Sector Finance

Smart Contract Security Audit

Prepared by: **Halborn**

Date of Engagement: **November 7th, 2022 - December 12th, 2022**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	11/07/2022	Grzegorz Trawinski
0.2	Document Update	11/16/2022	Grzegorz Trawinski
0.3	Document Update	12/09/2022	Grzegorz Trawinski
0.4	Document Update	12/09/2022	Luis Arroyo
0.5	Draft Review	12/12/2022	Roberto Reigada
0.6	Draft Review	12/12/2022	Piotr Cielas
0.7	Draft Review	12/12/2022	Gabi Urrutia
1.0	Remediation Plan	12/16/2022	Grzegorz Trawinski
1.1	Remediation Plan Review	12/16/2022	Roberto Reigada
1.2	Remediation Plan Review	12/16/2022	Piotr Cielas
1.3	Remediation Plan Review	12/16/2022	Gabi Urrutia

CONTACTS

CONTACT	COMPANY	EMAIL
Rob Behnke	Halborn	Rob.Behnke@halborn.com
Steven Walbroehl	Halborn	Steven.Walbroehl@halborn.com
Gabi Urrutia	Halborn	Gabi.Urrutia@halborn.com
Piotr Cielas	Halborn	Piotr.Cielas@halborn.com
Kubilay Onur Gungor	Halborn	Kubilay.Gungor@halborn.com
Grzegorz Trawinski	Halborn	Grzegorz.Trawinski@halborn.com
Luis Arroyo	Halborn	Luis.Arroyo@halborn.com
Roberto Reigada	Halborn	Roberto.Reigada@halborn.com

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

Sector Finance is an on-chain, permissionless hedge fund/farm that offers high-yield market-neutral investment strategies.

Sector Finance engaged Halborn to conduct a security audit on their smart contracts beginning on November 7th, 2022 and ending on December 12th, 2022 . The security assessment was scoped to the smart contracts provided to the Halborn team.

1.2 AUDIT SUMMARY

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

The purpose of this audit is to:

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

In summary, Halborn identified some security risks that were addressed by the Sector Finance team.

1.3 TEST APPROACH & METHODOLOGY

Halborn performed a combination of manual and automated security testing to balance efficiency, timeliness, practicality, and accuracy in regard to the scope of this audit. While manual testing is recommended to uncover flaws in logic, process, and implementation; automated testing techniques help enhance coverage of the bridge code and can quickly identify items

that do not follow security best practices. The following phases and associated tools were used throughout the term of the audit:

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

RISK METHODOLOGY:

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

RISK SCALE - LIKELIHOOD

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

RISK SCALE - IMPACT

5 - May cause devastating and unrecoverable impact or loss.
4 - May cause a significant level of impact or loss.
3 - May cause a partial impact or loss to many.
2 - May cause temporary impact or loss.
1 - May cause minimal or un-noticeable impact.

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

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL

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

1.4 SCOPE

IN-SCOPE:

The security assessment was scoped to the following sector-contracts:

- SectorRegistry.sol
- SectorTimelock.sol
- Accounting.sol
- Auth.sol
- AuthU.sol
- Fees.sol
- FeesU.sol
- StratAuth.sol
- SafeETH.sol
- UniUtils.sol
- CompMultiFarm.sol
- Compound.sol
- CompoundFarm.sol
- MasterChefFarm.sol
- MiniChef2Farm.sol
- MiniChefFarm.sol
- StarChefFarm.sol
- HLPCore.sol
- MasterChefCompMulti.sol
- IMX.sol
- IMXCore.sol
- IMXFarm.sol
- IBase.sol
- ICompound.sol
- IFarmable.sol
- IIMXFarm.sol
- ILending.sol
- ILp.sol
- IUniFarm.sol
- IUniLp.sol
- ERC4626
- BatchedWithdraw.sol

- ERC4626.sol
- SectorBase.sol
- ERC5115
- SCYBase.sol
- SCYStrategy.sol
- SCYVault.sol
- AggregatorVault.sol
- HLPVault.sol
- IMXLend.sol
- IMXVault.sol
- Stargate.sol
- Synapse.sol

Commit ID: 24f5bf391c3c2a33eed0fbbe3ad07307724a9ef9

In addition, the Sector Finance team provided the following fix commit IDs:
- on November 16, Commit ID: 73b990d9e856c027c45926835db780f93a63a4d9
- on December 5, Commit ID: 46a73e34fd1b9c0e81cd671e4b10253e4db43f3d

OUT-OF-SCOPE:
Other smart contracts in the repository, external libraries and economical attacks.

2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
7	2	2	2	2

LIKELIHOOD

IMPACT

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

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
HAL-01 - REDEEM NOT POSSIBLE DUE TO INTEGER UNDERFLOW	Critical	SOLVED - 11/15/2022
HAL-02 - MULTIPLE REQUESTREDEEM CALLS MAY RETURN AN UNDERESTIMATED AMOUNT OF TOKENS	Critical	SOLVED - 11/15/2022
HAL-03 - EMERGENCYREDEEM MAY RETURN AN UNDERESTIMATED AMOUNT OF TOKENS	Critical	SOLVED - 11/15/2022
HAL-04 - PROFIT THEFT POSSIBLE VIA DEPOSIT IN SCYVAULT	Critical	SOLVED - 11/30/2022
HAL-05 - BROKEN IMX CALCULATIONS CAUSE DENIAL OF SERVICE	Critical	SOLVED - 12/06/2022
HAL-06 - GETANDUPDATETVL REVERTS DUE TO INTEGER UNDERFLOW IN IMX	Critical	SOLVED - 12/06/2022
HAL-07 - CLOSEPOSITION IN THE HLP STRATEGY REVERTS WHEN SHORTBALANCE > SHORTPOSITION	Critical	SOLVED - 12/16/2022
HAL-08 - STRATEGYTVL IN THE SYNAPSE SOLUTION REVERTS FOR EMPTY DEPOSIT	High	SOLVED - 11/30/2022
HAL-09 - GETMAXTVL MAY REVERT DUE TO INTEGER UNDERFLOW IN IMX	High	SOLVED - 12/06/2022
HAL-10 - PROFIT HARVESTING IN THE SYNAPSE SOLUTION MAY REVERT	Medium	SOLVED - 12/16/2022
HAL-11 - THE MINT FUNCTION LACKS LOCK FOR MINIMUM LIQUIDITY	Medium	SOLVED - 12/06/2022
HAL-12 - LOANHEALTH MAY REVERT DUE TO INTEGER UNDERFLOW IN IMX	Low	SOLVED - 12/16/2022
HAL-13 - EMERGENCYACTION IS NOT MARKED AS PAYABLE	Low	SOLVED - 12/16/2022
HAL-14 - UNNEEDED INITIALIZATION OF UINT256 VARIABLES TO 0	Informational	SOLVED - 12/16/2022
HAL-15 - GAS OVER-CONSUMPTION IN LOOPS	Informational	SOLVED - 12/16/2022

FINDINGS & TECH DETAILS

3.1 (HAL-01) REDEEM NOT POSSIBLE DUE TO INTEGER UNDERFLOW - CRITICAL

Description:

The BatchedWithdraw contract allows asset tokens to be redeemed in a two-step process with the requestRedeem() and redeem() functions. The assessment revealed that multiple subsequent redeems can lock solutions in a state where no more redeems could be possible. This happens because the shares property in the WithdrawRecord structure is not reset after finalizing a redeem, and in fact, the property is accumulating redeemed values. As a result, the subtraction pendingRedeem -= shares; causes integer underflow, disallowing further asset redeeming. It must be noted that the cancelRedeem() function reverts for the same reason.

Listing 1: BatchedWithdraw.sol (Lines 53,56)

```
45     function _requestRedeem(
46         uint256 shares,
47         address owner,
48         address redeemer
49     ) internal {
50         _transfer(owner, address(this), shares);
51         WithdrawRecord storage withdrawRecord = withdrawLedger[
 ↳ redeemer];
52         withdrawRecord.timestamp = block.timestamp;
53         withdrawRecord.shares += shares;
54         uint256 value = convertToAssets(shares);
55         withdrawRecord.value = value;
56         pendingRedeem += shares;
57         emit RequestWithdraw(msg.sender, owner, shares);
58     }
```

Listing 2: BatchedWithdraw.sol (Lines 126,130)

```
111         function _redeem(address account) internal returns (uint256
 ↳  amountOut, uint256 shares) {
112         WithdrawRecord storage withdrawRecord = withdrawLedger[
 ↳ account];
113
```

```
114         if (withdrawRecord.value == 0) revert ZeroAmount();
115         if (withdrawRecord.timestamp >= lastHarvestTimestamp)
 ↳ revert NotReady();
116
117         shares = withdrawRecord.shares;
118         // value of shares at time of redemption request
119         uint256 redeemValue = withdrawRecord.value;
120
121         // actual amount out is the smaller of currentValue and
 ↳ redeemValue
122         amountOut = _getWithdrawAmount(shares, redeemValue);
123
124         // update total pending withdraw
125         // pendingWithdraw -= redeemValue;
126         pendingRedeem -= shares;
127
128         // important pendingRedeem should update prior to
 ↳ beforeWithdraw call
129         beforeWithdraw(amountOut, shares);
130         withdrawRecord.value = 0;
131         _burn(address(this), shares);
132     }
```

Proof of Concept:

1. All necessary contracts are deployed: AggregatorVault, XChainLib, WETH, MockERC20.
2. As user_1 deposit() 10^{18} asset tokens.
3. As user_1 requestRedeem() of $0,5 * 10^{18}$ asset tokens.
4. Forward chain time for 1 second.
5. As a vault's manager harvest().
6. As user_1 redeem() asset tokens.
7. As user_1 requestRedeem() of $0,5 * 10^{18}$ asset tokens.
8. Forward chain time for 1 second.
9. As a vault's manager harvest().
10. As user_1 attempt to redeem() asset tokens. Observe that transaction reverts with Integer overflow error message.
11. As user_1 attempt to cancelRedeem(). Observe that transaction reverts with Integer overflow error message.

```
aggregatorVault.withdrawLedger(user1) (1668680820, 10000000000000000000, 5000000000000000000)
wETHMock.balanceOf(user1) 50,000,000,000,000,000,000
aggregatorVault.balanceOf(user1) 0
aggregatorVault.underlyingBalance(user1) 0
aggregatorVault.pendingRedeem() 50,000,000,000,000,000,000
aggregatorVault.redeem({'from': user1})
Transaction sent: 0x79026f261aa7e0639a1c15a31d87c5aed4861ab3f3cd949db3463cdbdd9c2ab5
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 22
  AggregatorVault.redeem confirmed (Integer overflow)   Block: 91   Gas used: 29325 (0.24%)

aggregatorVault.withdrawLedger(user1) (1668680820, 10000000000000000000, 5000000000000000000)
wETHMock.balanceOf(user1) 50,000,000,000,000,000,000
aggregatorVault.balanceOf(user1) 0
aggregatorVault.underlyingBalance(user1) 0
aggregatorVault.pendingRedeem() 50,000,000,000,000,000,000
aggregatorVault.cancelRedeem({'from': user1})
Transaction sent: 0x47a565715697c689db614d25a001423de4cfdac004dbd23a1a053100c48e9388
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 23
  AggregatorVault.cancelRedeem confirmed (Integer overflow)   Block: 92   Gas used: 31715 (0.26%)
```

Risk Level:

Likelihood - 5
Impact - 5

Recommendation:

It is recommended to reset the shares property in the WithdrawRecord structure after finalizing redeem.

Remediation Plan:

SOLVED: The Sector Finance team solved this issue in commit 32498d69daeb3437b3d19bf981e2c9ce5d4f55cc: the shares property in the WithdrawRecord structure is now reset after finalizing redeem.

3.2 (HAL-02) MULTIPLE REQUESTREDEEM CALLS MAY RETURN AN UNDERESTIMATED AMOUNT OF TOKENS - CRITICAL

Description:

The BatchedWithdraw contract allows redeeming asset tokens in a two-step process, with the requestRedeem() and redeem() functions. The assessment revealed that multiple subsequent calls to the requestRedeem() function made before calling the redeem() function can result in underestimated amount of tokens returned. This happens because the values property in the WithdrawRecord structure does not accumulate over subsequent calls to the requestRedeem() function, but it is overwritten. As a result, the final call to the redeem() function returns fewer tokens than expected.

