// HALBORN

SUSHISWAP - MISO

Smart Contract Security Audit

Prepared by: **Halborn**

Date of Engagement: **April 15th-27th, 2021**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY 6
CONTACTS 7
1 EXECUTIVE OVERVIEW 8
1.1 INTRODUCTION 9
1.2 AUDIT SUMMARY 10
1.3 TEST APPROACH & METHODOLOGY 10
RISK METHODOLOGY 11
1.4 SCOPE 13
2 ASSESSMENT SUMMARY & FINDINGS OVERVIEW 15
3 FINDINGS & TECH DETAILS 16
3.1 (HAL-01) USE OF BLOCK.TIMESTAMP - LOW 18
Description 18
Code Location 18
Risk Level 20
Recommendation 20
Remediation Plan 21
3.2 (HAL-02) UINT256 OVERFLOW - LOW 22
Description 22
Code Location 22
Risk Level 22
Recommendation 22
Remediation Plan 23
3.3 (HAL-03) FLOATING PRAGMA - LOW 24
Description 24

Code Location 24
Risk Level 24
Recommendations 24
Remediation Plan 24
3.4 (HAL-04) UNINITIALIZED VARIABLE - LOW 25
Description 25
Code Location 25
Risk Level 26
Recommendations 26
Remediation Plan 26
3.5 (HAL-05) IGNORE RETURN VALUES - LOW 27
Description 27
Code Location 27
Risk Level 28
Recommendation 28
Remediation Plan 28
3.6 (HAL-06) INIT FUNCTION SHOULD BE CALLED - INFORMATIONAL 29
Description 29
Risk Level 30
Recommendations 30
Remediation Plan 31
3.7 (HAL-07) POSSIBLE MISUSE OF PUBLIC FUNCTIONS - INFORMATIONAL 32
Description 32
Risk Level 32
Recommendations 32

Remediation Plan 32

3.8 (HAL-08) MISSING ADDING OPERATOR ROLE AND INTERFACE TO THE DEPLOYED CHILD CONTRACT - INFORMATIONAL 33

Description 33

Code Location 34

Risk Level 35

Recommendations 35

Remediation Plan 35

3.9 (HAL-09) POINTLIST FACTORY DEPRECATION HAS NO EFFECT - INFORMATIONAL 36

Description 36

Code Location 36

Risk Level 36

Recommendations 37

Remediation Plan 37

3.10 (HAL-10) INDEX OUT OF RANGE MISSING CHECKS - INFORMATIONAL 38

Description 38

Code Location 40

Risk Level 40

Recommendations 40

Remediation Plan 40

3.11 (HAL-11) SLOT0 NOT USED - INFORMATIONAL 41

Description 41

Code Location 43

Risk Level 45

Recommendations 45
Remediation Plan 45
3.12 (HAL-12) DISORDERED MATH OPERATIONS - INFORMATIONAL 46
Description 46
Code Location 46
Risk Level 47
Recommendations 47
Remediation Plan 47
3.13 (HAL-13) INTEGER TRUNCATION - INFORMATIONAL 48
Description 48
Code Location 48
Risk Level 50
Recommendations 50
Remediation Plan 50
3.14 (HAL-14) MISMATCHED STRUCTS AND INTERFACES - INFORMATIONAL 51
Description 51
Code Location 51
Risk Level 52
Recommendations 52
Remediation Plan 53
3.15 (HAL-15) MISSING CHECKS - INFORMATIONAL 54
Description 54
Code Location 54
Risk Level 54
Recommendations 55
Remediation Plan 55

4 MANUAL TESTING 56
4.1 ERC20 Token 59
Delegate By Signature 59
4.2 SushiToken 60
4.3 Uniswap 62
4.4 BoringBatchable 63
4.5 MISOAccessFactory 64
4.6 BatchAuction 64
4.7 TokenVault 65
4.8 MISOFermentor 67
4.9 MISOTokenFactory 68
4.10 Tokens 70
4.11 Call value checks 70
5 AUTOMATED TESTING 71
5.1 SYMBOLIC EXECUTION SECURITY ASSESSMENT 73
Description 73
Results 73
5.2 STATIC ANALYSIS REPORT 80
Description 80
Results 80
5.3 AUTOMATED SECURITY SCAN 84
Description 84
Results 84

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	04/20/2021	Gabi Urrutia
0.3	Document Edits	04/27/2021	Nishit Majithia
0.5	Document Edits	04/27/2021	Gabi Urrutia
0.7	Document Edits	04/28/2021	Ferran Celades
0.9	Document Edits	04/29/2021	Nishit Majithia
1.0	Final Draft	04/30/2021	Steven Walbroehl
1.1	Final Version	05/03/2021	Steven Walbroehl
1.2	Remediation Plan	05/10/2021	Nishit Majithia

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
Nishit Majithia	Halborn	nishit.majithia@halborn.com
Ferran Celades	Halborn	ferran.celades@halborn.com

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

Minimal Initial SushiSwap Offering, also known as MISO, is a set of open source smart contracts designed to simplify the process of launching new projects on the SushiSwap exchange. MISO aims to promote new capital and transactions on the exchange by increasing the attractiveness of SushiSwap as a place for token creators and communities to initiate new project tokens. In addition, MISO aims to create a launchpad for creators of technical and non-technical projects so that the community and projects can access all the options they need to safely and successfully deploy to the SushiSwap exchange

With this in mind, the development of MISO has led to the addition of some features that were not outlined in the original proposal. Similar to other exchanges in the cryptocurrency market, SushiSwap needs a launchpad that matches its user base and the items it wants to offer. MISO will be used as the personal platform for the SushiSwap community to start the project, so that the project and the community know that all aspects of the start-up are handled through a trusted open source contract.

SushiSwap engaged Halborn to conduct a security assessment on their Smart contracts beginning on April 15th, 2021 and ending April 27th, 2021. The security assessment was scoped to the smart contract provided in the Github repository MISO Smart Contracts and an audit of the security risk and implications regarding the changes introduced by the development team at SushiSwap prior to its production release shortly following the assessments deadline.

Though this security audit's outcome is satisfactory, only the most essential aspects were tested and verified to achieve objectives and deliverables set in the scope due to time and resource constraints. It is essential to note the use of the best practices for secure smart-contract development.

1.2 AUDIT SUMMARY

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

The purpose of this audit to achieve the following:

- Ensure that smart contract functions are intended.
- Identify potential security issues with the smart contracts.

In summary, Halborn identified few security risks, and recommends performing further testing to validate extended safety and correctness in context to the whole set of contracts. External threats, such as economic attacks, oracle attacks, and inter-contract functions and calls should be validated for expected logic and state.

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 the smart contract audit. While manual testing is recommended to uncover flaws in logic, process,and implementation; automated testing techniques help enhance coverage of smart contracts 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 (brownie console and manual deployments on Ganache)
- Manual testing by custom Python 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 (Goerli, Remix IDE)
- Symbolic Execution / EVM bytecode security assessment (Manticore)

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. It's quantitative model ensures repeatable and accurate measurement while enabling users to see the underlying vulnerability characteristics that was 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 smart contracts:

/

- MISOFermenter.sol
- MISOLiquidityLauncher.sol
- MISOTokenFactory.sol

Access/

- MISOAccessControls.sol
- MISOAccessFactory.sol
- MISOAdminAccess.sol
- PointListFactory.sol
- PointList.sol

Auctions/

- BatchAuction.sol
- DutchAuction.sol
- HyperbolicAuction.sol

Helper/

- MISOHelper.sol

Liquidity/

- LiquidityZAP.sol
- PoolLiquidity02.sol
- PoolLiquidity.sol

Utils/

- BoringBatchable.sol
- BoringERC20.sol
- BoringFactory.sol
- BoringMath.sol
- BoringOwnable.sol
- CloneFactory.sol
- Documents.sol
- Owned.sol

- SafeMathPlus.sol
- SafeTransfer.sol
- Timelock.sol
- WETH9.sol

Vault/
- GnosisSafeFactory.sol
- TokenVault.sol

Commit ID: 031e04ed0c2d987898c6b335f046d419e2688a98
Fix commit ID: 5377235cc727a9c5e4f2ff33d037069c076384c0

OUT-OF-SCOPE:
Market Smart Contracts:
- Crowdsale.sol
- MISOMarket.sol

Farming Smart Contracts
- Farms/MISOMasterChef.sol
- Farms/MISOMasterChefV2.sol
- MISOFarmFactory.sol

External libraries and economics attacks.

2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	0	0	5	10

LIKELIHOOD

IMPACT

(HAL-01)				
(HAL-03) (HAL-04)				
		(HAL-02) (HAL-05)		
(HAL-06) (HAL-07) (HAL-08) (HAL-09) (HAL-10) (HAL-11) (HAL-12) (HAL-13) (HAL-14) (HAL-15)				

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
USE OF BLOCK.TIMESTAMP	Low	SOLVED - 05/10/2021
UINT256 OVERFLOW	Low	SOLVED - 05/10/2021
FLOATING PRAGMA	Low	SOLVED - 05/10/2021
UNINITIALIZED VARIABLE	Low	SOLVED - 05/10/2021
IGNORE RETURN VALUES	Low	PARTIALLY SOLVED - 05/10/2021
INIT FUNCTION SHOULD BE CALLED	Informational	ACKNOWLEDGE
POSSIBLE MISUSE OF PUBLIC FUNCTIONS	Informational	ACKNOWLEDGE
MISSING ADDING OPERATOR ROLE AND INTERFACE TO THE DEPLOYED CHILD CONTRACT	Informational	SOLVED - 05/10/2021
POINTLIST FACTORY DEPRECATION HAS NO EFFECT	Informational	SOLVED - 05/10/2021
INDEX OUT OF RANGE MISSING CHECKS	Informational	PARTIALLY SOLVED - 05/10/2021
SLOT0 NOT USED	Informational	ACKNOWLEDGE
DISORDERED MATH OPERATIONS	Informational	SOLVED - 05/10/2021
INTEGER TRUNCATION	Informational	ACKNOWLEDGE
MISMATCHED STRUCTS AND INTERFACES	Informational	PARTIALLY SOLVED - 05/10/2021
MISSING CHECKS	Informational	SOLVED - 05/20/2021

FINDINGS & TECH DETAILS

3.1 (HAL-01) USE OF BLOCK.TIMESTAMP - LOW

Description:

During a manual static review, the tester noticed the use of block.timestamp in few contracts. The contract developers should be aware that this does not mean current time. Miners can influence the value of block.timestamp to perform Maximal Extractable Value (MEV) attacks. The use of now creates a risk that time manipulation can be performed to manipulate price oracles. Miners can modify the timestamp by up to 900 seconds.

Code Location:

Listing 1: PoolLiquidity.sol (Lines 141)

```
139     /// @notice Deposits ETH to the contract.
140     function depositETH() public payable {
141         require(block.timestamp < deadline, "PoolLiquidity:
                Contract has expired");
142         require(liquidityAdded == 0, "PoolLiquidity: Liquidity
                already added");
143         if (msg.value > 0 ) {
144             WETH.deposit{value : msg.value}();
145         }
146     }
```

Listing 2: PoolLiquidity.sol (Lines 153)

```
152     function depositTokens(uint256 amount) external returns (bool
            success) {
153         require(block.timestamp < deadline, "PoolLiquidity:
                Contract has expired");
154         require(amount > 0, "PoolLiquidity: Token amount must be
                greater than 0");
155         require(liquidityAdded == 0, "PoolLiquidity: Liquidity
                already added");
156         _safeTransferFrom(address(token), msg.sender, amount);
```

```
157     }
```

Listing 3: PoolLiquidity.sol (Lines 182)

```
180     function withdrawLPTokens() external returns (uint256
            liquidity) {
181         require(accessControls.hasOperatorRole(msg.sender), "
                PoolLiquidity: Sender must be operator");
182         require(block.timestamp >= unlock, "PoolLiquidity:
                Liquidity is locked");
183         liquidity = IERC20(tokenWETHPair).balanceOf(address(this))
                ;
```

Listing 4: PoolLiquidity.sol (Lines 192)

```
189     function withdrawDeposits() external {
190         require(accessControls.hasOperatorRole(msg.sender), "
                PoolLiquidity: Sender must be operator");
191         require(liquidityAdded == 0, "PoolLiquidity: Liquidity is
                locked");
192         require(block.timestamp > expiry, "PoolLiquidity: Timer
                has not yet expired");
```

Listing 5: PoolLiquidity.sol (Lines 220,221)

```
219         require(accessControls.hasOperatorRole(msg.sender), "
                PoolLiquidity: Sender must be operator");
220         require(block.timestamp > deadline, "PoolLiquidity:
                Deposit deadline has not passed");
221         require(block.timestamp < expiry, "PoolLiquidity: Contract
                 has expired");
```

Listing 6: PoolLiquidity02.sol (Lines 188)

```
186     function _deposit(uint amount, address token) private returns
            (bool success) {
187         // CC: shouldn't be require(block.timestamp < deadline, if
                 deadline is deposit deadline?
188         require(block.timestamp < expiry, "PoolLiquidity02:
                Contract has expired");
```

Listing 7: PoolLiquidity02.sol (Lines 217)

```
215     function withdrawLPTokens() external returns (uint256
            liquidity) {
216         require(accessControls.hasOperatorRole(msg.sender), "
                PoolLiquidity02: Sender must be operator");
217         require(block.timestamp >= unlock, "PoolLiquidity02:
                Liquidity is locked");
218         liquidity = IERC20(tokenPair).balanceOf(address(this));
```

Listing 8: PoolLiquidity02.sol (Lines 226)

```
224     function withdrawDeposits() external {
225         require(accessControls.hasOperatorRole(msg.sender), "
                PoolLiquidity02: Sender must be operator");
226         require(block.timestamp > expiry, "PoolLiquidity02: Timer
                has not yet expired");
```

Listing 9: PoolLiquidity02.sol (Lines 255,256)

```
252     function _launchLiquidityPool() internal returns (uint256
            liquidity) {
253         /// GP: Could add a flag to give the option for a
                trustless launch
254         require(accessControls.hasOperatorRole(msg.sender), "
                PoolLiquidity02: Sender must be operator");
255         require(block.timestamp > deadline, "PoolLiquidity02:
                Deposit deadline has not passed");
256         require(block.timestamp < expiry, "PoolLiquidity02:
                Contract has expired");
```

Risk Level:

Likelihood - 1
Impact - 4

Recommendation:

Use block.number instead of block.timestamp or now to reduce the risk of MEV attacks. Check if the timescale of the project occurs across years,

days and months rather than seconds. If possible, it is recommended to use Oracles.

Remediation Plan:

SOLVED: Although `block.timestamp` is still in use in contract `Liquidity/PostAuctionLauncher.sol`, timescales are higher than 900 seconds which is considered as safe.

3.2 (HAL-02) UINT256 OVERFLOW - LOW

Description:

An overflow happens when an arithmetic operation reaches the maximum or minimum size of a type. For instance here in PoolLiquidity.sol and PoolLiquidity02.sol, value received from users stored in the uint256 type. In computer programming, an integer overflow occurs when an arithmetic operation attempts to create a numeric value that is outside of the range that can be represented with a given number of bits -- either larger than the maximum or lower than the minimum representable value.

Code Location:

Listing 10: initPoolLiquidity() method in both PoolLiquidity.sol and in PoolLiquidity02.sol (Lines 126)

```
123     wallet = _wallet;
124     deadline = _deadline;
125     launchwindow = _launchwindow;
126     expiry = _deadline + _launchwindow;
127     locktime = _locktime;
```

Risk Level:

Likelihood - 3
Impact - 2

Recommendation:

It is recommended to use vetted safe math libraries for arithmetic operations consistently throughout the smart contract system

Reference:

Ethereum Smart Contract Best Practices - Integer Overflow and Underflow

Remediation Plan:

SOLVED: The Pool Liquidity contracts has been restricted to just auctions and timing variables(_expiry, _deadline, Launchwindow etc) have also been removed.

3.3 (HAL-03) FLOATING PRAGMA - LOW

Description:

The Smart Contract Liquidity/LiquidityZAP.sol uses the floating pragma ^0.6.12. Contracts should be deployed with the same compiler version and flags used during development and testing. Locking the **pragma** helps to ensure that contracts do not accidentally get deployed using another pragma. For example, an outdated pragma version might introduce bugs that affect the contract system negatively or recently released pragma versions may have unknown security vulnerabilities.

Code Location:

Listing 11: LiquidityZAP.sol

```
1 pragma solidity ^0.6.12;
```

Risk Level:

Likelihood - 1
Impact - 3

Recommendations:

Consider locking the pragma version. It is not recommended to use a floating pragma in production. Apart from just locking the pragma version in the code, the sign (>=) need to be removed. It is possible to lock the pragma by fixing the version both in truffle-config.js for Truffle framework or in hardhat.config.js for HardHat framework.

Remediation Plan:

SOLVED: LiquidityZAP.sol contract was removed.

3.4 (HAL-04) UNINITIALIZED VARIABLE - LOW

Description:

The withdrawLPTokens function in PoolLiquidity contract does use the variable unlock, as seen on Listing 12, whose the initial values is 0. The only time that variable is set is when calling the _launchLiquidityPool function as shown in Listing 13. This means that the unlock variable check on Listing 12 does always validate causing the _safeTransfer to be called.

Code Location:

Listing 12: PoolLiquidity.sol (Lines 182)

```
180 function withdrawLPTokens() external returns (uint256 liquidity) {
181     require(accessControls.hasOperatorRole(msg.sender), "
            PoolLiquidity: Sender must be operator");
182     require(block.timestamp >= unlock, "PoolLiquidity: Liquidity
            is locked");
183     liquidity = IERC20(tokenWETHPair).balanceOf(address(this));
184     require(liquidity > 0, "PoolLiquidity: Liquidity must be
            greater than 0");
185     _safeTransfer(tokenWETHPair, wallet, liquidity);
186 }
```

Listing 13: PoolLiquidity.sol (Lines 251)

```
249     liquidity = IUniswapV2Pair(tokenWETHPair).mint(address(this));
250     liquidityAdded = liquidityAdded.add(liquidity);
251     unlock = block.timestamp + locktime;
252     emit LiquidityAdded(liquidityAdded);
```

Risk Level:

Likelihood - 1
Impact - 3

Recommendations:

If is recommended to initialize all internal variables on the same function, either on the constructor or a custom `init` method. However, using unitialized variables an expecting them to have a value could cause unexpected behaviours on the execution flow.

Remediation Plan:

SOLVED: Calling `withdrawLPTokens` is not allowed without first calling `finalize` due to the `launched` required lock. The `unlock` variable is set on `finalize`

3.5 (HAL-05) IGNORE RETURN VALUES - LOW

Description:

The return value of an external call is not stored in a local or state variable. In contracts liquidity/LiquidityZAP.sol, Recipes/MISORecipe01.sol and Vault/TokenVault.sol, there are few instances where external methods are being called and return values(bool) is being ignored.

Code Location:

Listing 14: LiquidityZAP.sol (Lines 127)

```
125         uint256 outTokens = UniswapV2Library.getAmountOut(
                buyAmount, reserveWeth, reserveTokens);
126
127         _WETH.transfer(_tokenWETHPair, buyAmount);
```

Listing 15: LiquidityZAP.sol (Lines 171)

```
169         address to
170     ) public returns (uint amountA, uint amountB) {
171         IUniswapV2Pair(_tokenWETHPair).transferFrom(msg.sender,
                _tokenWETHPair, liquidity); // send liquidity to pair
172         (uint amount0, uint amount1) = IUniswapV2Pair(
                _tokenWETHPair).burn(to);
```

Listing 16: LiquidityZAP.sol (Lines 207)

```
205         uint256 outTokens = UniswapV2Library.getAmountOut(
                amountETH, reserveWeth, reserveTokens);
206
207         _WETH.transfer(_tokenWETHPair, amountETH);
```

Listing 17: MISORecipe01.sol (Lines 129)

```
127         ISushiToken token = ISushiToken(tokenFactory.createToken(
                _name, _symbol, 1, msg.sender, tokensToMint));
128
129         token.approve(address(misoMarket), tokensToMarket);
```

Listing 18: MISORecipe01.sol

```
154         token.transfer(address(poolLiquidity),tokensToLiquidity);
```

Listing 19: MISORecipe01.sol (Lines 193)

```
193         token.transfer(address(farm),tokensToFarm);
194         uint256 allocPoint = 10;
```

Listing 20: TokenVault.sol (Lines 76)

```
75          UserInfo storage userItem = users[_withdrawer];
76          userItem.lockedItemsWithUser.add(_tokenAddress);
77          userItem.lockToItems[_tokenAddress].push(_id);
```

Risk Level:

Likelihood - 3
Impact - 2

Recommendation:

Add return value check to avoid unexpected crash of the contract. Return value check will help in handling the exceptions better way.

