SyntheticToken.sol#315)
        - _repayFee = controller.repayFee() (contracts/SyntheticToken.sol#236)
        - _debtFloorInUsd = controller.debtFloorInUsd() (contracts/SyntheticToken.sol#246)
        - whenNotShutdown() (contracts/SyntheticToken.sol#229)
                - require(bool,string)(! controller.everythingStopped(),not-shutdown) (contracts/access/Manageable.sol#45)
        State variables written after the call(s):
        - _burn(_payer,_amountToRepay) (contracts/SyntheticToken.sol#256)
                - balanceOf[account] = accountBalance - amount (contracts/SyntheticToken.sol#156)
        - _burn(_payer,_amountToRepay) (contracts/SyntheticToken.sol#256)
                - totalSupply -= amount (contracts/SyntheticToken.sol#158)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1

SyntheticToken.repay(address,uint256)._feeAmount (contracts/SyntheticToken.sol#238) is a local variable never initialized
SyntheticToken.issue(uint256,address)._feeAmount (contracts/SyntheticToken.sol#210) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables

Reentrancy in SyntheticToken.accrueInterest() (contracts/SyntheticToken.sol#314-321):
        External calls:
        - _interestAmountAccrued = debtToken.accrueInterest() (contracts/SyntheticToken.sol#315)
        State variables written after the call(s):
        - _mint(address(controller.treasury()),_interestAmountAccrued) (contracts/SyntheticToken.sol#319)
                - balanceOf[account] += amount (contracts/SyntheticToken.sol#146)
        - _mint(address(controller.treasury()),_interestAmountAccrued) (contracts/SyntheticToken.sol#319)
                - totalSupply += amount (contracts/SyntheticToken.sol#145)
Reentrancy in SyntheticToken.issue(uint256,address) (contracts/SyntheticToken.sol#179-221):
        External calls:
        - whenNotShutdown() (contracts/SyntheticToken.sol#182)
                - require(bool,string)(! controller.everythingStopped(),not-shutdown) (contracts/access/Manageable.sol#45)
        State variables written after the call(s):
        - nonReentrant() (contracts/SyntheticToken.sol#183)
                - _status = _ENTERED (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#56)
                - _status = _NOT_ENTERED (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#62)
Reentrancy in SyntheticToken.repay(address,uint256) (contracts/SyntheticToken.sol#229-260):
        External calls:
        - whenNotShutdown() (contracts/SyntheticToken.sol#229)
                - require(bool,string)(! controller.everythingStopped(),not-shutdown) (contracts/access/Manageable.sol#45)
        State variables written after the call(s):
        - nonReentrant() (contracts/SyntheticToken.sol#229)
                - _status = _ENTERED (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#56)
                - _status = _NOT_ENTERED (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#62)
Reentrancy in SyntheticToken.updateInterestRate(uint256) (contracts/SyntheticToken.sol#303-309):
        External calls:
        - accrueInterest() (contracts/SyntheticToken.sol#304)
                - _interestAmountAccrued = debtToken.accrueInterest() (contracts/SyntheticToken.sol#315)
        State variables written after the call(s):
        - interestRate = _newInterestRate (contracts/SyntheticToken.sol#308)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2

Reentrancy in SyntheticToken.accrueInterest() (contracts/SyntheticToken.sol#314-321):
        External calls:
        - _interestAmountAccrued = debtToken.accrueInterest() (contracts/SyntheticToken.sol#315)
        Event emitted after the call(s):
        - Transfer(address(0),account,amount) (contracts/SyntheticToken.sol#147)
                - _mint(address(controller.treasury()),_interestAmountAccrued) (contracts/SyntheticToken.sol#319)
Reentrancy in SyntheticToken.issue(uint256,address) (contracts/SyntheticToken.sol#179-221):
        External calls:
        - accrueInterest() (contracts/SyntheticToken.sol#191)
                - _interestAmountAccrued = debtToken.accrueInterest() (contracts/SyntheticToken.sol#315)
        - whenNotShutdown() (contracts/SyntheticToken.sol#182)
                - require(bool,string)(! controller.everythingStopped(),not-shutdown) (contracts/access/Manageable.sol#45)
        Event emitted after the call(s):
        - Transfer(address(0),account,amount) (contracts/SyntheticToken.sol#147)
                - accrueInterest() (contracts/SyntheticToken.sol#191)
Reentrancy in SyntheticToken.issue(uint256,address) (contracts/SyntheticToken.sol#179-221):
        External calls:
        - accrueInterest() (contracts/SyntheticToken.sol#191)
                - _interestAmountAccrued = debtToken.accrueInterest() (contracts/SyntheticToken.sol#315)
        - _debtFloorInUsd = controller.debtFloorInUsd() (contracts/SyntheticToken.sol#199)
        - _issueFee = controller.issueFee() (contracts/SyntheticToken.sol#208)
        - whenNotShutdown() (contracts/SyntheticToken.sol#182)
                - require(bool,string)(! controller.everythingStopped(),not-shutdown) (contracts/access/Manageable.sol#45)
        Event emitted after the call(s):
        - Transfer(address(0),account,amount) (contracts/SyntheticToken.sol#147)
                - _mint(address(controller.treasury()),_feeAmount) (contracts/SyntheticToken.sol#213)
        - Transfer(address(0),account,amount) (contracts/SyntheticToken.sol#147)
                - _mint(_to,_amountToIssue) (contracts/SyntheticToken.sol#217)
Reentrancy in SyntheticToken.issue(uint256,address) (contracts/SyntheticToken.sol#179-221):
        External calls:
        - accrueInterest() (contracts/SyntheticToken.sol#191)
                - _interestAmountAccrued = debtToken.accrueInterest() (contracts/SyntheticToken.sol#315)
        - _debtFloorInUsd = controller.debtFloorInUsd() (contracts/SyntheticToken.sol#199)
        - _issueFee = controller.issueFee() (contracts/SyntheticToken.sol#208)
        - debtToken.mint(_account,_amount) (contracts/SyntheticToken.sol#218)
        - whenNotShutdown() (contracts/SyntheticToken.sol#182)
                - require(bool,string)(! controller.everythingStopped(),not-shutdown) (contracts/access/Manageable.sol#45)
        Event emitted after the call(s):
        - SyntheticTokenIssued(_account,_to,_amount,_feeAmount) (contracts/SyntheticToken.sol#220)
Reentrancy in SyntheticToken.repay(address,uint256) (contracts/SyntheticToken.sol#229-260):
        External calls:
        - accrueInterest() (contracts/SyntheticToken.sol#232)
                - _interestAmountAccrued = debtToken.accrueInterest() (contracts/SyntheticToken.sol#315)
        - whenNotShutdown() (contracts/SyntheticToken.sol#229)
                - require(bool,string)(! controller.everythingStopped(),not-shutdown) (contracts/access/Manageable.sol#45)
        Event emitted after the call(s):
        - Transfer(address(0),account,amount) (contracts/SyntheticToken.sol#147)
                - accrueInterest() (contracts/SyntheticToken.sol#232)
Reentrancy in SyntheticToken.repay(address,uint256) (contracts/SyntheticToken.sol#229-260):
        External calls:
        - accrueInterest() (contracts/SyntheticToken.sol#232)
                - _interestAmountAccrued = debtToken.accrueInterest() (contracts/SyntheticToken.sol#315)
        - _repayFee = controller.repayFee() (contracts/SyntheticToken.sol#236)
        - whenNotShutdown() (contracts/SyntheticToken.sol#229)
                - require(bool,string)(! controller.everythingStopped(),not-shutdown) (contracts/access/Manageable.sol#45)
        Event emitted after the call(s):
        - Transfer(sender,recipient,amount) (contracts/SyntheticToken.sol#137)
                - _transfer(_payer,address(controller.treasury()),_feeAmount) (contracts/SyntheticToken.sol#243)
Reentrancy in SyntheticToken.repay(address,uint256) (contracts/SyntheticToken.sol#229-260):
        External calls:
```

