



# Vesper Finance

Smart Contract Security Audit  
- OneOracle

Prepared by: Halborn

Date of Engagement: May 31st, 2022 - June 7th, 2022

Visit: [Halborn.com](https://Halborn.com)

DOCUMENT REVISION HISTORY	3
CONTACTS	3
1 EXECUTIVE OVERVIEW	4
1.1 INTRODUCTION	5
1.2 AUDIT SUMMARY	5
1.3 TEST APPROACH & METHODOLOGY	5
RISK METHODOLOGY	6
1.4 SCOPE	8
2 ASSESSMENT SUMMARY & FINDINGS OVERVIEW	9
3 FINDINGS & TECH DETAILS	10
3.1 (HAL-01) SAFEAPPROVE(0) IS NOT CALLED BEFORE SETTING A NEW APPROVAL - <span style="color: purple;">CRITICAL</span>	12
Description	12
Proof of Concept	14
Risk Level	14
Recommendation	14
3.2 (HAL-02) SWAPPER.GETBESTAMOUNT FUNCTIONS MAY RETURN INCORRECT MAX AND MIN AMOUNTS - <span style="color: red;">HIGH</span>	16
Description	16
Proof of Concept	19
Risk Level	19
Recommendation	19
3.3 (HAL-03) INCOMPATIBILITY WITH TRANSFER-ON-FEE OR DEFLATIONARY TOKENS - <span style="color: red;">HIGH</span>	21
Description	21

Proof of Concept	22
Risk Level	22
Recommendation	22
<b>3.4 (HAL-04) MAX DEVIATION TOO HIGH IN SWAPPERORACLE - MEDIUM</b>	<b>23</b>
Description	23
Risk Level	24
Recommendation	25
<b>3.5 (HAL-05) POSSIBLE MISUSE OF PUBLIC FUNCTIONS - INFORMATIONAL</b>	<b>26</b>
Description	26
Risk Level	26
Recommendation	26
<b>3.6 (HAL-06) STATE VARIABLES MISSING IMMUTABLE MODIFIER - INFORMATIONAL</b>	<b>27</b>
Description	27
Code Location	27
Risk Level	27
Recommendation	27
<b>4 AUTOMATED TESTING</b>	<b>28</b>
<b>4.1 STATIC ANALYSIS REPORT</b>	<b>29</b>
Description	29
Slither results	29
<b>4.2 AUTOMATED SECURITY SCAN</b>	<b>40</b>
Description	40
MythX results	40

## DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	05/31/2022	Roberto Reigada
0.2	Document Updates	06/07/2022	Roberto Reigada
0.3	Draft Review	06/07/2022	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
Roberto Reigada	Halborn	Roberto.Reigada@halborn.com

# EXECUTIVE OVERVIEW

DRAFT

## 1.1 INTRODUCTION

Vesper Finance engaged Halborn to conduct a security audit on their smart contracts beginning on May 31st, 2022 and ending on June 7th, 2022. The security assessment was scoped to the smart contracts provided in the GitHub repository [bloqpriv/one-oracle](#).

## 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 should be addressed by [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 code and can quickly identify items that do not follow the security best practices. The following phases and associated tools were used during 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 hot-spots 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:

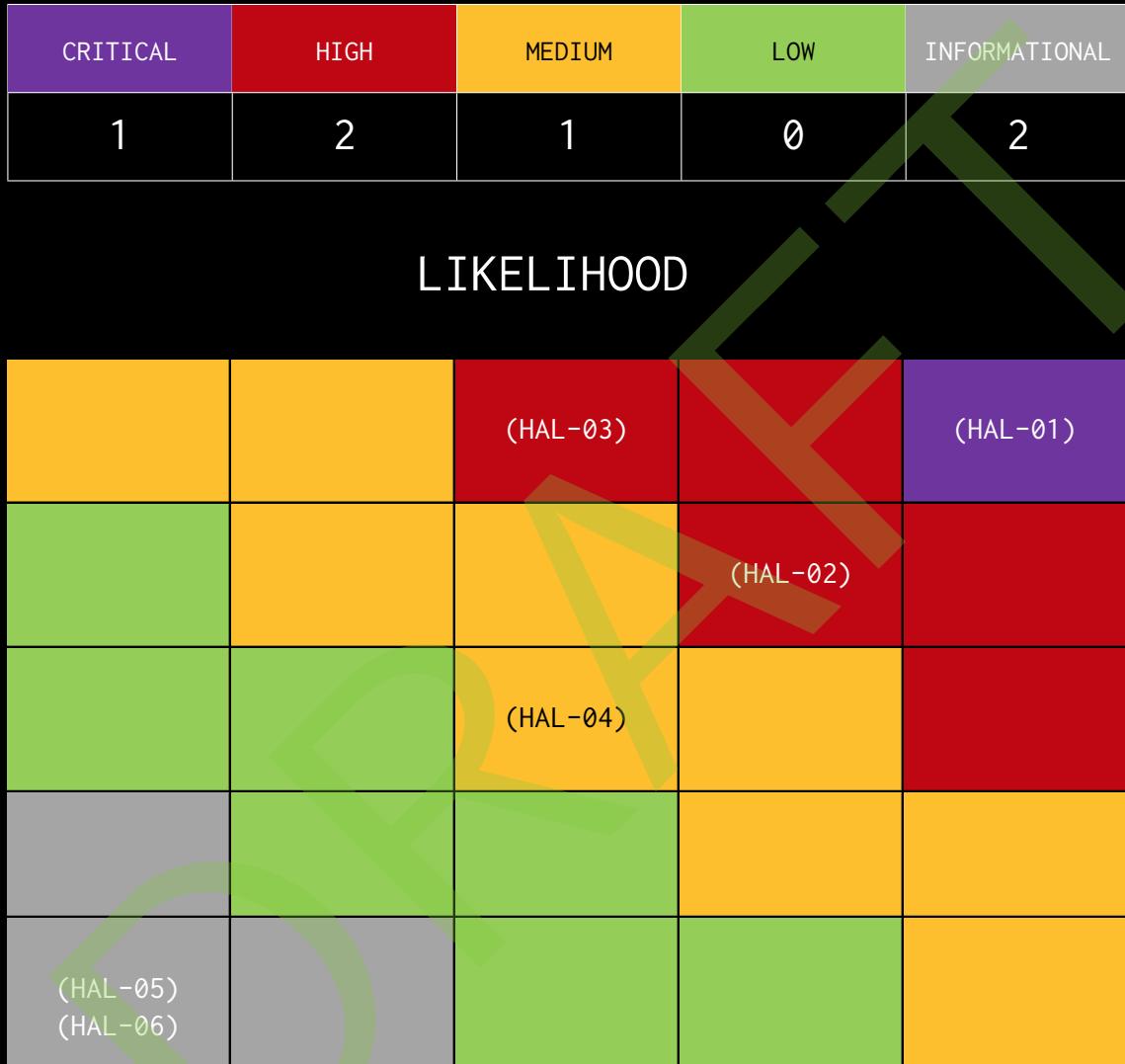
The security assessment was scoped to the following smart contracts:

- Governable.sol
- ChainlinkArbitrumPriceProvider.sol
- ChainlinkAvalanchePriceProvider.sol
- ChainlinkMainnetPriceProvider.sol
- ChainlinkPolygonPriceProvider.sol
- ChainlinkPriceProvider.sol
- PriceProvidersAggregator.sol
- UniswapV2LikePriceProvider.sol
- UniswapV3PriceProvider.sol
- Oracle.sol
- SwapperOracle.sol
- Swapper.sol
- UniswapV2LikeExchange.sol
- CTokenOracle.sol
- CurveLpTokenOracle.sol
- SynthDefaultOracle.sol
- UniswapV3CrossPoolOracle.sol

Initial Commit ID:

- 84c39f610c30283ea23358f27304ef7783921c44

## 2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW



## EXECUTIVE OVERVIEW

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
HAL01 - SAFEAPPROVE(0) IS NOT CALLED BEFORE SETTING A NEW APPROVAL	Critical	-
HAL02 - SWAPPER.GETBESTAMOUNT FUNCTIONS MAY RETURN INCORRECT MAX AND MIN AMOUNTS	High	-
HAL03 - INCOMPATIBILITY WITH TRANSFER-ON-FEE OR DEFLATIONARY TOKENS	High	-
HAL04 - MAX DEVIATION TOO HIGH IN SWAPPERORACLE	Medium	-
HAL05 - POSSIBLE MISUSE OF PUBLIC FUNCTIONS	Informational	-
HAL06 - STATE VARIABLES MISSING IMMUTABLE MODIFIER	Informational	-