Remediation Plan:

PARTIALLY SOLVED: LiquidityZAP.sol contract was removed. Return value check is still missing in MISORecipe01.sol and TokenVault.sol contracts.

3.6 (HAL-06) INIT FUNCTION SHOULD BE CALLED - INFORMATIONAL

Description:

All of the `init` methods present on the contracts include a locking variable that is set when the `init` function is fully executed. This prevents calling `init` a second time or by accident.

The following contracts are part of the `Access` directory and they require to initialise the underlaying `AccessControl` contract or the `DEFAULT_ADMIN_ROLE` role and the ownership of the contract could be taken by a 3rd party:

- `MISOAccessControls` should call `initAccessControls` .
- `MISOAccessFactory` should call `initMISOAccessFactory` .
- `MISOAdminAccess` should call `initAccessControls`.
- `PointList` should call `initPointList`.
- `PointListFactory` should call `initPointListFactory`.

The following contracts contain `init` methods that initialise global variables used during the contract, if those `init` methods are not called before interacting with the contract unexpected behaviours and the possibility of contract takeover could occur:

Auction:

- `BatchAuction` should call `initAuction`.
- `Crowdsale` (OUT OF SCOPE).
- `DutchAuction` should call `initAuction`.
- `HyperbolicAuction` should call `initAuction`.

Liquidity:

- `LiquidityZAP` should call `initUniswapZAP`.

- PoolLiquidity should call initPoolLiquidity and launchLiquidityPool (otherwise the unlock variable is uninitialized)
- PoolLiquidity02 should call initPoolLiquidity.

Tokens:

- FixedToken should call initToken.
- MintableToken should call initToken.
- SushiToken should call initToken.

Contracts:

- MISOFarmFactory should call initMISOFarmFactory.
- MISOFermenter should call _initMISOFermenter.
- MISOLiquidityLauncher should call initMISOLiquidityLauncher.
- MISOMarket (OUT OF SCOPE).
- MISOTokenFactory should call initMISOTokenFactory.

Risk Level:

Likelihood - 1
Impact - 1

Recommendations:

As an example, an attacker could create an automated tool that scans all contract factories for new contracts being deployed. Once a new contract is created automatically call the init method gaining control of it.

It is recommended to initialize internal state variables and access controls during the creation of the contract. This would prevent an attacker to initializing the contract and taking the control of it. If the init function is the desired behaviour then a guideline on deployment should be given to the customers reminding them to always call the init function.

Remediation Plan:

The SushiSwap team accepts the risk.

3.7 (HAL-07) POSSIBLE MISUSE OF PUBLIC FUNCTIONS - INFORMATIONAL

Description:

In smart contracts Access/MISOAdminAccess.sol, Access/PointListFactory.sol, Auction/BatchAuction.sol, Auction/DutchAuction.sol, Auction/HyperbolicAuction.sol, Token/FixedToken.sol, Token/MintableToken.sol, Token/SushiToken.sol, Helper/MISOHelper.sol, Utils/BoringBatchable.sol, Utils/BoringFactory.sol, Utils/BoringOwnable.sol, Utils/Owned.sol, Utils/Timeclock.sol, Utils/WETH9.sol, Vault/GnosisSafeFactory.sol and Vault/TokenVault.sol there are methods like initAccessControls(), initPointListFactory(), deprecateFactory(), setMinimumFee(), setDividends(), deployPointList(), transferAnyERC20Token(), commitEth(), commitTokens(), finalize(), withdrawTokens(), initMarket() etc. are marked as public but they are never directly called within the same contract or in any of its descendants.

Risk Level:

Likelihood - 1
Impact - 1

Recommendations:

If the function is not intended to be called internally or by descendants, it is better to mark all these functions as external instead of public to save gas.

Remediation Plan:

The SushiSwap team accepts the risk.

3.8 (HAL-08) MISSING ADDING OPERATOR ROLE AND INTERFACE TO THE DEPLOYED CHILD CONTRACT - INFORMATIONAL

Description:

PointList is checking for OperatorRole in setPoints, that means that the admin would have to add itself as operator in order to set the points. This is causing all PointList's created with the PointListFactory to not allow the admin to setPoints as shown in Listing 22.

Deploying a PointList from the factory and setting points i by calling setPoints is giving the error of missing operator role as shown in Figure Figure 1.

```
>>> PointListFactory[0].deployPointList(accounts[0], [], [])
Transaction sent: 0x6ff55df2a182b365075aecaa29db4ad53bf3ef37cb74d58b174f984b9d2ea2b3
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 202
  PointListFactory.deployPointList confirmed - Block: 204   Gas used: 242208 (3.60%)

<Transaction '0x6ff55df2a182b365075aecaa29db4ad53bf3ef37cb74d58b174f984b9d2ea2b3'>
>>> PointListFactory[0].lists(0)
'0x9E28c6dBb2e268aCF172e6908A40B56968aA3792'
>>> PointList.at(PointListFactory[0].lists(0)).setPoints([], [])
  File "<console>", line 1, in <module>
  File "brownie/network/contract.py", line 1676, in __call__
    return self.transact(*args)
  File "brownie/network/contract.py", line 1545, in transact
    raise AttributeError(
AttributeError: Final argument must be a dict of transaction parameters that includes a `from` f
ield specifying the sender of the transaction
>>> PointList.at(PointListFactory[0].lists(0)).setPoints([], [], {'from':accounts[0]})
Transaction sent: 0x748be4b1aec87fc10ec4fc409342c4a17016efe2b4afa48494dde09975c6b310
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 203
  PointList.setPoints confirmed (PointList.setPoints: Sender must be operator) - Block: 205   Ga
s used: 24591 (0.37%)

<Transaction '0x748be4b1aec87fc10ec4fc409342c4a17016efe2b4afa48494dde09975c6b310'>
>>>
```

Figure 1: Require error returned when calling setPoints after the contract has been deployed using PointListFactory

After adding the operator role with addOperatorRole it is possible to call setPoints as shown in Figure Figure 2.

Furthermore, It was noticed that the address(this) is also added as an operator to the under-laying created child contract as shown in line 129 on Listing 21. The operator role does only allow the setPoints function to

```
>>> PointList.at(PointListFactory[0].lists(0)).addOperatorRole(accounts[0], {'from':accounts[0]})
Transaction sent: 0x3c2f147346250eb25f5aa1923fee9e4603dc05d30a6a426d6f057c8cf3a39138
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 208
  PointList.addOperatorRole confirmed - Block: 210   Gas used: 27353 (0.41%)

<Transaction '0x3c2f147346250eb25f5aa1923fee9e4603dc05d30a6a426d6f057c8cf3a39138'>
>>> PointList.at(PointListFactory[0].lists(0)).setPoints([accounts[0]], [123], {'from':accounts[0]
})
Transaction sent: 0x0383fd9e1e18bef7fd292d3d3e0d2a89b49c37a951963cf279036193f1d00cc4
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 209
  PointList.setPoints confirmed - Block: 211   Gas used: 26218 (0.39%)

<Transaction '0x0383fd9e1e18bef7fd292d3d3e0d2a89b49c37a951963cf279036193f1d00cc4'>
>>> PointList.at(PointListFactory[0].lists(0)).points(accounts[0])
123
>>>
```

Figure 2: Successfully calling the setPoints after the operator role has been added

be called. However, there exists no interface from the PointListFactory to call any of the methods present on the created child PointList.

Moreover, the PointListFactory is adding itself as an operator on the created PointList, but the creator of the PointList (list owner and admin) can easily remove it by calling removeOperatorRole.

Code Location:

Listing 21: PointListFactory.sol (Lines 134,129)

```
126 pointList = createClone(pointListTemplate);
127 if (_accounts.length > 0) {
128     IPointList(pointList).initPointList(address(this));
129     MISOAccessControls(pointList).addOperatorRole(address(this));
130     IPointList(pointList).setPoints(_accounts, _amounts);
131     MISOAccessControls(pointList).addAdminRole(_listOwner);
132     MISOAccessControls(pointList).removeAdminRole(address(this));
133 } else {
134     IPointList(pointList).initPointList(_listOwner);
135 }
```

Listing 22: PointList.sol (Lines 70)

```
 69 function setPoints(address[] memory _accounts, uint256[] memory
      _amounts) external override {
 70 require(hasOperatorRole(msg.sender), "PointList.setPoints: Sender
```

```
      must be operator");
71 require(_accounts.length != 0);
72 require(_accounts.length == _amounts.length);
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendations:

If that's an intended functionality it should be explained to the customers when deploying the PointListFactory contract and using it.

Furthermore, if no interface will be provided to the PointListFactory is better to remove access control to under-laying contract since compromising the PointListFactory would allow any child PointList to be compromised as well, this will make sure that the least privilege security principle is meet.

Remediation Plan:

SOLVED: Removed addOperatorRole for the Factory address, and checking for hasAdminRole on the deployed child contract

3.9 (HAL-09) POINTLIST FACTORY DEPRECATION HAS NO EFFECT - INFORMATIONAL

Description:

Deprecating a PointListFactory does not do anything aside from emitting an event an updating the internal newAddress variable as shown in Figure 23. The newAddress variable is not checked anywhere inside the contract, meaning that the Factory can still create pointList childs based on the pointListTemplate

Code Location:

Listing 23: PointListFactory.sol (Lines 89)

```
81 /**
82  * @notice Deprecates factory.
83  * @param _newAddress Blank address.
84  */
85 function deprecateFactory(address _newAddress) public {
86     require(accessControls.hasAdminRole(msg.sender), "
           PointListFactory: Sender must be admin");
87     require(newAddress == address(0));
88     emit FactoryDeprecated(_newAddress);
89     newAddress = _newAddress;
90 }
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendations:

The deprecation funcionallity seams to not be in use so the recommendation is to remove the code from the contract if that's the case. If not using the deprecation feature is unintended make sure to use the `newAddress` to validate if the Factory has been deprecated when creating new `PointList` childs. The deprecation functionality should be denying the creation of new contracts.

Remediation Plan:

SOLVED and INFORMED: The deprecate functionality is used to inform on the client side that a new address of the factory is present without breaking the compatibility with the old factory

3.10 (HAL-10) INDEX OUT OF RANGE MISSING CHECKS - INFORMATIONAL

Description:

On TokenVault the withdrawTokens function does not check for out-of-bounds access on the lockToItems mapping. After withdrawing all the funds on the vault and having the balance to 0 the element is removed from the mapping using pop. When the user tries to call the withdrawTokens on a previously used _tokenAddress and _index and “Index out of range” error is thrown as seen on Figure Figure 3.

```
>>> TokenVault[0].getItemAtUserIndex(0, SushiToken[0], accounts[0])
(500000, 1619514367, "0x7f02b78D06A2e767136c83C3f698C8Fd949A4d24", 5)
>>> TokenVault[0].lockedItem(5)
(500000, 1619514367, "0x7f02b78D06A2e767136c83C3f698C8Fd949A4d24", 0)
>>> TokenVault[0].withdrawTokens(SushiToken[0], 0, 5, 500000)
Transaction sent: 0x0152ae50184117db635b598d817cda24cacf77adddac9b9b8b60a840c67da348
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 140
  TokenVault.withdrawTokens confirmed - Block: 142   Gas used: 31856 (0.47%)

<Transaction '0x0152ae50184117db635b598d817cda24cacf77adddac9b9b8b60a840c67da348'>
>>> TokenVault[0].withdrawTokens(SushiToken[0], 0, 5, 500000)
Transaction sent: 0xb96bb018a5551553a56f1682b72bebb0c03673f341a83d0033b2c62e754d8ea4
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 141
  TokenVault.withdrawTokens confirmed (Index out of range) - Block: 143   Gas used: 6721975 (100
.00%)

<Transaction '0xb96bb018a5551553a56f1682b72bebb0c03673f341a83d0033b2c62e754d8ea4'>
>>> TokenVault[0].getItemAtUserIndex(0, SushiToken[0], accounts[0])
  File "<console>", line 1, in <module>
  File "brownie/network/contract.py", line 1712, in __call__
    return self.call(*args, block_identifier=block_identifier)
  File "brownie/network/contract.py", line 1517, in call
    raise VirtualMachineError(e) from None
VirtualMachineError: invalid opcode
```

Figure 3: Out of range error on TokenVault withdrawTokens function

Furthermore, the function named getDocumentName in the Documents contract does not validate the length of the array, causing out-of-bounds access if a bigger index is provided as shown in Listing 25. This could be easily hit since removing a document from the Documents contract causes the last element to take its place shifting and changing the id of the last element as shown in Figure 4.

```
>>> BatchAuction[0].getAllDocuments()
("name0", "name1", "name2", "name3", "name4", "name5")
>>> BatchAuction[0].getDocumentName(0)
'name0'
>>> BatchAuction[0].getDocumentName(1)
'name1'
>>> BatchAuction[0].getDocumentName(2)
'name2'
>>> BatchAuction[0].getDocumentName(3)
'name3'
>>> BatchAuction[0].getDocumentName(4)
'name4'
>>> BatchAuction[0].getDocumentName(5)
'name5'
>>> BatchAuction[0].getDocumentName(6)
  File "<console>", line 1, in <module>
  File "brownie/network/contract.py", line 1712, in __call__
    return self.call(*args, block_identifier=block_identifier)
  File "brownie/network/contract.py", line 1517, in call
    raise VirtualMachineError(e) from None
VirtualMachineError: invalid opcode: Index out of range
>>> BatchAuction[0].removeDocument('name2')
Transaction sent: 0x70cad3ebcb84f0921413b4765e0e166a9d540cf0f5e54a685548d56a4a91c527
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 37
  BatchAuction.removeDocument confirmed - Block: 38   Gas used: 35625 (0.53%)

<Transaction '0x70cad3ebcb84f0921413b4765e0e166a9d540cf0f5e54a685548d56a4a91c527'>
>>> BatchAuction[0].getAllDocuments()
("name0", "name1", "name5", "name3", "name4")
>>> BatchAuction[0].getDocumentName(0)
'name0'
>>> BatchAuction[0].getDocumentName(1)
'name1'
>>> BatchAuction[0].getDocumentName(2)
'name5'
>>> BatchAuction[0].getDocumentName(3)
'name3'
>>> BatchAuction[0].getDocumentName(4)
'name4'
>>> BatchAuction[0].getDocumentName(5)
  File "<console>", line 1, in <module>
  File "brownie/network/contract.py", line 1712, in __call__
    return self.call(*args, block_identifier=block_identifier)
  File "brownie/network/contract.py", line 1517, in call
    raise VirtualMachineError(e) from None
VirtualMachineError: invalid opcode: Index out of range
>>>
```

Figure 4: Shifted Documents ids when removing a middle placed document

Code Location:

Listing 24: TokenVault.sol (Lines 100)

```
 99 require(_amount > 0, 'token amount is Zero');
100 uint256 id = users[msg.sender].lockToItems[_tokenAddress][_index];
101 Item storage userItem = lockedItem[id];
102 require(id == _id && userItem.owner == msg.sender, 'LOCK MISMATCH'
      );
103 require(userItem.unlockTime < block.timestamp, 'Not unlocked yet')
      ;
104 userItem.amount = userItem.amount.sub(_amount);
```

Listing 25: Documents.sol (Lines 97)

```
 92 /**
 93  * @notice Used to retrieve the document name from index in the
       smart contract.
 94  * @return string Name of the document name.
 95  */
 96 function getDocumentName(uint256 _index) external view returns (
      string memory) {
 97     return _docNames[_index];
 98 }
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendations:

It is recommended to perform out of bounds checks when accessing Solidity arrays. Furthermore, reporting an error when fetching out of bounds data will let the user know what caused the execution reversion.

Remediation Plan:

PARTIALLY SOLVED: still missing on withdrawTokens (TokenVault)

3.11 (HAL-11) SLOT0 NOT USED - INFORMATIONAL

Description:

The following contracts are increase the index counter used to access array data before storing the value. This results in the first slot of the array, index `0` not being used. Moreover, Solidity array indices all start from `0` which could produce an off-by-one issue in further development if this unused slot is not kept in mind.

- `TokenVault` inside `lockTokens` is increasing the `depositId` counter before using it, shown in Listing 26. Causing slot `0` on `lockedItem` to not be used as shown in Figure 5.
- `MISOFarmFactory` inside `addFarmTemplate` is increasing the `farmTemplateId` counter before using it, shown in Listing 27. Causing slot `0` on `farmTemplates` to not be used as shown in Figure 6.
- `MISOFermentor` inside `addEscrowTemplate` is increasing the `escrowTemplateId` counter before using it, shown in Listing 28. Causing slot `0` on `escrowTemplates` to not be used as shown in Figure 7.
- `MISOLiquidityLauncher` inside `addLiquidityLauncherTemplate` is increasing the `launcherTemplateId` counter before using it, shown in Listing **??**. Causing slot `0` on `launcherTemplates` to not be used as shown in Figure 8.
- `MISOTokenFactory` inside `addTokenTemplate` is increasing the `tokenTemplateId` counter before using it, shown in Listing 30. Causing slot `0` on `tokenTemplates` to not be used as shown in Figure 9.