```
        - accrueInterest() (contracts/SyntheticToken.sol#232)
                - _interestAmountAccrued = debtToken.accrueInterest() (contracts/SyntheticToken.sol#315)
        - _repayFee = controller.repayFee() (contracts/SyntheticToken.sol#236)
        - _debtFloorInUsd = controller.debtFloorInUsd() (contracts/SyntheticToken.sol#246)
        - whenNotShutdown() (contracts/SyntheticToken.sol#229)
                - require(bool,string)(! controller.everythingStopped(),not-shutdown) (contracts/access/Manageable.sol#45)
        Event emitted after the call(s):
        - Transfer(account,address(0),amount) (contracts/SyntheticToken.sol#160)
                - _burn(_payer,_amountToRepay) (contracts/SyntheticToken.sol#256)
Reentrancy in SyntheticToken.repay(address,uint256) (contracts/SyntheticToken.sol#229-260):
        External calls:
        - accrueInterest() (contracts/SyntheticToken.sol#232)
                - _interestAmountAccrued = debtToken.accrueInterest() (contracts/SyntheticToken.sol#315)
        - _repayFee = controller.repayFee() (contracts/SyntheticToken.sol#236)
        - _debtFloorInUsd = controller.debtFloorInUsd() (contracts/SyntheticToken.sol#246)
        - debtToken.burn(_onBehalfOf,_amountToRepay) (contracts/SyntheticToken.sol#257)
        - whenNotShutdown() (contracts/SyntheticToken.sol#229)
                - require(bool,string)(! controller.everythingStopped(),not-shutdown) (contracts/access/Manageable.sol#45)
        Event emitted after the call(s):
        - DebtRepaid(_payer,_onBehalfOf,_amount,_feeAmount) (contracts/SyntheticToken.sol#259)
Reentrancy in SyntheticToken.updateInterestRate(uint256) (contracts/SyntheticToken.sol#303-309):
        External calls:
        - accrueInterest() (contracts/SyntheticToken.sol#304)
                - _interestAmountAccrued = debtToken.accrueInterest() (contracts/SyntheticToken.sol#315)
        Event emitted after the call(s):
        - InterestRateUpdated(_currentInterestRate,_newInterestRate) (contracts/SyntheticToken.sol#307)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Address.isContract(address) (contracts/dependencies/openzeppelin/utils/Address.sol#26-36) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#32-34)
Address.verifyCallResult(bool,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#195-215) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#207-210)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['0.8.9', '^0.8.0']
        - 0.8.9 (contracts/SyntheticToken.sol#3)
        - 0.8.9 (contracts/access/Manageable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3)
        - 0.8.9 (contracts/interface/IController.sol#3)
        - 0.8.9 (contracts/interface/IDebtToken.sol#3)
        - 0.8.9 (contracts/interface/IDepositToken.sol#3)
        - 0.8.9 (contracts/interface/IGovernable.sol#3)
        - 0.8.9 (contracts/interface/IPausable.sol#3)
        - 0.8.9 (contracts/interface/IRewardsDistributor.sol#3)
        - 0.8.9 (contracts/interface/ISyntheticToken.sol#3)
        - 0.8.9 (contracts/interface/ITreasury.sol#3)
        - 0.8.9 (contracts/interface/oracle/IMasterOracle.sol#3)
        - 0.8.9 (contracts/lib/WadRayMath.sol#3)
        - 0.8.9 (contracts/storage/SyntheticTokenStorage.sol#3)
        - 0.8.9 (contracts/utils/TokenHolder.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#108-114) is never used and should be removed
Address.functionDelegateCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#168-170) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187) is never used and should be removed
Address.functionStaticCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#141-143) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160) is never used and should be removed
Context._msgData() (contracts/dependencies/openzeppelin/utils/Context.sol#20-22) is never used and should be removed
ReentrancyGuard.__ReentrancyGuard_init() (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#40-42) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#44-57) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#68-79) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#59-66) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#26-35) is never used and should be removed
TokenHolder._requireCanSweep() (contracts/utils/TokenHolder.sol#17) is never used and should be removed
WadRayMath.rayDiv(uint256,uint256) (contracts/lib/WadRayMath.sol#63-65) is never used and should be removed
WadRayMath.rayMul(uint256,uint256) (contracts/lib/WadRayMath.sol#49-55) is never used and should be removed
WadRayMath.rayToWad(uint256) (contracts/lib/WadRayMath.sol#72-74) is never used and should be removed
WadRayMath.wadToRay(uint256) (contracts/lib/WadRayMath.sol#81-83) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version0.8.9 (contracts/SyntheticToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/access/Manageable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3) allows old versions
Pragma version0.8.9 (contracts/interface/IController.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IDebtToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IDepositToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IGovernable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IPausable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IRewardsDistributor.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/ISyntheticToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/ITreasury.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/oracle/IMasterOracle.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/lib/WadRayMath.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/storage/SyntheticTokenStorage.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/utils/TokenHolder.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.9 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#54-59):
        - (success) = recipient.call{value: amount}() (contracts/dependencies/openzeppelin/utils/Address.sol#57)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (contracts/dependencies/openzeppelin/utils/Address.sol#122-133):
        - (success,returndata) = target.call{value: value}(data) (contracts/dependencies/openzeppelin/utils/Address.sol#131)
Low level call in Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160):
        - (success,returndata) = target.staticcall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#158)
Low level call in Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187):
        - (success,returndata) = target.delegatecall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#185)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter SyntheticToken.initialize(string,string,uint8,IController,IDebtToken,uint256,uint256)._name (contracts/SyntheticToken.sol#43) is not in mixedCase
Parameter SyntheticToken.initialize(string,string,uint8,IController,IDebtToken,uint256,uint256)._symbol (contracts/SyntheticToken.sol#44) is not in mixedCase
Parameter SyntheticToken.initialize(string,string,uint8,IController,IDebtToken,uint256,uint256)._decimals (contracts/SyntheticToken.sol#45) is not in mixedCase
Parameter SyntheticToken.initialize(string,string,uint8,IController,IDebtToken,uint256,uint256)._controller (contracts/SyntheticToken.sol#46) is not in mixedCase
Parameter SyntheticToken.initialize(string,string,uint8,IController,IDebtToken,uint256,uint256)._debtToken (contracts/SyntheticToken.sol#47) is not in mixedCase
Parameter SyntheticToken.initialize(string,string,uint8,IController,IDebtToken,uint256,uint256)._interestRate (contracts/SyntheticToken.sol#48) is not in mixedCase
Parameter SyntheticToken.initialize(string,string,uint8,IController,IDebtToken,uint256,uint256)._maxTotalSupplyInUsd (contracts/SyntheticToken.sol#49) is not in mixedCase
Parameter SyntheticToken.issue(uint256,address)._amount (contracts/SyntheticToken.sol#179) is not in mixedCase
Parameter SyntheticToken.issue(uint256,address)._to (contracts/SyntheticToken.sol#179) is not in mixedCase
Parameter SyntheticToken.repay(address,uint256)._onBehalfOf (contracts/SyntheticToken.sol#229) is not in mixedCase
Parameter SyntheticToken.repay(address,uint256)._amount (contracts/SyntheticToken.sol#229) is not in mixedCase
Parameter SyntheticToken.mint(address,uint256)._to (contracts/SyntheticToken.sol#267) is not in mixedCase
Parameter SyntheticToken.mint(address,uint256)._amount (contracts/SyntheticToken.sol#267) is not in mixedCase
Parameter SyntheticToken.burn(address,uint256)._from (contracts/SyntheticToken.sol#276) is not in mixedCase
Parameter SyntheticToken.burn(address,uint256)._amount (contracts/SyntheticToken.sol#276) is not in mixedCase
Parameter SyntheticToken.updateMaxTotalSupplyInUsd(uint256)._newMaxTotalSupplyInUsd (contracts/SyntheticToken.sol#284) is not in mixedCase
Parameter SyntheticToken.updateInterestRate(uint256)._newInterestRate (contracts/SyntheticToken.sol#303) is not in mixedCase
Function Manageable.__Manageable_init() (contracts/access/Manageable.sol#21) is not in mixedCase
Parameter Manageable.setController(IController)._controller (contracts/access/Manageable.sol#59) is not in mixedCase
Variable Manageable.__gap (contracts/access/Manageable.sol#64) is not in mixedCase
Function ReentrancyGuard.__ReentrancyGuard_init() (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#40-42) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._token (contracts/utils/TokenHolder.sol#26) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._to (contracts/utils/TokenHolder.sol#27) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._amount (contracts/utils/TokenHolder.sol#28) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Manageable.__gap (contracts/access/Manageable.sol#64) is never used in SyntheticToken (contracts/SyntheticToken.sol#13-322)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

initialize(string,string,uint8,IController,IDebtToken,uint256,uint256) should be declared external:
        - SyntheticToken.initialize(string,string,uint8,IController,IDebtToken,uint256,uint256) (contracts/SyntheticToken.sol#42-64)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