Listing 3: BatchedWithdraw.sol (Line 55)

```
45     function _requestRedeem(
46         uint256 shares,
47         address owner,
48         address redeemer
49     ) internal {
50         _transfer(owner, address(this), shares);
51         WithdrawRecord storage withdrawRecord = withdrawLedger[
↳ redeemer];
52         withdrawRecord.timestamp = block.timestamp;
53         withdrawRecord.shares += shares;
54         uint256 value = convertToAssets(shares);
55         withdrawRecord.value = value;
56         pendingRedeem += shares;
57         emit RequestWithdraw(msg.sender, owner, shares);
58     }
```

Proof of Concept:

1. All necessary contracts are deployed: AggregatorVault, XChainLib, WETH, MockERC20.

2. As user_1 deposit() 10^{18} asset tokens.
3. As user_1 requestRedeem() of $0,5 * 10^{18}$ asset tokens.
4. As user_1 requestRedeem() of $0,5 * 10^{18}$ asset tokens.
5. Forward chain time for 1 second.
6. As a vault's manager harvest().
7. As user_1 redeem() asset tokens. Observe that transaction finishes successfully. Note that user received only $0,5 * 10^{18}$ asset tokens.

```
aggregatorVault.withdrawLedger(user1) (1669719051, 0, 0)
aggregatorVault.balanceOf(user1) 100,000,000,000,000,000,000
aggregatorVault.underlyingBalance(user1) 100,000,000,000,000,000,000
aggregatorVault.pendingRedeem() 0
aggregatorVault.requestRedeem(shares / 2, {'from': user1})
Transaction sent: 0x1f73f1adc7f6154211a1d3ffd94aab717b2f3a3281c87fed347fd84c03451644
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 85
  AggregatorVault.requestRedeem confirmed   Block: 318   Gas used: 118033 (0.98%)

aggregatorVault.withdrawLedger(user1) (1669719051, 5000000000000000000, 5000000000000000000)
aggregatorVault.balanceOf(user1) 50,000,000,000,000,000,000
aggregatorVault.underlyingBalance(user1) 50,000,000,000,000,000,000
aggregatorVault.pendingRedeem() 50,000,000,000,000,000,000
aggregatorVault.requestRedeem(shares / 2, {'from': user1})
Transaction sent: 0x4add626979f685d7f38e3e57627ee4c6569923377b72213446e95889c16c62ba
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 86
  AggregatorVault.requestRedeem confirmed   Block: 319   Gas used: 38833 (0.32%)

aggregatorVault.withdrawLedger(user1) (1669719051, 10000000000000000000, 5000000000000000000)
wETHMock.balanceOf(user1) 0
aggregatorVault.balanceOf(user1) 0
aggregatorVault.underlyingBalance(user1) 0
aggregatorVault.pendingRedeem() 100,000,000,000,000,000,000
chain.sleep(1), chain.mine(1)
aggregatorVault.harvest(expectedTvl, maxDelta)
Transaction sent: 0xce95a13db04dcec328b40974fa53e4ff6d36519967eeac3e454a254f6bf49351
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 203
  AggregatorVault.harvest confirmed   Block: 321   Gas used: 56691 (0.47%)

aggregatorVault.withdrawLedger(user1) (1669719051, 10000000000000000000, 5000000000000000000)
wETHMock.balanceOf(user1) 0
aggregatorVault.balanceOf(user1) 0
aggregatorVault.underlyingBalance(user1) 0
aggregatorVault.pendingRedeem() 100,000,000,000,000,000,000
aggregatorVault.redeem({'from': user1})
Transaction sent: 0xa0b5eaa4fefa1f3f0709a3ad6ce708ba06b5fbcc04ee633791951622c19e8bd4
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 87
  AggregatorVault.redeem confirmed   Block: 322   Gas used: 53336 (0.44%)

aggregatorVault.withdrawLedger(user1) (1669719051, 10000000000000000000, 0)
wETHMock.balanceOf(user1) 50,000,000,000,000,000,000
aggregatorVault.balanceOf(user1) 0
aggregatorVault.underlyingBalance(user1) 0
aggregatorVault.pendingRedeem() 0
```

Risk Level:

Likelihood - 5

Impact - 5

Recommendation:

It is recommended to accumulate the `value` property in the `WithdrawRecord` structure in the subsequent `requestRedeem()` function call.

Remediation Plan:

SOLVED: The Sector Finance team solved this issue in commit `32498d69daeb3437b3d19bf981e2c9ce5d4f55cc`: the `value` field in the `WithdrawRecord` structure is being accumulated on subsequent calls to the `requestRedeem()` function.

3.3 (HAL-03) EMERGENCYREDEEM MAY RETURN AN UNDERESTIMATED AMOUNT OF TOKENS - CRITICAL

Description:

The emergencyRedeem() function within the AggregatorVault.sol contract allows a user to perform an immediate redeem of their asset tokens. However, in a certain situation, the function may return an underestimated amount of tokens. This happens due to the fact a user calls the requestRedeem() function prior to calling the emergencyRedeem() function. Then, the user record contains decreased amount of shares, which has an impact on subtraction done for calculating the underlyingShare parameter.

After consulting the issue with the Sector Finance team, it turned out that the burn operation was being done too early, before the actual accounting calculation.

Listing 4: AggregatorVault.sol (Lines 194-196)

```
180     /// @dev this method allows direct redemption of shares in
 ↳ exchange for
181     /// a portion of the float amount + a portion of all the
 ↳ strategy shares the vault holds
182     /// deposits are paused when we are in the emergency redeem
 ↳ state
183     function emergencyRedeem() public {
184         if (block.timestamp - lastHarvestTimestamp <
 ↳ maxHarvestInterval) revert RecentHarvest();
185         uint256 _totalSupply = totalSupply();
186         uint256 shares = balanceOf(msg.sender);
187         if (shares == 0) return;
188         _burn(msg.sender, shares);
189
190         // redeem proportional share of vault's underlying float
 ↳ balance
191         // (minus pendingWithdraw)
192         uint256 pendingWithdraw = convertToAssets(pendingRedeem);
193         if (floatAmnt > pendingWithdraw) {
```

```
194             uint256 availableFloat = floatAmnt - pendingWithdraw;
195             uint256 underlyingShare = (availableFloat * shares) /
 ↳ _totalSupply;
196             beforeWithdraw(underlyingShare, 0);
197             asset.safeTransfer(msg.sender, underlyingShare);
198         }
199
200         uint256 l = strategyIndex.length;
201
202         // redeem proportional share of each strategy
203         for (uint256 i; i < l; ++i) {
204             ERC20 stratToken = ERC20(strategyIndex[i]);
205             uint256 balance = stratToken.balanceOf(address(this));
206             uint256 userShares = (shares * balance) / _totalSupply
 ↳ ;
207             if (userShares == 0) continue;
208             stratToken.safeTransfer(msg.sender, userShares);
209         }
210     }
```

Proof of Concept:

1. All necessary contracts are deployed: AggregatorVault, XChainLib, WETH, MockERC20.
2. As user_1 deposit() 10^{18} asset tokens.
3. As user_1 requestRedeem() of $0,5 * 10^{18}$ asset tokens.
4. As user_1 cancelRedeem().
5. Forward chain time for 1 second.
6. As a vault's manager harvest().
7. As user_1 requestRedeem() of $0,5 * 10^{18}$ asset tokens.
8. Forward chain time for 1 second.
9. As a vault's manager harvest().
10. As user_1 call the emergencyRedeem() function. Observe that transaction finishes successfully. Note that user received only 999 asset tokens.

```
aggregatorVault.withdrawLedger(user1) (1669719345, 5000000000000000000, 0)
wETHMock.balanceOf(user1) 0
wETHMock.balanceOf(aggregatorVault) 100,000,000,000,000,000,000
aggregatorVault.balanceOf(user1) 9999999999999999000
aggregatorVault.balanceOf(aggregatorVault) 0
aggregatorVault.underlyingBalance(user1) 100,000,000,000,000,000,000
aggregatorVault.pendingRedeem() 0
aggregatorVault.floatAmnt() 100,000,000,000,000,000,000
aggregatorVault.totalSupply() 100,000,000,000,000,000,000
aggregatorVault.convertToAssets(aggregatorVault.pendingRedeem()) 0
aggregatorVault.requestRedeem(shares / 2, {'from': user1})
Transaction sent: 0xf96a9a29ddeefbcbaa518c509e31844ed78819c9dc838c992c994cc9d2815b81
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 97
  AggregatorVault.requestRedeem confirmed   Block: 357   Gas used: 107233 (0.89%)

aggregatorVault.withdrawLedger(user1) (1669719347, 10000000000000000000, 5000000000000000000)
wETHMock.balanceOf(user1) 0
wETHMock.balanceOf(aggregatorVault) 100,000,000,000,000,000,000
aggregatorVault.balanceOf(user1) 4999999999999999000
aggregatorVault.balanceOf(aggregatorVault) 50,000,000,000,000,000,000
aggregatorVault.underlyingBalance(user1) 50,000,000,000,000,000,000
aggregatorVault.pendingRedeem() 50,000,000,000,000,000,000
aggregatorVault.floatAmnt() 100,000,000,000,000,000,000
aggregatorVault.totalSupply() 100,000,000,000,000,000,000
aggregatorVault.convertToAssets(aggregatorVault.pendingRedeem()) 50,000,000,000,000,000,000
chain.sleep(1), chain.mine(1)
aggregatorVault.harvest(expectedTvl, maxDelta)
Transaction sent: 0x69bb310ab409d05d200c4053b71b6a7a46629e2148658a981c75ef90d751f035
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 227
  AggregatorVault.harvest confirmed   Block: 359   Gas used: 54817 (0.46%)

aggregatorVault.withdrawLedger(user1) (1669719347, 10000000000000000000, 5000000000000000000)
wETHMock.balanceOf(user1) 0
wETHMock.balanceOf(aggregatorVault) 100,000,000,000,000,000,000
aggregatorVault.balanceOf(user1) 4999999999999999000
aggregatorVault.balanceOf(aggregatorVault) 50,000,000,000,000,000,000
aggregatorVault.underlyingBalance(user1) 50,000,000,000,000,000,000
aggregatorVault.pendingRedeem() 50,000,000,000,000,000,000
aggregatorVault.floatAmnt() 100,000,000,000,000,000,000
aggregatorVault.totalSupply() 100,000,000,000,000,000,000
aggregatorVault.convertToAssets(aggregatorVault.pendingRedeem()) 50,000,000,000,000,000,000
aggregatorVault.emergencyRedeem({'from': user1})
Transaction sent: 0xb853ba6e4f3ebb80b844e9a71baf2b844a63f91d32a784861daf1cbbff6bff6d
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 98
  AggregatorVault.emergencyRedeem confirmed   Block: 361   Gas used: 85347 (0.71%)

aggregatorVault.withdrawLedger(user1) (1669719347, 10000000000000000000, 5000000000000000000)
wETHMock.balanceOf(user1) 999
wETHMock.balanceOf(aggregatorVault) 100,000,000,000,000,000,000
aggregatorVault.balanceOf(user1) 0
aggregatorVault.balanceOf(aggregatorVault) 50,000,000,000,000,000,000
aggregatorVault.underlyingBalance(user1) 0
aggregatorVault.pendingRedeem() 50,000,000,000,000,000,000
aggregatorVault.floatAmnt() 100,000,000,000,000,000,000
aggregatorVault.totalSupply() 50,000,000,000,000,000,000
aggregatorVault.convertToAssets(aggregatorVault.pendingRedeem()) 100,000,000,000,000,000,000
```

Risk Level:

Likelihood - 5

Impact - 5

Recommendation:

It is recommended to adjust the order of operations done within the `emergencyRedeem()` function.

Remediation Plan:

SOLVED: The Sector Finance team solved this issue in commit `32498d69daeb3437b3d19bf981e2c9ce5d4f55cc`: the `emergencyRedeem()` function now returns a correct number of tokens.

3.4 (HAL-04) PROFIT THEFT POSSIBLE VIA DEPOSIT IN SCYVAULT - CRITICAL

Description:

The SCYVault solution offers the opportunity to deposit Ether in the vault. The process is two-step: first, the Ether is transferred to the contract, secondly, the deposit() function is called to account for the Ether transferred in the first step, based on the contract's balance. The solution is designed to be used by other smart contracts; therefore, the two-step process should be done within a single transaction.
However, the assessment revealed that it is possible to abuse the deposit() function to steal the profits from the vault. The vault must be configured that it supports native token (acceptsNativeToken=True) and the yield token is set to native (yieldToken=NATIVE).
When the vault obtains the profit (yield tokens) in Ether it directly increases the contract's balance. Finally, the attacker can call the deposit() function to collect the shares based on the contract's balance and then redeem the liquidity.