```
>>>
>>> TokenVault[0].allDepositIds(0)
1
>>> TokenVault[0].lockedItem(0)
(0, 0, "0x0000000000000000000000000000000000000000", 0)
>>> TokenVault[0].lockedItem(1)
(500000, 1619514367, "0x7f02b78D06A2e767136c83C3f698C8Fd949A4d24", 0)
>>>
```

Figure 5: Unused first slot on TokenVault lockedItem

```
<Transaction '0x1a7e5cc72ec15d92100d3c3abb810baf1ca705723d63d55be19a79d2fc2fa6b8'>
>>> MISOFarmFactory[0].addFarmTemplate(accounts[2])
Transaction sent: 0x728ae46b85a4bea186d246b16c470dc3c13e7f9fc612c629d0a6b499dc1217e7
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 151
  MISOFarmFactory.addFarmTemplate confirmed - Block: 153   Gas used: 90538 (1.35%)

<Transaction '0x728ae46b85a4bea186d246b16c470dc3c13e7f9fc612c629d0a6b499dc1217e7'>
>>> MISOFarmFactory[0].farmTemplateId()
1
>>> MISOFarmFactory[0].getFarmTemplate(1)
'0xdEc2B17dE4DC6ad6D2647B81E1B7BF4A08761353'
>>> MISOFarmFactory[0].getFarmTemplate(0)
'0x0000000000000000000000000000000000000000'
>>>
```

Figure 6: Unused first slot on MISOFarmFactory farmTemplates

```
>>> MISOFermenter[0].addEscrowTemplate(accounts[2])
Transaction sent: 0x01d869394538e300fdb93315b91ec12747a273ba47f64cfbe724a982063842b8
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 161
  MISOFermenter.addEscrowTemplate confirmed - Block: 163   Gas used: 89697 (1.33%)

<Transaction '0x01d869394538e300fdb93315b91ec12747a273ba47f64cfbe724a982063842b8'>
>>> MISOFermenter[0].getEscrowTemplate(0)
'0x0000000000000000000000000000000000000000'
>>> MISOFermenter[0].getEscrowTemplate(1)
'0xdEc2B17dE4DC6ad6D2647B81E1B7BF4A08761353'
>>>
```

Figure 7: Unused first slot on MISOFermentor escrowTemplates

```
>>> MISOLiquidityLauncher[0].addLiquidityLauncherTemplate(accounts[2])
Transaction sent: 0xf06e1a725fa396981fc26e06d8f6746a28afab990086dd69c293ecf8b0639e3b
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 173
  MISOLiquidityLauncher.addLiquidityLauncherTemplate confirmed - Block: 175   Gas used: 90583 (1
.35%)

<Transaction '0xf06e1a725fa396981fc26e06d8f6746a28afab990086dd69c293ecf8b0639e3b'>
>>> MISOLiquidityLauncher[0].getLiquidityLauncherTemplate(0)
'0x0000000000000000000000000000000000000000'
>>> MISOLiquidityLauncher[0].getLiquidityLauncherTemplate(1)
'0xdEc2B17dE4DC6ad6D2647B81E1B7BF4A08761353'
```

Figure 8: Unused first slot on MISOLiquidityLauncher launcherTemplates

```
>>> MISOTokenFactory[0].initMISOTokenFactory(MISOAccessControls[0])
Transaction sent: 0xdb199e6f6f742000311ce1e14683748988ff80b549e642334d8bbaa5685f803c
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 176
  MISOTokenFactory.initMISOTokenFactory confirmed - Block: 178   Gas used: 44541 (0.66%)

<Transaction '0xdb199e6f6f742000311ce1e14683748988ff80b549e642334d8bbaa5685f803c'>
>>> MISOTokenFactory[0].accessControls
<ContractCall 'accessControls()'>
>>> MISOTokenFactory[0].accessControls()
'0x7A7Dba3Ba924425fd6207FF37EFfbFC1Ddb3CEdA'
>>> MISOTokenFactory[0].addTokenTemplate(accounts[0])
Transaction sent: 0xbe35bf08ff40c5005b245bd735dcb988ace2812f75e5f69ed830130cf4115a62
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 177
  MISOTokenFactory.addTokenTemplate confirmed - Block: 179   Gas used: 90517 (1.35%)

<Transaction '0xbe35bf08ff40c5005b245bd735dcb988ace2812f75e5f69ed830130cf4115a62'>
>>> MISOTokenFactory[0].getTokenTemplate(0)
'0x0000000000000000000000000000000000000000'
>>> MISOTokenFactory[0].getTokenTemplate(1)
'0x7f02b78D06A2e767136c83C3f698C8Fd949A4d24'
>>>
```

Figure 9: Unused first slot on MISOTokenFactory tokenTemplates

Code Location:

Listing 26: TokenVault.sol (Lines 65)

```
65      _id = ++depositId;
66
67      lockedItem[_id].amount = _amount;
68      lockedItem[_id].unlockTime = _unlockTime;
69      lockedItem[_id].owner = _withdrawer;
```

Listing 27: MISOFarmFactory.sol (Lines 200)

```
194     function addFarmTemplate(address _template) external {
195         require(
196             accessControls.hasOperatorRole(msg.sender),
197             "MISOFarmFactory: Sender must be operator"
198         );
199         require(farmTemplateToId[_template] == 0);
200         farmTemplateId++;
201         farmTemplates[farmTemplateId] = _template;
202         farmTemplateToId[_template] = farmTemplateId;
203         emit FarmTemplateAdded(_template, farmTemplateId);
204     }
```

Listing 28: MISOFermentor.sol (Lines 98)

```
93     function addEscrowTemplate(address _escrowTemplate) external {
94          require(
95             accessControls.hasOperatorRole(msg.sender),
96             "MISOFermenter: Sender must be operator"
97         );
98         escrowTemplateId++;
99         escrowTemplates[escrowTemplateId] = _escrowTemplate;
100        escrowTemplateToId[_escrowTemplate] = escrowTemplateId;
101        emit EscrowTemplateAdded(_escrowTemplate, escrowTemplateId
               );
102    }
```

Listing 29: MISOLiquidityLauncher.sol (Lines 100)

```
93 function addLiquidityLauncherTemplate(address _template) external
       {
94     require(
95         accessControls.hasOperatorRole(msg.sender),
96         "MISOLiquidityLauncher: Sender must be operator"
97     );
98     require(liquidityTemplateToId[_template] == 0);
99     // GP: Check exisiting / duplicates
100    launcherTemplateId++;
101    launcherTemplates[launcherTemplateId] = _template;
102    liquidityTemplateToId[_template] = launcherTemplateId;
103
104    emit LiquidityTemplateAdded(_template, launcherTemplateId);
105 }
```

Listing 30: MISOTokenFactory.sol (Lines 200)

```
194 function addTokenTemplate(address _template) external {
195     require(
196         accessControls.hasOperatorRole(msg.sender),
197         "MISOTokenFactory: Sender must be operator"
198     );
199     require(tokenTemplateToId[_template] == 0);
200     tokenTemplateId++;
201     tokenTemplates[tokenTemplateId] = _template;
202     tokenTemplateToId[_template] = tokenTemplateId;
203     emit TokenTemplateAdded(_template, tokenTemplateId);
```

```
204 }
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendations:

It is recommended to always use the full storage when dealing with decentralized technology. If having the first index as 1 is an intended feature further development on the contracts should keep always in mind this to protect against off-by-one issues. Consistency on the code is a key priority when dealing with custom index values.

Remediation Plan:

The SushiSwap team accepts the risk.

3.12 (HAL-12) DISORDERED MATH OPERATIONS - INFORMATIONAL

Description:

```
>>>
>>> PointList[0].setPoints([accounts[0]],[0xfffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffe])
Transaction sent: 0xc7fcca36734d466104a74180e99c3e1b0726be1e79e82e886b10138ea6f576a0
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 25
  PointList.setPoints confirmed - Block: 26   Gas used: 25791 (0.38%)

<Transaction '0xc7fcca36734d466104a74180e99c3e1b0726be1e79e82e886b10138ea6f576a0'>
>>> hex(PointList[0].totalPoints())
'0xfffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffe'
>>> PointList[0].setPoints([accounts[0]],[0x1234])
Transaction sent: 0xcaf207fca053a12e2c97839e06678e2d8e4ac339b9ea1f8e96b4431ddd94db5e
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 26
  PointList.setPoints confirmed (SafeMath: addition overflow) - Block: 27   Gas used: 31436 (0.47%)

<Transaction '0xcaf207fca053a12e2c97839e06678e2d8e4ac339b9ea1f8e96b4431ddd94db5e'>
>>> hex(PointList[0].totalPoints())
'0xfffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffe'
>>>
```

Figure 10: Overflow produced when reaching the end of the uint256 integer due to the math operation order

Code Location:

On the `setPoints` function in the `PointList` contract when setting the `totalPoints` internal variable the math operation should be reverted as shown on Listing 31. Reverting the operations would allow bigger numbers to be added. As an example lets take the following expected scenario:

- The value of `totalPoints` is `0xfffff....fe`, only a user has that amount of points.
- The user wants to set the new `points` to a lower value, for example `0x1234`.
- The `totalPoints`, since only one user is contributing, should be `0x1234` at the end.

Since the `add` operation is done first, when the addition of `totalPoints` and `amount` reaches the upper limit the operation will overflow causing the `totalPoints` to not be updated.

Listing 31: PointList.sol (Lines 80)

```
78      if (amount != previousPoints) {
79          points[account] = amount;
80          totalPoints = totalPoints.add(amount).sub(previousPoints);
81          emit PointsUpdated(account, previousPoints, amount);
82      }
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendations:

Having the sub and add operations reverted would solve the previously exposed scenario and allow more room for upper limit large numbers. No underflow should be present since the previousPoints is already contemplated in the totalPoints value. This means that the minimum value that the totalPoints.sub(previousPoints) could have is 0 if only one user has points.

Remediation Plan:

SOLVED: Math operations were put in the correct order.

3.13 (HAL-13) INTEGER TRUNCATION - INFORMATIONAL

Description:

The commitmentsTotal struct field is defined as an uint128. However the value is first casted to a uint256 before performing a math add operation and later casted back to a uint128. This casting is performed on the following contracts and places:

- BatchAuction (Listing 32)
- DutchAuction (Listing 33)
- HyperbolicAuction (Listing 34)

Furthermore, the issue presented on “MISMATCHED STRUCTS AND INTERFACES” does lead to an integer truncation on the lastModified field as well.

Code Location:

Listing 32: BatchAuction.sol (Lines 209)

```
201 function _addCommitment(address _addr, uint256 _commitment)
        internal {
202     require(block.timestamp >= marketInfo.startTime && block.
            timestamp <= marketInfo.endTime, "BatchAuction: outside
            auction hours");
203
204     uint256 newCommitment = commitments[_addr].add(_commitment);
205     if (marketStatus.usePointList) {
206         require(IPointList(pointList).hasPoints(_addr,
                newCommitment));
207     }
208     commitments[_addr] = newCommitment;
209     marketStatus.commitmentsTotal = uint128(uint256(marketStatus.
            commitmentsTotal).add(_commitment));
210     emit AddedCommitment(_addr, _commitment);
211 }
```

Listing 33: DutchAuction.sol (Lines 391)

```
381 function _addCommitment(address _addr, uint256 _commitment)
        internal {
382     require(block.timestamp >= uint256(marketInfo.startTime) &&
            block.timestamp <= uint256(marketInfo.endTime), "
            DutchAuction: outside auction hours");
383     MarketStatus storage status = marketStatus;
384
385     uint256 newCommitment = commitments[_addr].add(_commitment);
386     if (status.usePointList) {
387         require(IPointList(pointList).hasPoints(_addr,
                newCommitment));
388     }
389
390     commitments[_addr] = newCommitment;
391     status.commitmentsTotal = uint128(uint256(status.
            commitmentsTotal).add(_commitment));
392     emit AddedCommitment(_addr, _commitment);
393 }
```

Listing 34: DutchAuction.sol (Lines 344)

```
333 function _addCommitment(address _addr, uint256 _commitment)
        internal {
334     require(block.timestamp >= uint256(marketInfo.startTime) &&
            block.timestamp <= uint256(marketInfo.endTime), "
            HyperbolicAuction: outside auction hours");
335     MarketStatus storage status = marketStatus;
336     require(!status.finalized, "HyperbolicAuction: auction already
             finalized");
337
338     uint256 newCommitment = commitments[_addr].add(_commitment);
339     if (status.usePointList) {
340         require(IPointList(pointList).hasPoints(_addr,
                newCommitment));
341     }
342
343     commitments[_addr] = newCommitment;
344     status.commitmentsTotal = uint128(uint256(status.
            commitmentsTotal).add(_commitment));
345     emit AddedCommitment(_addr, _commitment);
346 }
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendations:

It is recommended to keep consistency with the variable types that are used. If the variable is added against other `uint256` variables the data type should be changed to the biggest data type in order to avoid integer truncation. When casting data type variables integer truncation should be kept in mind.

Remediation Plan:

The SushiSwap team accepts the risk.

3.14 (HAL-14) MISMATCHED STRUCTS AND INTERFACES - INFORMATIONAL

Description:

On the MISOHelper file inside the DocumentHepler contract, the internal declared Document struct mismatches the original struct declared under Documents as shown in Listing 35 and Listing 36. Furthermore, the getDocument interface (see Listing 37) is expected to return 3 values on MISOHelper but only returning 2 values on getDocument from the Documents contract, as shown in Listing 38 and Listing 39. This issue will cause a revert to alway happen if the amount of returned values from the functions do not match.

Code Location:

Listing 35: MISOHelper.sol (Lines 40)

```
38     struct Document {
39         bytes32 docHash;
40         uint256 lastModified;
41         string uri;
42     }
```

Listing 36: Documents.sol (Lines 12)

```
10     struct Document {
11         uint32 docIndex;    // Store the document name indexes
12         uint64 lastModified; // Timestamp at which document
               details was last modified
13         string data; // data of the document that exist off-chain
14     }
```

Listing 37: MISOHelper.sol (Lines 33)

```
32 interface IDocument {
33     function getDocument(bytes32 _name) external view returns (
           string memory, bytes32, uint256);
```

```
34     function getAllDocuments() external view returns (bytes32[]
           memory);
35 }
```

Listing 38: MISOHelper.sol (Lines 56)

```
49     Document[] memory documents = new Document[](documentNames.
           length);
50
51     for(uint256 i = 0; i < documentNames.length; i++) {
52         (
53             documents[i].uri,
54             documents[i].docHash,
55             documents[i].lastModified
56         ) = document.getDocument(documentNames[i]);
57     }
```

Listing 39: Documents.sol (Lines 69)

```
69 function getDocument(string calldata _name) external view returns
       (string memory, uint256) {
70     return (
71         _documents[_name].data,
72         uint256(_documents[_name].lastModified)
73     );
74 }
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendations:

It is recommended to keep consistency on the data types and declarations of the same struct on multiple files. If the same struct is needed consider declaring it on a separated file and importing it on the used contracts as an abstract contract or by tuple unpacking. Furthermore,

when declaring interfaces the expected returned values should always be checked with the original implementation, this can avoid issues on returned values being swapped and indirect truncation.

Remediation Plan:

PARTIALLY SOLVED: still in `lastModified` uint256 mismatchs with uint64.

3.15 (HAL-15) MISSING CHECKS - INFORMATIONAL

Description:

The PoolLiquidity02 contract does not validate the deposit deadline on the depositETH, depositToken1, depositToken2 functions. Those functions all delegate the responsability to the internal _deposit function. However, none of the stated functions are performing any checks on the deposit deadline variable allowing anyone to deposit past that time.

Code Location:

Listing 40: PoolLiquidity02.sol (Lines 187)

```
186 function _deposit(uint amount, address token) private returns (
        bool success) {
187     // CC: shouldn't be require(block.timestamp < deadline, if
            deadline is deposit deadline?
188     require(block.timestamp < expiry, "PoolLiquidity02: Contract
            has expired");
189     require(amount > 0, "PoolLiquidity02: Token amount must be
            greater than 0");
190     require(liquidityAdded == 0, "PoolLiquidity02: Liquidity
            already added");
191     _safeTransferFrom(token, msg.sender, amount);
192 }
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendations:

If is recommended to always use all declared variables for their purpose. If the intended functionality of the `deadline` variable differs from the one stated on the comments it should be properly renamed to not cause confusion. If the `deadline` variable checks are missing they should be added on the independent `deposit*` functions or the delegated `_deposit` function.

Remediation Plan:

SOLVED: `PoolLiquidity02.sol` contract was removed.

MANUAL TESTING

During the manual testing multiple questions where considered while evaluation each of the defined functions:

- Can it be re-called changing admin/roles and permissions?
- Can somehow an external controlled contract call again the function during the execution of it? (Re-entrancy)
- Can it be called twice in the same block and cause issues?
- Do we control sensitive or vulnerable parameters?
- Does the function check for boundaries on the parameters and internal values? Bigger than zero or equal? Argument count, array sizes, integer truncation . . .
- Are the function parameters and variables controlled by external contracts?
- Can extended contracts cause issues on the extender contract?

First of all the source code was reviewed for uncoded features and missing code, for example the Listing 41 shows all the TODO comments found.

Listing 41

```
tests/test_recipe_02.py
128:    pool_id = 1 # TODO poolLength is 2

contracts/Tokens/FixedToken.sol
15:    // TODO: remove owner

contracts/Recipes/MISORecipe02.vy
130:    paymentCurrency: address, # TODO: hardcode payment
    currency to ETH?

tests/test_crowdsale.py
318:    # TODO

tests/test_pool_liquidity_02.py
217:# TODO - uncomment
```

After initial recon, some understanding on the different components of the platform was performed. Some external dependencies on OpenZeppelin version 3.2.0 where found:

Listing 42: brownie-config.yaml

```
dependencies:
    - OpenZeppelin/openzeppelin-contracts@3.2.0
```

Checking the changelog for OpenZeppelin revealed the following:

> If you're using our implementation of ERC777 from version 3.3.0 or earlier, and you define a custom _beforeTokenTransfer function that writes to a storage variable, you may be vulnerable to a reentrancy attack.

However, looking for custom _beforeTokenTransfer implementations revealed that none were used:

Listing 43

```
contracts/Tokens/MintableToken.sol
96:    function _beforeTokenTransfer(address from, address to,
    uint256 amount) internal virtual override(ERC20, ERC20Pausable)
     {
97:        super._beforeTokenTransfer(from, to, amount);

contracts/Tokens/ERC20.sol
260:        _beforeTokenTransfer(sender, recipient, amount);
279:        _beforeTokenTransfer(address(0), account, amount);
300:        _beforeTokenTransfer(account, address(0), amount);
353:    function _beforeTokenTransfer(address from, address to,
    uint256 amount) internal virtual { }

contracts/Tokens/ERC20/ERC20Pausable.sol
17:     * @dev See {ERC20-_beforeTokenTransfer}.
23:    function _beforeTokenTransfer(address from, address to,
    uint256 amount) internal virtual override {
24:        super._beforeTokenTransfer(from, to, amount);
```

4.1 ERC20 Token

The _initERC20 function is protected using a locker, not allowing to call initToken again.

The EIP-2612 permit function was added as well (More information on ### EIP-2612: permit -- 712-signed approvals). It was mostly taken from Makerdao https://github.com/makerdao/dss/blob/4433006599962183 4b08de1edf3b962f6bbd74c6/src/dai.sol#L118-L140

Delegate By Signature:

The ERC20 allows delegating votes by signature as described in EIP-712.Delegate votes from the signatory to the delegatee. This method has the same purpose as Delegate but it instead enables offline signatures to participate in Compound governance vote delegation. For more details on how to create an offline signature.

> A key benefit to users of by-signature functionality is that they can create a signed delegate or vote transaction for free, and have a trusted third-party spend ETH on gas fees and write it to the blockchain for them. In this guide, we will focus on code examples around this type of functionality.

Listing 44

```
1 function delegateBySig(address delegatee, uint nonce, uint expiry,
      uint8 v, bytes32 r, bytes32 s)
```

- delegatee: The address in which the sender wishes to delegate their votes to.
- nonce: The contract state required to match the signature. This can be retrieved from the contract's public nonces mapping.
- expiry: The time at which to expire the signature. A block timestamp as seconds since the unix epoch (uint).

- v: The recovery byte of the signature.
- r: Half of the ECDSA signature pair.
- s: Half of the ECDSA signature pair.
- RETURN: No return, reverts on error.

A pseudo-code of the implementation looks like the following:

Listing 45

```
sig = secp256k1.sign(msgHash, privateKey)

r = sig.signature.slice(0, 32)
s = sig.signature.slice(32, 64)
v = sig.recovery + 27
```

4.2 SushiToken

Initially a race condition on `_writeCheckpoint` was considered, see Listing 47 for reference. The idea was that a miner could potentially cause the order of the transactions to be as desired forcing the least `newVotes` to be set last. This would allow a malicious miner to manipulate the voting power of any address if multiple votes are set on the same block. The idea was then refuted because of the following statement of the official ETH implementation documentation:

> The protocol requires that transactions from a single account are executed in nonce order. It's still possible for an account to have many pending transactions, and as long as the nonces are contiguous, a miner can include as many of them as desired in a single block, as long as they're executed in order.

Listing 46: SushiToken.sol (Lines 279)

```
278 if (nCheckpoints > 0 && checkpoints[delegatee][nCheckpoints - 1].
        fromBlock == blockNumber) {
279     checkpoints[delegatee][nCheckpoints - 1].votes = newVotes;
```

```
280 } else {
281     checkpoints[delegatee][nCheckpoints] = Checkpoint(blockNumber,
          newVotes);
282     numCheckpoints[delegatee] = nCheckpoints + 1;
283 }
```

The SushiToken contract is checking the variable nonce after its being used. This is causing unnecessary gas usage if the variable checks are not lately met. Moving the checks before the usage would reduce gas cost.