Treasury.sol

```
Treasury.migrateTo(address).i (contracts/Treasury.sol#51) is a local variable never initialized
Treasury.migrateTo(address).i_scope_0 (contracts/Treasury.sol#69) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables

Treasury.migrateTo(address) (contracts/Treasury.sol#47-76) has external calls inside a loop: _underlying = _depositToken.underlying() (contracts/Treasury.sol#53)
Treasury.migrateTo(address) (contracts/Treasury.sol#47-76) has external calls inside a loop: _balance = _depositToken.balanceOf(address(this)) (contracts/Treasury.sol#55)
Treasury.migrateTo(address) (contracts/Treasury.sol#47-76) has external calls inside a loop: _underlyingBalance = _underlying.balanceOf(address(this)) (contracts/Treasury.sol#56)
Treasury.migrateTo(address) (contracts/Treasury.sol#47-76) has external calls inside a loop: _balance_scope_1 = _vsAsset.balanceOf(address(this)) (contracts/Treasury.sol#71)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#calls-inside-a-loop

Address.isContract(address) (contracts/dependencies/openzeppelin/utils/Address.sol#26-36) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#32-34)
Address.verifyCallResult(bool,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#195-215) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#207-210)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['0.8.9', '^0.8.0']
        - 0.8.9 (contracts/Treasury.sol#3)
        - 0.8.9 (contracts/access/Manageable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3)
        - 0.8.9 (contracts/interface/IController.sol#3)
        - 0.8.9 (contracts/interface/IDebtToken.sol#3)
        - 0.8.9 (contracts/interface/IDepositToken.sol#3)
        - 0.8.9 (contracts/interface/IGovernable.sol#3)
        - 0.8.9 (contracts/interface/IPausable.sol#3)
        - 0.8.9 (contracts/interface/IRewardsDistributor.sol#3)
        - 0.8.9 (contracts/interface/ISyntheticToken.sol#3)
        - 0.8.9 (contracts/interface/ITreasury.sol#3)
        - 0.8.9 (contracts/interface/oracle/IMasterOracle.sol#3)
        - 0.8.9 (contracts/storage/TreasuryStorage.sol#3)
        - 0.8.9 (contracts/utils/TokenHolder.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#108-114) is never used and should be removed
Address.functionDelegateCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#168-170) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187) is never used and should be removed
Address.functionStaticCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#141-143) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160) is never used and should be removed
Context._msgData() (contracts/dependencies/openzeppelin/utils/Context.sol#20-22) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#44-57) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#68-79) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#59-66) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#28-35) is never used and should be removed
TokenHolder._requireCanSweep() (contracts/utils/TokenHolder.sol#17) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version0.8.9 (contracts/Treasury.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/access/Manageable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3) allows old versions
Pragma version0.8.9 (contracts/interface/IController.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IDebtToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IDepositToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IGovernable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IPausable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IRewardsDistributor.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/ISyntheticToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/ITreasury.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/oracle/IMasterOracle.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/storage/TreasuryStorage.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/utils/TokenHolder.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.9 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#54-59):
        - (success) = recipient.call{value: amount}() (contracts/dependencies/openzeppelin/utils/Address.sol#57)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (contracts/dependencies/openzeppelin/utils/Address.sol#122-133):
        - (success,returndata) = target.call{value: value}(data) (contracts/dependencies/openzeppelin/utils/Address.sol#131)
Low level call in Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160):
        - (success,returndata) = target.staticcall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#158)
Low level call in Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187):
        - (success,returndata) = target.delegatecall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#185)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter Treasury.initialize(IController)._controller (contracts/Treasury.sol#26) is not in mixedCase
Parameter Treasury.pull(address,uint256)._to (contracts/Treasury.sol#38) is not in mixedCase
Parameter Treasury.pull(address,uint256)._amount (contracts/Treasury.sol#38) is not in mixedCase
Parameter Treasury.migrateTo(address)._newTreasury (contracts/Treasury.sol#47) is not in mixedCase
Function Manageable.__Manageable_init() (contracts/access/Manageable.sol#21) is not in mixedCase
Parameter Manageable.setController(IController)._controller (contracts/access/Manageable.sol#59) is not in mixedCase
Variable Manageable.__gap (contracts/access/Manageable.sol#64) is not in mixedCase
Function ReentrancyGuard.__ReentrancyGuard_init() (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#40-42) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._token (contracts/utils/TokenHolder.sol#26) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._to (contracts/utils/TokenHolder.sol#27) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._amount (contracts/utils/TokenHolder.sol#28) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Manageable.__gap (contracts/access/Manageable.sol#64) is never used in Treasury (contracts/Treasury.sol#12-77)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

initialize(IController) should be declared external:
        - Treasury.initialize(IController) (contracts/Treasury.sol#26-33)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

TokenHolder.sol

```
Address.isContract(address) (contracts/dependencies/openzeppelin/utils/Address.sol#26-36) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#32-34)
Address.verifyCallResult(bool,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#195-215) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#207-210)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['0.8.9', '^0.8.0']
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3)
        - 0.8.9 (contracts/utils/TokenHolder.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#108-114) is never used and should be removed
Address.functionDelegateCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#168-170) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187) is never used and should be removed
Address.functionStaticCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#141-143) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#44-57) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#68-79) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#59-66) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#28-35) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3) allows old versions
Pragma version0.8.9 (contracts/utils/TokenHolder.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.9 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#54-59):
        - (success) = recipient.call{value: amount}() (contracts/dependencies/openzeppelin/utils/Address.sol#57)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (contracts/dependencies/openzeppelin/utils/Address.sol#122-133):
        - (success,returndata) = target.call{value: value}(data) (contracts/dependencies/openzeppelin/utils/Address.sol#131)
Low level call in Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160):
        - (success,returndata) = target.staticcall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#158)
Low level call in Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187):
        - (success,returndata) = target.delegatecall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#185)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter TokenHolder.sweep(IERC20,address,uint256)._token (contracts/utils/TokenHolder.sol#26) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._to (contracts/utils/TokenHolder.sol#27) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._amount (contracts/utils/TokenHolder.sol#28) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

TokenHolder (contracts/utils/TokenHolder.sol#11-52) does not implement functions:
        - TokenHolder._requireCanSweep() (contracts/utils/TokenHolder.sol#17)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unimplemented-functions
```

CTokenOracle.sol

```
CTokenOracle.getPriceInUsd(IERC20) (contracts/oracle/ib/CTokenOracle.sol#37-50) performs a multiplication on the result of a division:
        -_underlyingAmount = (ONE_CTOKEN * ICToken(address(_asset)).exchangeRateStored()) / 1e18 (contracts/oracle/ib/CTokenOracle.sol#48)
        -_priceInUsd = (_underlyingPriceInUsd * _underlyingAmount) / 10 ** IERC20Metadata(_underlyingAddress).decimals() (contracts/oracle/ib/CTokenOracle.sol#49)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-multiply

CTokenOracle.getPriceInUsd(IERC20)._underlying (contracts/oracle/ib/CTokenOracle.sol#40) is a local variable never initialized
CTokenOracle.getPriceInUsd(IERC20)._underlyingAddress (contracts/oracle/ib/CTokenOracle.sol#38) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables

CTokenOracle.getPriceInUsd(IERC20) (contracts/oracle/ib/CTokenOracle.sol#37-50) ignores return value by ICToken(address(_asset)).underlying() (contracts/oracle/ib/CTokenOracle.sol#40-42)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

CTokenOracle.constructor(IOracle,address)._wethLike (contracts/oracle/ib/CTokenOracle.sol#27) lacks a zero-check on :
        - wethLike = _wethLike (contracts/oracle/ib/CTokenOracle.sol#29)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

Variable 'CTokenOracle.getPriceInUsd(IERC20)._underlying (contracts/oracle/ib/CTokenOracle.sol#40)' in CTokenOracle.getPriceInUsd(IERC20) (contracts/oracle/ib/CTokenOracle.sol#37-50) potentially used before declara
/ib/CTokenOracle.sol#41)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#pre-declaration-usage-of-local-variables

Different versions of Solidity is used:
        - Version used: ['0.8.9', '>=0.6.2', '^0.8.0']
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3)
        - 0.8.9 (contracts/interface/external/ICToken.sol#3)
        - 0.8.9 (contracts/interface/oracle/IOracle.sol#3)
        - >=0.6.2 (contracts/lib/OracleHelpers.sol#3)
        - 0.8.9 (contracts/lib/WadRayMath.sol#3)
        - 0.8.9 (contracts/oracle/ib/CTokenOracle.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

OracleHelpers.normalizeUsdInput(address,uint256) (contracts/lib/OracleHelpers.sol#15-18) is never used and should be removed
OracleHelpers.normalizeUsdInput(uint256,uint256) (contracts/lib/OracleHelpers.sol#28-34) is never used and should be removed
OracleHelpers.normalizeUsdOutput(address,uint256) (contracts/lib/OracleHelpers.sol#10-13) is never used and should be removed
OracleHelpers.normalizeUsdOutput(uint256,uint256) (contracts/lib/OracleHelpers.sol#20-26) is never used and should be removed
WadRayMath.rayDiv(uint256,uint256) (contracts/lib/WadRayMath.sol#63-65) is never used and should be removed
WadRayMath.rayMul(uint256,uint256) (contracts/lib/WadRayMath.sol#49-55) is never used and should be removed
WadRayMath.rayToWad(uint256) (contracts/lib/WadRayMath.sol#72-74) is never used and should be removed
WadRayMath.wadDiv(uint256,uint256) (contracts/lib/WadRayMath.sol#39-41) is never used and should be removed
WadRayMath.wadMul(uint256,uint256) (contracts/lib/WadRayMath.sol#25-31) is never used and should be removed
WadRayMath.wadToRay(uint256) (contracts/lib/WadRayMath.sol#81-83) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3) allows old versions
Pragma version0.8.9 (contracts/interface/external/ICToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/oracle/IOracle.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version>=0.6.2 (contracts/lib/OracleHelpers.sol#3) allows old versions
Pragma version0.8.9 (contracts/lib/WadRayMath.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/oracle/ib/CTokenOracle.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.9 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Parameter OracleHelpers.normalizeUsdOutput(address,uint256)._usdToken (contracts/lib/OracleHelpers.sol#10) is not in mixedCase
Parameter OracleHelpers.normalizeUsdOutput(address,uint256)._amountInUsd (contracts/lib/OracleHelpers.sol#10) is not in mixedCase
Parameter OracleHelpers.normalizeUsdInput(address,uint256)._usdToken (contracts/lib/OracleHelpers.sol#15) is not in mixedCase
Parameter OracleHelpers.normalizeUsdInput(address,uint256)._amountInUsd (contracts/lib/OracleHelpers.sol#15) is not in mixedCase
Parameter OracleHelpers.normalizeUsdOutput(uint256,uint256)._usdTokenDecimals (contracts/lib/OracleHelpers.sol#20) is not in mixedCase
Parameter OracleHelpers.normalizeUsdOutput(uint256,uint256)._amountInUsd (contracts/lib/OracleHelpers.sol#20) is not in mixedCase
Parameter OracleHelpers.normalizeUsdInput(uint256,uint256)._usdTokenDecimals (contracts/lib/OracleHelpers.sol#28) is not in mixedCase
Parameter OracleHelpers.normalizeUsdInput(uint256,uint256)._amountInUsd (contracts/lib/OracleHelpers.sol#28) is not in mixedCase
Parameter CTokenOracle.getPriceInUsd(IERC20)._asset (contracts/oracle/ib/CTokenOracle.sol#37) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

ChainlinkPriceProvider.sol

```
Different versions of Solidity is used:
        - Version used: ['0.8.9', '^0.8.0']
        - ^0.8.0 (contracts/dependencies/chainlink/interfaces/AggregatorV3Interface.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/math/Math.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#3)
        - 0.8.9 (contracts/interface/oracle/IPriceProvider.sol#3)
        - 0.8.9 (contracts/oracle/ChainlinkPriceProvider.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Math.average(uint256,uint256) (contracts/dependencies/openzeppelin/utils/math/Math.sol#27-30) is never used and should be removed
Math.ceilDiv(uint256,uint256) (contracts/dependencies/openzeppelin/utils/math/Math.sol#38-41) is never used and should be removed
Math.max(uint256,uint256) (contracts/dependencies/openzeppelin/utils/math/Math.sol#12-14) is never used and should be removed
Math.min(uint256,uint256) (contracts/dependencies/openzeppelin/utils/math/Math.sol#19-21) is never used and should be removed
SafeCast.toInt128(int256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#151-154) is never used and should be removed
SafeCast.toInt16(int256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#205-208) is never used and should be removed
SafeCast.toInt256(uint256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#235-239) is never used and should be removed
SafeCast.toInt32(int256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#187-190) is never used and should be removed
SafeCast.toInt64(int256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#169-172) is never used and should be removed
SafeCast.toInt8(int256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#223-226) is never used and should be removed
SafeCast.toUint128(uint256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#46-49) is never used and should be removed
SafeCast.toUint16(uint256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#106-109) is never used and should be removed
SafeCast.toUint224(uint256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#31-34) is never used and should be removed
SafeCast.toUint32(uint256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#91-94) is never used and should be removed
SafeCast.toUint64(uint256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#76-79) is never used and should be removed
SafeCast.toUint8(uint256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#121-124) is never used and should be removed
SafeCast.toUint96(uint256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#61-64) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.0 (contracts/dependencies/chainlink/interfaces/AggregatorV3Interface.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/math/Math.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#3) allows old versions
Pragma version0.8.9 (contracts/interface/oracle/IPriceProvider.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/oracle/ChainlinkPriceProvider.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.9 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Parameter ChainlinkPriceProvider.getPriceInUsd(address)._aggregator (contracts/oracle/ChainlinkPriceProvider.sol#36) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

DefaultOracle.sol

```
DefaultOracle.getPriceInUsd(IERC20) (contracts/oracle/DefaultOracle.sol#175-186) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(block.timestamp - _lastUpdatedAt <= assets[_asset].stalePeriod,price-is-stale) (contracts/oracle/DefaultOracle.sol#185)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp

Address.isContract(address) (contracts/dependencies/openzeppelin/utils/Address.sol#26-36) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#32-34)
Address.verifyCallResult(bool,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#195-215) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#207-210)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['0.8.9', '^0.8.0']
        - 0.8.9 (contracts/access/Governable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3)
        - 0.8.9 (contracts/interface/IGovernable.sol#3)
        - 0.8.9 (contracts/interface/oracle/IOracle.sol#3)
        - 0.8.9 (contracts/interface/oracle/IPriceProvider.sol#3)
        - 0.8.9 (contracts/oracle/DefaultOracle.sol#3)
        - 0.8.9 (contracts/utils/TokenHolder.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#108-114) is never used and should be removed
Address.functionDelegateCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#168-170) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187) is never used and should be removed
Address.functionStaticCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#141-143) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160) is never used and should be removed
Context._msgData() (contracts/dependencies/openzeppelin/utils/Context.sol#20-22) is never used and should be removed
Governable.__Governable_init() (contracts/access/Governable.sol#39-43) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#44-57) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#68-79) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#59-66) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#28-35) is never used and should be removed
TokenHolder._requireCanSweep() (contracts/utils/TokenHolder.sol#17) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version0.8.9 (contracts/access/Governable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3) allows old versions
Pragma version0.8.9 (contracts/interface/IGovernable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/oracle/IOracle.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/oracle/IPriceProvider.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/oracle/DefaultOracle.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/utils/TokenHolder.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.9 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#54-59):
        - (success) = recipient.call{value: amount}() (contracts/dependencies/openzeppelin/utils/Address.sol#57)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (contracts/dependencies/openzeppelin/utils/Address.sol#122-133):
        - (success,returndata) = target.call{value: value}(data) (contracts/dependencies/openzeppelin/utils/Address.sol#131)
Low level call in Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160):
        - (success,returndata) = target.staticcall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#158)
Low level call in Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187):
        - (success,returndata) = target.delegatecall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#185)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function Governable.__Governable_init() (contracts/access/Governable.sol#39-43) is not in mixedCase
Parameter Governable.transferGovernorship(address)._proposedGovernor (contracts/access/Governable.sol#59) is not in mixedCase
Parameter DefaultOracle.setPriceProvider(DefaultOracle.Protocol,IPriceProvider)._protocol (contracts/oracle/DefaultOracle.sol#81) is not in mixedCase
Parameter DefaultOracle.setPriceProvider(DefaultOracle.Protocol,IPriceProvider)._priceProvider (contracts/oracle/DefaultOracle.sol#81) is not in mixedCase
Parameter DefaultOracle.addOrUpdateUsdAsset(IERC20)._asset (contracts/oracle/DefaultOracle.sol#112) is not in mixedCase
Parameter DefaultOracle.addOrUpdateAssetThatUsesChainlink(IERC20Metadata,address,uint256)._asset (contracts/oracle/DefaultOracle.sol#123) is not in mixedCase
Parameter DefaultOracle.addOrUpdateAssetThatUsesChainlink(IERC20Metadata,address,uint256)._aggregator (contracts/oracle/DefaultOracle.sol#124) is not in mixedCase
Parameter DefaultOracle.addOrUpdateAssetThatUsesChainlink(IERC20Metadata,address,uint256)._stalePeriod (contracts/oracle/DefaultOracle.sol#125) is not in mixedCase
Parameter DefaultOracle.addOrUpdateAssetThatUsesUniswapV2(IERC20,address,uint256)._asset (contracts/oracle/DefaultOracle.sol#140) is not in mixedCase
Parameter DefaultOracle.addOrUpdateAssetThatUsesUniswapV2(IERC20,address,uint256)._underlying (contracts/oracle/DefaultOracle.sol#141) is not in mixedCase
Parameter DefaultOracle.addOrUpdateAssetThatUsesUniswapV2(IERC20,address,uint256)._stalePeriod (contracts/oracle/DefaultOracle.sol#142) is not in mixedCase
Parameter DefaultOracle.addOrUpdateAssetThatUsesUniswapV3(IERC20,address)._asset (contracts/oracle/DefaultOracle.sol#154) is not in mixedCase
Parameter DefaultOracle.addOrUpdateAssetThatUsesUniswapV3(IERC20,address)._underlying (contracts/oracle/DefaultOracle.sol#154) is not in mixedCase
Parameter DefaultOracle.update(IERC20)._asset (contracts/oracle/DefaultOracle.sol#164) is not in mixedCase
Parameter DefaultOracle.getPriceInUsd(IERC20)._asset (contracts/oracle/DefaultOracle.sol#175) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._token (contracts/utils/TokenHolder.sol#26) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._to (contracts/utils/TokenHolder.sol#27) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._amount (contracts/utils/TokenHolder.sol#28) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

MasterOracle.sol

```
MasterOracle._updateOracles(address[],IOracle[]).i (contracts/oracle/MasterOracle.sol#55) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables

Address.isContract(address) (contracts/dependencies/openzeppelin/utils/Address.sol#26-36) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#32-34)
Address.verifyCallResult(bool,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#195-215) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#207-210)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['0.8.9', '^0.8.0']
        - 0.8.9 (contracts/access/Governable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3)
        - 0.8.9 (contracts/interface/IGovernable.sol#3)
        - 0.8.9 (contracts/interface/oracle/IMasterOracle.sol#3)
        - 0.8.9 (contracts/interface/oracle/IOracle.sol#3)
        - 0.8.9 (contracts/oracle/MasterOracle.sol#3)
        - 0.8.9 (contracts/utils/TokenHolder.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#108-114) is never used and should be removed
Address.functionDelegateCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#168-170) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187) is never used and should be removed
Address.functionStaticCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#141-143) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160) is never used and should be removed
Context._msgData() (contracts/dependencies/openzeppelin/utils/Context.sol#20-22) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#44-57) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#68-79) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#59-66) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#28-35) is never used and should be removed
TokenHolder._requireCanSweep() (contracts/utils/TokenHolder.sol#17) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version0.8.9 (contracts/access/Governable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3) allows old versions
Pragma version0.8.9 (contracts/interface/IGovernable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/oracle/IMasterOracle.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/oracle/IOracle.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/oracle/MasterOracle.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/utils/TokenHolder.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.9 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#54-59):
        - (success) = recipient.call{value: amount}() (contracts/dependencies/openzeppelin/utils/Address.sol#57)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (contracts/dependencies/openzeppelin/utils/Address.sol#122-133):
        - (success,returndata) = target.call{value: value}(data) (contracts/dependencies/openzeppelin/utils/Address.sol#131)
Low level call in Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160):
        - (success,returndata) = target.staticcall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#158)
Low level call in Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187):
        - (success,returndata) = target.delegatecall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#185)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function Governable.__Governable_init() (contracts/access/Governable.sol#39-43) is not in mixedCase
Parameter Governable.transferGovernorship(address)._proposedGovernor (contracts/access/Governable.sol#59) is not in mixedCase
Parameter MasterOracle.initialize(address[],IOracle[],IOracle)._assets (contracts/oracle/MasterOracle.sol#35) is not in mixedCase
Parameter MasterOracle.initialize(address[],IOracle[],IOracle)._oracles (contracts/oracle/MasterOracle.sol#36) is not in mixedCase
Parameter MasterOracle.initialize(address[],IOracle[],IOracle)._defaultOracle (contracts/oracle/MasterOracle.sol#37) is not in mixedCase
Parameter MasterOracle.addOrUpdate(address[],IOracle[])._assets (contracts/oracle/MasterOracle.sol#71) is not in mixedCase
Parameter MasterOracle.addOrUpdate(address[],IOracle[])._oracles (contracts/oracle/MasterOracle.sol#71) is not in mixedCase
Parameter MasterOracle.setDefaultOracle(IOracle)._newDefaultOracle (contracts/oracle/MasterOracle.sol#81) is not in mixedCase
Parameter MasterOracle.quoteTokenToUsd(IERC20,uint256)._asset (contracts/oracle/MasterOracle.sol#113) is not in mixedCase
Parameter MasterOracle.quoteTokenToUsd(IERC20,uint256)._amount (contracts/oracle/MasterOracle.sol#113) is not in mixedCase
Parameter MasterOracle.quoteUsdToToken(IERC20,uint256)._asset (contracts/oracle/MasterOracle.sol#124) is not in mixedCase
Parameter MasterOracle.quoteUsdToToken(IERC20,uint256)._amountInUsd (contracts/oracle/MasterOracle.sol#124) is not in mixedCase
Parameter MasterOracle.quote(IERC20,IERC20,uint256)._assetIn (contracts/oracle/MasterOracle.sol#137) is not in mixedCase
Parameter MasterOracle.quote(IERC20,IERC20,uint256)._assetOut (contracts/oracle/MasterOracle.sol#138) is not in mixedCase
Parameter MasterOracle.quote(IERC20,IERC20,uint256)._amountIn (contracts/oracle/MasterOracle.sol#139) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._token (contracts/utils/TokenHolder.sol#26) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._to (contracts/utils/TokenHolder.sol#27) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._amount (contracts/utils/TokenHolder.sol#28) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

UniswapV3PriceProvider.sol