Listing 5: SCYVault.sol (Lines 98,110)

```
97      function _depositNative() internal override {
98          uint256 balance = address(this).balance;
99          IWETH(address(underlying)).deposit{ value: balance }();
100         if (sendERC20ToStrategy) IERC20(underlying).safeTransfer(
 ↳ strategy, balance);
101     }
102
103     function _deposit(
104         address,
105         address token,
106         uint256 amount
107     ) internal override isInitialized returns (uint256 sharesOut)
 ↳ {
108         // if we have any float in the contract we cannot do
 ↳ deposit accounting
109         if (uBalance > 0) revert DepositsPaused();
110         if (token == NATIVE) _depositNative();
```

```
111         uint256 yieldTokenAdded = _stratDeposit(amount);
112         sharesOut = toSharesAfterDeposit(yieldTokenAdded);
113         vaultTvl += amount;
114     }
```

Listing 6: SCYBase.sol (Lines 52,60)

```
44 function deposit(
45         address receiver,
46         address tokenIn,
47         uint256 amountTokenToPull,
48         uint256 minSharesOut
49     ) external payable nonReentrant returns (uint256
↳ amountSharesOut) {
50         require(isValidBaseToken(tokenIn), "SCY: Invalid tokenIn")
↳ ;
51 
52         if (tokenIn == NATIVE && amountTokenToPull != 0) revert
↳ CantPullEth();
53         else if (amountTokenToPull != 0) _transferIn(tokenIn, msg.
↳ sender, amountTokenToPull);
54 
55         // this depends on strategy
56         // this supports depositing directly into strategy to save
↳  gas
57         uint256 amountIn = getFloatingAmount(tokenIn);
58         if (amountIn == 0) revert ZeroAmount();
59 
60         amountSharesOut = _deposit(receiver, tokenIn, amountIn);
61         if (amountSharesOut < minSharesOut) revert InsufficientOut
↳ (amountSharesOut, minSharesOut);
62 
63         // lock minimum liquidity if totalSupply is 0
64         if (totalSupply() == 0) {
65             if (MIN_LIQUIDITY > amountSharesOut) revert
↳ MinLiquidity();
66             amountSharesOut -= MIN_LIQUIDITY;
67             _mint(address(1), MIN_LIQUIDITY);
68         }
69 
70         _mint(receiver, amountSharesOut);
71         emit Deposit(msg.sender, receiver, tokenIn, amountIn,
↳ amountSharesOut);
72     }
```

Proof of Concept:

1. All necessary contracts are deployed: MockScyVault, XChainLib, WETH.
2. Configure MockScyVault that acceptsNativeToken=True and yieldToken =NATIVE.
3. As owner deposit() minimum liquidity asset tokens.
4. As user_1 deposit() $10 * 10^{18}$ asset tokens.
5. As user_2 deposit() $10 * 10^{18}$ asset tokens.
6. Observe that both users have $10 * 10^{18}$ of shares.

```
mockScyVault_1.balanceOf(user2) 10,000,000,000,000,000,000
mockScyVault_1.balanceOf(user1) 10,000,000,000,000,000,000
```

6. Simulate profit transfer to the vault by transferring 10^{18} ethereum tokens to the vault.
7. As user_1 call deposit() function, as input provide tokenIn=0x0 and amountTokenToPull=0. Observe that vault balance of user_1 increased.

```
Simulate vault profit in NATIVE (=yieldToken)
---
owner.transfer(mockScyVault_1, profit)
Transaction sent: 0x847c1183a8cab415998424fc3311b46ba91bd589231d251e6942ca0ea5a22b73
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 17
  Transaction confirmed   Block: 31   Gas used: 21055 (0.18%)

mockScyVault_1.deposit(user1, ZERO_ADDRESS, 0, 0, {'from': user1})
Transaction sent: 0xfe3773e8f0c48fcb747c8d6ebd0402d354fa1569628fb0cac8c246dc4dd18107
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 10
  MockScyVault.deposit confirmed   Block: 32   Gas used: 118633 (0.99%)
```

8. Observe that user_1 got additional 10^{18} of shares

```
mockScyVault_1.balanceOf(user2) 10,000,000,000,000,000,000
mockScyVault_1.balanceOf(user1) 11,000,000,000,000,000,000
```

7. As user_1 call redeem() function. Observe that user_1 balance increased with 10^{18} tokens.

```
---
Transaction sent: 0x1c01e67e1a1f47330c15dfd94e9ec988fba75707788d6ca840d8bc5733b744d4
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 5
  MockScyVault.redeem confirmed   Block: 16   Gas used: 117621 (0.98%)

Transaction sent: 0x01987a9254c55f68c9484e983ab385c6fe2632b0d35713a6d17d7c527f3e7874
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 1
  MockScyVault.redeem confirmed   Block: 17   Gas used: 117609 (0.98%)

---
user1.balance() 1,001,000,000,000,000,000,000
user2.balance() 1,000,000,000,000,000,000,000
```

Risk Level:

Likelihood - 5
Impact - 5

Recommendation:

It is recommended that the `_depositNative()` function implementation should refer to `msg.value` instead of the contract's balance.

Remediation Plan:

SOLVED: The Sector Finance team solved this issue in commit `77ccf856b7730ce3c5923ae318e817a7e69ba292`: the `_depositNative()` function now uses the `msg.value` instead of the contract balance.

3.5 (HAL-05) BROKEN IMX CALCULATIONS CAUSE DENIAL OF SERVICE - CRITICAL

Description:

The IMX solution is a vault basing on the Impermax lending services and yield farming. The assessment revealed that due to broken implementation of arithmetic operations within the IMXCore.sol contract, the solution breaks after first deposit.
Assuming the first deposit was done with 1 ETH, any subsequent deposits with amounts equal or higher than that is not possible (but a significantly smaller amount e.g. 0.1 ETH can be deposited), such transactions reverts with ERC20: transfer amount exceeds balance or STRAT: OVER_MAX_TVL message error. Also, afterwards, following functions revert with various error messages:
- harvest() with Integer overflow
- depositIntoStrategy() with ILM
- closePosition() with ILB

The above functions revert with the error message depending on the time elapsed between transactions.

Proof of Concept:

1. Fork Optimism blockchain with block number set to 45075376.
2. Deploy the IXMVault vault with IXM strategy using ETH-USDC-Tarot-Velo configuration prepared in strategies.json file.

Listing 7: strategies.json

```
107 "ETH-USDC-Tarot-Velo": {
108     "a1_underlying": "0x4200000000000000000000000000000000000006",
109     "a2_acceptsNativeToken": true,
110     "b_short": "0x7F5c764cBc14f9669B88837ca1490cCa17c31607",
111     "c0_uniPair": "0x79c912FEF520be002c2B6e57EC4324e260f38E50",
112     "c1_pairRouter": "0xa132DAB612dB5cB9fC9Ac426A0Cc215A3423F9c9",
113     "d_poolToken": "0x28fA49da55A54F02Bc4ff103adbd843B50B556F4",
```

```
114      "e_farmToken": "0x3c8B650257cFb5f272f799F5e2b4e65093a11a05",
115      "f_farmRouter": "0xa132DAB612dB5cB9fC9Ac426A0Cc215A3423F9c9",
116      "h_harvestPath": [
117        "0x3c8B650257cFb5f272f799F5e2b4e65093a11a05"
118      ],
119      "x_chain": "OP"
120    }
```

3. As user_1 deposit 1 WETH tokens to the IXMVault.
4. As user_1 attempt to deposit 1 WETH tokens to the IXMVault once again. Observe the error message.

```
 1 ether
Transaction sent: 0x8a6ebee0a2b19a04577ae30aaf2665f24566ddc97898892a85db041c7364206e
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 28
  IMXVault.deposit confirmed (ERC20: transfer amount exceeds balance)   Block: 45075408   Gas used: 532829 (0.18%)
```

5. As user_1 attempt to deposit 0.5 WETH tokens to the IXMVault. Observe the error message.

```
 0.5 ether
Transaction sent: 0x0b16d7d875fab535e86f31317336dfde9ecd4e70893efb2b15db9ab188ef776e
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 26
  IMXVault.deposit confirmed (ERC20: transfer amount exceeds balance)   Block: 45075406   Gas used: 532829 (0.18%)
```

6. As user_1 attempt to deposit 2 WETH tokens to the IXMVault. Observe the error message.

```
 2 ether
Transaction sent: 0xfabeb96978c77f56f797e99f7a8aa0b4d7afb6535e4b99a12ae48bc4fa2f6e27
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 30
  IMXVault.deposit confirmed (ERC20: transfer amount exceeds balance)   Block: 45075410   Gas used: 532829 (0.18%)
```

7. As user_1 attempt to deposit 10 WETH tokens to the IXMVault. Observe the error message.

```
 5 ether
Transaction sent: 0xbf8d465dfb46006d163dbc38f4bd23c6fa295eee4898ce2942ef30af566ec538
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 32
  IMXVault.deposit confirmed (ERC20: transfer amount exceeds balance)   Block: 45075412   Gas used: 532829 (0.18%)
```

Risk Level:

Likelihood - 5
Impact - 5

Recommendation:

It is recommended to review the implementation of IMX solution and fix arithmetic calculations that prevents the expected operations within the IMXVault.

Remediation Plan:

SOLVED: The Sector Finance team solved this issue in multiple commits, with the latest 46a73e34fd1b9c0e81cd671e4b10253e4db43f3d: the Sector Finance team fixed multiple functions; _addLp, _increasePosition(), rebalance() in terms of arithmetic operations. The solution now allows for multiple deposits and harvests. Vault arithmetic operations within HLP have also been updated.

3.6 (HAL-06) GETANDUPDATETVL REVERTS DUE TO INTEGER UNDERFLOW IN IMX - CRITICAL

Description:

The getAndUpdateTVL() function from IMXCore.sol contract calculates the current TVL stored within the IMX strategy. The assessment revealed that this function may revert due to integer underflow, causing denial of service for vital functions within the solution. Over the time, the borrowBalance parameter can represent a value greater than the double underlyingLp value, causing a revert due to subtraction. The errors occur non-deterministically and depends on the state of blockchain - on the previous deposits done within IMX.

The getAndUpdateTVL() function is used by:
- deposit()
- _increasePosition()
- harvest()
- rebalance()

Listing 8: IMXCore.sol (Line 321)

```
312     function getAndUpdateTVL() public returns (uint256 tvl) {
313         (uint256 uBorrow, uint256 shortPosition) =
 ↳ _updateAndGetBorrowBalances();
314         uint256 borrowBalance = _shortToUnderlying(shortPosition)
 ↳ + uBorrow;
315         uint256 shortP = _short.balanceOf(address(this));
316         uint256 shortBalance = shortP == 0
317             ? 0
318             : _shortToUnderlying(_short.balanceOf(address(this)));
319         (uint256 underlyingLp, ) = _getLPBalances();
320         uint256 underlyingBalance = _underlying.balanceOf(address(
 ↳ this));
321         tvl = underlyingLp * 2 - borrowBalance + underlyingBalance
 ↳ + shortBalance;
322     }
```

Proof of Concept:

1. Fork Optimism blockchain with block number set to 45077409.
2. Deploy the first instance of IXMVault vault with IXM strategy using ETH-USDC-Tarot-Velo configuration prepared in strategies.json file.

Listing 9: strategies.json

```
107  "ETH-USDC-Tarot-Velo": {
108      "a1_underlying": "0x4200000000000000000000000000000000000006",
109      "a2_acceptsNativeToken": true,
110      "b_short": "0x7F5c764cBc14f9669B88837ca1490cCa17c31607",
111      "c0_uniPair": "0x79c912FEF520be002c2B6e57EC4324e260f38E50",
112      "c1_pairRouter": "0xa132DAB612dB5cB9fC9Ac426A0Cc215A3423F9c9",
113      "d_poolToken": "0x28fA49da55A54F02Bc4ff103adbd843B50B556F4",
114      "e_farmToken": "0x3c8B650257cFb5f272f799F5e2b4e65093a11a05",
115      "f_farmRouter": "0xa132DAB612dB5cB9fC9Ac426A0Cc215A3423F9c9",
116      "h_harvestPath": [
117        "0x3c8B650257cFb5f272f799F5e2b4e65093a11a05"
118      ],
119      "x_chain": "OP"
120    }
```

3. As user_1 deposit 5 WETH tokens to the IXMVault.
4. Observe the IMX state. Note the value of borrowBalance is equal to 17,815,629,572,686,714,880

```
strategy._getBorrowBalances() (6407564074418082265, 14446393038)
strategy._updateAndGetBorrowBalances() (6407564074418082265, 14446393038)
strategy._shortToUnderlying() 11,408,065,498,268,631,040
strategy.borrowBalance() 17,815,629,572,686,714,880
strategy._getLPBalances() 11,406,923,272,898,971,648
Transaction sent: 0xe9dc1931bfe7d0c50b80355db362afd008b9c344dd43e12136c04d5b8413018f
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 13
  IMX.getAndUpdateTVL confirmed   Block: 45075395   Gas used: 73042 (0.02%)

strategy.getAndUpdateTVL() 4998216973111230538
```

5. Forward blockchain time for 30 days and mine 2000 block.
6. Observe the IMX state. Note the value of borrowBalance increased slightly and it is equal to 18,039,683,154,659,420,160.

```
strategy._getBorrowBalances() (6422385778560733612, 14711349962)
strategy._updateAndGetBorrowBalances() (6422385778560733612, 14711349962)
strategy._shortToUnderlying() 11,617,297,376,098,686,976
strategy.borrowBalance() 18,039,683,154,659,420,160
strategy._getLPBalances() 11,406,923,272,898,971,648
Transaction sent: 0x70bc2a5656589a8794799207d25b1541c12e99c6d6158338c104da7a6bd096d9
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 16
  IMX.getAndUpdateTVL confirmed   Block: 45077398   Gas used: 98666 (0.03%)

strategy.getAndUpdateTVL() 4774163303280111227
```

7. Deploy the second instnace of IXMVault vault with IXM strategy using ETH-USDC-Tarot-Velo configuration prepared in strategies.json file.
8. As user_1 deposit 1 WETH tokens to the IXMVault.
9. Observe the IMX state. Note the value of borrowBalance is equal to 3,563,132,806,893,267,968.

```
strategy._getBorrowBalances() (1281513809572667229, 2889288041)
strategy._updateAndGetBorrowBalances() (1281513606288927510, 2889286076)
strategy._shortToUnderlying() 2,281,618,997,320,600,576
strategy.borrowBalance() 3,563,132,806,893,267,968
strategy._getLPBalances() 2,281,384,654,265,000,960
Transaction sent: 0x53e2ac3b8918748f623128e0347bc3b6bff54af27d7a248f7e3d41b66979c1a6
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 26
  IMX.getAndUpdateTVL confirmed   Block: 45077410   Gas used: 73042 (0.02%)

strategy.getAndUpdateTVL() 999634949910386460
```

10. Forward blockchain time for 10 days and mine 2000 block. Note that chain time was forwarded three times less than in step 5.
11. Observe the IMX state. Note the value of borrowBalance increased significantly and it is equal to 5,079,333,160,769,501,184. Note that this value is higher than double LPBalances value.
12. Observe that getAndUpdateTVL() reverts due to Integer overflow error.

```
strategy._getBorrowBalances() (1457155895223912408, 4586880398)
strategy._updateAndGetBorrowBalances() (1457155664078451843, 4586877279)
strategy._shortToUnderlying() 3,622,177,265,545,588,736
strategy.borrowBalance() 5,079,333,160,769,501,184
strategy._getLPBalances() 2,281,384,654,265,000,960
Transaction sent: 0xceb68ebdcb613dff61a07e92139b2484b239dded4c83408d3b6af175c3e62220
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 29
  IMX.getAndUpdateTVL confirmed (Integer overflow)   Block: 45079413   Gas used: 72835 (0.02%)

strategy.getAndUpdateTVL() None
```

Risk Level:

Likelihood - 5
Impact - 5

Recommendation:

It is recommended to review the implementation of `getAndUpdateTVL()` function and update the order of arithmetic operations to prevent integer underflows.