# FINDINGS & TECH DETAILS

DRAFT

### 3.1 (HAL-01) SAFEAPPROVE(0) IS NOT CALLED BEFORE SETTING A NEW APPROVAL - CRITICAL

Description:

In the `UniswapV2LikeExchange` contract, the functions `swapExactInput()` and `swapExactOutput()` do a `SafeApprove` call before calling `swapExactTokensForTokens()` in the Router contract:

```
Listing 1: UniswapV2LikeExchange.sol (Lines 114,128)

108 function swapExactInput(
109     address[] calldata path_,
110     uint256 amountIn_,
111     uint256 amountOutMin_,
112     address outReceiver_
113 ) external returns (uint256 _amountOut) {
114     IERC20(path_[0]).safeApprove(address(router), amountIn_);
115     _amountOut = router.swapExactTokensForTokens(amountIn_,
116         amountOutMin_, path_, outReceiver_, block.timestamp)[
117         path_.length - 1
118     ];
119
120 /// @inheritdoc IExchange
121 function swapExactOutput(
122     address[] calldata path_,
123     uint256 amountOut_,
124     uint256 amountInMax_,
125     address inSender_,
126     address outRecipient_
127 ) external returns (uint256 _amountIn) {
128     IERC20(path_[0]).safeApprove(address(router), amountInMax_);
129     _amountIn = router.swapTokensForExactTokens(amountOut_,
130         amountInMax_, path_, outRecipient_, block.timestamp)[0];
131     // If swap end up costly less than _amountInMax then return
132     remaining
133     uint256 _remainingAmountIn = IERC20(path_[0]).balanceOf(
134         address(this));
135     if (_remainingAmountIn > 0) {
```

```

133         IERC20(path_[0]).safeTransfer(inSender_,
134             _remainingAmountIn);
135     }

```

`SafeERC20.safeApprove()` function is used:

**Listing 2: SafeERC20.safeApprove (Lines 53-56)**

```

45 function safeApprove(
46     IERC20 token,
47     address spender,
48     uint256 value
49 ) internal {
50     // safeApprove should only be called when setting an initial
51     // allowance, or when resetting it to zero. To increase and decrease it,
52     // use 'safeIncreaseAllowance' and 'safeDecreaseAllowance'
53     require(
54         (value == 0) || (token.allowance(address(this), spender)
55         == 0),
56         "SafeERC20: approve from non-zero to non-zero allowance"
57     );
58     _callOptionalReturn(token, abi.encodeWithSelector(token.
59         approve.selector, spender, value));
60 }

```

As per the `SafeERC20` contract's comment: `safeApprove` should only be called when setting an initial allowance, or when resetting it to zero. To increase and decrease it, use `safeIncreaseAllowance` and `safeDecreaseAllowance`.

In the line highlighted we can see that `SafeERC20.safeApprove()` checks that the allowance is zero before setting a new approval.

This means that every second call by a user to `Swapper.swapExactOutput()` with the same `tokenIn_` token will fail with the error: `SafeERC20: approve from non-zero to non-zero allowance` as the allowance after the first swap will be higher than zero.

## Proof of Concept:

In the image below, we can see how the second swap call for DAI as `tokenIn_` fails with the error: `SafeERC20: approve from non-zero to non-zero allowance`:

```

contract_DAI.balanceOf(user1) -> 40000000000000000000000000000000
contract_DAI.balanceOf(contract_Swapper) -> 0
contract_WETH.balanceOf(user1) -> 0
contract_WETH.balanceOf(contract_Swapper) -> 0
Calling -> contract_DAI.approve(contract_Swapper.address, 4000_0000000000000000000000000000, {'from': user1})
Transaction sent: 0xd614e5e3890f53fb61fe3c2ce69f32925180a258745db6e122cd42e7c56a9fc91
Gas price: 0.0 gwei Gas limit: 600000000 Nonce: 0
DAI.approve confirmed Block: 14898684 Gas used: 44050 (0.01%)

Calling -> contract_Swapper.swapExactOutput(contract_DAI.address, contract_WETH.address, 1_00000000000000000000000000000000, user1.address, {'from': user1})
Transaction sent: 0xcba91c9334f4fb0441f1dce5ebc1487a3857497d1df30dc8dd012e7d46ac600
Gas price: 0.0 gwei Gas limit: 600000000 Nonce: 1
Swapper.swapExactOutput confirmed Block: 14898685 Gas used: 278067 (0.05%)

contract_DAI.balanceOf(user1) -> 2248157218382782526549
contract_DAI.balanceOf(contract_Swapper) -> 0
contract_WETH.balanceOf(user1) -> 10000000000000000000000000000000
contract_WETH.balanceOf(contract_Swapper) -> 0

Calling -> contract_Swapper.swapExactOutput(contract_DAI.address, contract_WETH.address, 1_00000000000000000000000000000000, user1.address, {'from': user1})
Transaction sent: 0x4282ae95b68e6b49d49fa94d6340a123269915e0300771954f0e1773f5a4977b
Gas price: 0.0 gwei Gas limit: 600000000 Nonce: 2
Swapper.swapExactOutput confirmed (SafeERC20: approve from non-zero to non-zero allowance) Block: 14898686 Gas used: 160976 (0.03%)

contract_DAI.balanceOf(user1) -> 2248157218382782526549
contract_DAI.balanceOf(contract_Swapper) -> 0
contract_WETH.balanceOf(user1) -> 10000000000000000000000000000000
contract_WETH.balanceOf(contract_Swapper) -> 0

```

## Risk Level:

**Likelihood - 5**

**Impact - 5**

## Recommendation:

It is recommended to perform 2 different `SafeApprove()` calls in the swap functions. One resetting the allowance to 0 by calling `SafeApprove(0)` and another one with the amount needed to be approved:

**Listing 3: UniswapV2LikeExchange.sol (Lines 114,115,129,130)**

```

108 function swapExactInput(
109     address[] calldata path_,
110     uint256 amountIn_,
111     uint256 amountOutMin_,
112     address outReceiver_
113 ) external returns (uint256 _amountOut) {
114     IERC20(path_[0]).safeApprove(address(router), 0);
115     IERC20(path_[0]).safeApprove(address(router), amountIn_);

```

## FINDINGS & TECH DETAILS

```
116     _amountOut = router.swapExactTokensForTokens(amountIn_,
117         ↳ amountOutMin_, path_, outReceiver_, block.timestamp)[
118             path_.length - 1
119         ];
120     ];
121 }
122 /// @inheritdoc IExchange
123 function swapExactOutput(
124     address[] calldata path_,
125     uint256 amountOut_,
126     uint256 amountInMax_,
127     address inSender_,
128     address outRecipient_
129 ) external returns (uint256 _amountIn) {
130     IERC20(path_[0]).safeApprove(address(router), 0);
131     IERC20(path_[0]).safeApprove(address(router), amountInMax_);
132     _amountIn = router.swapTokensForExactTokens(amountOut_,
133         ↳ amountInMax_, path_, outRecipient_, block.timestamp)[0];
134     // If swap end up costly less than _amountInMax then return
135     ↳ remaining
136     uint256 _remainingAmountIn = IERC20(path_[0]).balanceOf(
137         ↳ address(this));
138     if (_remainingAmountIn > 0) {
139         IERC20(path_[0]).safeTransfer(inSender_,
140             ↳ _remainingAmountIn);
141     }
142 }
```

## 3.2 (HAL-02) SWAPPER.GETBESTAMOUNT FUNCTIONS MAY RETURN INCORRECT MAX AND MIN AMOUNTS - HIGH

Description:

In the `Swapper` contract, the function `getBestAmountIn()` given an input token, an output token and an output amount, calculates:

- The minimum amount of tokens that should be input by the user to get the amount of output tokens set => `_amountIn`
- The maximum amount of tokens that should be input by the user to get the amount of output tokens set => `_amountInMax`
- The exchange address from where the `_amountIn` data was pulled => `_exchange`
- The swap path that must be used in the `_exchange` to get that `_amountIn` => `_path`