```
Address.isContract(address) (contracts/dependencies/openzeppelin/utils/Address.sol#26-36) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#32-34)
Address.verifyCallResult(bool,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#195-215) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#207-210)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['0.8.9', '>=0.6.2', '^0.8.0']
        - 0.8.9 (contracts/access/Governable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3)
        - 0.8.9 (contracts/interface/IGovernable.sol#3)
        - 0.8.9 (contracts/interface/oracle/IPriceProvider.sol#3)
        - 0.8.9 (contracts/interface/oracle/IUniswapV3CrossPoolOracle.sol#3)
        - >=0.6.2 (contracts/lib/OracleHelpers.sol#3)
        - 0.8.9 (contracts/oracle/UniswapV3PriceProvider.sol#3)
        - 0.8.9 (contracts/utils/TokenHolder.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#108-114) is never used and should be removed
Address.functionDelegateCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#168-170) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187) is never used and should be removed
Address.functionStaticCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#141-143) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160) is never used and should be removed
Context._msgData() (contracts/dependencies/openzeppelin/utils/Context.sol#20-22) is never used and should be removed
Governable.__Governable_init() (contracts/access/Governable.sol#39-43) is never used and should be removed
OracleHelpers.normalizeUsdInput(address,uint256) (contracts/lib/OracleHelpers.sol#15-18) is never used and should be removed
OracleHelpers.normalizeUsdInput(uint256,uint256) (contracts/lib/OracleHelpers.sol#28-34) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#44-57) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#68-79) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#59-66) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#28-35) is never used and should be removed
TokenHolder._requireCanSweep() (contracts/utils/TokenHolder.sol#17) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version0.8.9 (contracts/access/Governable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3) allows old versions
Pragma version0.8.9 (contracts/interface/IGovernable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/oracle/IPriceProvider.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/oracle/IUniswapV3CrossPoolOracle.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version>=0.6.2 (contracts/lib/OracleHelpers.sol#3) allows old versions
Pragma version0.8.9 (contracts/oracle/UniswapV3PriceProvider.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/utils/TokenHolder.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.9 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#54-59):
        - (success) = recipient.call{value: amount}() (contracts/dependencies/openzeppelin/utils/Address.sol#57)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (contracts/dependencies/openzeppelin/utils/Address.sol#122-133):
        - (success,returndata) = target.call{value: value}(data) (contracts/dependencies/openzeppelin/utils/Address.sol#131)
Low level call in Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160):
        - (success,returndata) = target.staticcall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#158)
Low level call in Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187):
        - (success,returndata) = target.delegatecall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#185)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function Governable.__Governable_init() (contracts/access/Governable.sol#39-43) is not in mixedCase
Parameter Governable.transferGovernorship(address)._proposedGovernor (contracts/access/Governable.sol#59) is not in mixedCase
Parameter OracleHelpers.normalizeUsdOutput(address,uint256)._usdToken (contracts/lib/OracleHelpers.sol#10) is not in mixedCase
Parameter OracleHelpers.normalizeUsdOutput(address,uint256)._amountInUsd (contracts/lib/OracleHelpers.sol#10) is not in mixedCase
Parameter OracleHelpers.normalizeUsdInput(address,uint256)._usdToken (contracts/lib/OracleHelpers.sol#15) is not in mixedCase
Parameter OracleHelpers.normalizeUsdInput(address,uint256)._amountInUsd (contracts/lib/OracleHelpers.sol#15) is not in mixedCase
```

```
Parameter OracleHelpers.normalizeUsdOutput(uint256,uint256)._usdTokenDecimals (contracts/lib/OracleHelpers.sol#20) is not in mixedCase
Parameter OracleHelpers.normalizeUsdOutput(uint256,uint256)._amountInUsd (contracts/lib/OracleHelpers.sol#20) is not in mixedCase
Parameter OracleHelpers.normalizeUsdInput(uint256,uint256)._usdTokenDecimals (contracts/lib/OracleHelpers.sol#28) is not in mixedCase
Parameter OracleHelpers.normalizeUsdInput(uint256,uint256)._amountInUsd (contracts/lib/OracleHelpers.sol#28) is not in mixedCase
Parameter UniswapV3PriceProvider.updateTwapPeriod(uint32)._newTwapPeriod (contracts/oracle/UniswapV3PriceProvider.sol#55) is not in mixedCase
Parameter UniswapV3PriceProvider.getPriceInUsd(address)._token (contracts/oracle/UniswapV3PriceProvider.sol#66) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._token (contracts/utils/TokenHolder.sol#26) is not in mixedCase
```

Governable.sol

```
Address.isContract(address) (contracts/dependencies/openzeppelin/utils/Address.sol#26-36) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#32-34)
Address.verifyCallResult(bool,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#195-215) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#207-210)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['0.8.9', '^0.8.0']
        - 0.8.9 (contracts/access/Governable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3)
        - 0.8.9 (contracts/interface/IGovernable.sol#3)
        - 0.8.9 (contracts/utils/TokenHolder.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#108-114) is never used and should be removed
Address.functionDelegateCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#168-170) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187) is never used and should be removed
Address.functionStaticCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#141-143) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160) is never used and should be removed
Context._msgData() (contracts/dependencies/openzeppelin/utils/Context.sol#20-22) is never used and should be removed
Governable.__Governable_init() (contracts/access/Governable.sol#39-43) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#44-57) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#68-79) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#59-66) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#28-35) is never used and should be removed
TokenHolder._requireCanSweep() (contracts/utils/TokenHolder.sol#17) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version0.8.9 (contracts/access/Governable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3) allows old versions
Pragma version0.8.9 (contracts/interface/IGovernable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/utils/TokenHolder.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.9 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#54-59):
        - (success) = recipient.call{value: amount}() (contracts/dependencies/openzeppelin/utils/Address.sol#57)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (contracts/dependencies/openzeppelin/utils/Address.sol#122-133):
        - (success,returndata) = target.call{value: value}(data) (contracts/dependencies/openzeppelin/utils/Address.sol#131)
Low level call in Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160):
        - (success,returndata) = target.staticcall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#158)
Low level call in Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187):
        - (success,returndata) = target.delegatecall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#185)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function Governable.__Governable_init() (contracts/access/Governable.sol#39-43) is not in mixedCase
Parameter Governable.transferGovernship(address)._proposedGovernor (contracts/access/Governable.sol#59) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._token (contracts/utils/TokenHolder.sol#26) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._to (contracts/utils/TokenHolder.sol#27) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._amount (contracts/utils/TokenHolder.sol#28) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