Remediation Plan:

SOLVED: The Sector Finance team solved this issue in commit `46a73e34fd1b9c0e81cd671e4b10253e4db43f3d`: the `getAndUpdateTVL()` function now returns `0` instead of reverting due to integer underflow.

3.7 (HAL-07) CLOSEPOSITION IN THE HLP STRATEGY REVERTS WHEN SHORTBALANCE > SHORTPOSITION - CRITICAL

Description:

The _closePosition() function from HLPCore.sol contract removes all assets from the HLP strategy, then converts them to the underlying token and moves them to the SCYVault contract.

Listing 10: HLPCore.sol (Line 322)

```
312     function _closePosition() internal {
313         _decreaseULpTo(0);
314         uint256 shortPosition = _updateAndGetBorrowBalance();
315         uint256 shortBalance = _short.balanceOf(address(this));
316         if (shortPosition > shortBalance) {
317             pair()._swapTokensForExactTokens(
318                 shortPosition - shortBalance,
319                 address(_underlying),
320                 address(_short)
321             );
322         } else if (shortBalance > shortPosition) {
323             pair()._swapExactTokensForTokens(
324                 shortBalance - shortPosition,
325                 address(_short),
326                 address(_underlying)
327             );
328         }
329         _repay(_short.balanceOf(address(this)));
330         uint256 collateralBalance = _updateAndGetCollateralBalance
 ↳ ();
331         _redeem(collateralBalance);
332     }
```

It is possible for an attacker to manipulate the execution flow and force a condition where the closePosition function reverts when trying to repay the short token.

In a volatile market, closing positions at exact moments is important to avoid losing deposited funds. Also, the lack of possibility to close the position at a certain profit may have negative impact on customers' profits.

Proof of Concept:

1. Fork MoonRiver blockchain with block number set to 3131313.
2. Deploy the first instance of IXMVault vault with HLP strategy using USDC-MOVR-SOLAR-WELL configuration prepared in strategies.json file.

Listing 11: strategies.json

```
43    "USDC-MOVR-SOLAR-WELL": {
44      "a_underlying": "0xE3F5a90F9cb311505cd691a46596599aA1A0AD7D",
45      "b_short": "0x98878B06940aE243284CA214f92Bb71a2b032B8A",
46      "c_uniPair": "0xe537f70a8b62204832B8Ba91940B77d3f79AEb81",
47      "d1_cTokenLend": "0xd0670AEe3698F66e2D4dAf071EB9c690d978BFA8",
48      "d2_cTokenBorrow": "0x6a1A771C7826596652daDC9145fEAaE62b1cd07f
 ↳ ",
49      "e1_farmToken": "0x6bD193Ee6D2104F14F94E2cA6efefae561A4334B",
50      "e2_farmId": 10,
51      "e3_uniFarm": "0x0329867a8c457e9f75e25b0685011291cd30904f",
52      "f_farmRouter": "0xAA30eF758139ae4a7f798112902Bf6d65612045f",
53      "h_harvestPath": [
54        "0x6bD193Ee6D2104F14F94E2cA6efefae561A4334B",
55        "0x98878B06940aE243284CA214f92Bb71a2b032B8A",
56        "0xE3F5a90F9cb311505cd691a46596599aA1A0AD7D"
57      ],
58      "l1_comptroller": "0x0b7a0EAA884849c6Af7a129e899536dDDcA4905E"
 ↳ ,
59      "l2_lendRewardToken": "0
 ↳ xBb8d88bcD9749636BC4D2bE22aaC4Bb3B01A58F1",
60      "l3_lendRewardPath": [
61        "0xBb8d88bcD9749636BC4D2bE22aaC4Bb3B01A58F1",
62        "0x98878B06940aE243284CA214f92Bb71a2b032B8A",
63        "0xE3F5a90F9cb311505cd691a46596599aA1A0AD7D"
64      ],
65      "l4_lendRewardRouter": "0
 ↳ xAA30eF758139ae4a7f798112902Bf6d65612045f",
66      "n_nativeToken": 2,
67      "x_chain": "MOVR"
```

```
68    },
```

3. As user_1 deposit 1000 USDC tokens to the HLPVault.
4. Call harvest(),
5. Attacker sends an arbitrary amount of SHORT tokens to strategy to make shortBalance > shortPosition.
6. GUARDIAN tries to close position.

```
    ├─ [20962] 0x6a1A771C7826596652daDC9145fEAaE62b1cd07f::repayBorrow{value: 36974891141476688676}()
    │   ├─ [19228] 0x0b7a0EAA884849c6Af7a129e899536dDDcA4905E::repayBorrowAllowed(0x6a1A771C7826596652daDC9145fEAaE62b1cd07f, MasterChefCompMulti: [0x2e234DAe75C793f67A35089C9d99245E1C58470b], MasterChefCompMulti: [0x2e234DAe75C793f67A35089C9d99245E1C58470b], 36974891141476688676)
    │   │   ├─ [18542] 0x8529ea4DBDcA738aA928d682ea9c1382Bf2Ff098::repayBorrowAllowed(0x6a1A771C7826596652daDC9145fEAaE62b1cd07f, MasterChefCompMulti: [0x2e234DAe75C793f67A35089C9d99245E1C58470b], MasterChefCompMulti: [0x2e234DAe75C793f67A35089C9d99245E1C58470b], 36974891141476688676) [delegatecall]
    │   │   │   ├─ [454] 0x6a1A771C7826596652daDC9145fEAaE62b1cd07f::borrowIndex() [staticcall]
    │   │   │   │   └─ ← 0x0000000000000000000000000000000000000000000010268333863cb622
    │   │   │   ├─ [1382] 0x6a1A771C7826596652daDC9145fEAaE62b1cd07f::borrowBalanceStored(MasterChefCompMulti: [0x2e234DAe75C793f67A35089C9d99245E1C58470b]) [staticcall]
    │   │   │   │   └─ ← 0x00000000000000000000000000000000000000000000000202123397807 4a323
    │   │   │   ├─ emit topic 0: 0xa1b6a046664a0ecf068059f26de56878f8d0e799907ca2e42d9148ccbdc717a7
    │   │   │   │        topic 1: 0x0000000000000000000000000000000000000000000000000000000000000000
    │   │   │   │        topic 2: 0x000000000000000000000006a1a771c7826596652dadc9145feaae62b1cd07f
    │   │   │   │        topic 3: 0x000000000000000000000002e234dae75c793f67a35089c9d99245e1c58470b
    │   │   │   │           data: 0x0000000000000000000000000000000000000000000000000000000000000000000000001a30fa0fc528728d5bfbc2e0faf73252
    │   │   │   ├─ [1382] 0x6a1A771C7826596652daDC9145fEAaE62b1cd07f::borrowBalanceStored(MasterChefCompMulti: [0x2e234DAe75C793f67A35089C9d99245E1C58470b]) [staticcall]
    │   │   │   │   └─ ← 0x00000000000000000000000000000000000000000000000202123397807 4a323
    │   │   │   ├─ emit topic 0: 0xa1b6a046664a0ecf068059f26de56878f8d0e799907ca2e42d9148ccbdc717a7
    │   │   │   │        topic 1: 0x0000000000000000000000000000000000000000000000000000000000000001
    │   │   │   │        topic 2: 0x000000000000000000000006a1a771c7826596652dadc9145feaae62b1cd07f
    │   │   │   │        topic 3: 0x000000000000000000000002e234dae75c793f67a35089c9d99245e1c58470b
    │   │   │   │           data: 0x0000000000000000000000000000000000000000000000000000000000000000000000004215925302bedf47e63e2cd715ece2
    │   │   │   └─ ← 0x0000000000000000000000000000000000000000000000000000000000000000
    │   │   └─ ← 0x0000000000000000000000000000000000000000000000000000000000000000
    │   └─ ← "REPAY_BORROW_NEW_ACCOUNT_BORROW_BALANCE_CALCULATION_FAILED"
    └─ ← "REPAY_BORROW_NEW_ACCOUNT_BORROW_BALANCE_CALCULATION_FAILED"
└─ ← "REPAY_BORROW_NEW_ACCOUNT_BORROW_BALANCE_CALCULATION_FAILED"

Test result: FAILED. 0 passed; 1 failed; finished in 36.69ms

Failing tests:
Encountered 1 failing test in src/tests/strategy/hlp.t.sol:hlp
[FAIL. Reason: REPAY_BORROW_NEW_ACCOUNT_BORROW_BALANCE_CALCULATION_FAILED] test3() (gas: 5615245)
```

Risk Level:

Likelihood - 5
Impact - 5

Recommendation:

It is recommended to review the implementation of _closePosition() function and update the order of arithmetic operations to prevent integer underflow occurrence.

Remediation Plan:

SOLVED: The Sector Finance team solved this issue in commit e9fc38bae7438846b73fd640f539a8a6d41d5f0b: the _closePosition() function now repays the short token with _updateAndGetBorrowBalance() returning instead of the contract balance.

3.8 (HAL-08) STRATEGYTVL IN THE SYNAPSE SOLUTION REVERTS FOR EMPTY DEPOSIT - HIGH

Description:

The _strategyTvl() function within the Synapse.sol contract allows fetching the TVL value deposited in the farm. However, the assessment revealed if no deposit was made to the vault, this function always reverts with SafeMath: subtraction overflow error message. This happens because when balance returned by the farm.userInfo() function is equal to 0, then ISynapseSwap.calculateRemoveLiquidityOneToken() reverts due to overflow.

Listing 12: Synapse.sol

```
69     function _strategyTvl() internal view override returns (
↳ uint256) {
70         (uint256 balance, ) = farm.userInfo(uint256(farmId),
↳ address(this));
71         return ISynapseSwap(strategy).
↳ calculateRemoveLiquidityOneToken(balance, uint8(strategyId));
72     }
```

The _strategyTvl() function is used by:
- _stratGetAndUpdateTvl()
- getStrategyTvl()
- getTvl()
- getAndUpdateTvl()
- getUpdatedUnderlyingBalance()
- underlyingBalance()
- updateStrategy()
- underlyingToShares()
- sharesToUnderlying()
- harvest()

The majority of the above functions are used as prerequisites for input preparation to the deposit(), redeem(), harvest(), withdrawFromStrategy()

functions.

The solution is designed to be used by other smart contracts; therefore, a risk exists that flow execution of these smart contracts is being interrupted due to the revert in the _strategyTvl() function.

It was also possible to reproduce this vulnerability with integration tests prepared by the Sector Finance team in the Foundry framework.

```
Encountered 5 failing tests in src/tests/strategy/Integration.t.sol:IntegrationSynapse
[FAIL. Reason: SafeMath: subtraction overflow] testAccounting() (gas: 114521)
[FAIL. Reason: SafeMath: subtraction overflow] testClosePosition() (gas: 114520)
[FAIL. Reason: SafeMath: subtraction overflow] testFlashSwap() (gas: 114564)
[FAIL. Reason: SafeMath: subtraction overflow] testIntegrationFlow() (gas: 114564)
[FAIL. Reason: SafeMath: subtraction overflow] testManagerWithdraw() (gas: 114521)
```

Proof of Concept:

Instance 1:

1. Fork Arbitrum blockchain with block number set to 42392775.
2. Deploy the Synapse vault using USDC-Arbitrum-Synapse configuration prepared in strategies.json file.

Listing 13: strategies.json

```
81    "USDC-Arbitrum-Synapse": {
82      "a_underlying": "0xFF970A61A04b1cA14834A43f5dE4533eBDDB5CC8",
83      "b_strategy": "0x9Dd329F5411466d9e0C488fF72519CA9fEf0cb40",
84      "c_strategyId": 1,
85      "d_yieldToken": "0xcFd72be67Ee69A0dd7cF0f846Fc0D98C33d60F16",
86      "e_farmId": 3,
87      "f1_farm": "0x73186f2Cf2493f20836b17b21ae79fc12934E207",
88      "f2_farmToken": "0x080F6AEd32Fc474DD5717105Dba5ea57268F46eb",
89      "g_farmRouter": "0xE592427A0AEce92De3Edee1F18E0157C05861564",
90      "h_harvestPath": "0
 ↳ x080f6aed32fc474dd5717105dba5ea57268f46eb000bb8ff970a61a04b1ca14834a43f5de4533e
 ↳ ",
91      "x_chain": "ARBITRUM"
92    },
```

3. Attempt to call getAndUpdateTvl() function. Observe that transaction reverts with SafeMath: subtraction overflow error message.

```
Transaction sent: 0x15fcdbda60c7f2817beb62cc7aa2353db845dc743bfd2183d35ad60a74b563a4
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 5
  Synapse.getAndUpdateTvl confirmed (SafeMath: subtraction overflow)   Block: 42392785   Gas used: 83123 (0.03%)

synapse.getAndUpdateTvl() None
```

Risk Level:

Likelihood - 4
Impact - 5

Recommendation:

It is recommended to handle the situation when farm's balance is equal to 0 in the _strategyTvl() function.