Listing 4: `Swapper.sol` (Lines 71, 75, 89, 90)

```
56 function getBestAmountIn(
57     address tokenIn_,
58     address tokenOut_,
59     uint256 amountOut_
60 )
61 public
62 view
63 returns (
64     uint256 _amountIn,
65     uint256 _amountInMax ,
66     IExchange _exchange ,
67     address[] memory _path
68 )
69 {
70     _exchange = IExchange(exchanges.at(0));
71     (_amountIn, _path) = _getBestAmountIn(_exchange, tokenIn_,
72     tokenOut_, amountOut_);
73     uint256 _len = exchanges.length();
74     for (uint256 i = 1; i < _len; ++i) {
```

```

74         IExchange _iExchange = IExchange(exchanges.at(i));
75         (uint256 _iAmountIn, address[] memory _iPath) =
L_     _getBestAmountIn(
76             _iExchange,
77             tokenIn_,
78             tokenOut_,
79             amountOut_
80         );
81         if (_iAmountIn > 0 && _iAmountIn < _amountIn) {
82             _amountIn = _iAmountIn;
83             _exchange = _iExchange;
84             _path = _iPath;
85         }
86     }
87     require(_path.length > 0, "no-path-found");
88
89     uint256 _amountInFromOracle = oracle.quote(tokenOut_, tokenIn_
L_     , amountOut_);
90     _amountInMax = (_amountInFromOracle * (1e18 + maxSlippage)) /
L_     1e18;
91 }
```

In order to retrieve all that data, this function calls the `getBestAmountIn()` function in the Exchange Router Contract:

**Listing 5: UniswapV2LikeExchange.sol (Lines 51,64)**

```

42 function getBestAmountIn(
43     address tokenIn_,
44     address tokenOut_,
45     uint256 amountOut_
46 ) external view returns (uint256 _amountIn, address[] memory _path
L_ ) {
47     // 1. Check IN-OUT pair
48     address[] memory _pathA = new address[](2);
49     _pathA[0] = tokenIn_;
50     _pathA[1] = tokenOut_;
51     uint256 _amountInA = _getAmountsIn(amountOut_, _pathA);
52
53     if (tokenIn_ == wethLike || tokenOut_ == wethLike) {
54         // Returns if one of the token is WETH-Like
55         require(_amountInA > 0, "invalid-swap");
56         return (_amountInA, _pathA);
```

```

57     }
58
59     // 2. Check IN-WETH-OUT path
60     address[] memory _pathB = new address[](3);
61     _pathB[0] = tokenIn_;
62     _pathB[1] = wethLike;
63     _pathB[2] = tokenOut_;
64     uint256 _amountInB = _getAmountsIn(amountOut_, _pathB); XXXXXXXXXX
65
66     // 3. Get best route between paths A and B
67     require(_amountInA > 0 || _amountInB > 0, "invalid-swap");
68
69     // Returns A if it's valid and better than B or if B isn't
70     if ((_amountInA > 0 && _amountInA < _amountInB) || _amountInB
71     == 0) {
72         return (_amountInA, _pathA);
73     }
74     return (_amountInB, _pathB);

```

Although occasionally the direct path (`tokenIn_ => tokenOut_`) will not be available, as it occurs here when trying to swap WBTC into ApeCoins:

```

>>> contract_UNISWAPV2_ROUTER.getAmountsIn(5000_0000000000000000, [contract_WBTC, contract_APECOIN])
File "<console>", line 1, in <module>
File "/usr/local/lib/python3.8/dist-packages/brownie/network/contract.py", line 1665, in __call__
    return self._call(*args, block_identifier=block_identifier)
File "/usr/local/lib/python3.8/dist-packages/brownie/network/contract.py", line 1461, in call
    raise VirtualMachineError(e) from None
VirtualMachineError: revert
>>> contract_SUSHISWAP_ROUTER.getAmountsIn(5000_0000000000000000, [contract_WBTC, contract_APECOIN])
File "<console>", line 1, in <module>
File "/usr/local/lib/python3.8/dist-packages/brownie/network/contract.py", line 1665, in __call__
    return self._call(*args, block_identifier=block_identifier)
File "/usr/local/lib/python3.8/dist-packages/brownie/network/contract.py", line 1461, in call
    raise VirtualMachineError(e) from None
VirtualMachineError: revert

```

In this case, the returned values will be taken from the path (`tokenIn_ => WETH => tokenOut_`):

```

>>> contract_UNISWAPV2_ROUTER.getAmountsIn(5000_0000000000000000, [contract_WBTC, contract_WETH, contract_APECOIN])
(129019022, 21202769103392587143, 50000000000000000000)
>>> contract_SUSHISWAP_ROUTER.getAmountsIn(5000_0000000000000000, [contract_WBTC, contract_WETH, contract_APECOIN])
(10936212, 50951847326946404459, 500000000000000000)

```

The `_amountIn` retrieved when this last path is used is way higher, as there is an extra jump in the swap path. On the other hand, `SwapperOracle` is used to retrieve the `_amountInMax`. Chainlink price feeds are used here as primary oracle. This results in an `_amountInMax` way lower than the `_amountIn` value returned.

The same issue is also present in the `Swapper.getBestAmountOut()` function. But in this function, the `_amountOutMin` is higher than the `_amountOut` returned.

#### Proof of Concept:

1. `getBestAmountIn()`: USDC => WETH, 1\_00000000000000000000000000000000
2. `getBestAmountIn()`: USDC => DAI, 1\_00000000000000000000000000000000
3. `getBestAmountIn()`: WBTC => APECOIN, 5000\_000000000000000000000000000000: Issue occurs here.

```
contract Swapper.getBestAmountIn(contract USDC.address, contract WETH.address, 1_00000000000000000000000000000000) ->
(101506945, 105214765, "0x1597109021769040d0e1012204d910c6d93e2c24", ("0xa0b66991cd21b34c1d19f4a2e9200c33606eB48", "0xC02aa39b223FE6D0Ae5C4F27eAD9083C756Cc2"))

contract Swapper.getBestAmountIn(contract USDC.address, contract DAI.address, 1_00000000000000000000000000000000) ->
(1010107, 1009587, "0x0cd2De9A792d91CB1081e29340867663DD755AC", ("0xb0b6991c621b36c1d19f4a2e9200c33606eB48", "0x6B175474E89094c44d96b954EdeCA495271d0F"))

contract Swapper.getBestAmountIn(contract WETH.address, contract APECOIN.address, 5000_000000000000000000000000000000) ->
(129019022, 104629050, "0x774b4F4565ebcB2Fd033641C75D7F9c90159e02", ("0xC02aa39b223FE8D0Ae5C4F27eAD9083C756Cc2", "0x4d22452801AcEd8B2F0aebE155379bb5D594381"))
```

We can see how in the last call `_amountInMax` is lower than `_amountIn`:

- `_amountIn` => 1\_29019022
- `_amountInMax` => 1\_04629050

And also below, we can appreciate the same issue in the `getBestAmountOut()` function:

```
contract Swapper.getBestAmountOut(contract APECOIN.address, contract WBTC.address, 5000_000000000000000000000000000000) ->
(10550625, 105597187, "0x774b4F4565ebcB2Fd033641C75D7F9c90159e02", ("0x4d22452801AcEd8B2F0aebE155379bb5D594381", "0xC02aa39b223FE8D0Ae5C4F27eAD9083C756Cc2", "0x2260FAC5E5542a773a4ffBCfeD7C193bc2C599"))
```

In this case, we are getting an `_amountOutMin` higher than `_amountOut`.

#### Risk Level:

**Likelihood - 4**

**Impact - 4**

#### Recommendation:

An `_amountIn` higher than `_amountInMax` in the `getBestAmountIn()` return values will cause that the `swapExactOutput()` call fails, as the user would not have approved enough tokens. Something similar occurs with `getBestAmountOut()` and `swapExactInput()` with an `_amountOutMin` higher than `_amountOut`. Users in this case may receive an amount of tokens way lower

than what they expected from the swap.

It is recommended to revisit the logic of these functions, so these cases cannot happen.