Listing 47: SushiToken.sol (Lines 155,170)

```
151 bytes32 structHash = keccak256(
152     abi.encode(
153         DELEGATION_TYPEHASH,
154         delegatee,
155         nonce,
156         expiry
157     )
158 );
159
160 bytes32 digest = keccak256(
161     abi.encodePacked(
162         "\x19\x01",
163         domainSeparator,
164         structHash
165     )
166 );
167
168 address signatory = ecrecover(digest, v, r, s);
169 require(signatory != address(0), "SUSHI::delegateBySig: invalid
      signature");
170 require(nonce == sigNonces[signatory]++, "SUSHI::delegateBySig:
      invalid nonce");
171 require(now <= expiry, "SUSHI::delegateBySig: signature expired");
172 return _delegate(signatory, delegatee);
```

4.3 Uniswap

The uniswap contracts are used to provide liquidity to/from the Pool. The original contracts taken from https://github.com/Uniswap/uniswap-v2-core/tree/27f6354bae6685612c182c3bc7577e61bc8717e3/contracts are modified in the following way:

- Change contract version to 0.6.12 and do the necessary patching.
- Add migrator member in UniswapV2Factory which can be set by feeToSetter.
- Allow migrator to specify the amount of liquidity during the first mint. Disallow first mint if migrator is set.

The diff files can be seen in Figure 11 / Figure 12 for the Uniswapv2Factory and Figure 13 / Figure 14 for the Uniswapv2Pair contract.

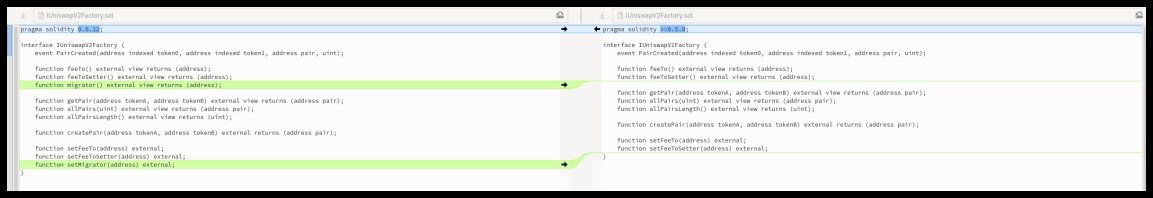

Figure 11: Diff from the original code on Uniswapv2Factory

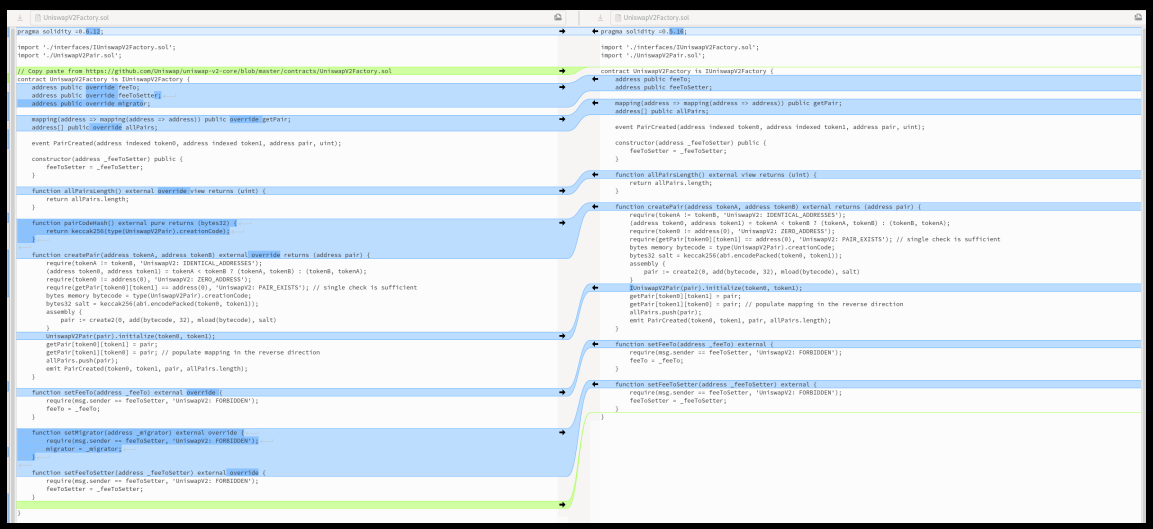

Figure 12: Diff from the original code on Uniswapv2Factory

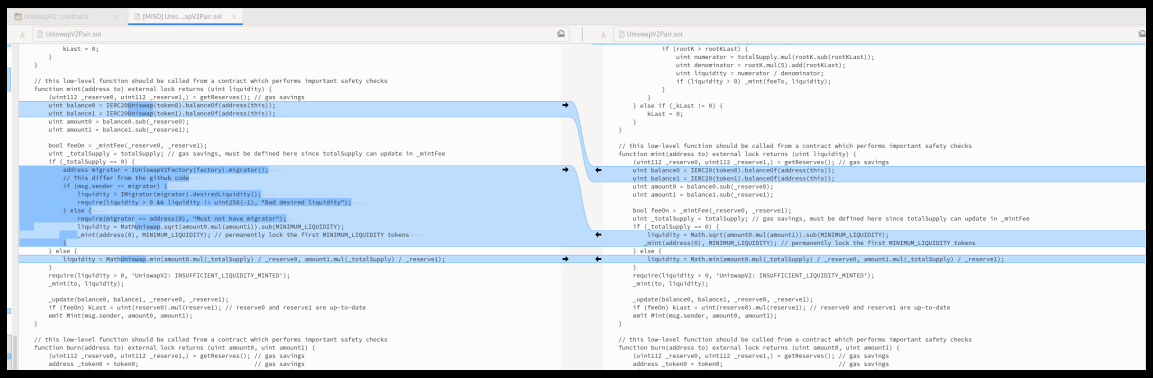

Figure 13: Diff from the original code on Uniswapv2Pair

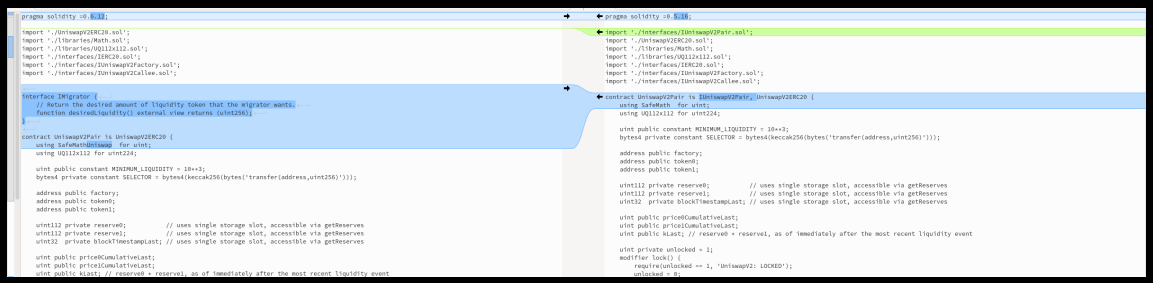

Figure 14: Diff from the original code on Uniswapv2Pair

4.4 BoringBatchable

The BoringBatchable uses delegatecall which allows calling a different contract but keeping the context and variables on the same slots. There exists several security concerns about using delegate call as seen on https://blog.sigmaprime.io/solidity-security.html#delegatecall. However, it was noticed that the usage is safe, since no external calls are performed and no external contracts are called as seen on Listing 48.

Listing 48: BoringBatchable.sol (Lines 39)

```
35 function batch(bytes[] calldata calls, bool revertOnFail) external
       payable returns (bool[] memory successes, bytes[] memory
      results) {
36     successes = new bool[](calls.length);
37     results = new bytes[](calls.length);
38     for (uint256 i = 0; i < calls.length; i++) {
```

```
39          (bool success, bytes memory result) = address(this).
                delegatecall(calls[i]);
40          require(success || !revertOnFail, _getRevertMsg(result));
41          successes[i] = success;
42          results[i] = result;
43      }
44  }
```

> Delegatecall is only used on the same contract, so it's safe. The BoringBatchable contract is used as an inherited contract in different places and never used alone. It is not possible to call internal methods since they are not declared in the function hash table and delegatecall does not have access to it.

4.5 MISOAccessFactory

All the methods of the contract were logically tested. Deploying a child AccessFactory was tested as seen on Figure 15. The created child contracts were casted to the interfaces and tested accordingly.

```
>>>
>>> MISOAccessFactory[1].deployAccessControl(accounts[0])
Transaction sent: 0x38bb9468113fb08bc3c30c9f635e0947052baa8cf4c5697f88bccba3533a0035
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 198
  MISOAccessFactory.deployAccessControl confirmed - Block: 200   Gas used: 205054 (3.05%)

<Transaction '0x38bb9468113fb08bc3c30c9f635e0947052baa8cf4c5697f88bccba3533a0035'>
>>> MISOAccessFactory[1].children(1)
'0x93bee4514cC329Ec8396Ad4319b343bda7D8aC37'
>>> MISOAccessFactory[1].children(0)
'0x0588A8ae333c89dC943b05D002D7914D04ADBA65'
>>>
```

Figure 15: Deployment of child AccessControls from the MISOAccessFactory

4.6 BatchAuction

There is no lock to the initAuction function. However the transaction will revert once the lock on initAccessControls is hit. That's causing

unecessary gas usage when calling the initAuction method a second time just to have the transaction revert. Some test were performed:

- Making sure that the points are correctly checked and cannot be bypassed, as shown in Figure 16.
- Making sure that committing to a finalized Auction or recalling the finalize method is forbidden, as shown Figure 17.

```
<Transaction '0xc4d9fc5c5d5b2060acb5761eb0827b3bc2080e37516c8a3aef006a3ea2d55c1f'>
>>> PointList[0].points(accounts[0])
247
>>> BatchAuction[0].commitments(accounts[0])
246
>>> BatchAuction[0].commitTokens(123, True)
Transaction sent: 0x2fa22391bc05c773a98db249f63bd1ed9d270b35d9f745a87c20eed2ab43b953
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 55
  BatchAuction.commitTokens confirmed (reverted) - Block: 56   Gas used: 61492 (0.91%)

<Transaction '0x2fa22391bc05c773a98db249f63bd1ed9d270b35d9f745a87c20eed2ab43b953'>
>>>
```

Figure 16: Points are correctly checked on BatchAuction commitment

```
280
>>> BatchAuction[0].finalize()
Transaction sent: 0xd95a7064a9467f287ca5e4ddd3ca8a2accedbc36b7d72f7616c1e633db906aef
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 63
  BatchAuction.finalize confirmed (BatchAuction: Auction has already finalized) - Block: 64   Gas
 used: 29344 (0.44%)

<Transaction '0xd95a7064a9467f287ca5e4ddd3ca8a2accedbc36b7d72f7616c1e633db906aef'>
>>>
```

Figure 17: The BatchAuction is correctly checking for the finalize state and denies commitment/re-finalization

4.7 TokenVault

All the contract functions were manually tested for logic error. The following tests were performed:

- Locking funds should increase the contract balance and decrease the balance of the account, shown in Figure 18.
- Withdrawal should be kept locked until the _unlockTime time is reached.

- Once the _unlockTime is reached the maximum allowed withdraw should be the user amount, this can be seen in Figure 19 and Figure 20.

During the test some other issues were found on the TokenVault contract, such as:

- Unused slot0 on lockedItem, explained on “SLOT0 NOT USED”.
- Out-of-bounds missing checks, explained on “INDEX OUT OF RANGE MISSING CHECKS”.

```
>>> TokenVault[0].lockTokens(SushiToken[0], 500000, 1619514367, accounts[0])
Transaction sent: 0x6971dbc1338c46d0d8cbf1251e5f9c75fc67e6595081e1a1da4a2af5ba74a18e
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 62
  TokenVault.lockTokens confirmed - Block: 64   Gas used: 262662 (3.91%)

<Transaction '0x6971dbc1338c46d0d8cbf1251e5f9c75fc67e6595081e1a1da4a2af5ba74a18e'>
>>> TokenVault[0]
<TokenVault Contract '0x405C18ceA5ECe124D9A0C016bE0eaab86AACc2eF'>
>>> SushiToken[0].balanceOf(accounts[0])
0
>>> SushiToken[0].balanceOf(TokenVault[0])
500000
>>>
```

Figure 18: Initialization of a Token vault an locking funds

```
>>>
>>> SushiToken[0].balanceOf(TokenVault[0])
1480000
>>> SushiToken[0].balanceOf(accounts[0])
199020000
>>> TokenVault[0].getUserLockedItemAtIndex(accounts[0],0)
'0x01E598CBc1c3759F83B22BCACDfd12d16BC799a9'
>>> TokenVault[0].getLockedItemAtId(1)
(480000, 1619514367, "0x7f02b78D06A2e767136c83C3f698C8Fd949A4d24", 0)
>>> TokenVault[0].withdrawTokens(SushiToken[0], 0, 1, 20000)
Transaction sent: 0x6086553172f357384180bf4a617bfb3705ba3df16834906d32149c9e03f6a548
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 74
  TokenVault.withdrawTokens confirmed - Block: 76   Gas used: 49250 (0.73%)

<Transaction '0x6086553172f357384180bf4a617bfb3705ba3df16834906d32149c9e03f6a548'>
>>> SushiToken[0].balanceOf(accounts[0])
199040000
>>> SushiToken[0].balanceOf(TokenVault[0])
1460000
>>>
```

Figure 19: Successful withdrawal of locked funds when the unlockTime is reached

```
>>>
>>> TokenVault[0].getItemAtUserIndex(0, SushiToken[0], accounts[0])
(500000, 1619514367, "0x7f02b78D06A2e767136c83C3f698C8Fd949A4d24", 5)
>>> TokenVault[0].lockedItem(5)
(500000, 1619514367, "0x7f02b78D06A2e767136c83C3f698C8Fd949A4d24", 0)
>>> TokenVault[0].withdrawTokens(SushiToken[0], 0, 5, 500000)
Transaction sent: 0x0152ae50184117db635b598d817cda24cacf77adddac9b9b8b60a840c67da348
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 140
  TokenVault.withdrawTokens confirmed - Block: 142   Gas used: 31856 (0.47%)

<Transaction '0x0152ae50184117db635b598d817cda24cacf77adddac9b9b8b60a840c67da348'>
>>>
```

Figure 20: Successful withdrawal until the balance is 0 of the locked funds

4.8 MISOFermentor

All the contract functions were manually tested for logic error. The following tests were performed:

- Creating an escrow template and adding it to the Factory, seen in Figure 21.
- Removing templates and checking for out-of-bounds access and re-deletion, seen in Figure 22.

During the test some other issues were found on the MISOFermentor contract, such as:

- Unused slot0 on lockedItem, explained on “SLOT0 NOT USED” .

```
>>> MISOFermenter[0].createEscrow(2)
Transaction sent: 0xe000cefc8e448efcf9334fb404686e49e59426de07d76907efe0137e49522b8b
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 169
  MISOFermenter.createEscrow confirmed - Block: 171   Gas used: 136445 (2.03%)

<Transaction '0xe000cefc8e448efcf9334fb404686e49e59426de07d76907efe0137e49522b8b'>
>>> MISOFermenter[0].escrows(1)
'0xcBf1FD5face55875f71d5A2D4d8606b32498d547'
>>>
```

Figure 21: Creating escrow templates on MISOFermentor

```
>>> MISOFermenter[0].getTemplateId('0xdEc2B17dE4DC6ad6D2647B81E1B7BF4A08761353')
1
>>> MISOFermenter[0].removeEscrowTemplate(1)
Transaction sent: 0x8552b7f1144a2f59b22074ad48c4a03b9f69e99695826ca9fe389b2faf53f244
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 162
  MISOFermenter.removeEscrowTemplate confirmed - Block: 164   Gas used: 17341 (0.26%)

<Transaction '0x8552b7f1144a2f59b22074ad48c4a03b9f69e99695826ca9fe389b2faf53f244'>
>>> MISOFermenter[0].getTemplateId('0xdEc2B17dE4DC6ad6D2647B81E1B7BF4A08761353')
0
>>> MISOFermenter[0].getTemplateId('0xdEc2B17dE4DC6ad6D2647B81E1B7BF4A08761356')
0
>>> MISOFermenter[0].removeEscrowTemplate(1)
Transaction sent: 0x67f1f3f4ad64b16f16eb803ce3588217a03428c40870dde20ff61b085c36f8c5
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 163
  MISOFermenter.removeEscrowTemplate confirmed (reverted) - Block: 165   Gas used: 22332 (0.33%)

<Transaction '0x67f1f3f4ad64b16f16eb803ce3588217a03428c40870dde20ff61b085c36f8c5'>
>>> MISOFermenter[0].getEscrowTemplate(1)
'0x0000000000000000000000000000000000000000'
>>> MISOFermenter[0].getEscrowTemplate(0)
'0x0000000000000000000000000000000000000000'
>>>
```

Figure 22: Removing an escrow templates from MISOFermentor and checking re-deletion

4.9 MISOTokenFactory

All the contract functions were manually tested for logic error. The following tests were performed:

- Adding a template to the MISOTokenFactory, show in Figure 23
- Removing a template from the MISOTokenFactory and trying to access it later, show in Figure 24.
- Creation of a child Token using InitData, and checking functionality, show in Figure 25 and Figure 26.