Remediation Plan:

SOLVED: The Sector Finance team solved this issue in commit 87834bb1babf7a44a21fc7a913241bac6fff7521: the _strategyTvl() function now returns 0 if the farm balance is 0.

3.9 (HAL-09) GETMAXTVL MAY REVERT DUE TO INTEGER UNDERFLOW IN IMX - HIGH

Description:

The getMaxTvl() function from IMXCore.sol contract calculates the maximum possible TVL stored within the IMX strategy. The assessment revealed that this function may revert due to integer underflow, causing denial of service for the deposit() function. The error occurs non-deterministically and depends on the state of the blockchain - on the Tarot pool's balances of underlying and short tokens. The solution works with the Tarot pool that has very low liquidity and the error happens when pool is close to 100% utilization.

Listing 14: IMXCore.sol (Line 300)

```
298     function getMaxTvl() public view returns (uint256) {
299         (, uint256 sBorrow) = _getBorrowBalances();
300         uint256 availableToBorrow = sBorrowable().totalSupply() -
 ↳ sBorrowable().totalBorrows();
301         return
302             min(
303                 _maxTvl,
304                 // adjust the availableToBorrow to account for
 ↳ leverage
305                 _shortToUnderlying(
306                     sBorrow + (availableToBorrow * 1e18) / (
 ↳ _optimalUBorrow() + 1e18)
307                 )
308             );
309     }
```

The solution is designed to be used by other smart contracts. Therefore, a risk exists that flow execution of these smart contracts are interrupted due to integer underflow within the getMaxTvl() function.

Proof of Concept:

1. Fork Optimism blockchain with block number set to 46077599.
2. Deploy the IXMVault vault with IXM strategy using ETH-USDC-Tarot-Velo configuration prepared in strategies.json file.

Listing 15: strategies.json

```
107 "ETH-USDC-Tarot-Velo": {
108     "a1_underlying": "0x4200000000000000000000000000000000000006",
109     "a2_acceptsNativeToken": true,
110     "b_short": "0x7F5c764cBc14f9669B88837ca1490cCa17c31607",
111     "c0_uniPair": "0x79c912FEF520be002c2B6e57EC4324e260f38E50",
112     "c1_pairRouter": "0xa132DAB612dB5cB9fC9Ac426A0Cc215A3423F9c9",
113     "d_poolToken": "0x28fA49da55A54F02Bc4ff103adbd843B50B556F4",
114     "e_farmToken": "0x3c8B650257cFb5f272f799F5e2b4e65093a11a05",
115     "f_farmRouter": "0xa132DAB612dB5cB9fC9Ac426A0Cc215A3423F9c9",
116     "h_harvestPath": [
117       "0x3c8B650257cFb5f272f799F5e2b4e65093a11a05"
118     ],
119     "x_chain": "OP"
120   }
```

3. Manipulate Tarot pool liquidity in such a way that utilization is close to 100%, e.g. as presented on the below image.

```
_sBorrowable.underlying() 0x7F5c764cBc14f9669B88837ca1490cCa17c31607
_uBorrowable.underlying() 0x4200000000000000000000000000000000000006
_sBorrowable.totalSupply() 57,446,521,880
_sBorrowable.totalBorrows() 50,404,488,459
_uBorrowable.totalSupply() 56,157,255,495,520,337,920
_uBorrowable.totalBorrows() 38,500,708,356,332,994,560
```

4. As user_1 attempt to call the getMaxTvl() function. Observe that the transaction reverts with Integer overflow error message.

```
Transaction sent: 0xb269682b65330931f8c56cb9045ae6481155824bfd10d9b677683c152aeff7a8
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 59
  IMX.getMaxTvl_ confirmed (Integer overflow)   Block: 46081671   Gas used: 33747 (0.01%)
```

5. As user_1 attempt to deposit 1 WETH tokens to the IXMVault. Observe that the transaction reverts with Integer overflow error message. Note that function reverts due to issue in the getMaxTvl() function.

```
Transaction sent: 0xf3bde01100168a5eece6189bc148fd183118c6a5ad4888142bff8580d6641e59
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 5
  IMXVault.deposit confirmed (reverted)   Block: 46081673   Gas used: 156739 (0.05%)
```

```
Call trace for '0xf3bde01100168a5eece6189bc148fd183118c6a5ad4888142bff8580d6641e59':
Initial call cost  [21912 gas]
IMXVault.deposit  0:5695  [15072 / 119827 gas]
├── <UnknownContract>.0x23b872dd  [CALL]  520:806  [23500 gas]
├── <UnknownContract>.0x70a08231  [STATICCALL]  1033:1122  [1857 gas]
└── IMX.deposit  [CALL]  1268:5683  [4688793 / 79398 gas]
    ├── IMXCore.deposit  1355:4446  [9325 / 62619 gas]
    │   ├── IMXCore.getAndUpdateTVL  1393:3272  [133 / 39531 gas]
    │   │   ├── IMXFarm._updateAndGetBorrowBalances  1400:3226  [61 / 39353 gas]
    │   │   │   ├── IMXFarm.accrueInterest  1406:2670  [3401 / 32811 gas]
    │   │   │   │   ├── <UnknownContract>.0xa6afed95  [CALL]  1460:2028  [14705 gas]
    │   │   │   │   └── <UnknownContract>.0xa6afed95  [CALL]  2091:2659  [14705 gas]
    │   │   │   └── IMXFarm._getBorrowBalances  2674:3218  [2171 / 6481 gas]
    │   │   │       ├── <UnknownContract>.0x4d73e9ba  [STATICCALL]  2716:2894  [2155 gas]
    │   │   │       └── <UnknownContract>.0x4d73e9ba  [STATICCALL]  2982:3160  [2155 gas]
    │   │   └── IUniLp._shortToUnderlying  3237:3250  [45 gas]
    │   ├── <UnknownContract>.0x70a08231  [STATICCALL]  3315:3444  [1985 gas]
    │   ├── IUniLp._getLPBalances  3511:3749  [41 / 3158 gas]
    │   │   └── IMXFarm._getLiquidity  3518:3744  [1138 / 3117 gas]
    │   │       └── <UnknownContract>.0x70a08231  [STATICCALL]  3560:3678  [1979 gas]
    │   ├── IUniLp.getUnderlyingShortReserves  3757:4180  [3431 / 6857 gas]
    │   │   └── <UnknownContract>.0x0902f1ac  [STATICCALL]  3907:3996  [3426 gas]
    │   └── <UnknownContract>.0x18160ddd  [STATICCALL]  4226:4296  [1763 gas]
    ├── <UnknownContract>.0x70a08231  [STATICCALL]  4490:4579  [1857 gas]
    └── IMXCore.getMaxTvl  4744:5683  [-4684007 / -4673871 gas]
        ├── IMXFarm._getBorrowBalances  4750:5294  [2171 / 6481 gas]
        │   ├── <UnknownContract>.0x4d73e9ba  [STATICCALL]  4792:4970  [2155 gas]
        │   └── <UnknownContract>.0x4d73e9ba  [STATICCALL]  5058:5236  [2155 gas]
        ├── <UnknownContract>.0x47bd3718  [STATICCALL]  5339:5438  [1866 gas]
        └── <UnknownContract>.0x18160ddd  [STATICCALL]  5525:5602  [1789 gas]
```

Risk Level:

Likelihood - 4

Impact - 5

Recommendation:

It is recommended to review the implementation of getMaxTvl() function and update the order of arithmetic operations to prevent integer underflow occurrence.

Remediation Plan:

SOLVED: The Sector Finance team solved this issue in commit 46a73e34fd1b9c0e81cd671e4b10253e4db43f3d: the getMaxTvl() function now returns 0 if there is no liquidity available to borrow in the Tarot pool, also the deposit() function reverts with the expected error message STRAT: OVER_MAX_TVL.

3.10 (HAL-10) PROFIT HARVESTING IN THE SYNAPSE SOLUTION MAY REVERT - MEDIUM

Description:

The harvest() function within the SCYVault.sol contract allows users with manager's role to collect the profits earned in a farm within the Synapse solution. However, the assessment revealed that while testing the solution on the forked environment - Arbitrum blockchain - this function may revert with various error messages:
- BoringERC20: Transfer failed
- SPL (SPL: Square root price limit)

The first error occurs in farm.harvest(farmId, address(this)); within the _harvestFarm() function. The second error occurs in amountOut = farmRouter.exactInput(params); within the _harvestFarm() function.

Listing 16: MiniChef2Farm.sol (Line 51)

```
47      function _harvestFarm(HarvestSwapParams calldata swapParams)
48          internal
49          returns (uint256 harvested, uint256 amountOut)
50      {
51          farm.harvest(farmId, address(this));
52          harvested = farmToken.balanceOf(address(this));
53          if (harvested == 0) return (0, 0);
54
55          ISwapRouter.ExactInputParams memory params = ISwapRouter.
↳ ExactInputParams({
56              path: swapParams.pathData,
57              recipient: address(this),
58              deadline: block.timestamp,
59              amountIn: harvested,
60              amountOutMinimum: swapParams.min
61          });
62          amountOut = farmRouter.exactInput(params);
63          emit HarvestedToken(address(farmToken), harvested,
↳ amountOut);
```

```
64      }
```

The farm is a contract implementing IMiniChefV2 interface, which is considered a black box from the perspective of the security assessment. The errors occur non-deterministically and depend on the state of the blockchain.
The Synapse solution is managed by the SectorTimelock contract that schedules operations and executes them timely, which may be an additional handicapping factor.
It was also possible to reproduce this vulnerability by integration tests prepared by the Sector Finance team in the Foundry framework.

```
Running 5 tests for src/tests/strategy/Integration.t.sol:IntegrationSynapse
[PASS] testAccounting() (gas: 1181293)
[PASS] testClosePosition() (gas: 774160)
[PASS] testFlashSwap() (gas: 1183197)
[FAIL. Reason: BoringERC20: Transfer failed] testIntegrationFlow() (gas: 1775012)
[PASS] testManagerWithdraw() (gas: 908756)
Test result: FAILED. 4 passed; 1 failed; finished in 260.34ms
```

In this instance, accordingly to Discord's group discussion, this issue was happening because Synapse team had not refilled the funds in the farm.

As a result, the Synapse solution is incapable to collect the profits earned from deposit in the farm. Though, the strategy participants still can redeem the liquidity.

Proof of Concept:

Instance 1:

1. Fork Arbitrum blockchain with block number set to 41961716.
2. Deploy the Synapse vault using USDC-Arbitrum-Synapse configuration prepared in strategies.json file.

Listing 17: strategies.json

```
81    "USDC-Arbitrum-Synapse": {
82      "a_underlying": "0xFF970A61A04b1cA14834A43f5dE4533eBDDB5CC8",
83      "b_strategy": "0x9Dd329F5411466d9e0C488fF72519CA9fEf0cb40",
84      "c_strategyId": 1,
85      "d_yieldToken": "0xcFd72be67Ee69A0dd7cF0f846Fc0D98C33d60F16",
86      "e_farmId": 3,
87      "f1_farm": "0x73186f2Cf2493f20836b17b21ae79fc12934E207",
88      "f2_farmToken": "0x080F6AEd32Fc474DD5717105Dba5ea57268F46eb",
89      "g_farmRouter": "0xE592427A0AEce92De3Edee1F18E0157C05861564",
90      "h_harvestPath": "0
 ↳ x080f6aed32fc474dd5717105dba5ea57268f46eb000bb8ff970a61a04b1ca14834a43f5de4533e
 ↳ ",
91      "x_chain": "ARBITRUM"
92    },
```

3. As owner deposit() 1,000,000,000 USDC tokens.
4. As user_1 deposit() 1,000,000,000 USDC tokens.
5. As user_2 deposit() 1,000,000,000 USDC tokens.
6. Forward blockchain time for 1 day and blocks for 100.
7. As owner attempt to harvest(). Observe that transaction reverts with BoringERC20: Transfer failed error message.

```
Tvl 2,999,710,651
---
synapse.balance() 0
synapse.strategy().balance() 0
USDC_ERC20.balance() 0
user1.balance() 99,999,999,999,999,991,611,392
user2.balance() 99,999,999,999,999,991,611,392
---
USDC_ERC20.balanceOf(treasury) 0
USDC_ERC20.balanceOf(synapse) 0
USDC_ERC20.balanceOf(user2) 3,000,000,000
USDC_ERC20.balanceOf(user1) 3,000,000,000
USDC_ERC20.balanceOf(owner) 3,000,000,000
USDC_ERC20.balanceOf(synapse.strategy()) 2,859,903,423,526
USDC_ERC20.balanceOf(synapse.strategy()) 2859903423526
---
synapse.balanceOf(treasury) 0
synapse.balanceOf(user2) 996,162,356,688,423,616,512
synapse.balanceOf(user1) 996,162,951,006,856,282,112
synapse.balanceOf(owner) 996,163,545,619,209,125,888
synapse.underlyingBalance(user2) 999,902,953
synapse.sharesToUnderlying(user2) 999,902,953
synapse.underlyingBalance(user1) 999,903,550
synapse.sharesToUnderlying(user1) 999,903,550
synapse.underlyingBalance(owner) 999,904,147
synapse.sharesToUnderlying(owner) 999,904,147
synapse.balanceOf(synapse.strategy()) 0
synapse.underlyingBalance(synapse.strategy()) 0
---
synapse.totalSupply() 2,988,488,853,314,488,762,368
synapse.convertToAssets(shares) 996,162,951,006,856,282,112
synapse.uBalance() 0
synapse.underlyingBalance() 999,903,550
synapse.getFloatingAmount() 0
Transaction sent: 0x9202e593115101d4cefa280518f95e9ccfd3570e22b2e24e3d4820b2f139734d
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 14
  Synapse.getAndUpdateTvl confirmed   Block: 41961739   Gas used: 90867 (0.03%)

synapse.getAndUpdateTvl() 2,999,710,651
synapse.getStrategyTvl() 2,999,710,651
---
chain.sleep(1 day), chain.mine(100)
Transaction sent: 0xa000c0f58c4dd10cb14aa1963073cb8cbfdbb2ca95d7a9c72a36ed05e4845ba2
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 15
  Synapse.getAndUpdateTvl confirmed   Block: 41961840   Gas used: 90867 (0.03%)

synapse.harvest(synapse.getTvl(), 0, [params1], [params2])
Transaction sent: 0x93c7180dba8865e51e6affe41a4110c949bb87042a5c2c6f97ec0aa66f5113a2
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 16
  Synapse.harvest confirmed (BoringERC20: Transfer failed)   Block: 41961841   Gas used: 130576 (0.04%)
```

Instance 2:

1. Fork Arbitrum blockchain with block number set to 42417795.
2. Deploy the Synapse vault using USDC-Arbitrum-Synapse configuration prepared in strategies.json file.