DRAFT

### 3.3 (HAL-03) INCOMPATIBILITY WITH TRANSFER-ON-FEE OR DEFLATIONARY TOKENS - HIGH

Description:

In the Swapper contract, the functions `swapExactInput()` and `swapExactOutput()` are used to swap tokens in an exchange:

Listing 6: Swapper.sol (Lines 148,149,165,166)

```
137 function swapExactInput(
138     address tokenIn_,
139     address tokenOut_,
140     uint256 amountIn_,
141     address receiver_
142 ) external returns (uint256 _amountOut) {
143     (, uint256 _amountOutMin, IExchange _exchange, address[]
144      memory _path) = getBestAmountOut(
145         tokenIn_,
146         tokenOut_,
147         amountIn_
148     );
149     IERC20(tokenIn_).safeTransferFrom(_msgSender(), address(
150      _exchange), amountIn_);
151     return _exchange.swapExactInput(_path, amountIn_,
152      _amountOutMin, receiver_);
153 }
154 /// @inheritdoc ISwapper
155 function swapExactOutput(
156     address tokenIn_,
157     address tokenOut_,
158     uint256 amountOut_,
159     address receiver_
160 ) external returns (uint256 _amountIn) {
161     (, uint256 _amountInMax, IExchange _exchange, address[] memory
162      _path) = getBestAmountIn(
163         tokenIn_,
164         tokenOut_,
165         amountOut_
166 }
```

```

163      );
164      address _caller = _msgSender();
165      IERC20(tokenIn_).safeTransferFrom(_caller, address(_exchange),
166      ↳ _amountInMax);
166      return _exchange.swapExactOutput(_path, amountOut_,
167      ↳ _amountInMax, _caller, receiver_);
167 }

```

The functions assume that the `IERC20(tokenIn_).safeTransferFrom()` call will transfer the full `amountIn_` and `_amountInMax` to the Swapper contract.

However, this may not be true if the `tokenIn_` is a transfer-on-fee token or a deflationary/rebasing token, causing the received amount to be less than the accounted amount.

If this occurs the swap call will revert as we can see in the proof of concept below.

#### Proof of Concept:

```

Calling -> contract_USDT.setParams(10, 20, {'from': contract_USDT.owner()}) SETTING A 1% FEE
Transaction sent: 0x5d2935a26f699f1796cf9d785ef1bd486fb50a3f0e8f7ebf7604367ab7ff71
Gas price: 0.0 gwei Gas limit: 600000000 Nonce: 1
USDT.setParams confirmed Block: 14891371 Gas used: 66957 (0.01%)

contract_USDT.balanceOf(user1) -> 50000000000
Calling -> contract_USDT.approve(contract_Swapper.address, 5000_000000, {'from': user1})
Transaction sent: 0x5f902b03d2e1bd4d9f9b5999be2aceec1298bca22a4e5e92bf973a3bece8177d3
Gas price: 0.0 gwei Gas limit: 600000000 Nonce: 2
USDT.approve confirmed Block: 14891372 Gas used: 45961 (0.01%)

Calling -> contract_Swapper.swapExactInput(contract_USDT.address, contract_WETH.address, 5000_000000, user1.address, {'from': user1})
Transaction sent: 0x39b3d83670e11de125ab434b06a0cd9af8b67aca745b1fe51c6b40969cd83ed
Gas price: 0.0 gwei Gas limit: 600000000 Nonce: 3
Swapper.swapExactInput confirmed (TransferHelper: TRANSFER_FAILED) Block: 14891373 Gas used: 572322191 (95.39%)

```

#### Risk Level:

**Likelihood - 3**

**Impact - 5**

#### Recommendation:

It is recommended to get the exact received amount of `tokenIn_` by calculating the difference of `tokenIn_` balance before and after the transfer and use it in the `swapExactInput()` and `swapExactOutput()` calls.

## 3.4 (HAL-04) MAX DEVIATION TOO HIGH IN SWAPPERORACLE - MEDIUM

Description:

In the contract `SwapperOracle`, the constant `MAX_DEVIATION` sets the max deviation accepted between 2 different price providers:

**Listing 7: SwapperOracle.sol**

```
16 uint256 internal constant MAX_DEVIATION = 0.1e18;
```

**Listing 8: SwapperOracle.sol (Line 102)**

```
62 function quote(
63     address tokenIn_,
64     address tokenOut_,
65     uint256 amountIn_
66 ) external view returns (uint256 _amountOut) {
67     // 1. Get price from chainlink
68     uint256 _lastUpdatedAt;
69     (_amountOut, _lastUpdatedAt) = _quote(DataTypes.Provider.
↳ CHAINLINK, tokenIn_, tokenOut_, amountIn_);
70
71     // 2. If price from chainlink is OK return it
72     if (_amountOut > 0 && !_priceIsStale(_lastUpdatedAt)) {
73         return _amountOut;
74     }
75
76     // 3. Get price from fallback A
77     (uint256 _amountOutA, uint256 _lastUpdatedAtA) = _quote(
↳ fallbackProviderA, tokenIn_, tokenOut_, amountIn_);
78
79     // 4. If price from fallback A is OK and there isn't a
↳ fallback B, return price from fallback A
80     bool _aPriceOK = _amountOutA > 0 && !_priceIsStale(
↳ _lastUpdatedAtA);
81     if (fallbackProviderB == DataTypes.Provider.NONE) {
82         require(_aPriceOK, "fallback-a-failed");
83         return _amountOutA;
84     }
```

```

85
86     // 5. Get price from fallback B
87     (uint256 _amountOutB, uint256 _lastUpdatedAtB) = _quote(
88         ↳ fallbackProviderB, tokenIn_, tokenOut_, amountIn_);
89
90     // 6. If only one price from fallbacks is valid, return it
91     bool _bPriceOK = _amountOutB > 0 && !_priceIsStale(
92         ↳ _lastUpdatedAtB);
93     if (!_bPriceOK && _aPriceOK) {
94         return _amountOutA;
95     } else if (_bPriceOK && !_aPriceOK) {
96         return _amountOutB;
97     }
98
99     // 7. Check fallback prices deviation
100    require(_aPriceOK && _bPriceOK, "fallbacks-failed");
101    uint256 _deviation = _amountOutA > _amountOutB
102        ? ((_amountOutA - _amountOutB) * 1e18) / _amountOutA
103        : ((_amountOutB - _amountOutA) * 1e18) / _amountOutB;
104    require(_deviation <= MAX_DEVIATION, "prices-deviation-too-
105    ↳ high");
106
107    // 8. If deviation is OK, return price from fallback A
108    return _amountOutA;
109 }
```

If the initial price provider does not return a correct price and there are 2 different fallback price providers set and both return correct prices, the price returned by these two will be compared with each other and if the difference is less than the 10% the price will be considered valid. Then the price of the fallback price provider A will be returned.

A 10% price deviation is way too high and may mean that the prices received are not correct.

**Risk Level:**

**Likelihood - 3**

**Impact - 3**

Recommendation:

It is recommended to lower the MAX\_DEVIATION constant to a max. of 1-2%, instead of 10:

**Listing 9: SwapperOracle.sol**

```
16 uint256 internal constant MAX_DEVIATION = 0.02e18;
```

### 3.5 (HAL-05) POSSIBLE MISUSE OF PUBLIC FUNCTIONS - INFORMATIONAL

#### Description:

In the contracts below, there are some functions marked as `public` that are never called directly within the contract itself or in any of their descendants:

#### `ChainlinkPriceProvider.sol`

- `getPriceInUsd()` (`ChainlinkPriceProvider.sol#31-33`)

#### `UniswapV2LikePriceProvider.sol`

- `updateDefaultTwapPeriod()` (`UniswapV2LikePriceProvider.sol#218-221`)

#### `SwapperOracle.sol`

- `updateFallbackProviders()` (`SwapperOracle.sol#131-139`)
- `updateProvidersAggregator()` (`SwapperOracle.sol#144-148`)
- `updateStalePeriod()` (`SwapperOracle.sol#153-156`)

#### `UniswapV3CrossPoolOracle.sol`

- `assetToAssetThruRoute()` (`UniswapV3CrossPoolOracle.sol#57-83`)

#### Risk Level:

**Likelihood** - 1

**Impact** - 1

#### Recommendation:

If the functions are not intended to be called internally or by their descendants, it is better to mark them as `external` to reduce gas costs.