Manageable.sol

```
Address.isContract(address) (contracts/dependencies/openzeppelin/utils/Address.sol#26-36) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#32-34)
Address.verifyCallResult(bool,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#195-215) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#207-210)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['0.8.9', '^0.8.0']
        - 0.8.9 (contracts/access/Manageable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3)
        - 0.8.9 (contracts/interface/IController.sol#3)
        - 0.8.9 (contracts/interface/IDebtToken.sol#3)
        - 0.8.9 (contracts/interface/IDepositToken.sol#3)
        - 0.8.9 (contracts/interface/IGovernable.sol#3)
        - 0.8.9 (contracts/interface/IPausable.sol#3)
        - 0.8.9 (contracts/interface/IRewardsDistributor.sol#3)
        - 0.8.9 (contracts/interface/ISyntheticToken.sol#3)
        - 0.8.9 (contracts/interface/ITreasury.sol#3)
        - 0.8.9 (contracts/interface/oracle/IMasterOracle.sol#3)
        - 0.8.9 (contracts/utils/TokenHolder.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#108-114) is never used and should be removed
Address.functionDelegateCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#168-170) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187) is never used and should be removed
Address.functionStaticCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#141-143) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160) is never used and should be removed
Context._msgData() (contracts/dependencies/openzeppelin/utils/Context.sol#20-22) is never used and should be removed
Manageable.__Manageable_init() (contracts/access/Manageable.sol#21) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#44-57) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#68-79) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#59-66) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#28-35) is never used and should be removed
TokenHolder._requireCanSweep() (contracts/utils/TokenHolder.sol#17) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version0.8.9 (contracts/access/Manageable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3) allows old versions
Pragma version0.8.9 (contracts/interface/IController.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IDebtToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IDepositToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IGovernable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IPausable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IRewardsDistributor.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/ISyntheticToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/ITreasury.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/oracle/IMasterOracle.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/utils/TokenHolder.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.9 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#54-59):
        - (success) = recipient.call{value: amount}() (contracts/dependencies/openzeppelin/utils/Address.sol#57)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (contracts/dependencies/openzeppelin/utils/Address.sol#122-133):
        - (success,returndata) = target.call{value: value}(data) (contracts/dependencies/openzeppelin/utils/Address.sol#131)
Low level call in Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160):
        - (success,returndata) = target.staticcall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#158)
Low level call in Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187):
        - (success,returndata) = target.delegatecall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#185)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function Manageable.__Manageable_init() (contracts/access/Manageable.sol#21) is not in mixedCase
Parameter Manageable.setController(IController)._controller (contracts/access/Manageable.sol#59) is not in mixedCase
Variable Manageable.__gap (contracts/access/Manageable.sol#64) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._token (contracts/utils/TokenHolder.sol#26) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._to (contracts/utils/TokenHolder.sol#27) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._amount (contracts/utils/TokenHolder.sol#28) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Manageable.__gap (contracts/access/Manageable.sol#64) is never used in Manageable (contracts/access/Manageable.sol#14-65)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable
```