Listing 18: strategies.json

```
81    "USDC-Arbitrum-Synapse": {
82      "a_underlying": "0xFF970A61A04b1cA14834A43f5dE4533eBDDB5CC8",
83      "b_strategy": "0x9Dd329F5411466d9e0C488fF72519CA9fEf0cb40",
84      "c_strategyId": 1,
85      "d_yieldToken": "0xcFd72be67Ee69A0dd7cF0f846Fc0D98C33d60F16",
```

```
86      "e_farmId": 3,
87      "f1_farm": "0x73186f2Cf2493f20836b17b21ae79fc12934E207",
88      "f2_farmToken": "0x080F6AEd32Fc474DD5717105Dba5ea57268F46eb",
89      "g_farmRouter": "0xE592427A0AEce92De3Edee1F18E0157C05861564",
90      "h_harvestPath": "0
↳ x080f6aed32fc474dd5717105dba5ea57268f46eb000bb8ff970a61a04b1ca14834a43f5de4533e
↳ ",
91      "x_chain": "ARBITRUM"
92    },
```

3. As owner deposit() 1,000,000,000 USDC tokens.
4. As user_1 deposit() 1,000,000,000 USDC tokens.
5. As user_2 deposit() 2,000,000,000 USDC tokens.
6. Forward blockchain time for 7 days and blocks for 1000.
7. As owner attempt to harvest(). Observe that transaction reverts with SPL error message.

```
Tvl 4,999,548,267
---
synapse.balance() 0
synapse.strategy().balance() 0
USDC_ERC20.balance() 0
user1.balance() 99,999,999,999,999,991,611,392
user2.balance() 99,999,999,999,999,991,611,392
---
USDC_ERC20.balanceOf(treasury) 0
USDC_ERC20.balanceOf(synapse) 0
USDC_ERC20.balanceOf(user2) 2,000,000,000
USDC_ERC20.balanceOf(user1) 2,000,000,000
USDC_ERC20.balanceOf(owner) 3,000,000,000
USDC_ERC20.balanceOf(synapse.strategy()) 2,964,765,153,210
USDC_ERC20.balanceOf(synapse.strategy()) 2964765153210
---
synapse.balanceOf(treasury) 0
synapse.balanceOf(user2) 1,992,132,257,015,061,020,672
synapse.balanceOf(user1) 1,992,126,472,864,204,324,864
synapse.balanceOf(owner) 996,067,551,711,495,454,720
synapse.underlyingBalance(user2) 1,999,811,759
synapse.sharesToUnderlying(user2) 1,999,811,759
synapse.underlyingBalance(user1) 1,999,805,953
synapse.sharesToUnderlying(user1) 1,999,805,953
synapse.underlyingBalance(owner) 999,907,308
synapse.sharesToUnderlying(owner) 999,907,308
synapse.balanceOf(synapse.strategy()) 0
synapse.underlyingBalance(synapse.strategy()) 0
---
synapse.totalSupply() 4,980,326,281,590,761,062,400
synapse.convertToAssets(shares) 1,992,138,051,539,426,607,104
synapse.uBalance() 0
synapse.underlyingBalance() 1,999,805,953
synapse.getFloatingAmount() 0
Transaction sent: 0xf57388a42de56b95dccb2abba4e9b31280594f0c0b9e8d059e78291542447a44
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 22
  Synapse.getAndUpdateTvl confirmed   Block: 42418828   Gas used: 90867 (0.03%)

synapse.getAndUpdateTvl() 4,999,548,267
---
chain.sleep(7 days), chain.mine(1000)
Transaction sent: 0x5027f13b4133c2b87b4fc9cc7ee4a51d7317b4f4dbb8981cc67f9c62e7e9e43c
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 23
  Synapse.getAndUpdateTvl confirmed   Block: 42419829   Gas used: 90867 (0.03%)

synapse.harvest(synapse.getTvl(), 0, [params1], [params2])
Transaction sent: 0x43b3699d3ef96f47ce37931fc4055de3bc0659f9838ad8eb9213f54e8ba8ee14
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 24
  Synapse.harvest confirmed (SPL)   Block: 42419830   Gas used: 162132 (0.05%)
```

Risk Level:

Likelihood - 2

Impact - 4

Recommendation:

It is recommended to prepare internal incident response procedures and communication for clients, to prevent e.g. panic redeem, in case of unpredictable state of farm.

Remediation Plan:

SOLVED: The Sector Finance team solved this issue in commit 404af4b1266ea814c76d578f700a5cb629dfe61a: the _stratHarvest() function expected to be reverted if there is any problem with external farm; however, the team updated the harvest() function, so it is now possible to execute it without triggering the _stratHarvest() function call, if this function may revert.

3.11 (HAL-11) THE MINT FUNCTION LACKS LOCK FOR MINIMUM LIQUIDITY - MEDIUM

Description:

The deposit() and mint() functions within the ERC4626.sol contract allow users to deposit liquidity in the vault. The deposit() function locks minimum liquidity during the initial deposit. However, the alternative function mint() lacks similar lock implementation. As a result, a user can take advantage of the function mint() to omit the locking part of the deposit. The MIN_LIQUIDITY is a constant value, and it is set to 1000.

Listing 19: ERC4626.sol (Lines 80-84)

```
61 function deposit(uint256 assets, address receiver)
62         public
63         payable
64         virtual
65         nonReentrant
66         returns (uint256 shares)
67     {
68         if (totalAssets() + assets > maxTvl) revert OverMaxTvl();
69
70         // This check is no longer necessary because we use
↳ MIN_LIQUIDITY
71         // Check for rounding error since we round down in
↳ previewDeposit.
72         // require((shares = previewDeposit(assets)) != 0, "
↳ ZERO_SHARES");
73         shares = previewDeposit(assets);
74
75         // Need to transfer before minting or ERC777s could
↳ reenter.
76         if (useNativeAsset && msg.value == assets) IWETH(address(
↳ asset)).deposit{ value: assets }();
77         else asset.safeTransferFrom(msg.sender, address(this),
↳ assets);
78
79         // lock minimum liquidity if totalSupply is 0
```

```
80          if (totalSupply() == 0) {
81              if (MIN_LIQUIDITY > shares) revert MinLiquidity();
82              shares -= MIN_LIQUIDITY;
83              _mint(address(1), MIN_LIQUIDITY);
84          }
85
86          _mint(receiver, shares);
87
88          emit Deposit(msg.sender, receiver, assets, shares);
89
90          afterDeposit(assets, shares);
91      }
92
93      function mint(uint256 shares, address receiver)
94          public
95          payable
96          virtual
97          nonReentrant
98          returns (uint256 assets)
99      {
100         assets = previewMint(shares); // No need to check for
 ↳ rounding error, previewMint rounds up.
101         if (totalAssets() + assets > maxTvl) revert OverMaxTvl();
102
103         // Need to transfer before minting or ERC777s could
 ↳ reenter.
104         if (useNativeAsset && msg.value == assets) IWETH(address(
 ↳ asset)).deposit{ value: assets }();
105         else asset.safeTransferFrom(msg.sender, address(this),
 ↳ assets);
106
107         _mint(receiver, shares);
108
109         emit Deposit(msg.sender, receiver, assets, shares);
110
111         afterDeposit(assets, shares);
112     }
```

Proof of Concept:

1. All necessary contracts are deployed: AggregatorVault, XChainLib, WETH, MockERC20.

2. As user_1 mint() $100 * 10^{18}$ shares.
3. Observe that transaction finishes successfully. Note that user received exactly $100 * 10^{18}$ shares.

```
Transaction sent: 0xefd184ef748ca9ef2dc5d2d7be34609ed8d57e0d03fb3d9cb5ff924827f52f45
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 1
  AggregatorVault.mint confirmed   Block: 23   Gas used: 108096 (0.90%)

tx.return_value 100,000,000,000,000,000,000
wETHMock.balanceOf(user1) 0
aggregatorVault.balanceOf(user1) 100,000,000,000,000,000,000
aggregatorVault.balanceOf(user1) 100000000000000000000
aggregatorVault.underlyingBalance(user1) 100,000,000,000,000,000,000
wETHMock.balanceOf(aggregatorVault) 100,000,000,000,000,000,000
aggregatorVault.balanceOf(aggregatorVault) 0
aggregatorVault.underlyingBalance(aggregatorVault) 0
aggregatorVault.pendingRedeem() 0
aggregatorVault.totalSupply() 100,000,000,000,000,000,000
aggregatorVault.convertToAssets(shares) 50,000,000,000,000,000,000
aggregatorVault.floatAmnt() 100,000,000,000,000,000,000
```

Risk Level:

Likelihood - 5
Impact - 1

Recommendation:

It is recommended to add minimum liquidity lock to the mint() function, as it is done in the deposit() function.

Remediation Plan:

SOLVED: The Sector Finance team solved this issue in commit de360d71c21ae72cfab3ca04f9be4d1bbf2e99ba: the minimum liquidity lock is now added to the mint() function.

3.12 (HAL-12) LOANHEALTH MAY REVERT DUE TO INTEGER UNDERFLOW IN IMX - LOW

Description:

The loanHealth() function from IMXCore.sol contract calculates an integer value that represents the health of the loan. The assessment revealed that this function may revert due to integer underflow when the underlying loan is unhealthy. The error occurs when shortfall variable is not equal to 0 and it is higher than liq variable.

Listing 20: IMXCore.sol (Line 339)

```
334     function loanHealth() public view override returns (uint256) {
335         uint256 balance = IERC20(address(_collateralToken)).
 ↳ balanceOf(address(this));
336         if (balance == 0) return 100e18;
337         uint256 liq = (balance * _collateralToken.exchangeRate())
 ↳ / 1e18;
338         (uint256 available, uint256 shortfall) = _collateralToken.
 ↳ accountLiquidity(address(this));
339         return shortfall == 0 ? (1e18 * (liq + available)) / liq :
 ↳  (1e18 * (liq - shortfall)) / liq;
340     }
```

The solution is designed to be used by other smart contracts; therefore, a risk exists that flow execution of these smart contracts is interrupted due to integer underflow within the loanHealth() function.

Risk Level:

Likelihood - 1
Impact - 3

Recommendation:

It is recommended to review the implementation of the loanHealth() function and update the order of arithmetic operations to prevent integer underflow occurrence.

Remediation Plan:

SOLVED: The Sector Finance team solved this issue in commit e7f83a783e8049b1d5608ef7cd4139b087744bc6: the loanHealth() now includes a check that prevents an integer underflow from happening.

3.13 (HAL-13) EMERGENCYACTION IS NOT MARKED AS PAYABLE - LOW

Description:

The StratAuth, SCYVault and SCYVault contracts implement the emergencyAction() function that allows the vault's owner to execute any action on the behalf of the vault. This function also allows sending ETH with the low level call() function. However, the emergencyAction() function is not marked as payable; therefore it cannot call any function that requires ETH immediately and ETH must be transferred to the contract before calling that function.

Listing 21: StratAuth.sol (Lines 20,25)

```
20     function emergencyAction(EAction[] calldata actions) public
↳ onlyOwner {
21         uint256 l = actions.length;
22         for (uint256 i = 0; i < l; i++) {
23             address target = actions[i].target;
24             bytes memory data = actions[i].data;
25             (bool success, ) = target.call{ value: actions[i].
↳ value }(data);
26             require(success, "emergencyAction failed");
27             emit EmergencyAction(target, data);
28         }
29     }
```

Risk Level:

Likelihood - 2
Impact - 2

Recommendation:

It is recommended to mark all implementations of the emergencyAction() function as payable within the solution.

Remediation Plan:

SOLVED: The Sector Finance team solved this issue in commit 404af4b1266ea814c76d578f700a5cb629dfe61a: all instances of the emergencyAction() functions are now decorated with the payable modifier.

3.14 (HAL-14) UNNEEDED INITIALIZATION OF UINT256 VARIABLES TO 0 - INFORMATIONAL

Description:

As `i` is an `uint256`, it is already initialized to 0. `uint256 i = 0` reassigns the 0 to `i` which wastes gas.