## 3.6 (HAL-06) STATE VARIABLES MISSING IMMUTABLE MODIFIER - INFORMATIONAL

### Description:

Some contracts have state variables that can be declared as `immutable` to save some gas.

The `immutable` keyword was added to Solidity in 0.6.5. State variables can be marked `immutable` which causes them to be read-only, but only assignable in the constructor.

### Code Location:

`CTokenOracle.sol`

- Line 18: `ISynthOracle public underlyingOracle;`
- Line 23: `address public wethLike;`

`CurveLpTokenOracle.sol`

- Line 13: `ICurveAddressProvider public addressProvider;`

`SynthDefaultOracle.sol`

- Line 31: `IChainlinkPriceProvider public chainlinkProvider;`

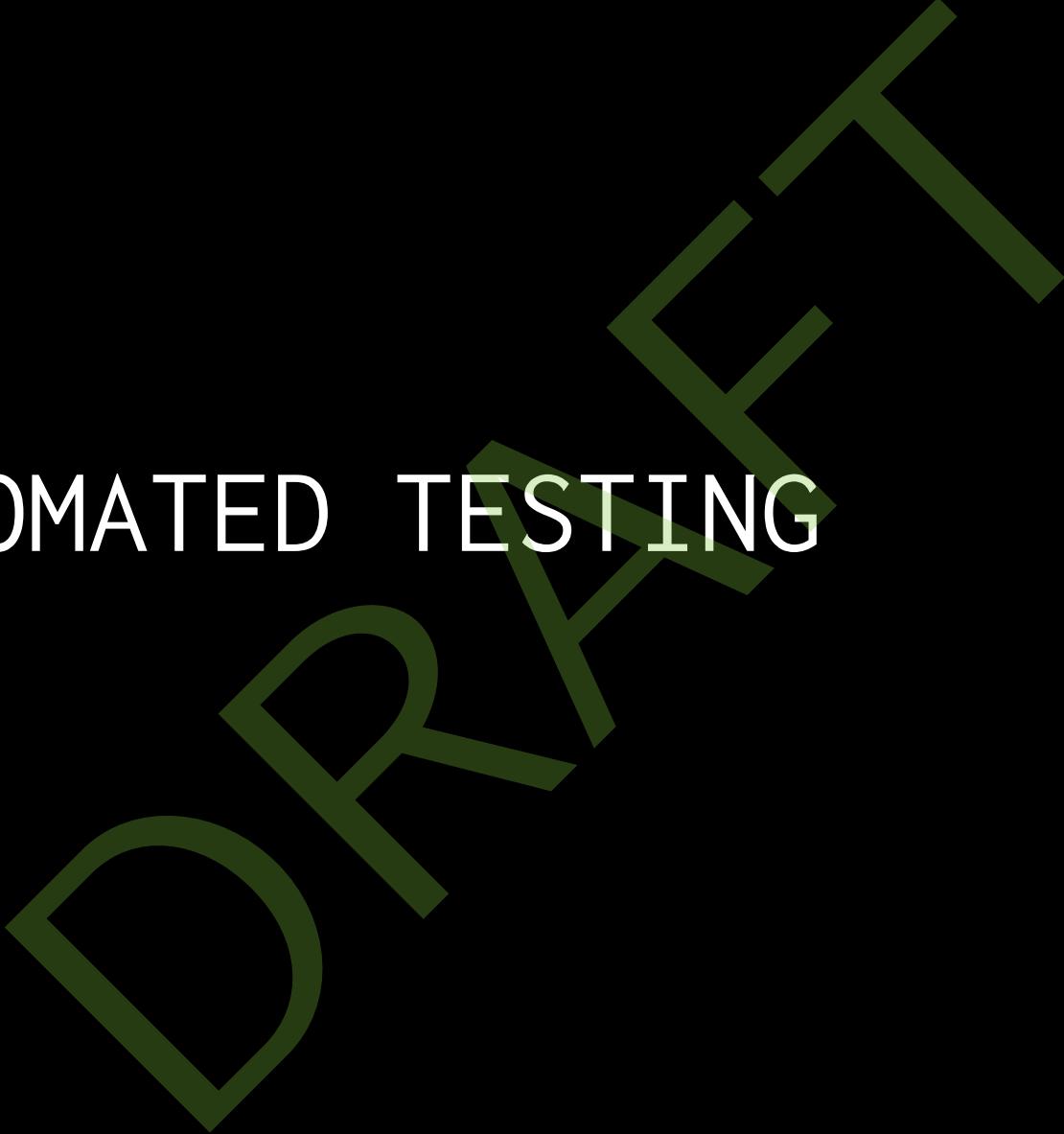
### Risk Level:

`Likelihood - 1`

`Impact - 1`

### Recommendation:

It is recommended to add the `immutable` modifier to the state variables mentioned.



# AUTOMATED TESTING

## 4.1 STATIC ANALYSIS REPORT

### Description:

Halborn used automated testing techniques to enhance the coverage of certain areas of the smart contracts in scope. Among the tools used was Slither, a Solidity static analysis framework. After Halborn verified the smart contracts in the repository and was able to compile them correctly into their ABIS and binary format, Slither was run against the 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:

#### Governable.sol

```
Address.verifyAllResult(bool,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#201-221) uses assembly
  Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-useage

Different versions of Solidity is used:
  - Version used: ['0.6.9', '^0.6.0', '^0.8.1', '^0.8.2']
  - 0.8.1 (node_modules/@openzeppelin/contracts/proxy/utils/Initializable.sol#4)
  - 0.8.0 (node_modules/@openzeppelin/contracts/utils/Address.sol#4)
  - 0.8.9 (contracts/access/Governable.sol#3)

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#87) is never used and should be removed
Address.functionCall(address,bytes,Uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#11-101) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#114-120) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#128-139) is never used and should be removed
Address.functionDelegateCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#14-16) is never used and should be removed
Address.functionDelegateCall(address,bytes,Uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#14-16) is never used and should be removed
Address.functionStaticCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#18-19) is never used and should be removed
Address.functionStaticCall(address,bytes,Uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#18-19) is never used and should be removed
Address.isContract(address) (node_modules/@openzeppelin/contracts/contracts/Address.sol#4-6) is never used and should be removed
Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts/contracts/Address.sol#40-45) is never used and should be removed
Address.sendValue(address,uint256,bytes) (node_modules/@openzeppelin/contracts/contracts/Address.sol#40-45) is never used and should be removed
Address.sendValue(address,uint256,bytes,Uint256) (node_modules/@openzeppelin/contracts/contracts/Address.sol#40-45) is never used and should be removed
Governable._Governable_init() (contracts/access/Governable.sol#3-4) is never used and should be removed
Instantiable._instantiable_init() (contracts/introspection/Instantiable.sol#13-15) is never used and should be removed
Initializable._initializable_init() (node_modules/@openzeppelin/contracts/proxy/utils/Initializable.sol#133-140) is never used and should be removed
Initializable._initializable_init() (node_modules/@openzeppelin/contracts/proxy/utils/Initializable.sol#133-140) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Pragma version="0.8.2" (node_modules/@openzeppelin/contracts/proxy/utils/Initializable.sol#8) allows old versions
Pragma version="0.8.1" (node_modules/@openzeppelin/contracts/proxy/utils/Initializable.sol#8) allows old versions
Pragma version="0.8.0" (node_modules/@openzeppelin/contracts/proxy/utils/Initializable.sol#8) allows old versions
Pragma version="0.8.9" (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/interfaces/IGovernable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Solidity 0.8.9 is not recommended
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.appendValue(address,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#65):
  - (success) = recipient.call{value: amount!} (node_modules/@openzeppelin/contracts/utils/Address.sol#42)
Low level call in Address.appendValue(address,bytes,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#128-139):
  - (success,returnData) = target.delegatecall(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#137)
Low level call in Address.appendValue(address,bytes,uint256,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#157-166):
  - (success,returnData) = target.staticcall(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#164)
Low level call in Address.functionDelegateCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#154-159):
  - (success,returnData) = target.delegatecall(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#159)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function Governor._Governable_init() (contracts/access/Governable.sol#3-4) is not in mixedCase
Parameter Governor.transferOwnership(address)_proposeGovernor (contracts/access/Governable.sol#6) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

#### ChainlinkArbitrumPriceProvider.sol