AUTOMATED TESTING

RewardsDistributor.sol

```
RewardsDistributor.claimRewards(address[],IERC20[]).i_scope_0 (contracts/RewardsDistributor.sol#198) is a local variable never initialized
RewardsDistributor.claimRewards(address[],IERC20[]).i (contracts/RewardsDistributor.sol#187) is a local variable never initialized
RewardsDistributor.updateTokenSpeeds(IERC20[],uint256[]).i (contracts/RewardsDistributor.sol#231) is a local variable never initialized
RewardsDistributor.claimRewards(address[],IERC20[]).j (contracts/RewardsDistributor.sol#192) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables

RewardsDistributor._updateTokenIndex(IERC20) (contracts/RewardsDistributor.sol#103-117) has external calls inside a loop: _totalSupply = _token.totalSupply() (contracts/RewardsDistributor.sol#109)
RewardsDistributor._updateTokensAccruedOf(IERC20,address) (contracts/RewardsDistributor.sol#122-136) has external calls inside a loop: _delta = _token.balanceOf(_account).wadMul(_deltaIndex) (contracts/RewardsDistr
RewardsDistributor._transferRewardIfEnoughTokens(address,uint256) (contracts/RewardsDistributor.sol#208-215) has external calls inside a loop: _balance = rewardToken.balanceOf(address(this)) (contracts/RewardsDistr
Address.functionCallWithValue(address,bytes,uint256,string) (contracts/dependencies/openzeppelin/utils/Address.sol#122-133) has external calls inside a loop: (success,returndata) = target.call{value: value}(data) (
1)
RewardsDistributor.onlyIfDistributorExists() (contracts/RewardsDistributor.sol#34-46) has external calls inside a loop: _rewardsDistributors = controller.getRewardsDistributors() (contracts/RewardsDistributor.sol#3
RewardsDistributor.onlyIfTokenExists(IERC20) (contracts/RewardsDistributor.sol#52-59) has external calls inside a loop: ! controller.isDepositTokenExists(IDepositToken(address(_token))) (contracts/RewardsDistributo
RewardsDistributor.onlyIfTokenExists(IERC20) (contracts/RewardsDistributor.sol#52-59) has external calls inside a loop: _syntheticToken = IDebtToken(address(_token)).syntheticToken() (contracts/RewardsDistributor.s
RewardsDistributor.onlyIfTokenExists(IERC20) (contracts/RewardsDistributor.sol#52-59) has external calls inside a loop: require(bool,string)(controller.isSyntheticTokenExists(_syntheticToken),invalid-token) (contra
RewardsDistributor.onlyIfTokenExists(IERC20) (contracts/RewardsDistributor.sol#52-59) has external calls inside a loop: require(bool,string)(address(_syntheticToken.debtToken()) == address(_token),invalid-token) (c
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#calls-inside-a-loop

Reentrancy in RewardsDistributor._transferRewardIfEnoughTokens(address,uint256) (contracts/RewardsDistributor.sol#208-215):
        External calls:
        - rewardToken.safeTransfer(_account,_amount) (contracts/RewardsDistributor.sol#212)
        Event emitted after the call(s):
        - RewardClaimed(_account,_amount) (contracts/RewardsDistributor.sol#213)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

RewardsDistributor._updateTokenIndex(IERC20) (contracts/RewardsDistributor.sol#103-117) uses timestamp for comparisons
        Dangerous comparisons:
        - _deltaTimestamps > 0 && _speed > 0 (contracts/RewardsDistributor.sol#108)
        - _deltaTimestamps > 0 && _supplyState.index > 0 (contracts/RewardsDistributor.sol#114)
RewardsDistributor._updateTokensAccruedOf(IERC20,address) (contracts/RewardsDistributor.sol#122-136) uses timestamp for comparisons
        Dangerous comparisons:
        - _accountIndex == 0 && _tokenIndex > 0 (contracts/RewardsDistributor.sol#127)
RewardsDistributor.updateBeforeMintOrBurn(IERC20,address) (contracts/RewardsDistributor.sol#142-147) uses timestamp for comparisons
        Dangerous comparisons:
        - tokenStates[_token].index > 0 (contracts/RewardsDistributor.sol#143)
RewardsDistributor.updateBeforeTransfer(IERC20,address,address) (contracts/RewardsDistributor.sol#153-163) uses timestamp for comparisons
        Dangerous comparisons:
        - tokenStates[_token].index > 0 (contracts/RewardsDistributor.sol#158)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp

Address.isContract(address) (contracts/dependencies/openzeppelin/utils/Address.sol#26-36) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#32-34)
Address.verifyCallResult(bool,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#195-215) uses assembly
        - INLINE ASM (contracts/dependencies/openzeppelin/utils/Address.sol#207-210)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['0.8.9', '^0.8.0']
        - 0.8.9 (contracts/RewardsDistributor.sol#3)
        - 0.8.9 (contracts/access/Manageable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3)
        - ^0.8.0 (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#3)
        - 0.8.9 (contracts/interface/IController.sol#3)
        - 0.8.9 (contracts/interface/IDebtToken.sol#3)
        - 0.8.9 (contracts/interface/IDepositToken.sol#3)
        - 0.8.9 (contracts/interface/IGovernable.sol#3)
        - 0.8.9 (contracts/interface/IPausable.sol#3)
        - 0.8.9 (contracts/interface/IRewardsDistributor.sol#3)
        - 0.8.9 (contracts/interface/ISyntheticToken.sol#3)
        - 0.8.9 (contracts/interface/ITreasury.sol#3)
        - 0.8.9 (contracts/interface/oracle/IMasterOracle.sol#3)
        - 0.8.9 (contracts/lib/WadRayMath.sol#3)
        - 0.8.9 (contracts/storage/RewardsDistributorStorage.sol#3)
        - 0.8.9 (contracts/utils/TokenHolder.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used
Address.functionCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#108-114) is never used and should be removed
Address.functionDelegateCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#168-170) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187) is never used and should be removed
Address.functionStaticCall(address,bytes) (contracts/dependencies/openzeppelin/utils/Address.sol#141-143) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160) is never used and should be removed
Context._msgData() (contracts/dependencies/openzeppelin/utils/Context.sol#20-22) is never used and should be removed
ReentrancyGuard.__ReentrancyGuard_init() (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#40-42) is never used and should be removed
SafeCast.toInt128(int256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#151-154) is never used and should be removed
SafeCast.toInt16(int256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#205-208) is never used and should be removed
SafeCast.toInt256(uint256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#235-239) is never used and should be removed
SafeCast.toInt32(int256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#187-190) is never used and should be removed
SafeCast.toInt64(int256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#169-172) is never used and should be removed
SafeCast.toInt8(int256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#223-226) is never used and should be removed
SafeCast.toUint128(uint256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#46-49) is never used and should be removed
SafeCast.toUint16(uint256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#106-109) is never used and should be removed
SafeCast.toUint256(int256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#133-136) is never used and should be removed
SafeCast.toUint64(uint256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#76-79) is never used and should be removed
SafeCast.toUint8(uint256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#121-124) is never used and should be removed
SafeCast.toUint96(uint256) (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#61-64) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#44-57) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#68-79) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#59-66) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#28-35) is never used and should be removed
TokenHolder._requireCanSweep() (contracts/utils/TokenHolder.sol#17) is never used and should be removed
WadRayMath.rayDiv(uint256,uint256) (contracts/lib/WadRayMath.sol#63-65) is never used and should be removed
WadRayMath.rayMul(uint256,uint256) (contracts/lib/WadRayMath.sol#40-55) is never used and should be removed
WadRayMath.rayToWad(uint256) (contracts/lib/WadRayMath.sol#72-74) is never used and should be removed
WadRayMath.wadToRay(uint256) (contracts/lib/WadRayMath.sol#81-83) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version0.8.9 (contracts/RewardsDistributor.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/access/Manageable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/proxy/utils/Initializable.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/IERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/extensions/IERC20Metadata.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/token/ERC20/utils/SafeERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Address.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/Context.sol#3) allows old versions
Pragma version^0.8.0 (contracts/dependencies/openzeppelin/utils/math/SafeCast.sol#3) allows old versions
Pragma version0.8.9 (contracts/interface/IController.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IDebtToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IDepositToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IGovernable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IPausable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/IRewardsDistributor.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/ISyntheticToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/ITreasury.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/interface/oracle/IMasterOracle.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/lib/WadRayMath.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/storage/RewardsDistributorStorage.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.9 (contracts/utils/TokenHolder.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.9 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/dependencies/openzeppelin/utils/Address.sol#54-59):
        - (success) = recipient.call{value: amount}() (contracts/dependencies/openzeppelin/utils/Address.sol#57)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (contracts/dependencies/openzeppelin/utils/Address.sol#122-133):
        - (success,returndata) = target.call{value: value}(data) (contracts/dependencies/openzeppelin/utils/Address.sol#131)
Low level call in Address.functionStaticCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#151-160):
        - (success,returndata) = target.staticcall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#158)
Low level call in Address.functionDelegateCall(address,bytes,string) (contracts/dependencies/openzeppelin/utils/Address.sol#178-187):
        - (success,returndata) = target.delegatecall(data) (contracts/dependencies/openzeppelin/utils/Address.sol#185)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter RewardsDistributor.initialize(IController,IERC20)._controller (contracts/RewardsDistributor.sol#61) is not in mixedCase
Parameter RewardsDistributor.initialize(IController,IERC20)._rewardToken (contracts/RewardsDistributor.sol#61) is not in mixedCase
Parameter RewardsDistributor.updateBeforeMintOrBurn(IERC20,address)._token (contracts/RewardsDistributor.sol#142) is not in mixedCase
Parameter RewardsDistributor.updateBeforeMintOrBurn(IERC20,address)._account (contracts/RewardsDistributor.sol#142) is not in mixedCase
Parameter RewardsDistributor.updateBeforeTransfer(IERC20,address,address)._token (contracts/RewardsDistributor.sol#154) is not in mixedCase
Parameter RewardsDistributor.updateBeforeTransfer(IERC20,address,address)._from (contracts/RewardsDistributor.sol#155) is not in mixedCase
```

```
Parameter RewardsDistributor.updateBeforeTransfer(IERC20,address,address)._to (contracts/RewardsDistributor.sol#156) is not in mixedCase
Parameter RewardsDistributor.claimRewards(address)._account (contracts/RewardsDistributor.sol#168) is not in mixedCase
Parameter RewardsDistributor.claimRewards(address,IERC20[])._account (contracts/RewardsDistributor.sol#175) is not in mixedCase
Parameter RewardsDistributor.claimRewards(address,IERC20[])._tokens (contracts/RewardsDistributor.sol#175) is not in mixedCase
Parameter RewardsDistributor.claimRewards(address[],IERC20[])._accounts (contracts/RewardsDistributor.sol#184) is not in mixedCase
Parameter RewardsDistributor.claimRewards(address[],IERC20[])._tokens (contracts/RewardsDistributor.sol#184) is not in mixedCase
Parameter RewardsDistributor.updateTokenSpeed(IERC20,uint256)._token (contracts/RewardsDistributor.sol#220) is not in mixedCase
Parameter RewardsDistributor.updateTokenSpeed(IERC20,uint256)._newSpeed (contracts/RewardsDistributor.sol#220) is not in mixedCase
Parameter RewardsDistributor.updateTokenSpeeds(IERC20[],uint256[])._tokens (contracts/RewardsDistributor.sol#227) is not in mixedCase
Parameter RewardsDistributor.updateTokenSpeeds(IERC20[],uint256[])._speeds (contracts/RewardsDistributor.sol#227) is not in mixedCase
Function Manageable.__Manageable_init() (contracts/access/Manageable.sol#21) is not in mixedCase
Parameter Manageable.setController(IController)._controller (contracts/access/Manageable.sol#59) is not in mixedCase
Variable Manageable.__gap (contracts/access/Manageable.sol#64) is not in mixedCase
Function ReentrancyGuard.__ReentrancyGuard_init() (contracts/dependencies/openzeppelin/security/ReentrancyGuard.sol#40-42) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._token (contracts/utils/TokenHolder.sol#26) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._to (contracts/utils/TokenHolder.sol#27) is not in mixedCase
Parameter TokenHolder.sweep(IERC20,address,uint256)._amount (contracts/utils/TokenHolder.sol#28) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Manageable.__gap (contracts/access/Manageable.sol#64) is never used in RewardsDistributor (contracts/RewardsDistributor.sol#14-235)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable
```

- As a result of the tests carried out with the Slither tool, some results were obtained and reviewed by Halborn. Based on the results reviewed, some vulnerabilities were determined to be false positives. The actual vulnerabilities found by Slither are already included in the report findings.

4.2 AUTOMATED SECURITY SCAN

Description:

Halborn used automated security scanners to assist with detection of well-known security issues, and to identify low-hanging fruits on the targets for this engagement. Among the tools used was MythX, a security analysis service for Ethereum smart contracts. MythX performed a scan on all the contracts and sent the compiled results to the analyzers to locate any vulnerabilities.

MythX results:

Controller.sol

Line	SWC Title	Severity	Short Description
401	(SWC-123) Requirement Violation	Low	Requirement violation.

DebtToken.sol

Line	SWC Title	Severity	Short Description
215	(SWC-116) Timestamp Dependence	Low	A control flow decision is made based on The block.timestamp environment variable.

NativeTokenGateway.sol

Line	SWC Title	Severity	Short Description
53	(SWC-123) Requirement Violation	Low	Requirement violation.

RewardsDistributor.sol

Line	SWC Title	Severity	Short Description
209	(SWC-123) Requirement Violation	Low	Requirement violation.

SyntheticToken.sol

Line	SWC Title	Severity	Short Description
319	(SWC-107) Reentrancy	Low	Read of persistent state following external call

TokenHolder.sol

Line	SWC Title	Severity	Short Description
43	(SWC-123) Requirement Violation	Low	Requirement violation.

- No major issues found by Mythx.

THANK YOU FOR CHOOSING

// HALBORN