During the test some other issues were found on the MISOTokenFactory contract, such as:

- Unused slot0 on lockedItem, explained on “SLOT0 NOT USED”.

```
>>> MISOTokenFactory[0].initMISOTokenFactory(MISOAccessControls[0])
Transaction sent: 0xdb199e6f6f742000311ce1e14683748988ff80b549e642334d8bbaa5685f803c
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 176
  MISOTokenFactory.initMISOTokenFactory confirmed - Block: 178   Gas used: 44541 (0.66%)

<Transaction '0xdb199e6f6f742000311ce1e14683748988ff80b549e642334d8bbaa5685f803c'>
>>> MISOTokenFactory[0].accessControls
<ContractCall 'accessControls()'>
>>> MISOTokenFactory[0].accessControls()
'0x7A7Dba3Ba924425fd6207FF37EFfbFC1Ddb3CEdA'
>>> MISOTokenFactory[0].addTokenTemplate(accounts[0])
Transaction sent: 0xbe35bf08ff40c5005b245bd735dcb988ace2812f75e5f69ed830130cf4115a62
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 177
  MISOTokenFactory.addTokenTemplate confirmed - Block: 179   Gas used: 90517 (1.35%)

<Transaction '0xbe35bf08ff40c5005b245bd735dcb988ace2812f75e5f69ed830130cf4115a62'>
>>> MISOTokenFactory[0].getTokenTemplate(0)
'0x0000000000000000000000000000000000000000'
>>> MISOTokenFactory[0].getTokenTemplate(1)
'0x7f02b78D06A2e767136c83C3f698C8Fd949A4d24'
>>>
```

Figure 23: Creating of a MISOTokenFactory and addition of a template to it

```
>>> MISOTokenFactory[0].removeTokenTemplate(1)
Transaction sent: 0xdfda9b004ecf6d97971f56f6250a9a6f072a54af068187ba49fe6ba6bdc9eab9
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 186
  MISOTokenFactory.removeTokenTemplate confirmed - Block: 188   Gas used: 19317 (0.29%)

<Transaction '0xdfda9b004ecf6d97971f56f6250a9a6f072a54af068187ba49fe6ba6bdc9eab9'>
>>> MISOTokenFactory[0].removeTokenTemplate(0)
Transaction sent: 0x79b0a8c6eb524835645d9db11436238e617268faba1352b7bed678741ac67d70
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 187
  MISOTokenFactory.removeTokenTemplate confirmed (reverted) - Block: 189   Gas used: 26281 (0.39
%)

<Transaction '0x79b0a8c6eb524835645d9db11436238e617268faba1352b7bed678741ac67d70'>
>>> MISOFermenter[0].getTemplateId('0xdEc2B17dE4DC6ad6D2647B81E1B7BF4A08761356')
0
>>> MISOTokenFactory[0].getTokenTemplate(3)
'0x0000000000000000000000000000000000000000'
>>> MISOTokenFactory[0].getTokenTemplate(0)
'0x0000000000000000000000000000000000000000'
>>> MISOTokenFactory[0].getTokenTemplate(1)
'0x0000000000000000000000000000000000000000'
>>>
```

Figure 24: Deletion of a template and re-deletion test

```
>>> MISOTokenFactory[0].createToken(2, accounts[2], SushiToken[0].getInitData('Name', 'NAME', ac
counts[0], 100000))
Transaction sent: 0x274d8214f181724d22c5c9ea13f4bbf05cb1a7ec68c635d30b048f383b92506d
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 184
  MISOTokenFactory.createToken confirmed - Block: 186   Gas used: 434593 (6.47%)

<Transaction '0x274d8214f181724d22c5c9ea13f4bbf05cb1a7ec68c635d30b048f383b92506d'>
>>> SushiToken[0].getInitData('Name', 'NAME', accounts[0], 100000)
0x0000000000000000000000000000000000000000000000000000000000000080000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000c0000000000000000000000007f02b78d06a2e767136c83c3f698c8fd949a4d
2400000000000000000000000000000000000000000000000000000000000000186a0000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000044e616d650000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000044e414d450000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000
>>>
```

Figure 25: InitData used to create a Token on the MISOTokenFactory

```
>>> MISOTokenFactory[0].tokens(0)
'0x2017644993D6C35b4CA64A726b2927C2108A3bfE'
>>> SushiToken.at(MISOTokenFactory[0].tokens(0)).name()
'Name'
>>>
```

Figure 26: Checking functionality on the created Token child

4.10 Tokens

The code is successfully checking for the MINT_ROLE before any mint operation is performed as shown in Figure 27.

```
contracts/Recipes/MISORecipe03.vy
37:    def mint(owner: address, amount: uint256) : nonpayable
38-    def approve(spender: address, amount: uint256) -> bool: nonpayable

contracts/Recipes/MISORecipe02.vy
13:    def mint(owner: address, amount: uint256) : nonpayable
14-    def approve(spender: address, amount: uint256) -> bool: nonpayable

contracts/Tokens/SushiToken.sol
61:    function mint(address _to, uint256 _amount) public  {
62-        require(hasRole(MINTER_ROLE, _msgSender()), "SushiToken: must have minter role to mint");

contracts/Tokens/MintableToken.sol
63:    function mint(address to, uint256 amount) public virtual {
64-        require(hasRole(MINTER_ROLE, _msgSender()), "MintableToken: must have minter role to mint");

interfaces/ISushiToken.sol
5:    function mint(address owner, uint256 amount) external;
6-    function name() external view returns (string memory);
```

Figure 27: All mint functions are validating the MINT_ROLE before they take action

4.11 Call value checks

During manual testing two controlled .call{value:value} where found on Timelock (executeTransaction) and SafeTransfer (safeTransferETH). Usually those type of calls can lead to re-entrancy issues if they are not correctly manipulated. On both of the cases, the call is the last statement executed during the function flow, the only time safeTransferETH is called is inside LiquidityZAP (removeLiquidityETHOnly), as shown in Listing 49. Furthermore, all funds and tokens are correctly decreased before that call happens not allowing a re-entrancy

to maliciously alter user funds.

Listing 49: LiquidityZAP.sol (Lines 192)

```
178 function removeLiquidityETHOnly(address payable to, uint256
        liquidity) public returns (uint amountOut){
179     require(to != address(0), "LiquidityZAP: Invalid address");
180     (uint amountToken, uint amountETH) = removeLiquidity( _token,
            address(_WETH),liquidity,address(this));
181
182     (uint256 reserveWeth, uint256 reserveTokens) = getPairReserves
            ();
183     uint256 outETH = UniswapV2Library.getAmountOut(amountToken,
            reserveTokens, reserveWeth);
184
185     _safeTransfer(_token, _tokenWETHPair, amountToken);
186
187     (address token0, address token1) = UniswapV2Library.sortTokens
            (address(_WETH), _token);
188     IUniswapV2Pair(_tokenWETHPair).swap(address(_WETH) == token0 ?
             outETH : 0, address(_WETH) == token1 ? outETH : 0, address
            (this), "");
189
190     amountOut = IERC20(address(_WETH)).balanceOf(address(this));
191     _WETH.withdraw(amountOut);
192     _safeTransferETH(to, amountOut);
193 }
```

AUTOMATED TESTING

5.1 SYMBOLIC EXECUTION SECURITY ASSESSMENT

Description:

The tool used to perform the symbolic execution for analyzing Smart Contract is Manticore. In general, obtaining all possible states of a Smart Contract by symbolic execution is the main goal for using this tool. Briefly, concrete values are replaced by symbolic values and variables which are used to generate path conditions which are logic formulas that represent the state of the program and the transformations between program states. Some detectors were used in the audit to find some vulnerabilities such as: Integer Overflow, Simple and Advance Reentrancy and Delegate calls.

Results:

MISOTokenFactoy.sol

```
2021-04-28 15:48:44,502: [105] m.main:INFO: Beginning analysis
2021-04-28 15:48:44,505: [105] m.e.manticore:INFO: Starting symbolic create contract
2021-04-28 15:48:50,577: [105] m.e.manticore:INFO: Starting symbolic transaction: 0
2021-04-28 15:49:36,651: [105] m.e.detectors:WARNING: INVALID instruction
2021-04-28 16:02:19,232: [105] m.p.evm:WARNING: Concretizing LOG's 2 argument and dropping its taints: the value might not be tracked properly (This may affect detectors)
2021-04-28 16:02:44,272: [105] m.e.manticore:INFO: 11 alive states, 33 terminated states
2021-04-28 16:02:44,810: [285] m.c.manticore:INFO: Generated testcase No. 0 - RETURN(2 txs)
2021-04-28 16:02:44,811: [285] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-28 16:02:44,871: [287] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-28 16:02:44,964: [286] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-28 16:02:45,030: [289] m.c.manticore:INFO: Generated testcase No. 1 - STOP(2 txs)
2021-04-28 16:02:45,030: [288] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-28 16:02:45,032: [289] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-28 16:02:45,143: [290] m.c.manticore:INFO: Generated testcase No. 2 - RETURN(2 txs)
2021-04-28 16:02:45,144: [290] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-28 16:02:45,149: [292] m.c.manticore:INFO: Generated testcase No. 3 - RETURN(2 txs)
2021-04-28 16:02:45,151: [292] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-28 16:02:45,220: [293] m.c.manticore:INFO: Generated testcase No. 4 - RETURN(2 txs)
2021-04-28 16:02:45,222: [293] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-28 16:02:45,296: [297] m.c.manticore:INFO: Generated testcase No. 5 - RETURN(2 txs)
2021-04-28 16:02:45,297: [297] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-28 16:02:45,339: [298] m.c.manticore:INFO: Generated testcase No. 6 - RETURN(2 txs)
2021-04-28 16:02:45,341: [298] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-28 16:02:45,606: [287] m.c.manticore:INFO: Generated testcase No. 7 - RETURN(2 txs)
2021-04-28 16:02:45,708: [286] m.c.manticore:INFO: Generated testcase No. 8 - RETURN(2 txs)
2021-04-28 16:02:45,796: [288] m.c.manticore:INFO: Generated testcase No. 9 - RETURN(2 txs)
2021-04-28 16:03:00,507: [293] m.c.manticore:INFO: Generated testcase No. 10 - RETURN(2 txs)
2021-04-28 16:03:37,665: [105] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-28 16:04:22,359: [105] m.c.manticore:INFO: Results in /share/mcore_nwa86zgs
```

Listing 50

```
1 - INVALID instruction -
2   Contract: 0xc06496d2feb76cfdfcd1958b3a47e0e336c05899  EVM
        Program counter: 0xe64
3   Solidity snippet:
4     40  address[] public tokens
5
```

MISOFermenter.sol

```
2021-04-29 10:52:09,175: [569] m.main:INFO: Beginning analysis
2021-04-29 10:52:09,177: [569] m.e.manticore:INFO: Starting symbolic create contract
2021-04-29 10:52:11,416: [569] m.e.manticore:INFO: Starting symbolic transaction: 0
2021-04-29 10:52:14,217: [569] m.e.detectors:WARNING: INVALID instruction
2021-04-29 10:54:15,512: [569] m.e.manticore:INFO: 7 alive states, 8 terminated states
2021-04-29 10:54:15,630: [668] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 10:54:15,695: [669] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 10:54:15,762: [673] m.c.manticore:INFO: Generated testcase No. 0 - RETURN(2 txs)
2021-04-29 10:54:15,764: [673] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 10:54:15,781: [675] m.c.manticore:INFO: Generated testcase No. 2 - STOP(2 txs)
2021-04-29 10:54:15,781: [674] m.c.manticore:INFO: Generated testcase No. 1 - RETURN(2 txs)
2021-04-29 10:54:15,782: [675] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 10:54:15,783: [674] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 10:54:15,789: [676] m.c.manticore:INFO: Generated testcase No. 3 - RETURN(2 txs)
2021-04-29 10:54:15,790: [676] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 10:54:15,792: [670] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 10:54:15,922: [682] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 10:54:15,934: [679] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 10:54:16,012: [668] m.c.manticore:INFO: Generated testcase No. 4 - RETURN(2 txs)
2021-04-29 10:54:16,121: [669] m.c.manticore:INFO: Generated testcase No. 5 - RETURN(2 txs)
2021-04-29 10:54:16,182: [670] m.c.manticore:INFO: Generated testcase No. 6 - RETURN(2 txs)
2021-04-29 10:54:40,257: [569] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 10:54:43,271: [569] m.c.manticore:INFO: Results in /share/mcore_tyg79gmp
```

Listing 51

```
- INVALID instruction -
  Contract: 0xe05d862759d96988dd6e24a44853ca8d8843b633  EVM
      Program counter: 0x3a8
  Solidity snippet:
    35  address[] public escrows
```

MISOLiquidityLauncher.sol

```
2021-04-29 10:56:50,021: [846] m.main:INFO: Beginning analysis
2021-04-29 10:56:50,023: [846] m.e.manticore:INFO: Starting symbolic create contract
2021-04-29 10:56:52,968: [846] m.e.manticore:INFO: Starting symbolic transaction: 0
2021-04-29 10:56:58,011: [846] m.e.detectors:WARNING: INVALID instruction
2021-04-29 11:00:36,315: [846] m.e.manticore:INFO: 8 alive states, 9 terminated states
2021-04-29 11:00:36,540: [952] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:00:36,639: [953] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:00:36,731: [957] m.c.manticore:INFO: Generated testcase No. 0 - RETURN(2 txs)
2021-04-29 11:00:36,732: [957] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:00:36,733: [955] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:00:36,766: [956] m.c.manticore:INFO: Generated testcase No. 1 - STOP(2 txs)
2021-04-29 11:00:36,768: [956] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:00:36,778: [959] m.c.manticore:INFO: Generated testcase No. 2 - RETURN(2 txs)
2021-04-29 11:00:36,780: [959] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:00:36,825: [960] m.c.manticore:INFO: Generated testcase No. 3 - RETURN(2 txs)
2021-04-29 11:00:36,826: [960] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:00:36,840: [961] m.c.manticore:INFO: Generated testcase No. 4 - RETURN(2 txs)
2021-04-29 11:00:36,842: [961] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:00:37,108: [952] m.c.manticore:INFO: Generated testcase No. 5 - RETURN(2 txs)
2021-04-29 11:00:37,132: [953] m.c.manticore:INFO: Generated testcase No. 6 - RETURN(2 txs)
2021-04-29 11:00:37,217: [955] m.c.manticore:INFO: Generated testcase No. 7 - RETURN(2 txs)
2021-04-29 11:00:37,330: [963] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:00:51,137: [846] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:00:55,931: [846] m.c.manticore:INFO: Results in /share/mcore_vkzth6si
```

Listing 52

```
- INVALID instruction -
  Contract: 0x1a8f7b65edf810af2f5b3c7da345e13765fb87e5  EVM
      Program counter: 0x3f9
  Solidity snippet:
    27  address[] public launchers
```

MISOAccessControls.sol

```
2021-04-29 11:05:17,489: [1146] m.main:INFO: Beginning analysis
2021-04-29 11:05:17,491: [1146] m.e.manticore:INFO: Starting symbolic create contract
2021-04-29 11:05:21,748: [1146] m.e.manticore:INFO: Starting symbolic transaction: 0
2021-04-29 11:05:27,721: [1146] m.e.manticore:INFO: Found a concrete globalsha3 b'\x97fpp\xc5N\xf1\x82\xb0\xf5\x85\x8b\x03K\xea\xc1\xb6\xf3\x08\x9a\xa2\xd3\x18\x8b\xb1\xe8\x92\x9fO\xa9\xb9)\x00\x00\x00\x00\x00\x00\x00\x00\x00
\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00' -> 107805592017746300286356689973669261979332378511177658323151461781878711726495
2021-04-29 11:05:29,536: [1146] m.e.manticore:INFO: Found a concrete globalsha3 b'\x9f-\xf0\xfe\xd2\xc7vH\xdeX`\xa4\xccP\x8c\xd0\x81\x8c\x85\xb8\xb8\xa1\xabL\xee\xef\x8d\x98\x1c\x89V\xa6\x00\x00\x00\x00\x00\x00\x00\x00\x00\x0
0\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00' -> 33956025540612463420913754845728157836108004539506646855579507491750302920 76
2021-04-29 11:05:33,367: [1146] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x0
0\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00' -> 78338746147236970124700731725183845421594913511827187288591969170390706184117
2021-04-29 11:05:36,408: [1146] m.e.manticore:INFO: Found a concrete globalsha3 b'\x9dI\xf3\x97\xae\x9e\xf1\xa84\xb5i\xac\xb9gy\x9a6pa\xe3\x05\x93!\x81\xa4OWs\xda\x87;\xfd\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x
00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00' -> 58691200583646296702422031135575587854415580653341377753113499911664758366186
2021-04-29 11:05:36,507: [1146] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\r\xed\x86J\xa3\x80\x1878\xb9\x86\x1b(,\x86W\xe7\xc1\xbfy\xad2(\xb6v\xf7\xd3\xcdB\x84\xa5D?\x17
\xf1\x96+6\xe4\x91\xb3\n@\xb2@XI\xe5\x97\xba_\xb6' -> 75747826566981395943807448412316615630342422239967101656874720293306082442793
2021-04-29 11:14:49,974: [1146] m.e.manticore:INFO: Found a concrete globalsha3 b'\xad2(\xb6v\xf7\xd3\xcdB\x84\xa5D?\x17\xf1\x96+6\xe4\x91\xb3\n@\xb2@XI\xe5\x97\xba_\xb5' -> 108949517105633263939327814466789839344308686945441
099439098060136067734755317
2021-04-29 11:27:15,628: [1146] m.e.detectors:WARNING: Potentially reading uninitialized storage
2021-04-29 11:30:09,017: [1146] m.e.manticore:INFO: 13 alive states, 18 terminated states
2021-04-29 11:30:10,139: [1372] m.c.manticore:INFO: Generated testcase No. 0 - RETURN(2 txs)
2021-04-29 11:30:10,141: [1372] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:30:10,206: [1373] m.c.manticore:INFO: Generated testcase No. 1 - RETURN(2 txs)
2021-04-29 11:30:10,207: [1373] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:30:10,658: [1368] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:30:10,862: [1367] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:30:10,938: [1369] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:30:11,156: [1370] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:30:11,481: [1377] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:30:11,708: [1375] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:30:12,180: [1366] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:30:12,809: [1368] m.c.manticore:INFO: Generated testcase No. 2 - RETURN(2 txs)
2021-04-29 11:30:12,858: [1371] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:30:12,994: [1367] m.c.manticore:INFO: Generated testcase No. 3 - RETURN(2 txs)
2021-04-29 11:30:13,061: [1369] m.c.manticore:INFO: Generated testcase No. 4 - RETURN(2 txs)
2021-04-29 11:30:13,381: [1370] m.c.manticore:INFO: Generated testcase No. 5 - STOP(2 txs)
2021-04-29 11:30:13,613: [1377] m.c.manticore:INFO: Generated testcase No. 6 - RETURN(2 txs)
2021-04-29 11:30:13,799: [1375] m.c.manticore:INFO: Generated testcase No. 7 - RETURN(2 txs)
2021-04-29 11:30:17,062: [1366] m.c.manticore:INFO: Generated testcase No. 8 - RETURN(2 txs)
2021-04-29 11:30:17,838: [1371] m.c.manticore:INFO: Generated testcase No. 9 - STOP(2 txs)
2021-04-29 11:30:24,648: [1373] m.c.manticore:INFO: Generated testcase No. 10 - RETURN(2 txs)
2021-04-29 11:30:27,907: [1372] m.c.manticore:INFO: Generated testcase No. 11 - RETURN(2 txs)
2021-04-29 11:30:35,729: [1370] m.c.manticore:INFO: Generated testcase No. 12 - RETURN(2 txs)
2021-04-29 11:30:51,955: [1146] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:31:26,449: [1146] m.c.manticore:INFO: Results in /share/mcore_o_dxp3ka
```

Listing 53

```
- Potentially reading uninitialized storage -
  Contract: 0x2ffb1df6723fbee1cfdca9d129cbb103fe66c7f3  EVM
      Program counter: 0xfc3
  Solidity snippet:
    34  indexed caller
        35
```

MISOAccessFactory.sol

```
2021-04-29 11:52:11,229: [1731] m.main:INFO: Beginning analysis
2021-04-29 11:52:11,231: [1731] m.e.manticore:INFO: Starting symbolic create contract
2021-04-29 11:52:14,462: [1731] m.e.manticore:INFO: Starting symbolic transaction: 0
2021-04-29 11:52:35,018: [1731] m.e.detectors:WARNING: INVALID instruction
2021-04-29 11:54:08,521: [1731] m.e.manticore:INFO: 7 alive states, 19 terminated states
2021-04-29 11:54:08,700: [1845] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:54:08,744: [1846] m.c.manticore:INFO: Generated testcase No. 0 - STOP(2 txs)
2021-04-29 11:54:08,747: [1846] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:54:08,838: [1848] m.c.manticore:INFO: Generated testcase No. 1 - RETURN(2 txs)
2021-04-29 11:54:08,839: [1848] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:54:08,850: [1851] m.c.manticore:INFO: Generated testcase No. 2 - RETURN(2 txs)
2021-04-29 11:54:08,852: [1851] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:54:08,874: [1850] m.c.manticore:INFO: Generated testcase No. 3 - RETURN(2 txs)
2021-04-29 11:54:08,875: [1850] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:54:08,887: [1855] m.c.manticore:INFO: Generated testcase No. 5 - RETURN(2 txs)
2021-04-29 11:54:08,888: [1854] m.c.manticore:INFO: Generated testcase No. 4 - RETURN(2 txs)
2021-04-29 11:54:08,888: [1855] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:54:08,890: [1854] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:54:09,194: [1845] m.c.manticore:INFO: Generated testcase No. 6 - RETURN(2 txs)
2021-04-29 11:54:18,130: [1731] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 11:54:23,922: [1731] m.c.manticore:INFO: Results in /share/mcore_084m0vbh
```

Listing 54

```
- INVALID instruction -
  Contract: 0xd771845f3de93ba366c96f49eb51f70c26e0c2b4  EVM
      Program counter: 0xbb6
```

```
Solidity snippet:
  25  address[] public children
```