```
Address.verifyAllResult(bool,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#201-221) uses assembly
  Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-useage

Different versions of Solidity is used:
  - Version used: ['0.6.9', '^0.6.0', '^0.8.1', '^0.8.2']
  - 0.8.0 (node_modules/@chainlink/contracts/src/v0.8/interface/AggregatorV3Interface.sol#2)
  - 0.8.0 (node_modules/@openzeppelin/contracts/proxy/utils/Initializable.sol#4)
  - 0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#4)
  - 0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol#4)
  - 0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#4)
  - 0.8.0 (node_modules/@openzeppelin/contracts/math/Math.sol#4)
  - 0.8.0 (node_modules/@openzeppelin/contracts/math/SafeCast.sol#4)
  - 0.8.0 (node_modules/@openzeppelin/contracts/math/SafeMath.sol#4)
  - 0.8.9 (contracts/core/ChainlinkArbitrumPriceProvider.sol#3)
  - 0.8.9 (contracts/interfaces/ICore/ChainlinkArbitrumPriceProvider.sol#3)
  - 0.8.9 (contracts/interfaces/ICore/IChainlinkPriceProvider.sol#3)
  - 0.8.9 (contracts/interfaces/ICore/IPriceProvider.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#87) is never used and should be removed
Address.functionCall(address,bytes,Uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#11-101) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#114-120) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#128-139) is never used and should be removed
Address.functionDelegatedCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#14-16) is never used and should be removed
Address.functionDelegatedCall(address,bytes,Uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#14-16) is never used and should be removed
Address.functionStaticCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#18-19) is never used and should be removed
Address.functionStaticCall(address,bytes,Uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#18-19) is never used and should be removed
Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#40-45) is never used and should be removed
Address.sendValue(address,uint256,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#40-45) is never used and should be removed
Address.verifyAllResult(bool,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#201-221) is never used and should be removed
Governable._Governable_init() (contracts/access/Governable.sol#3) is never used and should be removed
```







## UniswapV2LikePriceProvider.sol

```
UniswapV2OracleLibrary.currentBlockTimestamp() (node_modules/uniswap/v2-periphery/contracts/libraries/UniswapV2OracleLibrary.sol#1-18) uses a weak PRNG: uint32(block.timestamp % 4) // 32 (node_modules/uniswap/v2-periphery/contracts/libraries/UniswapV2OracleLibrary.sol#12)
Reference: https://github.com/crytic/slither/wlkt/Detector-Documentation#weak-PRNG

FixedPoint.mul(FixedPoint.uint128, uint256) (node_modules/uniswap/lib/contracts/libraries/FixedPoint.sol#4) is a local variable never initialized
Reference: https://github.com/crytic/slither/wlkt/Detector-Documentation#uninitialized-local-variables

UniswapV2LikePriceProvider.constructor(address,uint256,address).nativeToken_ (contracts/core/UniswapV2LikePriceProvider.sol#5) lacks a reto-check on :
    - nativeToken_ = nativeToken_ (contract/core/UniswapV2LikePriceProvider.sol#16)
Reference: https://github.com/crytic/slither/wlkt/Detector-Documentation#missing-zero-edits-validation

UniswapV2OracleLibrary.currentCumulativePrice(address) (node_modules/uniswap/v2-periphery/contracts/libraries/UniswapV2OracleLibrary.sol#16-36) uses timestamp for comparisons
    - blockTimestamp_ = blockTimestamp (node_modules/uniswap/v2-periphery/contracts/libraries/UniswapV2OracleLibrary.sol#25)
Reference: https://github.com/crytic/slither/wlkt/Detector-Documentation#block-timestamp

Address.verifyCallResult(bool,bytes,string) (node_modules/openzeppelin/contracts/utils/Address.sol#1-22) uses assembly
    - INLINE ASM (node_modules/openzeppelin/contracts/utils/Address.sol#11-21)
Reference: https://github.com/crytic/slither/wlkt/Detector-Documentation#assembly-by-use

Different versions of Solidity is used:
    - Version used: F0.8.0+commit.4e04f4d..F0.8.0+commit.4e04f4d (node_modules/openzeppelin/contracts/Context.sol#1)
    - 0.8.1 (node_modules/openzeppelin/contracts/utils/Address.sol#4)
    - 0.8.0 (node_modules/openzeppelin/contracts/utils/Context.sol#4)
    - 0.8.0+commit.4e04f4d (node_modules/openzeppelin/contracts/utils/Context.sol#1)
    - 0.8.0+commit.4e04f4d (node_modules/uniswap/lib/contracts/libraries/Babylonian.sol#3)
    - >=0.4.0 (node_modules/uniswap/lib/contracts/libraries/FixedPoint.sol#18)
    - >=0.4.0 (node_modules/uniswap/lib/contracts/libraries/Math.sol#1)
    - >=0.4.0 (node_modules/uniswap/lib/contracts/libraries/Math.sol#14)
    - >=0.4.0 (node_modules/uniswap/lib/contracts/libraries/Math.sol#15)
    - >=0.5.0 (node_modules/uniswap/v2-core/contracts/interfaces/ILiquidityProviderFactory.sol#1)
    - >=0.5.0 (node_modules/uniswap/v2-periphery/contracts/libraries/UniswapV2Pair.sol#1)
    - >=0.5.0 (node_modules/uniswap/v2-periphery/contracts/libraries/UniswapV2OracleLibrary.sol#1)
    - >=0.5.0 (node_modules/uniswap/v2-periphery/contracts/libraries/UniswapV2OracleLibrary.sol#2)
    - >=0.5.0 (node_modules/uniswap/v2-periphery/contracts/libraries/UniswapV2OracleLibrary.sol#3)
    - >=0.5.0 (node_modules/uniswap/v2-periphery/contracts/libraries/UniswapV2OracleLibrary.sol#4)
    - 0.8.9 (contracts/core/UniswapV2LikePriceProvider.sol#3)
    - 0.8.9 (contracts/interfaces/Governable.sol#3)
    - 0.8.9 (contracts/interfaces/IERC20.sol#14)
    - 0.8.9 (contracts/interfaces/IUniswapV2LikePriceProvider.sol#3)
Reference: https://github.com/crytic/slither/wlkt/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (node_modules/openzeppelin/contracts/utils/Address.sol#8-10) is never used and should be removed
Address.functionCall(address,bytes,string) (node_modules/openzeppelin/contracts/utils/Address.sol#19-20) is never used and should be removed
Address.functionCallWithValue(address,bytes,int256,string) (node_modules/openzeppelin/contracts/utils/Address.sol#12-19) is never used and should be removed
Address.functionDelegatecall(address,bytes) (node_modules/openzeppelin/contracts/utils/Address.sol#17-18) is never used and should be removed
Address.functionStaticcall(address,bytes) (node_modules/openzeppelin/contracts/utils/Address.sol#15-16) is never used and should be removed
Address.functionStaticcall(address,bytes,bytes) (node_modules/openzeppelin/contracts/utils/Address.sol#13-14) is never used and should be removed
Address.functionStaticcall(address,bytes,bytes,bytes) (node_modules/openzeppelin/contracts/utils/Address.sol#11-12) is never used and should be removed
Address.license() (node_modules/openzeppelin/contracts/Initializable.sol#1-2) is never used and should be removed
Address.verifyCallResult(bool,bytes,string) (node_modules/openzeppelin/contracts/utils/Address.sol#20-22) is never used and should be removed
Babylonian.sqrt(uint256) (node_modules/uniswap/lib/contracts/libraries/Babylonian.sol#20-23) is never used and should be removed
Babylonian.sqrt(uint256) (node_modules/uniswap/lib/contracts/libraries/Babylonian.sol#24-25) is never used and should be removed
FixedPoint.decode(FixedPoint.uint128) (node_modules/uniswap/lib/contracts/libraries/FixedPoint.sol#57-59) is never used and should be removed
FixedPoint.div(FixedPoint.uint128,uint128) (node_modules/uniswap/lib/contracts/libraries/FixedPoint.sol#34-39) is never used and should be removed
FixedPoint.reciprocal(FixedPoint.uint128) (node_modules/uniswap/lib/contracts/libraries/FixedPoint.sol#67-70) is never used and should be removed
FixedPoint.sqrt(FixedPoint.uint128) (node_modules/uniswap/lib/contracts/libraries/FixedPoint.sol#73-75) is never used and should be removed
Initializable._disableInitializers() (node_modules/openzeppelin/contracts/proxy/Initializable.sol#12-13) is never used and should be removed
Initializable._setInitializersVersion(uint32) (node_modules/openzeppelin/contracts/proxy/Initializable.sol#13-14) is never used and should be removed
Math.callInt256(uint256,uint256) (node_modules/openzeppelin/contracts/math/Math.sol#9-10) is never used and should be removed
Math.max(uint256,uint256) (node_modules/openzeppelin/contracts/math/Math.sol#15) is never used and should be removed
Reference: https://github.com/crytic/slither/wlkt/Detector-Documentation#head-code

Pragma version 0.8.2 (node_modules/openzeppelin/contracts/proxy/utils/Initializable.sol#4) allows old versions
Pragma version 0.8.4 (node_modules/openzeppelin/contracts/utils/Address.sol#4) allows old versions
Pragma version 0.8.4 (node_modules/openzeppelin/contracts/utils/math/Math.sol#4) allows old versions
Pragma version 0.8.4 (node_modules/openzeppelin/contracts/utils/math/Math.sol#18) allows old versions
Pragma version 0.4.0 (node_modules/uniswap/lib/contracts/libraries/Babylonian.sol#3) allows old versions
Pragma version 0.4.0 (node_modules/uniswap/lib/contracts/libraries/Babylonian.sol#14) allows old versions
Pragma version 0.5.0 (node_modules/uniswap/v2-core/contracts/interfaces/ILiquidityProviderFactory.sol#1) allows old versions
Pragma version 0.5.0 (node_modules/uniswap/v2-core/contracts/interfaces/ILiquidityProviderFactory.sol#2) allows old versions
Pragma version 0.8.9 (contracts/contracts/Governable.sol#17) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version 0.9.9 (contracts/core/UniswapV2LikePriceProvider.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version 0.9.9 (contracts/interfaces/IERC20.sol#1) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version 0.9.9 (contracts/interfaces/IERC20.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version 0.9.9 (contracts/interfaces/core/IUniswapV2LikePriceProvider.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
pragma 0.8.9 is not recommended for deployment
Reference: https://github.com/crytic/slither/wlkt/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (node_modules/openzeppelin/contracts/utils/Address.sol#10-13):
    - (success,returnData) = target.call{value:(uint256)}(address,amount);
        + success = target.call{value:(uint256)}(address,amount);
        + returnData = target.call{value:(uint256)}(address,amount);
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/openzeppelin/contracts/utils/Address.sol#12-19):
    - (success,returnData) = target.functionCall{value:(bytes)}(address,calldatasize);
        + success = target.functionCall{value:(bytes)}(address,calldatasize);
        + returnData = target.functionCall{value:(bytes)}(address,calldatasize);
Low level call in Address.functionDelegatecall(address,bytes,string) (node_modules/openzeppelin/contracts/utils/Address.sol#15-19):
    - (success,returnData) = target.functionDelegatecall(address,bytes)(address,calldatasize);
        + success = target.functionDelegatecall(address,bytes)(address,calldatasize);
        + returnData = target.functionDelegatecall(address,bytes)(address,calldatasize);
Reference: https://github.com/crytic/slither/wlkt/Detector-Documentation#low-level-calls
```

## UniswapV3PriceProvider.sol

Oracle.sol

## SwapperOracle.sol

# AUTOMATED TESTING

UniswapV2IkeExchange.sol

# AUTOMATED TESTING

```

Pragma version:0.8 (contracts/interfaces/IGovernable.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 (contracts/interfaces/external/ICurveAddressProvider.sol#8) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8-7
Pragma version:0.8 (contracts/interfaces/external/IERC20Metadata.sol#1) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8-7
Pragma version:0.8 (contracts/interfaces/periphery/synth/ISynthOracle.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 (contracts/periphery/auxiliary/CurveLpTokenOracle.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8-7
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendWithValue(address,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#40-65):
  - (success) = recipient.call.value(amount)() (node_modules/@openzeppelin/contract/utils/Address.sol#63)
Low level call in Address.functionCallWithValue(address,bytes,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#128-139):
  - (success) = target.call.value(amount)() (node_modules/@openzeppelin/contract/utils/Address.sol#137)
Low level call in Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#157-166):
  - (success,returnData) = target.staticcall(data) (node_modules/@openzeppelin/contract/utils/Address.sol#164)
Low level call in Address.functionDelegatecall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#184-193):
  - (success,returnData) = target.delegatecall(data) (node_modules/@openzeppelin/contract/utils/Address.sol#192)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function Governorable _Governable init() (contracts/access/Governable.sol#33-43) is not in mixedCase
Parameter Governorable._transferGovernorship(address) (proposeGovernor) (contracts/access/Governable.sol#6) is not in mixedCase
Function ICurveAddressProvider._register() (contract/interfaces/external/ICurveAddressProvider.sol#8) is not in mixedCase
Function ISynthOracle._getPriceInEth(address,uint256) (contract/interfaces/periphery/synth/ISynthOracle.sol#24) is not in mixedCase
Parameters CurveLpTokenOracle._lpTokenPriceInEth(address) (contract/interfaces/periphery/auxiliary/CurveLpTokenOracle.sol#24) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

SynthDefaultOracle.sol

SynthDefaultOracle.getPriceInEth(IErc20) (contract/interfaces/synth/SynthDefaultOracle.sol#85-95) uses timestamp for comparisons
  Dangerous comparisons:
    - require(bool,address)_assetbeta_stalebefore > block.timestamp - lastUpdatedSt,price-is-stale (contracts/periphery/synth/SynthDefaultOracle.sol#95)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp

Address.verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#201-221) uses assembly
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-use

Different Versions of Solidity is used:
  Version used: ('0.6.9', '+0.8.0', '+0.8.1', '+0.8.2')
  - +0.8.2 (node_modules/@openzeppelin/contracts/proxy/utils/Initializable.sol#4)
  - +0.8.1 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#1)
  - +0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20Metadata.sol#4)
  - +0.8.1 (node_modules/@openzeppelin/contracts/contracts/utils/Address.sol#4)
  - +0.8.0 (node_modules/@openzeppelin/contracts/contracts/utils/Context.sol#4)
  - +0.8.0 (node_modules/@openzeppelin/contracts/math/Math.sol#4)
  - +0.8.9 (contracts/interfaces/IGovernable.sol#3)
  - +0.8.9 (contract/interfaces/core/IChainlinkProxyFrontier.sol#3)
  - +0.8.9 (contract/interfaces/external/IERC20.sol#3)
  - +0.8.9 (contracts/interfaces/periphery/synth/ISynthOracle.sol#3)
  - +0.8.9 (contracts/interfaces/periphery/synth/SynthDefaultOracle.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (node_modules/@openzeppelin/contract/utils/Address.sol#95-107) is never used and should be removed
Address.functionCall(address,bytes,string) (node_modules/@openzeppelin/contract/utils/Address.sol#108-119) is never used and should be removed
Address.functionStaticCall(address,bytes,uint256) (node_modules/@openzeppelin/contract/utils/Address.sol#120-131) is never used and should be removed
Address.functionStaticCall(address,bytes,uint256,bytes) (node_modules/@openzeppelin/contract/utils/Address.sol#132-139) is never used and should be removed
Address.functionDelegateCall(address,bytes) (node_modules/@openzeppelin/contract/utils/Address.sol#140-147) is never used and should be removed
Address.functionDelegateCall(address,bytes,uint256) (node_modules/@openzeppelin/contract/utils/Address.sol#148-166) is never used and should be removed
Address.functionDelegateCall(address,bytes,uint256,bytes) (node_modules/@openzeppelin/contract/utils/Address.sol#167-176) is never used and should be removed
Address.functionCall(address,bytes,uint256) (node_modules/@openzeppelin/contract/utils/Address.sol#177-186) is never used and should be removed
Address.functionCall(address,bytes,uint256,bytes) (node_modules/@openzeppelin/contract/utils/Address.sol#187-202) is never used and should be removed
Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#45) is never used and should be removed
Address.verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#201-221) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (node_modules/@openzeppelin/contract/utils/Address.sol#222-233) is never used and should be removed
Address.functionStaticCallWithValue(address,bytes,uint256) (node_modules/@openzeppelin/contract/utils/Address.sol#234-245) is never used and should be removed
Governable._Governable init() (contracts/access/Governable.sol#33) is never used and should be removed
Initializable._disableInitializers() (node_modules/@openzeppelin/contract/proxy/utils/Initializable.sol#19-31) is never used and should be removed
Initializable._setInitializersVersion(uint) (node_modules/@openzeppelin/contracts/proxy/utils/Initializable.sol#33-149) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#decrements