MISOAdminAccess.sol

```
2021-04-29 12:02:43,906: [2017] m.main:INFO: Beginning analysis
2021-04-29 12:02:43,908: [2017] m.e.manticore:INFO: Starting symbolic create contract
2021-04-29 12:02:46,836: [2017] m.e.manticore:INFO: Starting symbolic transaction: 0
2021-04-29 12:02:52,670: [2017] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x0
0\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00' -> 78338746147236970124700731725183845421594913511827187288591969170390706184117
2021-04-29 12:03:12,188: [2017] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x92r\xc5\xe8N\xa9\xd7__?gHc\x85\xab\xe6\xcf\x7f\xc7^\xad2(\xb6v\xf7\xd3\xcdB\x84\xa5D?\x17\xf1
\x96+6\xe4\x91\xb3\n@\xb2@XI\xe5\x97\xba_\xb6' -> 110358265247489776265680614812032941922355087728031115947097882211986241511275
2021-04-29 12:06:06,865: [2017] m.e.manticore:INFO: Found a concrete globalsha3 b'\xad2(\xb6v\xf7\xd3\xcdB\x84\xa5D?\x17\xf1\x96+6\xe4\x91\xb3\n@\xb2@XI\xe5\x97\xba_\xb5' -> 108949517105633263939327814466789839344308686945441
099439098060136067734755317
2021-04-29 12:11:23,145: [2017] m.e.detectors:WARNING: Potentially reading uninitialized storage
2021-04-29 12:11:32,271: [2017] m.e.manticore:INFO: 7 alive states, 10 terminated states
2021-04-29 12:11:33,049: [2179] m.c.manticore:INFO: Generated testcase No. 0 - RETURN(2 txs)
2021-04-29 12:11:33,050: [2179] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 12:11:33,187: [2172] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 12:11:33,296: [2175] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 12:11:33,405: [2178] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 12:11:33,445: [2174] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 12:11:33,856: [2182] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 12:11:33,884: [2185] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 12:11:33,967: [2173] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 12:11:34,352: [2180] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 12:11:34,396: [2177] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 12:11:34,472: [2172] m.c.manticore:INFO: Generated testcase No. 1 - STOP(2 txs)
2021-04-29 12:11:34,554: [2175] m.c.manticore:INFO: Generated testcase No. 2 - RETURN(2 txs)
2021-04-29 12:11:34,674: [2178] m.c.manticore:INFO: Generated testcase No. 3 - RETURN(2 txs)
2021-04-29 12:11:34,755: [2174] m.c.manticore:INFO: Generated testcase No. 4 - RETURN(2 txs)
2021-04-29 12:11:36,487: [2173] m.c.manticore:INFO: Generated testcase No. 5 - RETURN(2 txs)
2021-04-29 12:11:36,921: [2177] m.c.manticore:INFO: Generated testcase No. 6 - STOP(2 txs)
2021-04-29 12:11:55,875: [2017] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 12:12:08,939: [2017] m.c.manticore:INFO: Results in /share/mcore_og69k6db
```

Listing 55

```
- Potentially reading uninitialized storage -
Contract: 0xf5e00a87c3725500647841f6f9376450367c5377  EVM
    Program counter: 0x81d
Solidity snippet:
  34  initAccess = true

```

DutchAuction.sol

```
2021-04-29 14:46:02,727: [473] m.main:INFO: Beginning analysis
2021-04-29 14:46:02,729: [473] m.e.manticore:INFO: Starting symbolic create contract
2021-04-29 14:46:28,458: [473] m.e.manticore:INFO: Starting symbolic transaction: 0
2021-04-29 14:49:38,978: [473] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
2021-04-29 14:50:02,663: [473] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
2021-04-29 14:50:51,739: [473] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
2021-04-29 14:51:18,290: [473] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
2021-04-29 14:54:24,425: [473] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x
00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00' -> 78338746
147236970124700731725183845421594913511827187288591969170390706184117
2021-04-29 14:54:47,169: [473] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
2021-04-29 14:55:19,123: [473] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
2021-04-29 14:55:21,263: [473] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
2021-04-29 14:55:27,176: [473] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00.\xf0\x13\xf5-\x90\xcfk\xe2\x9e\x8e\xba\xd7\xf
58>\xd9d)`\xad2(\xb6v\xf7\xd3\xcdB\x84\xa5D?\x17\xf1\x96+6\xe4\x91\xb3\n@\xb2@XI\xe5\x97\xba_\xb6' -> 114255025405949350345530878573045796171788899672233836316374617182380129
016235
```

HyperbolicAuction.sol

```
2021-04-29 15:38:19,854: [764] m.e.detectors:WARNING: Potentially reading uninitialized storage
2021-04-29 15:38:29,037: [764] m.e.detectors:WARNING: Potentially reading uninitialized storage
2021-04-29 15:38:43,753: [764] m.e.manticore:INFO: Found a concrete globalsha3 b'\x97fpp\xc5N\xf1\x8z\xb0\xf5\x85\x8b\x03K\xea\xc1\xb6\xf3\x08\x9a\xa2\xd3\x18\x8b\xb1\xe8\x92\
x9fO\xa9\xb9)\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00' -> 107805592017746300286356689973
366926197933237851117765832315146178187871726495
2021-04-29 15:40:44,262: [764] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x006\xae\xb2hL(\xe6)\xc6V\xbd\x9d&=M\xb6\x8c\xcfF
\x91\xad2(\xb6v\xf7\xd3\xcdB\x84\xa5D?\x17\xf1\x96+6\xe4\x91\xb3\n@\xb2@XI\xe5\x97\xba_\xb6' -> 111352176424578406654942707019684485426114695074745053145793698640480647807357
2021-04-29 15:40:48,870: [764] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
2021-04-29 15:40:49,731: [764] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
2021-04-29 15:55:11,512: [764] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
2021-04-29 15:55:12,660: [764] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
2021-04-29 16:06:07,720: [764] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
2021-04-29 16:06:08,583: [764] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
2021-04-29 16:06:52,944: [764] m.e.detectors:WARNING: INVALID instruction
2021-04-29 16:13:01,126: [764] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
2021-04-29 16:13:02,131: [764] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
2021-04-29 16:13:47,390: [764] m.e.detectors:WARNING: Potentially reading uninitialized storage
2021-04-29 16:14:36,088: [764] m.e.detectors:WARNING: Potentially reading uninitialized storage
2021-04-29 16:26:51,041: [764] m.e.detectors:WARNING: Potentially reading uninitialized storage
2021-04-29 16:28:01,321: [764] m.e.detectors:WARNING: Potentially reading uninitialized storage
2021-04-29 16:38:58,404: [764] m.e.manticore:INFO: Found a concrete globalsha3 b'\x9dI\xf3\x97\xae\x9e\xf1\xa84\xb5i\xac\xb9gy\x9a6pa\xe3\x05\x93!\x81\xa4OWs\xda\x87;\xfd\x00\
x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00' -> 5869120058364629670242203113557558785441558065334137775311349991166475836618 6
2021-04-29 16:39:00,815: [764] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x006\xae\xb2hL(\xe6)\xc6V\xbd\x9d&=M\xb6\x8c\xcfF
\x91\x81\xc2\n\xd5\xf5\xfd\x8e$\x0fI\xed\xb0F\xc3-;\x10W\xba\x93\xff\x8a\x19\x82^\xdc%\x83\xc5\xc3\x1f\xeb' -> 1156772609075435561192796851611816741031347784142011570059197051338 6583294828
2021-04-29 16:39:06,812: [764] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
2021-04-29 16:39:25,857: [764] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
2021-04-29 16:39:29,194: [764] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
2021-04-29 16:40:36,374: [764] m.e.detectors:WARNING: Potentially reading uninitialized storage
2021-04-29 16:51:29,569: [764] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
2021-04-29 16:51:51,704: [764] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
```

FixedToken.sol

```
2021-04-29 17:24:49,540: [30] m.main:INFO: Beginning analysis
2021-04-29 17:24:49,542: [30] m.e.manticore:INFO: Starting symbolic create contract
2021-04-29 17:24:55,773: [30] m.e.manticore:INFO: Starting symbolic transaction: 0
2021-04-29 17:25:46,525: [30] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00ys\xae#MH\x97e\xdb\xa3\xc3;-\xc3\x8e\xcf/\xc8\x
c7\xcf\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x03' -> 5936097343991899888368989798267819408
703294771061972398428625082692694958227
2021-04-29 17:28:02,588: [30] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00ys\xae#MH\x97e\xdb\xa3\xc3;-\xc3\x8e\xcf/\xc8\x
c7\xcf\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x01' -> 7542207184770167497258186229687406346
68709328533072004859857774099468865034842
2021-04-29 17:37:54,955: [30] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
```

MintableToken.sol

```
2021-04-29 17:26:46,780: [91] m.main:INFO: Beginning analysis
2021-04-29 17:26:46,781: [91] m.e.manticore:INFO: Starting symbolic create contract
2021-04-29 17:26:56,429: [91] m.e.manticore:INFO: Starting symbolic transaction: 0
2021-04-29 17:28:03,222: [91] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00@\xdeg\xab\xeb\xaf\x1e,L\xb2ac}k\x96\xa3\x97\xb
b\xaa\xca\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00' -> 1120912213801401266057876009286317716296763872644975347734707204914183281125 53
2021-04-29 17:28:49,018: [91] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x0
0\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00' -> 783387461472369701247007317251838454215949135118271872885919691703907061841 17
2021-04-29 17:28:53,059: [91] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00@\xdeg\xab\xeb\xaf\x1e,L\xb2ac}k\x96\xa3\x97\xb
b\xaa\xca\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x02' -> 6799564979635064662570401734682640228843572300946600123661626624837618670 269
2021-04-29 17:29:29,531: [91] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00@\xdeg\xab\xeb\xaf\x1e,L\xb2ac}k\x96\xa3\x97\xb
b\xaa\xca\xad2(\xb6v\xf7\xd3\xcdB\x84\xa5D?\x17\xf1\x96+6\xe4\x91\xb3\n@\xb2@XI\xe5\x97\xba_\xb6' -> 8266069127018952459091539996784824668360412085284313363603166731614537173691
2021-04-29 18:17:55,107: [91] m.e.manticore:INFO: Found a concrete globalsha3 b'e\xd7\xa2\x8e2e\xb3zdt\x92\x9f3e!\xb32\xc1h\x1b\x93?l\xb9\xf37fsD\r\x86*\x00\x00\x00\x00\x00\x0
0\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00' -> 1120769908023974508943377164250082809153724287466338374543195232295110411815 19
2021-04-29 18:17:56,671: [91] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00@\xdeg\xab\xeb\xaf\x1e,L\xb2ac}k\x96\xa3\x97\xb
b\xaa\xca\xf7\xc9T,Y\x10\x17\xa2\x1ct\xb6\xf3\xfa\xb6&<yR\xfc\n\xaf\x9d\xb4\xc2**\x04\xdd\xc7\xf8gP' -> 5991431928722924019166066645300586003433268030231124549624212649399257984 0310
2021-04-29 18:39:39,120: [91] m.e.manticore:INFO: Found a concrete globalsha3 b'\x9f-\xf0\xfe\xd2\xc7vH\xdeX`\xa4\xccP\x8c\xd0\x81\x8c\x85\xb8\xb8\xa1\xabL\xee\xef\x8d\x98\x1c
\x89V\xa6\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00' -> 3395602554061246342091375484572815783618080045395066468555795074917503029207 6
2021-04-29 18:39:40,043: [91] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00@\xdeg\xab\xeb\xaf\x1e,L\xb2ac}k\x96\xa3\x97\xb
b\xaa\xca\x07\x81\xd7\xca\xc9\xc3x\xef\xa2*t\x81\xe4\xd4\xd2\x97\x04\xa6\x80\xdd\xf5\x04\xb3\xbcP\xb5\x17p\x0e\xe1\x1em' -> 515820485065479149226679210357888436510916592681902932873364143070681690560885
2021-04-29 18:55:54,329: [91] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
```

TokenVault.sol

```
2021-04-29 19:52:21,958: [546] m.e.manticore:INFO: 3 alive states, 11 terminated states
2021-04-29 19:52:22,599: [663] m.c.manticore:INFO: Generated testcase No. 0 - RETURN(2 txs)
2021-04-29 19:52:22,601: [663] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 19:52:22,823: [661] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 19:52:22,881: [662] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 19:52:23,605: [669] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 19:52:23,611: [678] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 19:52:23,753: [661] m.c.manticore:INFO: Generated testcase No. 1 - RETURN(2 txs)
2021-04-29 19:52:23,758: [662] m.c.manticore:INFO: Generated testcase No. 2 - RETURN(2 txs)
2021-04-29 19:52:24,129: [672] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 19:53:00,537: [546] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 19:53:08,329: [546] m.c.manticore:INFO: Results in /share/MISO/mcore_vxx7s5v7
```

Listing 56

```
- INVALID instruction -
  Contract: 0x466b00dccb6c8f9513b5230e1a612445e2fc1dcc  EVM
      Program counter: 0x889
  Solidity snippet:
    100  users[msg.sender].lockToItems[_tokenAddress][_index]

- INVALID instruction -
  Contract: 0x466b00dccb6c8f9513b5230e1a612445e2fc1dcc  EVM
      Program counter: 0xd34
  Solidity snippet:
    35  uint256[] public allDepositIds

- INVALID instruction -
  Contract: 0x466b00dccb6c8f9513b5230e1a612445e2fc1dcc  EVM
      Program counter: 0xcba
  Solidity snippet:
    131  users[_user].lockToItems[_tokenAddress][_index]
```

SushiToken.sol

```
2021-04-29 19:19:31,796: [254] m.main:INFO: Beginning analysis
2021-04-29 19:19:31,797: [254] m.e.manticore:INFO: Starting symbolic create contract
2021-04-29 19:19:43,404: [254] m.e.manticore:INFO: Starting symbolic transaction: 0
2021-04-29 19:20:43,850: [254] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00&\xeb7\x9fS\xa2/\x8e*\xda\xaf\xdf\xd5\xc1<g7~\
xaf\x9e\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\t' -> 3576969833931666009251062816571513168
22592329189401815099833264433165584048 66
2021-04-29 19:21:12,283: [254] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00&\xeb7\x9fS\xa2/\x8e*\xda\xaf\xdf\xd5\xc1<g7~\
xaf\x9e\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x02' -> 11504311185096745105875621394427823
7381171057784163530198232347618208474712 15
2021-04-29 19:22:03,686: [254] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00&\xeb7\x9fS\xa2/\x8e*\xda\xaf\xdf\xd5\xc1<g7~\
xaf\x9e\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x04' -> 73887915257533345478929720630497624
2384566705541715875124774657303858049537 71
2021-04-29 19:22:12,036: [254] m.e.manticore:INFO: Found a concrete globalsha3 b'' -> 89477152217924674838424037953991966239322087453347756267410168184682657981552
2021-04-29 19:22:34,277: [254] m.e.manticore:INFO: Found a concrete globalsha3 b'\x9f-\xf0\xfe\xd2\xc7vH\xdeX`\xa4\xccP\x8c\xd0\x81\x8c\x85\xb8\xb8\xa1\xabL\xee\xef\x8d\x98\x1
c\x89V\xa6\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00' -> 33956025540612463420913754845728
157836108004539506646855579507491750302920 76
2021-04-29 19:22:36,665: [254] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00&\xeb7\x9fS\xa2/\x8e*\xda\xaf\xdf\xd5\xc1<g7~\
xaf\x9e\x07\x81\xd7\xca\xc9\xc3x\xef\xa2*t\x81\xe4\xd4\xd2\x97\x04\xa6\x80\xdd\xf5\x04\xb3\xbcP\xb5\x17p\x0e\xe1\x1em' -> 29811837382829081519049040879703703551541090927856597
94558743735329710768260 0
2021-04-29 19:52:34,001: [254] m.e.detectors:WARNING: Warning NUMBER instruction used
2021-04-29 19:54:07,940: [254] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x
00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00' -> 78338746
1472369701247007317251838454215949135118271872885919691703907061841 17
2021-04-29 19:54:09,874: [254] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00&\xeb7\x9fS\xa2/\x8e*\xda\xaf\xdf\xd5\xc1<g7~\
xaf\x9e\xad2(\xb6v\xf7\xd3\xcdB\x84\xa5D?\x17\xf1\x96+6\xe4\x91\xb3\n@\xb2@XI\xe5\x97\xba_\xb6' -> 4820353491592311171149924492212927033622808856182423310111966132587057374215
7
```

PoolLiquidity.sol

```
2021-04-29 21:46:05,238: [1054] m.main:INFO: Beginning analysis
2021-04-29 21:46:05,242: [1054] m.e.manticore:INFO: Starting symbolic create contract
2021-04-29 21:46:19,653: [1054] m.e.manticore:INFO: Starting symbolic transaction: 0
2021-04-29 21:47:37,266: [1054] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
2021-04-29 21:49:52,410: [1054] m.e.detectors:WARNING: Warning TIMESTAMP instruction used
2021-04-29 21:50:00,108: [1054] m.e.detectors:WARNING: Unsigned integer overflow at ADD instruction
2021-04-29 22:10:22,316: [1054] m.p.evm:WARNING: Concretizing EXTCODESIZE's 1 argument and dropping its taints: the value might not be tracked properly (This may affect detect
ors)
2021-04-29 22:12:01,520: [1054] m.e.manticore:INFO: 16 alive states, 46 terminated states
2021-04-29 22:12:02,663: [1289] m.c.manticore:INFO: Generated testcase No. 0 - RETURN(2 txs)
2021-04-29 22:12:02,665: [1289] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 22:12:02,791: [1290] m.c.manticore:INFO: Generated testcase No. 1 - RETURN(2 txs)
2021-04-29 22:12:02,793: [1290] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 22:12:02,852: [1291] m.c.manticore:INFO: Generated testcase No. 2 - RETURN(2 txs)
2021-04-29 22:12:02,854: [1291] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 22:12:03,018: [1293] m.c.manticore:INFO: Generated testcase No. 3 - RETURN(2 txs)
2021-04-29 22:12:03,020: [1293] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 22:12:03,030: [1292] m.c.manticore:INFO: Generated testcase No. 4 - RETURN(2 txs)
2021-04-29 22:12:03,031: [1292] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 22:12:03,116: [1294] m.c.manticore:INFO: Generated testcase No. 5 - RETURN(2 txs)
2021-04-29 22:12:03,117: [1294] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 22:12:03,155: [1296] m.c.manticore:INFO: Generated testcase No. 7 - RETURN(2 txs)
2021-04-29 22:12:03,156: [1296] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 22:12:03,160: [1295] m.c.manticore:INFO: Generated testcase No. 6 - RETURN(2 txs)
2021-04-29 22:12:03,162: [1295] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 22:12:03,306: [1297] m.c.manticore:INFO: Generated testcase No. 8 - RETURN(2 txs)
2021-04-29 22:12:03,308: [1297] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 22:12:03,398: [1299] m.c.manticore:INFO: Generated testcase No. 9 - RETURN(2 txs)
2021-04-29 22:12:03,400: [1299] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-04-29 22:12:26,999: [1294] m.c.manticore:INFO: Generated testcase No. 10 - RETURN(2 txs)
2021-04-29 22:12:27,393: [1292] m.c.manticore:INFO: Generated testcase No. 11 - RETURN(2 txs)
2021-04-29 22:12:27,593: [1295] m.c.manticore:INFO: Generated testcase No. 12 - RETURN(2 txs)
2021-04-29 22:12:27,611: [1291] m.c.manticore:INFO: Generated testcase No. 13 - RETURN(2 txs)
2021-04-29 22:12:27,800: [1293] m.c.manticore:INFO: Generated testcase No. 14 - RETURN(2 txs)
2021-04-29 22:12:28,350: [1296] m.c.manticore:INFO: Generated testcase No. 15 - RETURN(2 txs)
2021-04-29 22:13:34,073: [1054] m.c.manticore:INFO: Results in /share/MISO/mcore_0br7n14t
```

Listing 57

```
- Unsigned integer overflow at ADD instruction -
  Contract: 0xc247ef81acb7901f62f7bae2777bd6449890a4b  EVM Program
      counter: 0xc16
  Solidity snippet:
    126  expiry

- Warning TIMESTAMP instruction used -
  Contract: 0xc247ef81acb7901f62f7bae2777bd6449890a4b  EVM Program
      counter: 0x7f2
  Solidity snippet:
    141  block.timestamp

- Warning TIMESTAMP instruction used -
  Contract: 0xc247ef81acb7901f62f7bae2777bd6449890a4b  EVM Program
      counter: 0x1541
  Solidity snippet:
    153  block.timestamp
```

5.2 STATIC ANALYSIS REPORT

Description:

Halborn used automated testing techniques to enhance coverage of certain areas of the scoped contract. 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. 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.