Pragma version:0.8.0 (node_modules/@openzeppelin/contracts/proxy/Initializable.sol#4) allows old versions
Pragma version:0.8.0 (node_modules/@openzeppelin/contracts/proxy/UUPSUpgradeable.sol#4) allows old versions
Pragma version:0.8.0 (node_modules/@openzeppelin/contracts/token/IERC20/IERC20.sol#4) allows old versions
Pragma version:0.8.1 (node_modules/@openzeppelin/contracts/token/IERC20/IERC20Metadata.sol#4) allows old versions
Pragma version:0.8.1 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#4) allows old versions
Pragma version:0.8.1 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20Metadata.sol#4) allows old versions
Pragma version:0.8.5 (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:0.8.5 (contracts/interfaces/IGovernable.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8-7
Pragma version:0.8.5 (contract/interfaces/IFeeRecipient.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8-7
Pragma version:0.8.5 (contract/interfaces/ICurveAddressProvider.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8-7
Pragma version:0.8.5 (contract/interfaces/ISynthOracle.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8-7
Pragma version:0.8.5 (contract/interfaces/periphery/synth/ISynthOracle.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8-7
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendWithValue(address,uint256) (node_modules/@openzeppelin/contract/utils/Address.sol#40-65):
  - (success) = recipient.call.value(amount)() (node_modules/@openzeppelin/contract/utils/Address.sol#63)
Low level call in Address.functionCallWithValue(address,bytes,uint256) (node_modules/@openzeppelin/contract/utils/Address.sol#128-139):
  - (success) = target.call.value(amount)() (node_modules/@openzeppelin/contract/utils/Address.sol#137)
Low level call in Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contract/utils/Address.sol#157-166):
  - (success,returnData) = target.staticcall(data) (node_modules/@openzeppelin/contract/utils/Address.sol#164)
Low level call in Address.functionDelegatecall(address,bytes,string) (node_modules/@openzeppelin/contract/utils/Address.sol#184-193):
  - (success,returnData) = target.delegatecall(data) (node_modules/@openzeppelin/contract/utils/Address.sol#192)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function Governorable _Governable init() (contracts/access/Governable.sol#33-43) is not in mixedCase
Parameter Governorable._transferGovernorship(address) (proposeGovernor) (contracts/access/Governable.sol#6) is not in mixedCase
Parameter SynthDefaultOracle._addOrDeleteAsset(IErc20)_asset (contract/interfaces/synth/SynthDefaultOracle.sol#6) is not in mixedCase
Parameter SynthDefaultOracle._addOrDeleteAsset(IErc20)_asset (contract/interfaces/synth/SynthDefaultOracle.sol#7) is not in mixedCase
Parameter SynthDefaultOracle._addOrDeleteAsset(IErc20)_asset (contract/interfaces/synth/SynthDefaultOracle.sol#8) is not in mixedCase
Parameter SynthDefaultOracle._addOrDeleteAsset(IErc20)_asset (contract/interfaces/synth/SynthDefaultOracle.sol#9) is not in mixedCase
Parameter SynthDefaultOracle._addOrDeleteAsset(IErc20)_asset (contract/interfaces/synth/SynthDefaultOracle.sol#10) is not in mixedCase

```

UniswapV3CrossPoolOracle.sol

- Some unused returns were correctly flagged by Slither.
  - The weak PRNG flagged by Slither is a false positive.
  - No major issues found by Slither.

## 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 the smart contracts and sent the compiled results to the analyzers to locate any vulnerabilities.

### MythX results:

#### Governable.sol

No issues found by MythX.

#### ChainlinkArbitrumPriceProvider.sol

Report for contracts/core/ChainlinkArbitrumPriceProvider.sol https://dashboard.mythx.io/#/console/analyses/d22900c1-95c3-4c59-9a67-4e03fffa9f38d			
Line	SWC Title	Severity	Short Description
10	(SWC-123) Requirement Violation	Low	Requirement violation.

#### ChainlinkAvalanchePriceProvider.sol

Report for contracts/core/ChainlinkAvalanchePriceProvider.sol https://dashboard.mythx.io/#/console/analyses/d690ffbe-f5dd-417b-8f45-9218d72d0957			
Line	SWC Title	Severity	Short Description
10	(SWC-123) Requirement Violation	Low	Requirement violation.

#### ChainlinkMainnetPriceProvider.sol

No issues found by MythX.

#### ChainlinkPolygonPriceProvider.sol

Report for contracts/core/ChainlinkPolygonPriceProvider.sol https://dashboard.mythx.io/#/console/analyses/fb18196d-c02a-442a-bfba-0db934f64646			
Line	SWC Title	Severity	Short Description
10	(SWC-123) Requirement Violation	Low	Requirement violation.

#### ChainlinkPriceProvider.sol

Report for contracts/core/ChainlinkPriceProvider.sol https://dashboard.mythx.io/#/console/analyses/3df55a56-9594-47af-a044-4d7800c70187			
Line	SWC Title	Severity	Short Description
16	(SWC-123) Requirement Violation	Low	Requirement violation.
86	(SWC-123) Requirement Violation	Low	Requirement violation.

**PriceProvidersAggregator.sol**

No issues found by MythX.

**UniswapV2LikePriceProvider.sol**

Report for contracts/core/UniswapV2LikePriceProvider.sol  
<https://dashboard.mythx.io/#/console/analyses/c399d6c2-2540-4ef4-9def-96207b0ecab3>

Line	SWC Title	Severity	Short Description
199	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
206	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
206	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
209	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
209	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered

**UniswapV3PriceProvider.sol**

No issues found by MythX.

**Oracle.sol**

No issues found by MythX.

**SwapperOracle.sol**

Report for contracts/periphery/SwapperOracle.sol  
<https://dashboard.mythx.io/#/console/analyses/61c53b2b-714d-4360-80d8-205fc0453e51>

Line	SWC Title	Severity	Short Description
100	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "***" discovered
100	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
100	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
101	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "***" discovered
101	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
101	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
164	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered

**Swapper.sol**

Report for contracts/swapper/Swapper.sol  
<https://dashboard.mythx.io/#/console/analyses/d66039ec-898c-4a17-b3bf-d290a3b39ec9>

Line	SWC Title	Severity	Short Description
73	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered
90	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
90	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
90	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "***" discovered
111	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered
128	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "***" discovered
128	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
128	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered

**UniswapV2LikeExchange.sol**

Report for contracts/swapper/UniswapV2LikeExchange.sol  
<https://dashboard.mythx.io/#/console/analyses/0e1bb03e-5302-4b4e-bc9e-58ba657dc87b>

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

### CTokenOracle.sol

Report for contracts/periphery/synth/CTokenOracle.sol  
<https://dashboard.mythx.io/#/console/analyses/40cb94a4-395e-4b66-8226-5f3bal679b46>

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

### CurveLpTokenOracle.sol

No issues found by MythX.

### SynthDefaultOracle.sol

Report for contracts/periphery/synth/SynthDefaultOracle.sol  
<https://dashboard.mythx.io/#/console/analyses/0517de04-ed66-4b35-8ddd-7d62dfde5e57>

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

### UniswapV3CrossPoolOracle.sol

No issues found by MythX.

- The requirement violations and assert violations are all false positives.
- Integer Overflows and Underflows flagged by MythX are false positives, as those contracts are using Solidity ^0.8.0 version. After the Solidity version 0.8.0 Arithmetic operations revert to underflow and overflow by default.
- No major issues found by MythX.