Results:

MISOTokenFactory.sol

```
INFO:Detectors:
SafeTransfer._safeTransfer(address,address,uint256) (contracts/Utils/SafeTransfer.sol#52-64) uses a dangerous strict equality:
        - require(bool)(success && (data.length == 0 || abi.decode(data,(bool)))) (contracts/Utils/SafeTransfer.sol#63)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities
INFO:Detectors:
Address.isContract(address) (contracts/OpenZeppelin/utils/Address.sol#26-35) uses assembly
        - INLINE ASM (contracts/OpenZeppelin/utils/Address.sol#33)
Address._verifyCallResult(bool,bytes,string) (contracts/OpenZeppelin/utils/Address.sol#171-188) uses assembly
        - INLINE ASM (contracts/OpenZeppelin/utils/Address.sol#180-183)
CloneFactory.createClone(address) (contracts/Utils/CloneFactory.sol#34-43) uses assembly
        - INLINE ASM (contracts/Utils/CloneFactory.sol#36-42)
CloneFactory.isClone(address,address) (contracts/Utils/CloneFactory.sol#45-60) uses assembly
        - INLINE ASM (contracts/Utils/CloneFactory.sol#47-59)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
```

In that case, the use of strict equality is safe. (== 0). So, it is a false positive.

BatchAution.sol

```
INFO:Detectors:
Reentrancy in BatchAuction.finalize() (contracts/Auctions/BatchAuction.sol#238-258):
        External calls:
        - _tokenPayment(paymentCurrency,wallet,uint256(marketStatus.commitmentsTotal)) (contracts/Auctions/BatchAuction.sol#249)
                - (success,data) = token.call(abi.encodeWithSelector(0xa9059cbb,to,amount)) (contracts/Utils/SafeTransfer.sol#58-62)
        - _tokenPayment(auctionToken,wallet,marketInfo.totalTokens) (contracts/Auctions/BatchAuction.sol#254)
                - (success,data) = token.call(abi.encodeWithSelector(0xa9059cbb,to,amount)) (contracts/Utils/SafeTransfer.sol#58-62)
        External calls sending eth:
        - _tokenPayment(paymentCurrency,wallet,uint256(marketStatus.commitmentsTotal)) (contracts/Auctions/BatchAuction.sol#249)
                - _to.transfer(_amount) (contracts/Utils/SafeTransfer.sol#32)
        - _tokenPayment(auctionToken,wallet,marketInfo.totalTokens) (contracts/Auctions/BatchAuction.sol#254)
                - _to.transfer(_amount) (contracts/Utils/SafeTransfer.sol#32)
        State variables written after the call(s):
        - marketStatus.finalized = true (contracts/Auctions/BatchAuction.sol#256)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities
INFO:Detectors:
BatchAuction.ETH_ADDRESS (contracts/Auctions/BatchAuction.sol#37) shadows:
        - SafeTransfer.ETH_ADDRESS (contracts/Utils/SafeTransfer.sol#5)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variable-shadowing
INFO:Detectors:
Documents._setDocument(string,string) (contracts/Utils/Documents.sol#33-43) uses a dangerous strict equality:
        - _documents[_name].lastModified == uint64(0) (contracts/Utils/Documents.sol#37)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities
INFO:Detectors:
BatchAuction.enableList(bool)._status (contracts/Auctions/BatchAuction.sol#354) shadows:
        - ReentrancyGuard._status (contracts/OpenZeppelin/utils/ReentrancyGuard.sol#36) (state variable)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
```

BatchAuction.finalize() is having nonReentrant guard so this Re-entrancy issue is false positive

DutchAuction.sol

```
INFO:Detectors:
Reentrancy in DutchAuction.finalize() (contracts/Auctions/DutchAuction.sol#404-432):
        External calls:
        - _tokenPayment(paymentCurrency,wallet,uint256(status.commitmentsTotal)) (contracts/Auctions/DutchAuction.sol#417)
                - (success,data) = token.call(abi.encodeWithSelector(0xa9059cbb,to,amount)) (contracts/Utils/SafeTransfer.sol#58-62)
        - _tokenPayment(auctionToken,wallet,uint256(marketInfo.totalTokens)) (contracts/Auctions/DutchAuction.sol#422)
                - (success,data) = token.call(abi.encodeWithSelector(0xa9059cbb,to,amount)) (contracts/Utils/SafeTransfer.sol#58-62)
        - _tokenPayment(auctionToken,wallet,uint256(marketInfo.totalTokens)) (contracts/Auctions/DutchAuction.sol#427)
                - (success,data) = token.call(abi.encodeWithSelector(0xa9059cbb,to,amount)) (contracts/Utils/SafeTransfer.sol#58-62)
        External calls sending eth:
        - _tokenPayment(paymentCurrency,wallet,uint256(status.commitmentsTotal)) (contracts/Auctions/DutchAuction.sol#417)
                - _to.transfer(_amount) (contracts/Utils/SafeTransfer.sol#32)
        - _tokenPayment(auctionToken,wallet,uint256(marketInfo.totalTokens)) (contracts/Auctions/DutchAuction.sol#422)
                - _to.transfer(_amount) (contracts/Utils/SafeTransfer.sol#32)
        - _tokenPayment(auctionToken,wallet,uint256(marketInfo.totalTokens)) (contracts/Auctions/DutchAuction.sol#427)
                - _to.transfer(_amount) (contracts/Utils/SafeTransfer.sol#32)
        State variables written after the call(s):
        - status.finalized = true (contracts/Auctions/DutchAuction.sol#429)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities
INFO:Detectors:
DutchAuction.ETH_ADDRESS (contracts/Auctions/DutchAuction.sol#36) shadows:
        - SafeTransfer.ETH_ADDRESS (contracts/Utils/SafeTransfer.sol#5)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variable-shadowing
INFO:Detectors:
SafeTransfer._safeTransfer(address,address,uint256) (contracts/Utils/SafeTransfer.sol#52-64) uses a dangerous strict equality:
        - require(bool)(success && (data.length == 0 || abi.decode(data,(bool)))) (contracts/Utils/SafeTransfer.sol#63)
SafeTransfer._safeTransferFrom(address,address,uint256) (contracts/Utils/SafeTransfer.sol#66-78) uses a dangerous strict equality:
        - require(bool)(success && (data.length == 0 || abi.decode(data,(bool)))) (contracts/Utils/SafeTransfer.sol#77)
Documents._setDocument(string,string) (contracts/Utils/Documents.sol#33-43) uses a dangerous strict equality:
        - _documents[_name].lastModified == uint64(0) (contracts/Utils/Documents.sol#37)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities
INFO:Detectors:
Reentrancy in DutchAuction.commitTokensFrom(address,uint256,bool) (contracts/Auctions/DutchAuction.sol#266-282):
        External calls:
        - _safeTransferFrom(paymentCurrency,_from,tokensToTransfer) (contracts/Auctions/DutchAuction.sol#279)
                - (success,data) = token.call(abi.encodeWithSelector(0x23b872dd,from,address(this),amount)) (contracts/Utils/SafeTransfer.sol#72-76)
        State variables written after the call(s):
        - _addCommitment(_from,tokensToTransfer) (contracts/Auctions/DutchAuction.sol#280)
                - status.commitmentsTotal = uint128(uint256(status.commitmentsTotal).add(_commitment)) (contracts/Auctions/DutchAuction.sol#391)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1
```

DutchAuction.finalize() is having nonReentrant guard so this Re-entrancy issue is false positive

HyperbolicAuction.sol

```
INFO:Detectors:
Reentrancy in HyperbolicAuction.finalize() (contracts/Auctions/HyperbolicAuction.sol#387-415):
        External calls:
        - _tokenPayment(paymentCurrency,wallet,uint256(status.commitmentsTotal)) (contracts/Auctions/HyperbolicAuction.sol#401)
                - (success,data) = token.call(abi.encodeWithSelector(0xa9059cbb,to,amount)) (contracts/Utils/SafeTransfer.sol#58-62)
        - _tokenPayment(auctionToken,wallet,uint256(info.totalTokens)) (contracts/Auctions/HyperbolicAuction.sol#406)
                - (success,data) = token.call(abi.encodeWithSelector(0xa9059cbb,to,amount)) (contracts/Utils/SafeTransfer.sol#58-62)
        - _tokenPayment(auctionToken,wallet,uint256(info.totalTokens)) (contracts/Auctions/HyperbolicAuction.sol#411)
                - (success,data) = token.call(abi.encodeWithSelector(0xa9059cbb,to,amount)) (contracts/Utils/SafeTransfer.sol#58-62)
        External calls sending eth:
        - _tokenPayment(paymentCurrency,wallet,uint256(status.commitmentsTotal)) (contracts/Auctions/HyperbolicAuction.sol#401)
                - _to.transfer(_amount) (contracts/Utils/SafeTransfer.sol#32)
        - _tokenPayment(auctionToken,wallet,uint256(info.totalTokens)) (contracts/Auctions/HyperbolicAuction.sol#406)
                - _to.transfer(_amount) (contracts/Utils/SafeTransfer.sol#32)
        - _tokenPayment(auctionToken,wallet,uint256(info.totalTokens)) (contracts/Auctions/HyperbolicAuction.sol#411)
                - _to.transfer(_amount) (contracts/Utils/SafeTransfer.sol#32)
        State variables written after the call(s):
        - status.finalized = true (contracts/Auctions/HyperbolicAuction.sol#413)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities
INFO:Detectors:
HyperbolicAuction.ETH_ADDRESS (contracts/Auctions/HyperbolicAuction.sol#58) shadows:
        - SafeTransfer.ETH_ADDRESS (contracts/Utils/SafeTransfer.sol#5)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variable-shadowing
INFO:Detectors:
SafeTransfer._safeTransfer(address,address,uint256) (contracts/Utils/SafeTransfer.sol#52-64) uses a dangerous strict equality:
        - require(bool)(success && (data.length == 0 || abi.decode(data,(bool)))) (contracts/Utils/SafeTransfer.sol#63)
SafeTransfer._safeTransferFrom(address,address,uint256) (contracts/Utils/SafeTransfer.sol#66-78) uses a dangerous strict equality:
        - require(bool)(success && (data.length == 0 || abi.decode(data,(bool)))) (contracts/Utils/SafeTransfer.sol#77)
Documents._setDocument(string,string) (contracts/Utils/Documents.sol#33-43) uses a dangerous strict equality:
        - _documents[_name].lastModified == uint64(0) (contracts/Utils/Documents.sol#37)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities
INFO:Detectors:
Reentrancy in HyperbolicAuction.commitTokensFrom(address,uint256,bool) (contracts/Auctions/HyperbolicAuction.sol#287-304):
        External calls:
        - _safeTransferFrom(paymentCurrency,_from,tokensToTransfer) (contracts/Auctions/HyperbolicAuction.sol#301)
                - (success,data) = token.call(abi.encodeWithSelector(0x23b872dd,from,address(this),amount)) (contracts/Utils/SafeTransfer.sol#72-76)
        State variables written after the call(s):
        - _addCommitment(_from,tokensToTransfer) (contracts/Auctions/HyperbolicAuction.sol#302)
                - status.commitmentsTotal = uint128(uint256(status.commitmentsTotal).add(_commitment)) (contracts/Auctions/HyperbolicAuction.sol#344)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1
```

HyperbolicAuction.finalize() is having nonReentrant guard so this Re-entrancy issue is false positive

MISOHelper.sol

```
INFO:Detectors:
MarketHelper.getCrowdsaleInfo(address).info (contracts/Helper/MISOHelper.sol#460) is a local variable never initialized
FarmHelper.getPools(address,address).userInfo (contracts/Helper/MISOHelper.sol#218) is a local variable never initialized
MarketHelper.getHyperbolicAuctionInfo(address).info (contracts/Helper/MISOHelper.sol#520) is a local variable never initialized
MarketHelper.getBatchAuctionInfo(address).info (contracts/Helper/MISOHelper.sol#500) is a local variable never initialized
FarmHelper.getUserPoolInfo(address,address,uint256).userInfo (contracts/Helper/MISOHelper.sol#257) is a local variable never initialized
MarketHelper.getDutchAuctionInfo(address).info (contracts/Helper/MISOHelper.sol#477) is a local variable never initialized
TokenHelper.getTokenInfo(address).info (contracts/Helper/MISOHelper.sol#86) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
Address.isContract(address) (contracts/OpenZeppelin/utils/Address.sol#26-35) uses assembly
        - INLINE ASM (contracts/OpenZeppelin/utils/Address.sol#33)
Address._verifyCallResult(bool,bytes,string) (contracts/OpenZeppelin/utils/Address.sol#171-188) uses assembly
        - INLINE ASM (contracts/OpenZeppelin/utils/Address.sol#180-183)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
```

Variables are initialized when the contract is deployed. Then, it is a false positive.

LiquidityZAP.sol

```
INFO:Detectors:
LiquidityZAP.getLPTokenPerEthUnit(uint256) (contracts/Liquidity/LiquidityZAP.sol#222-231) performs a multiplication on the result of a division:
        -(amount0,amount1) = (outTokens,ethAmt.div(2)) (contracts/Liquidity/LiquidityZAP.sol#228)
        -liquidity = SafeMathPlus.min(amount0.mul(_totalsupply) / _reserve0,amount1.mul(_totalsupply) / _reserve1) (contracts/Liquidity/LiquidityZAP.sol#230)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-multiply
INFO:Detectors:
SafeTransfer._safeTransfer(address,address,uint256) (contracts/Utils/SafeTransfer.sol#52-64) uses a dangerous strict equality:
        - require(bool)(success && (data.length == 0 || abi.decode(data,(bool)))) (contracts/Utils/SafeTransfer.sol#63)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities
INFO:Detectors:
UniswapV2Library.getAmountsOut(address,uint256,address[]).i (contracts/UniswapV2/UniswapV2Library.sol#66) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
LiquidityZAP.addLiquidityETHOnly(address) (contracts/Liquidity/LiquidityZAP.sol#117-134) ignores return value by _WETH.transfer(_tokenWETHPair,buyAmount) (contracts/Liquidity/LiquidityZAP.sol#127)
LiquidityZAP.removeLiquidity(address,address,uint256,address) (contracts/Liquidity/LiquidityZAP.sol#165-175) ignores return value by IUniswapV2Pair(_tokenWETHPair).transferFrom(msg.sender,_tokenWETHPair,liquidity) (contracts/Liquidity/LiquidityZAP.sol#171)
LiquidityZAP.removeLiquidityTokenOnly(address,uint256) (contracts/Liquidity/LiquidityZAP.sol#200-213) ignores return value by _WETH.transfer(_tokenWETHPair,amountETH) (contracts/Liquidity/LiquidityZAP.sol#207)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
```

Division and multiplication are in the correct order.
In that case, the use of strict equality is safe. (== 0).

PoolLiquidity02.sol

```
INFO:Detectors:
PoolLiquidity02._launchLiquidityPool() (contracts/Liquidity/PoolLiquidity02.sol#251-288) uses a dangerous strict equality:
        - token1Amount == 0 || token2Amount == 0 (contracts/Liquidity/PoolLiquidity02.sol#277)
SafeTransfer._safeTransfer(address,address,uint256) (contracts/Utils/SafeTransfer.sol#52-64) uses a dangerous strict equality:
        - require(bool)(success && (data.length == 0 || abi.decode(data,(bool)))) (contracts/Utils/SafeTransfer.sol#63)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities
INFO:Detectors:
Reentrancy in PoolLiquidity02.initPoolLiquidity(address,address,address,address,address,address,uint256,uint256,uint256,uint256,bool) (contracts/Liquidity/PoolLiquidity02.sol#111-148):
        External calls:
        - _setTokenPair() (contracts/Liquidity/PoolLiquidity02.sol#144)
                - tokenPair = factory.createPair(address(token1),address(token2)) (contracts/Liquidity/PoolLiquidity02.sol#301)
        State variables written after the call(s):
        - initialised = true (contracts/Liquidity/PoolLiquidity02.sol#145)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1
```

In that case, the use of strict equality is safe. (== 0).
Although Re-entrancy guard is not implemented in PoolLiquidity02.sol, intiPoolLiquidity() method can be called once only while setting the liquidity pool.

MISORecipe01.sol

```
INFO:Detectors:
MISORecipe01.prepareMiso(string,string,address) (contracts/Recipes/MISORecipe01.sol#117-200) ignores return value by token.approve(address(misoMarket),tokensToMarket) (contracts/Recipes/MISORecipe01.sol#129)
MISORecipe01.prepareMiso(string,string,address) (contracts/Recipes/MISORecipe01.sol#117-200) ignores return value by token.transfer(address(poolLiquidity),tokensToLiquidity) (contracts/Recipes/MISORecipe01.sol#154)
MISORecipe01.prepareMiso(string,string,address) (contracts/Recipes/MISORecipe01.sol#117-200) ignores return value by token.transfer(address(farm),tokensToFarm) (contracts/Recipes/MISORecipe01.sol#193)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
```

Ignore return value issue is already mentioned above in the report.

TokenVault.sol

```
INFO:Detectors:
TokenVault.lockTokens(address,uint256,uint256,address) (contracts/Vault/TokenVault.sol#53-82) ignores return value by userItem.lockedItemsWithUser.add(_tokenAddress) (contracts/Vault/TokenVault.sol#76)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
INFO:Detectors:
Reentrancy in TokenVault.lockTokens(address,uint256,uint256,address) (contracts/Vault/TokenVault.sol#53-82):
        External calls:
        - _safeTransferFrom(_tokenAddress,msg.sender,_amount) (contracts/Vault/TokenVault.sol#63)
                - (success,data) = token.call(abi.encodeWithSelector(0x23b872dd,from,address(this),amount)) (contracts/Utils/SafeTransfer.sol#72-76)
        State variables written after the call(s):
        - allDepositIds.push(_id) (contracts/Vault/TokenVault.sol#71)
        - _id = ++ depositId (contracts/Vault/TokenVault.sol#65)
        - lockedItem[_id].amount = _amount (contracts/Vault/TokenVault.sol#67)
        - lockedItem[_id].unlockTime = _unlockTime (contracts/Vault/TokenVault.sol#68)
        - lockedItem[_id].owner = _withdrawer (contracts/Vault/TokenVault.sol#69)
        - lockedItem[_id].userIndex = userIndex (contracts/Vault/TokenVault.sol#79)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
```

Ignore return value issue is already mentioned above in the report.

5.3 AUTOMATED SECURITY SCAN

Description:

Halborn used automated security scanners to assist with detection of well-known security issues, and to identify low-hanging fruit 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 testers machine and sent the compiled results to the analyzers to locate any vulnerabilities. In addition, security detections are only in scope.

Results:

Access/MISOAccessControls.sol

Report for Access/MISOAccessControls.sol
https://dashboard.mythx.io/#/console/analyses/4f2ca3fb-e487-493e-a111-1f1b8cf3b1b6

Line	SWC Title	Severity	Short Description
63	(SWC-000) Unknown	Medium	Function could be marked as external.
72	(SWC-000) Unknown	Medium	Function could be marked as external.
81	(SWC-000) Unknown	Medium	Function could be marked as external.

Access/MISOAccessFactory.sol

Report for Access/MISOAccessFactory.sol
https://dashboard.mythx.io/#/console/analyses/4f2ca3fb-e487-493e-a111-1f1b8cf3b1b6

Line	SWC Title	Severity	Short Description
85	(SWC-134) Message call with hardcoded gas amount	Low	Call with hardcoded gas amount.

Access/MISOAdminAccess.sol

Report for Access/MISOAdminAccess.sol
https://dashboard.mythx.io/#/console/analyses/4f2ca3fb-e487-493e-a111-1f1b8cf3b1b6

Line	SWC Title	Severity	Short Description
31	(SWC-000) Unknown	Medium	Function could be marked as external.
46	(SWC-000) Unknown	Medium	Function could be marked as external.

Access/PointListFactory.sol

Report for contracts/Access/PointListFactory.sol
https://dashboard.mythx.io/#/console/analyses/36a5ee4a-815e-49de-8c80-f71058008a21

Line	SWC Title	Severity	Short Description
64	(SWC-000) Unknown	Medium	Function could be marked as external.
77	(SWC-000) Unknown	Medium	Function could be marked as external.
85	(SWC-000) Unknown	Medium	Function could be marked as external.
96	(SWC-000) Unknown	Medium	Function could be marked as external.
106	(SWC-000) Unknown	Medium	Function could be marked as external.
118	(SWC-000) Unknown	Medium	Function could be marked as external.
118	(SWC-127) Arbitrary Jump with Function Type Variable	High	The caller can redirect execution to arbitrary bytecode locations.
128	(SWC-113) DoS with Failed Call	Medium	Multiple calls are executed in the same transaction.
134	(SWC-113) DoS with Failed Call	Medium	Multiple calls are executed in the same transaction.
136	(SWC-107) Reentrancy	Low	Read of persistent state following external call.
137	(SWC-107) Reentrancy	Low	Read of persistent state following external call.
138	(SWC-107) Reentrancy	Low	Read of persistent state following external call.
140	(SWC-134) Message call with hardcoded gas amount	Low	Call with hardcoded gas amount.
140	(SWC-107) Reentrancy	Low	Write to persistent state following external call.
140	(SWC-107) Reentrancy	Low	Read of persistent state following external call.
150	(SWC-000) Unknown	Medium	Function could be marked as external.

Auction/BatchAuction.sol

Report for contracts/Auctions/BatchAuction.sol
https://dashboard.mythx.io/#/console/analyses/ca813714-3746-49c1-80da-e94d8cebf71c

Line	SWC Title	Severity	Short Description
145	(SWC-000) Unknown	Medium	Function could be marked as external.
159	(SWC-000) Unknown	Medium	Function could be marked as external.
173	(SWC-000) Unknown	Medium	Function could be marked as external.
202	(SWC-107) Reentrancy	Low	Read of persistent state following external call.
204	(SWC-107) Reentrancy	Low	Read of persistent state following external call.
205	(SWC-107) Reentrancy	Low	Read of persistent state following external call.
208	(SWC-107) Reentrancy	Low	Write to persistent state following external call.
209	(SWC-107) Reentrancy	Low	Read of persistent state following external call.
238	(SWC-000) Unknown	Medium	Function could be marked as external.
261	(SWC-000) Unknown	Medium	Function could be marked as external.
310	(SWC-000) Unknown	Medium	Function could be marked as external.
422	(SWC-131) Presence of unused variables	Low	Unused function parameter "_data".
425	(SWC-000) Unknown	Medium	Function could be marked as external.

Auction/DutchAuction.sol

Report for contracts/Auctions/DutchAuction.sol
https://dashboard.mythx.io/#/console/analyses/86c451b9-0207-46ef-9d3b-69fc3b796748

Line	SWC Title	Severity	Short Description
213	(SWC-000) Unknown	Medium	Function could be marked as external.
227	(SWC-000) Unknown	Medium	Function could be marked as external.
247	(SWC-134) Message call with hardcoded gas amount	Low	Call with hardcoded gas amount.
255	(SWC-000) Unknown	Medium	Function could be marked as external.
294	(SWC-110) Assert Violation	Low	An assertion violation was triggered.
312	(SWC-000) Unknown	Medium	Function could be marked as external.
333	(SWC-000) Unknown	Medium	Function could be marked as external.
349	(SWC-000) Unknown	Medium	Function could be marked as external.
356	(SWC-000) Unknown	Medium	Function could be marked as external.
404	(SWC-000) Unknown	Medium	Function could be marked as external.
435	(SWC-000) Unknown	Medium	Function could be marked as external.
571	(SWC-131) Presence of unused variables	Low	Unused function parameter "_data".
575	(SWC-000) Unknown	Medium	Function could be marked as external.

Auction/HyperbolicAuction.sol

Report for contracts/Auctions/HyperbolicAuction.sol
https://dashboard.mythx.io/#/console/analyses/b8b9d0eb-2231-49f8-82ae-2b12038c0e78

Line	SWC Title	Severity	Short Description
134	(SWC-131) Presence of unused variables	Low	Unused function parameter "_factor".
196	(SWC-116) Timestamp Dependence	Low	A control flow decision is made based on The block.timestamp environment variable.
239	(SWC-000) Unknown	Medium	Function could be marked as external.
253	(SWC-000) Unknown	Medium	Function could be marked as external.
274	(SWC-134) Message call with hardcoded gas amount	Low	Call with hardcoded gas amount.
282	(SWC-000) Unknown	Medium	Function could be marked as external.
310	(SWC-000) Unknown	Medium	Function could be marked as external.
365	(SWC-000) Unknown	Medium	Function could be marked as external.
372	(SWC-000) Unknown	Medium	Function could be marked as external.
387	(SWC-000) Unknown	Medium	Function could be marked as external.
429	(SWC-000) Unknown	Medium	Function could be marked as external.
557	(SWC-131) Presence of unused variables	Low	Unused function parameter "_data".
564	(SWC-000) Unknown	Medium	Function could be marked as external.

Liquidity/LiquidityZAP.sol

Report for contracts/Liquidity/LiquidityZAP.sol
https://dashboard.mythx.io/#/console/analyses/5fd9f01c-38b5-4c8e-a19f-3d2ba7fca16e

Line	SWC Title	Severity	Short Description
1	(SWC-103) Floating Pragma	Low	A floating pragma is set.
92	(SWC-123) Requirement Violation	Low	Requirement violation.
121	(SWC-123) Requirement Violation	Low	Requirement violation.
160	(SWC-134) Message call with hardcoded gas amount	Low	Call with hardcoded gas amount.
172	(SWC-107) Reentrancy	Low	Read of persistent state following external call
172	(SWC-107) Reentrancy	Low	Write to persistent state following external call
172	(SWC-113) DoS with Failed Call	Low	Multiple calls are executed in the same transaction.
180	(SWC-131) Presence of unused variables	Low	Unused local variable "amountETH".
202	(SWC-131) Presence of unused variables	Low	Unused local variable "amountToken".
234	(SWC-107) Reentrancy	Low	Read of persistent state following external call.
235	(SWC-107) Reentrancy	Low	Read of persistent state following external call.
235	(SWC-113) DoS with Failed Call	Medium	Multiple calls are executed in the same transaction.

Liquidity/PoolLiquidity.sol

Report for contracts/Liquidity/PoolLiquidity.sol
https://dashboard.mythx.io/#/console/analyses/462f67d8-98ee-4c2c-ac8b-0005a91c9f32

Line	SWC Title	Severity	Short Description
126	(SWC-101) Integer Overflow and Underflow	High	The arithmetic operator can overflow.
129	(SWC-107) Reentrancy	Medium	Write to persistent state following external call
141	(SWC-116) Timestamp Dependence	Low	A control flow decision is made based on The block.timestamp environment variable.
153	(SWC-116) Timestamp Dependence	Low	A control flow decision is made based on The block.timestamp environment variable.
182	(SWC-116) Timestamp Dependence	Low	A control flow decision is made based on The block.timestamp environment variable.
183	(SWC-113) DoS with Failed Call	Low	Multiple calls are executed in the same transaction.
192	(SWC-116) Timestamp Dependence	Low	A control flow decision is made based on The block.timestamp environment variable.
199	(SWC-107) Reentrancy	Low	Read of persistent state following external call
220	(SWC-116) Timestamp Dependence	Low	A control flow decision is made based on The block.timestamp environment variable.
221	(SWC-116) Timestamp Dependence	Low	A control flow decision is made based on The block.timestamp environment variable.
266	(SWC-113) DoS with Failed Call	Low	Multiple calls are executed in the same transaction.
266	(SWC-107) Reentrancy	Medium	Read of persistent state following external call
266	(SWC-107) Reentrancy	Low	A call to a user-supplied address is executed.
292	(SWC-113) DoS with Failed Call	Low	Multiple calls are executed in the same transaction.
300	(SWC-107) Reentrancy	Low	Read of persistent state following external call

Liquidity/PoolLiquidity02.sol

Report for contracts/Liquidity/PoolLiquidity02.sol
https://dashboard.mythx.io/#/console/analyses/64740105-cc19-4cee-96f4-8852b1a4c301

Line	SWC Title	Severity	Short Description
142	(SWC-101) Integer Overflow and Underflow	High	The arithmetic operator can overflow.
146	(SWC-107) Reentrancy	Medium	Write to persistent state following external call
188	(SWC-116) Timestamp Dependence	Low	A control flow decision is made based on The block.timestamp environment variable.
217	(SWC-116) Timestamp Dependence	Low	A control flow decision is made based on The block.timestamp environment variable.
218	(SWC-113) DoS with Failed Call	Low	Multiple calls are executed in the same transaction.
226	(SWC-116) Timestamp Dependence	Low	A control flow decision is made based on The block.timestamp environment variable.
233	(SWC-107) Reentrancy	Low	Read of persistent state following external call
254	(SWC-116) Timestamp Dependence	Low	A control flow decision is made based on The block.timestamp environment variable.
255	(SWC-116) Timestamp Dependence	Low	A control flow decision is made based on The block.timestamp environment variable.
301	(SWC-107) Reentrancy	Low	A call to a user-supplied address is executed.
301	(SWC-113) DoS with Failed Call	Low	Multiple calls are executed in the same transaction.
301	(SWC-107) Reentrancy	Medium	Read of persistent state following external call
327	(SWC-113) DoS with Failed Call	Low	Multiple calls are executed in the same transaction.
335	(SWC-107) Reentrancy	Low	Read of persistent state following external call

Token/ERC20/ERC20Burnable.sol

Report for contracts/Tokens/ERC20/ERC20Burnable.sol
https://dashboard.mythx.io/#/console/analyses/7f45f2dc-b404-4cbc-924c-220b47a24ac4

Line	SWC Title	Severity	Short Description
3	(SWC-103) Floating Pragma	Low	A floating pragma is set.
18	(SWC-000) Unknown	Medium	Function could be marked as external.
33	(SWC-000) Unknown	Medium	Function could be marked as external.

Token/ERC20/ERC20Pausable.sol

```
Report for contracts/Tokens/ERC20/ERC20Pausable.sol
https://dashboard.mythx.io/#/console/analyses/a5ab1bd3-b120-4177-bd6d-9bf5df1d0a87
```

Line	SWC Title	Severity	Short Description
3	(SWC-103) Floating Pragma	Low	A floating pragma is set.

Token/FixedToken.sol

```
Report for contracts/Tokens/FixedToken.sol
https://dashboard.mythx.io/#/console/analyses/1e620a26-a714-4a32-bb76-1e903b58db9c
```

Line	SWC Title	Severity	Short Description
10	(SWC-131) Presence of unused variables	Low	Unused function parameter "_owner".
16	(SWC-000) Unknown	Medium	Function could be marked as external.

Token/MintableToken.sol

Report for contracts/Tokens/MintableToken.sol
https://dashboard.mythx.io/#/console/analyses/934d2a53-6173-45d1-b86e-5b70fef51c49

Line	SWC Title	Severity	Short Description
21	(SWC-000) Unknown	Medium	Function could be marked as external.
63	(SWC-000) Unknown	Medium	Function could be marked as external.
77	(SWC-000) Unknown	Medium	Function could be marked as external.
91	(SWC-000) Unknown	Medium	Function could be marked as external.

Token/SushiToken.sol

Report for contracts/Tokens/SushiToken.sol
https://dashboard.mythx.io/#/console/analyses/47256cd6-0acb-4b97-b98d-ec32befa9b27

Line	SWC Title	Severity	Short Description
26	(SWC-000) Unknown	Medium	Function could be marked as external.
59	(SWC-000) Unknown	Medium	Function could be marked as external.
199	(SWC-120) Weak Sources of Randomness from Chain Attributes	Low	Potential use of "block.number" as source of randonmness.
199	(SWC-120) Weak Sources of Randomness from Chain Attributes	Low	A control flow decision is made based on The block.number environment variable.
272	(SWC-120) Weak Sources of Randomness from Chain Attributes	Low	Potential use of "block.number" as source of randonmness.

Helper/MISOHelper.sol

Report for contracts/Helper/MISOHelper.sol
https://dashboard.mythx.io/#/console/analyses/8f089eeb-96c5-4b0d-8540-d69fd9b95d88

Line	SWC Title	Severity	Short Description
49	(SWC-128) DoS With Block Gas Limit	Low	Loop over unbounded data structure.
113	(SWC-000) Unknown	Medium	Function could be marked as external.
201	(SWC-128) DoS With Block Gas Limit	Low	Loop over unbounded data structure.
231	(SWC-000) Unknown	Medium	Function could be marked as external.
236	(SWC-128) DoS With Block Gas Limit	Low	Loop over unbounded data structure.
253	(SWC-000) Unknown	Medium	Function could be marked as external.
432	(SWC-000) Unknown	Medium	Function could be marked as external.
436	(SWC-128) DoS With Block Gas Limit	Low	Loop over unbounded data structure.
458	(SWC-000) Unknown	Medium	Function could be marked as external.
474	(SWC-000) Unknown	Medium	Function could be marked as external.
497	(SWC-000) Unknown	Medium	Function could be marked as external.
517	(SWC-000) Unknown	Medium	Function could be marked as external.
556	(SWC-000) Unknown	Medium	Function could be marked as external.

Utils/BoringBatchable.sol

Report for contracts/Utils/BoringBatchable.sol
https://dashboard.mythx.io/#/console/analyses/9ee9083f-5e5a-4de6-80c9-d315ff289e02

Line	SWC Title	Severity	Short Description
39	(SWC-104) Unchecked Call Return Value	Medium	Unchecked return value from external call.
47	(SWC-123) Requirement Violation	Low	Requirement violation.
52	(SWC-000) Unknown	Medium	Function could be marked as external.
62	(SWC-123) Requirement Violation	Low	Requirement violation.
62	(SWC-107) Reentrancy	Low	A call to a user-supplied address is executed.

Utils/BoringFactory.sol

Report for contracts/Utils/BoringFactory.sol
https://dashboard.mythx.io/#/console/analyses/d3e51921-f8c0-4cf3-a32a-fc0e10982949

Line	SWC Title	Severity	Short Description
7	(SWC-123) Requirement Violation	Low	Requirement violation.
20	(SWC-000) Unknown	Medium	Function could be marked as external.
49	(SWC-107) Reentrancy	Low	Read of persistent state following external call.
51	(SWC-123) Requirement Violation	Low	Requirement violation.
51	(SWC-113) DoS with Failed Call	Medium	Multiple calls are executed in the same transaction.
51	(SWC-107) Reentrancy	Low	Write to persistent state following external call.

Utils/BoringOwnable.sol

Report for contracts/Utils/BoringOwnable.sol
https://dashboard.mythx.io/#/console/analyses/10f96f75-b70b-42db-ba2f-df5c3acb0d3a

Line	SWC Title	Severity	Short Description
27	(SWC-000) Unknown	Medium	Function could be marked as external.
47	(SWC-000) Unknown	Medium	Function could be marked as external.

Utils/Documents.sol

Report for contracts/Utils/Documents.sol
https://dashboard.mythx.io/#/console/analyses/6c723d61-3aab-41c4-93db-be4652995190

Line	SWC Title	Severity	Short Description
21	(SWC-108) State Variable Default Visibility	Low	State variable visibility is not set.
81	(SWC-128) DoS With Block Gas Limit	Low	Implicit loop over unbounded data structure.

Utils/Owned.sol

Report for contracts/Utils/Owned.sol
https://dashboard.mythx.io/#/console/analyses/cbbd1354-1aca-40e4-94e5-b3398edf83aa

Line	SWC Title	Severity	Short Description
27	(SWC-000) Unknown	Medium	Function could be marked as external.
34	(SWC-000) Unknown	Medium	Function could be marked as external.
39	(SWC-000) Unknown	Medium	Function could be marked as external.

Utils/SafeMathPlus.sol

Report for contracts/Utils/SafeMathPlus.sol
https://dashboard.mythx.io/#/console/analyses/1f393386-1c78-44cd-a14e-a348698e1d5f

Line	SWC Title	Severity	Short Description
60	(SWC-128) DoS With Block Gas Limit	Low	Loop over unbounded data structure.

Utils/SafeTransfer.sol

Report for contracts/Utils/SafeTransfer.sol
https://dashboard.mythx.io/#/console/analyses/5b7b14fd-2e86-4f9c-854f-4eaa820be967

Line	SWC Title	Severity	Short Description
32	(SWC-134) Message call with hardcoded gas amount	Low	Call with hardcoded gas amount.

Utils/WETH9.sol

Report for contracts/Utils/WETH9.sol
https://dashboard.mythx.io/#/console/analyses/db535649-c502-4074-a303-1c247f83f581

Line	SWC Title	Severity	Short Description
38	(SWC-000) Unknown	Medium	Function could be marked as external.
41	(SWC-134) Message call with hardcoded gas amount	Low	Call with hardcoded gas amount.
45	(SWC-000) Unknown	Medium	Function could be marked as external.
49	(SWC-000) Unknown	Medium	Function could be marked as external.
55	(SWC-000) Unknown	Medium	Function could be marked as external.

Vault/GnosisSafeFactory.sol

Report for contracts/Vault/GnosisSafeFactory.sol
https://dashboard.mythx.io/#/console/analyses/912a14fd-4fe6-4fa9-af39-b60bd057677d

Line	SWC Title	Severity	Short Description
24	(SWC-108) State Variable Default Visibility	Low	State variable visibility is not set.
44	(SWC-000) Unknown	Medium	Function could be marked as external.
86	(SWC-000) Unknown	Medium	Function could be marked as external.

Vault/TokenVault.sol

Report for contracts/Vault/TokenVault.sol
https://dashboard.mythx.io/#/console/analyses/2c1d0d1d-7740-4e23-8568-0e3b9abc29a7

Line	SWC Title	Severity	Short Description
10	(SWC-123) Requirement Violation	Low	Requirement violation.
29	(SWC-108) State Variable Default Visibility	Low	State variable visibility is not set.
53	(SWC-000) Unknown	Medium	Function could be marked as external.
65	(SWC-107) Reentrancy	Low	Read of persistent state following external call.
67	(SWC-107) Reentrancy	Low	Write to persistent state following external call.
68	(SWC-107) Reentrancy	Low	Write to persistent state following external call.
69	(SWC-107) Reentrancy	Low	Read of persistent state following external call.
71	(SWC-107) Reentrancy	Low	Read of persistent state following external call.
77	(SWC-107) Reentrancy	Low	Read of persistent state following external call.
78	(SWC-107) Reentrancy	Low	Read of persistent state following external call.
79	(SWC-107) Reentrancy	Low	Write to persistent state following external call.
81	(SWC-107) Reentrancy	Low	Read of persistent state following external call.

MISOTokenFactory.sol

Report for contracts/MISOTokenFactory.sol
https://dashboard.mythx.io/#/console/analyses/8304b1cb-cae9-4f3a-9388-423fd99b8c40

Line	SWC Title	Severity	Short Description
21	(SWC-123) Requirement Violation	Low	Requirement violation.
158	(SWC-134) Message call with hardcoded gas amount	Low	Call with hardcoded gas amount.
161	(SWC-134) Message call with hardcoded gas amount	Low	Call with hardcoded gas amount.
213	(SWC-123) Requirement Violation	Low	Requirement violation.
232	(SWC-128) DoS With Block Gas Limit	Low	Implicit loop over unbounded data structure.

THANK YOU FOR CHOOSING

// HALBORN