Code Location:

Stargate.sol
- Line 52: `amntToTransfer = 0;`

IMXVault.sol
- Line 39: `amntToTransfer = 0;`

IMXLend.sol
- Line 41: `amntToTransfer = 0;`

HLPVault.sol
- Line 35: `amntToTransfer = 0;`

SCYVault.sol
- Line 284: `for (uint256 i = 0; i < l; i++){`

SectorBase.sol
- Line 76: `for (uint256 i = 0; i < l; i++){`

StratAuth.sol
- Line 22: `for (uint256 i = 0; i < l; i++){`

SectorTimelock.sol
- Line 84: `for (uint256 i = 0; i < proposers.length; ++i){`
- Line 89: `for (uint256 i = 0; i < executors.length; ++i){`
- Line 234: `for (uint256 i = 0; i < targets.length; ++i){`

- Line 308: `for (uint256 i = 0; i < targets.length; ++i){`

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

It is recommended to not initialize uint256 variables to 0 to save some gas. For example, use instead:
`for (uint256 i; i < proposal.targets.length; ++i)`.

Remediation Plan:

SOLVED: The Sector Finance team solved this issue in commit 404af4b1266ea814c76d578f700a5cb629dfe61a: all related instances of unit256 variables initialization are now removed.

3.15 (HAL-15) GAS OVER-CONSUMPTION IN LOOPS - INFORMATIONAL

Description:

In all the loops, the counter variable is incremented using `i++`. It is known that, in loops, using `++i` costs less gas per iteration than `i++`.

Code Location:

SCYVault.sol
- Line 284: `for (uint256 i = 0; i < l; i++){`

SectorBase.sol
- Line 76: `for (uint256 i = 0; i < l; i++){`

StratAuth.sol
- Line 22: `for (uint256 i = 0; i < l; i++){`

Proof of Concept:

For example, based in the following test contract:

Listing 22: Test.sol

```
1 //SPDX-License-Identifier: MIT
2 pragma solidity 0.8.9;
3
4 contract test {
5     function postiincrement(uint256 iterations) public {
6         for (uint256 i = 0; i < iterations; i++) {
7         }
8     }
9     function preiincrement(uint256 iterations) public {
10        for (uint256 i = 0; i < iterations; ++i) {
11        }
12    }
13 }
```

We can see the difference in the gas costs:

```
>>> test_contract.postiincrement(1)
Transaction sent: 0x1ecede6b109b707786d3685bd71dd9f22dc389957653036ca04c4cd2e72c5e0b
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 44
  test.postiincrement confirmed   Block: 13622335   Gas used: 21620 (0.32%)

<Transaction '0x1ecede6b109b707786d3685bd71dd9f22dc389957653036ca04c4cd2e72c5e0b'>
>>> test_contract.preiincrement(1)
Transaction sent: 0x205f09a4d2268de4c1a40f35bb2ec2847bf2ab8d584909b42c71a022b047614a
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 45
  test.preiincrement confirmed   Block: 13622336   Gas used: 21593 (0.32%)

<Transaction '0x205f09a4d2268de4c1a40f35bb2ec2847bf2ab8d584909b42c71a022b047614a'>
>>> test_contract.postiincrement(10)
Transaction sent: 0x98c04430526a59balcf947c114b62666a4417165947d31bf300cd6ae68328033
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 46
  test.postiincrement confirmed   Block: 13622337   Gas used: 22673 (0.34%)

<Transaction '0x98c04430526a59balcf947c114b62666a4417165947d31bf300cd6ae68328033'>
>>> test_contract.preiincrement(10)
Transaction sent: 0xf060d04714eff8482a828342414d5a20be9958c822d42860e7992aba20e1de05
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 47
  test.preiincrement confirmed   Block: 13622338   Gas used: 22601 (0.34%)

<Transaction '0xf060d04714eff8482a828342414d5a20be9958c822d42860e7992aba20e1de05'>
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

It is recommended to use `++i` instead of `i++` to increment the value of an `uint` variable inside a loop to save some gas. This is not applicable outside of loops.

Remediation Plan:

SOLVED: The Sector Finance team solved this issue in commit `404af4b1266ea814c76d578f700a5cb629dfe61a`: all related instances of `i++` are now updated to `++i`.

AUTOMATED TESTING

4.1 STATIC ANALYSIS REPORT

Description:

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

Slither results:

HLPVault.sol

```
HLPCore._rebalanceLoan(uint256) (src/strategies/hlp/HLPCore.sol#182-199) performs a multiplication on the result of a division:
        -targetHealth = (10000 * 1e18) / _safeCollateralRatio (src/strategies/hlp/HLPCore.sol#187)
        -addCollateral = (1e18 * ((collateral * targetHealth) / _loanHealth - collateral)) / ((targetHealth * 1e18) / _getCollateralFactor()
 + 1e18) (src/strategies/hlp/HLPCore.sol#188-189)
IMXFarm._optimalUBorrow() (src/strategies/imx/IMXFarm.sol#327-332) performs a multiplication on the result of a division:
        -s = (_collateralToken.safetyMarginSqrt() * safetyMarginSqrt()) / 1e18 (src/strategies/imx/IMXFarm.sol#330)
        -uBorrow = (1e18 * (2e18 - (l * s) / 1e18)) / ((l * 1e18) / s + (l * s) / 1e18 - 2e18) (src/strategies/imx/IMXFarm.sol#331)
SCYVault._redeem(address,address,uint256) (src/vaults/ERC5115/SCYVault.sol#117-154) performs a multiplication on the result of a division:
        -sharesToRedeem = (sharesToRedeem * _totalSupply) / (_totalSupply + lockedProfit()) (src/vaults/ERC5115/SCYVault.sol#127)
        -shareOfReserves = (reserves * sharesToRedeem) / _totalSupply (src/vaults/ERC5115/SCYVault.sol#133)
SCYVault.underlyingBalance(address) (src/vaults/ERC5115/SCYVault.sol#335-342) performs a multiplication on the result of a division:
        -adjustedShares = (userBalance * _totalSupply) / (_totalSupply + lockedProfit()) (src/vaults/ERC5115/SCYVault.sol#340)
        -(tvl * adjustedShares) / _totalSupply (src/vaults/ERC5115/SCYVault.sol#341)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-multiply
```

```
ILending._adjustCollateral(uint256) (src/strategies/mixins/ILending.sol#56-65) uses a dangerous strict equality:
        - collateralBalance == targetCollateral (src/strategies/mixins/ILending.sol#61)
HLPCore._decreaseULpTo(uint256) (src/strategies/hlp/HLPCore.sol#424-436) uses a dangerous strict equality:
        - removeLp == 0 (src/strategies/hlp/HLPCore.sol#435)
HLPCore._increaseLpPosition(uint256) (src/strategies/hlp/HLPCore.sol#475-514) uses a dangerous strict equality:
        - addUnderlying == 0 (src/strategies/hlp/HLPCore.sol#507)
IUniLp._quote(uint256,address,address) (src/strategies/mixins/IUniLp.sol#47-55) uses a dangerous strict equality:
        - amount == 0 (src/strategies/mixins/IUniLp.sol#52)
IUniLp._shortToUnderlying(uint256) (src/strategies/mixins/IUniLp.sol#76-78) uses a dangerous strict equality:
        - amount == 0 (src/strategies/mixins/IUniLp.sol#77)
IUniLp._underlyingToShort(uint256) (src/strategies/mixins/IUniLp.sol#81-83) uses a dangerous strict equality:
        - amount == 0 (src/strategies/mixins/IUniLp.sol#82)
HLPCore.getAndUpdateTVL() (src/strategies/hlp/HLPCore.sol#538-548) uses a dangerous strict equality:
        - shortP == 0 (src/strategies/hlp/HLPCore.sol#543)
HLPCore.getPositionOffset() (src/strategies/hlp/HLPCore.sol#596-609) uses a dangerous strict equality:
        - shortBalance == borrowBalance (src/strategies/hlp/HLPCore.sol#601)
HLPCore.getTVL() (src/strategies/hlp/HLPCore.sol#560-586) uses a dangerous strict equality:
        - shortPosition == 0 (src/strategies/hlp/HLPCore.sol#577)
IMXFarm._getLiquidity(uint256) (src/strategies/imx/IMXFarm.sol#300-306) uses a dangerous strict equality:
        - balance == 0 (src/strategies/imx/IMXFarm.sol#301)
IMXFarm._harvestFarm(HarvestSwapParams) (src/strategies/imx/IMXFarm.sol#276-294) uses a dangerous strict equality:
        - harvested == 0 (src/strategies/imx/IMXFarm.sol#290)
IMXCore.getAndUpdateTVL() (src/strategies/imx/IMXCore.sol#320-331) uses a dangerous strict equality:
        - shortP == 0 (src/strategies/imx/IMXCore.sol#324-326)
IMXCore.getPositionOffset() (src/strategies/imx/IMXCore.sol#363-375) uses a dangerous strict equality:
        - shortBalance == borrowBalance (src/strategies/imx/IMXCore.sol#367)
IMXCore.getTVL() (src/strategies/imx/IMXCore.sol#337-361) uses a dangerous strict equality:
        - shortPosition == 0 (src/strategies/imx/IMXCore.sol#354)
```

```
IMXFarm.loanHealth() (src/strategies/imx/IMXFarm.sol#334-340) uses a dangerous strict equality:
        - balance == 0 (src/strategies/imx/IMXFarm.sol#336)
IMXFarm.loanHealth_() (src/strategies/imx/IMXFarm.sol#342-348) uses a dangerous strict equality:
        - balance == 0 (src/strategies/imx/IMXFarm.sol#344)
Accounting.convertToAssets(uint256) (src/common/Accounting.sol#29-33) uses a dangerous strict equality:
        - supply == 0 (src/common/Accounting.sol#32)
Accounting.convertToShares(uint256) (src/common/Accounting.sol#23-27) uses a dangerous strict equality:
        - supply == 0 (src/common/Accounting.sol#26)
SCYBase.deposit(address,address,uint256,uint256) (src/vaults/ERC5115/SCYBase.sol#44-72) uses a dangerous strict equality:
        - amountIn == 0 (src/vaults/ERC5115/SCYBase.sol#58)
SCYBase.deposit(address,address,uint256,uint256) (src/vaults/ERC5115/SCYBase.sol#44-72) uses a dangerous strict equality:
        - totalSupply() == 0 (src/vaults/ERC5115/SCYBase.sol#64)
SCYVault.exchangeRateCurrent() (src/vaults/ERC5115/SCYVault.sol#401-405) uses a dangerous strict equality:
        - _totalSupply == 0 (src/vaults/ERC5115/SCYVault.sol#403)
SCYVault.exchangeRateUnderlying() (src/vaults/ERC5115/SCYVault.sol#320-325) uses a dangerous strict equality:
        - _totalSupply == 0 (src/vaults/ERC5115/SCYVault.sol#322)
SCYVault.getUpdatedUnderlyingBalance(address) (src/vaults/ERC5115/SCYVault.sol#327-333) uses a dangerous strict equality:
        - _totalSupply == 0 || userBalance == 0 (src/vaults/ERC5115/SCYVault.sol#330)
SCYVault.harvest(uint256,uint256,HarvestSwapParams[],HarvestSwapParams[]) (src/vaults/ERC5115/SCYVault.sol#157-211) uses a dangerous strict
equality:
        - profit == 0 (src/vaults/ERC5115/SCYVault.sol#178)
```

```
Accounting.previewMint(uint256) (src/common/Accounting.sol#39-43) uses a dangerous strict equality:
        - supply == 0 (src/common/Accounting.sol#42)
Accounting.previewWithdraw(uint256) (src/common/Accounting.sol#45-48) uses a dangerous strict equality:
        - supply == 0 (src/common/Accounting.sol#47)
SCYVault.sharesToUnderlying(uint256) (src/vaults/ERC5115/SCYVault.sol#350-355) uses a dangerous strict equality:
        - _totalSupply == 0 (src/vaults/ERC5115/SCYVault.sol#352)
Accounting.toSharesAfterDeposit(uint256) (src/common/Accounting.sol#16-21) uses a dangerous strict equality:
        - _totalAssets == 0 (src/common/Accounting.sol#19)
Accounting.toSharesAfterDeposit(uint256) (src/common/Accounting.sol#16-21) uses a dangerous strict equality:
        - supply == 0 (src/common/Accounting.sol#20)
SCYVault.underlyingBalance(address) (src/vaults/ERC5115/SCYVault.sol#335-342) uses a dangerous strict equality:
        - _totalSupply == 0 || userBalance == 0 (src/vaults/ERC5115/SCYVault.sol#338)
SCYVault.underlyingToShares(uint256) (src/vaults/ERC5115/SCYVault.sol#344-348) uses a dangerous strict equality:
        - _totalSupply == 0 (src/vaults/ERC5115/SCYVault.sol#346)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities

Reentrancy in HLPCore.__HedgedLP_init_(address,address,uint256,address) (src/strategies/hlp/HLPCore.sol#81-107):
        External calls:
        - _underlying.safeApprove(address(this),type()(uint256).max) (src/strategies/hlp/HLPCore.sol#92)
        State variables written after the call(s):
        - isInitialized = true (src/strategies/hlp/HLPCore.sol#106)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1
```

```
SCYVault.harvest(uint256,uint256,HarvestSwapParams[],HarvestSwapParams[]).newLockedProfit (src/vaults/ERC5115/SCYVault.sol#201) is a local v
ariable never initialized
SCYVault.harvest(uint256,uint256,HarvestSwapParams[],HarvestSwapParams[]).feeShares (src/vaults/ERC5115/SCYVault.sol#184) is a local variabl
e never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables

HLPCore._closePosition() (src/strategies/hlp/HLPCore.sol#402-422) ignores return value by pair()._swapTokensForExactTokens(shortPosition - s
hortBalance,address(_underlying),address(_short)) (src/strategies/hlp/HLPCore.sol#407-411)
HLPCore._closePosition() (src/strategies/hlp/HLPCore.sol#402-422) ignores return value by pair()._swapExactTokensForTokens(shortBalance - sh
ortPosition,address(_short),address(_underlying)) (src/strategies/hlp/HLPCore.sol#413-417)
IMXFarm._addIMXLiquidity(uint256,uint256,uint256,uint256) (src/strategies/imx/IMXFarm.sol#79-107) ignores return value by _sBorrowable.borro
wApprove(address(_sBorrowable),sBorrow) (src/strategies/imx/IMXFarm.sol#85)
IMXFarm._addLp(uint256,uint256) (src/strategies/imx/IMXFarm.sol#134-181) ignores return value by stakedToken.mint(address(_collateralToken))
 (src/strategies/imx/IMXFarm.sol#179)
IMXFarm._addLp(uint256,uint256) (src/strategies/imx/IMXFarm.sol#134-181) ignores return value by _collateralToken.mint(address(this)) (src/s
trategies/imx/IMXFarm.sol#180)
IMXFarm.impermaxRedeem(address,uint256,bytes) (src/strategies/imx/IMXFarm.sol#201-263) ignores return value by stakedToken.redeem(address(th
is)) (src/strategies/imx/IMXFarm.sol#212)
IMXFarm.impermaxRedeem(address,uint256,bytes) (src/strategies/imx/IMXFarm.sol#201-263) ignores return value by pair()._swapExactTokensForTok
ens(shortAmnt - d.repayShort,address(short()),address(underlying())) (src/strategies/imx/IMXFarm.sol#222-226)
IMXFarm.impermaxRedeem(address,uint256,bytes) (src/strategies/imx/IMXFarm.sol#201-263) ignores return value by pair()._swapTokensForExactTok
ens(d.repayShort - shortAmnt,address(underlying()),address(short())) (src/strategies/imx/IMXFarm.sol#231-235)
IFarmable._swap(IUniswapV2Router01,HarvestSwapParams,address,uint256) (src/strategies/mixins/IFarmable.sol#13-34) ignores return value by ro
uter.swapExactTokensForTokens(amount,swapParams.min,swapParams.path,address(this),swapParams.deadline) (src/strategies/mixins/IFarmable.sol#
27-33)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
```

```
StratAuth.emergencyAction(EAction[]).target (src/common/StratAuth.sol#23) lacks a zero-check on :
                - (success) = target.call{value: actions[i].value}(data) (src/common/StratAuth.sol#25)
Auth.transferOwnership(address)._pendingOwner (src/common/Auth.sol#59) lacks a zero-check on :
                - pendingOwner = _pendingOwner (src/common/Auth.sol#60)
SCYVault.initStrategy(address)._strategy (src/vaults/ERC5115/SCYVault.sol#82) lacks a zero-check on :
                - strategy = address(_strategy) (src/vaults/ERC5115/SCYVault.sol#84)
SCYVault.updateStrategy(address)._strategy (src/vaults/ERC5115/SCYVault.sol#89) lacks a zero-check on :
                - strategy = address(_strategy) (src/vaults/ERC5115/SCYVault.sol#92)
SCYVault.emergencyAction(EAction[]).target (src/vaults/ERC5115/SCYVault.sol#285) lacks a zero-check on :
                - (success) = target.call{value: actions[i].value}(data) (src/vaults/ERC5115/SCYVault.sol#287)
Fees.setTreasury(address)._treasury (src/common/Fees.sol#58) lacks a zero-check on :
                - treasury = _treasury (src/common/Fees.sol#59)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
```

```
StratAuth.emergencyAction(EAction[]) (src/common/StratAuth.sol#20-29) has external calls inside a loop: (success) = target.call{value: actio
ns[i].value}(data) (src/common/StratAuth.sol#25)
SCYVault.emergencyAction(EAction[]) (src/vaults/ERC5115/SCYVault.sol#282-291) has external calls inside a loop: (success) = target.call{valu
e: actions[i].value}(data) (src/vaults/ERC5115/SCYVault.sol#287)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-a-loop
```

```
SafeETH.safeTransferETH(address,uint256) (src/libraries/SafeETH.sol#5-14) uses assembly
        - INLINE ASM (src/libraries/SafeETH.sol#8-11)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

IBase.initializer() (src/strategies/mixins/IBase.sol#11-14) compares to a boolean constant:
        -require(bool,string)(isInitialized == false,INITIALIZED) (src/strategies/mixins/IBase.sol#12)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#boolean-equality

Low level call in StratAuth.emergencyAction(EAction[]) (src/common/StratAuth.sol#20-29):
        - (success) = target.call{value: actions[i].value}(data) (src/common/StratAuth.sol#25)
Low level call in SCYVault.emergencyAction(EAction[]) (src/vaults/ERC5115/SCYVault.sol#282-291):
        - (success) = target.call{value: actions[i].value}(data) (src/vaults/ERC5115/SCYVault.sol#287)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter Auth.transferOwnership(address)._pendingOwner (src/common/Auth.sol#59) is not in mixedCase
Parameter Fees.setPerformanceFee(uint256)._performanceFee (src/common/Fees.sol#40) is not in mixedCase
Parameter Fees.setManagementFee(uint256)._managementFee (src/common/Fees.sol#49) is not in mixedCase
Parameter Fees.setTreasury(address)._treasury (src/common/Fees.sol#58) is not in mixedCase
Function ISCYStrategy.MIN_LIQUIDITY() (src/interfaces/ERC5115/ISCYStrategy.sol#26) is not in mixedCase

Parameter Auth.transferOwnership(address)._pendingOwner (src/common/Auth.sol#59) is not in mixedCase
Parameter Fees.setPerformanceFee(uint256)._performanceFee (src/common/Fees.sol#40) is not in mixedCase
Parameter Fees.setManagementFee(uint256)._managementFee (src/common/Fees.sol#49) is not in mixedCase
Parameter Fees.setTreasury(address)._treasury (src/common/Fees.sol#58) is not in mixedCase
Function ISCYStrategy.MIN_LIQUIDITY() (src/interfaces/ERC5115/ISCYStrategy.sol#26) is not in mixedCase

Variable IMXFarm._pair (src/strategies/imx/IMXFarm.sol#18) is not in mixedCase
Function IIMXFarm._getBorrowBalances() (src/strategies/mixins/IIMXFarm.sol#25-29) is not in mixedCase
Function IIMXFarm._updateAndGetBorrowBalances() (src/strategies/mixins/IIMXFarm.sol#31-34) is not in mixedCase
Function IUniLp._shortToUnderlying(uint256) (src/strategies/mixins/IUniLp.sol#76-78) is not in mixedCase
Function IUniLp._underlyingToShort(uint256) (src/strategies/mixins/IUniLp.sol#81-83) is not in mixedCase
Parameter SCYVault.setMaxTvl(uint256)._maxTvl (src/vaults/ERC5115/SCYVault.sol#77) is not in mixedCase
Parameter SCYVault.initStrategy(address)._strategy (src/vaults/ERC5115/SCYVault.sol#82) is not in mixedCase
Parameter SCYVault.updateStrategy(address)._strategy (src/vaults/ERC5115/SCYVault.sol#89) is not in mixedCase
Constant SCYVault.vaultType (src/vaults/ERC5115/SCYVault.sol#22) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

HLPCore.harvestIsEnabled (src/strategies/hlp/HLPCore.sol#59) should be constant
HLPCore.lastHarvest (src/strategies/hlp/HLPCore.sol#43) should be constant
SCYBase.sendERC20ToStrategy (src/vaults/ERC5115/SCYBase.sol#27) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
```

AggregatorVault.sol

```
Reentrancy in ERC4626.deposit(uint256,address) (src/vaults/ERC4626/ERC4626.sol#61-91):
        External calls:
        - IWETH(address(asset)).deposit{value: assets}() (src/vaults/ERC4626/ERC4626.sol#76)
        - asset.safeTransferFrom(msg.sender,address(this),assets) (src/vaults/ERC4626/ERC4626.sol#77)
        External calls sending eth:
        - IWETH(address(asset)).deposit{value: assets}() (src/vaults/ERC4626/ERC4626.sol#76)
        State variables written after the call(s):
        - _mint(address(1),MIN_LIQUIDITY) (src/vaults/ERC4626/ERC4626.sol#83)
                - _totalSupply += amount (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#262)
        - _mint(receiver,shares) (src/vaults/ERC4626/ERC4626.sol#86)
                - _totalSupply += amount (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#262)
Reentrancy in ERC4626.mint(uint256,address) (src/vaults/ERC4626/ERC4626.sol#93-119):
        External calls:
        - IWETH(address(asset)).deposit{value: assets}() (src/vaults/ERC4626/ERC4626.sol#104)
        - asset.safeTransferFrom(msg.sender,address(this),assets) (src/vaults/ERC4626/ERC4626.sol#105)
        External calls sending eth:
        - IWETH(address(asset)).deposit{value: assets}() (src/vaults/ERC4626/ERC4626.sol#104)
        State variables written after the call(s):
        - _mint(address(1),MIN_LIQUIDITY) (src/vaults/ERC4626/ERC4626.sol#111)
                - _totalSupply += amount (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#262)
        - _mint(receiver,shares) (src/vaults/ERC4626/ERC4626.sol#114)
                - _totalSupply += amount (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#262)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities
```

```
Accounting.convertToAssets(uint256) (src/common/Accounting.sol#29-33) uses a dangerous strict equality:
        - supply == 0 (src/common/Accounting.sol#32)
Accounting.convertToShares(uint256) (src/common/Accounting.sol#23-27) uses a dangerous strict equality:
        - supply == 0 (src/common/Accounting.sol#26)
ERC4626.deposit(uint256,address) (src/vaults/ERC4626/ERC4626.sol#61-91) uses a dangerous strict equality:
        - totalSupply() == 0 (src/vaults/ERC4626/ERC4626.sol#80)
AggregatorVault.emergencyRedeem() (src/vaults/sectorVaults/AggregatorVault.sol#183-212) uses a dangerous strict equality:
        - shares == 0 (src/vaults/sectorVaults/AggregatorVault.sol#187)
AggregatorVault.emergencyRedeem() (src/vaults/sectorVaults/AggregatorVault.sol#183-212) uses a dangerous strict equality:
        - userShares == 0 (src/vaults/sectorVaults/AggregatorVault.sol#207)
ERC4626.mint(uint256,address) (src/vaults/ERC4626/ERC4626.sol#93-119) uses a dangerous strict equality:
        - useNativeAsset && msg.value == assets (src/vaults/ERC4626/ERC4626.sol#104)
ERC4626.mint(uint256,address) (src/vaults/ERC4626/ERC4626.sol#93-119) uses a dangerous strict equality:
        - totalSupply() == 0 (src/vaults/ERC4626/ERC4626.sol#108)
Accounting.previewMint(uint256) (src/common/Accounting.sol#39-43) uses a dangerous strict equality:
        - supply == 0 (src/common/Accounting.sol#42)
Accounting.previewWithdraw(uint256) (src/common/Accounting.sol#45-48) uses a dangerous strict equality:
        - supply == 0 (src/common/Accounting.sol#47)
AggregatorVault.sharesToUnderlying(uint256) (src/vaults/sectorVaults/AggregatorVault.sol#253-256) uses a dangerous strict equality:
        - supply == 0 (src/vaults/sectorVaults/AggregatorVault.sol#255)
Accounting.toSharesAfterDeposit(uint256) (src/common/Accounting.sol#16-21) uses a dangerous strict equality:
        - _totalAssets == 0 (src/common/Accounting.sol#19)
Accounting.toSharesAfterDeposit(uint256) (src/common/Accounting.sol#16-21) uses a dangerous strict equality:
        - supply == 0 (src/common/Accounting.sol#20)
AggregatorVault.underlyingToShares(uint256) (src/vaults/sectorVaults/AggregatorVault.sol#260-263) uses a dangerous strict equality:
        - supply == 0 (src/vaults/sectorVaults/AggregatorVault.sol#262)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities
```

```
Reentrancy in AggregatorVault.depositIntoStrategies(DepositParams[]) (src/vaults/sectorVaults/AggregatorVault.sol#109-138):
        External calls:
        - asset.safeTransfer(strategy.strategy(),amountIn) (src/vaults/sectorVaults/AggregatorVault.sol#125)
        - asset.safeTransfer(address(strategy),amountIn) (src/vaults/sectorVaults/AggregatorVault.sol#126)
        - strategy.deposit(address(this),address(asset),0,param.minSharesOut) (src/vaults/sectorVaults/AggregatorVault.sol#129)
        - asset.safeApprove(address(strategy),amountIn) (src/vaults/sectorVaults/AggregatorVault.sol#131)
        - strategy.deposit(amountIn,address(this)) (src/vaults/sectorVaults/AggregatorVault.sol#132)
        State variables written after the call(s):
        - totalChildHoldings += amountIn (src/vaults/sectorVaults/AggregatorVault.sol#134)
Reentrancy in AggregatorVault.emergencyRedeem() (src/vaults/sectorVaults/AggregatorVault.sol#183-212):
        External calls:
        - asset.safeTransfer(msg.sender,underlyingShare) (src/vaults/sectorVaults/AggregatorVault.sol#197)
        State variables written after the call(s):
        - _burn(msg.sender,shares) (src/vaults/sectorVaults/AggregatorVault.sol#211)
                - _balances[account] = accountBalance - amount (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#288)
        - _burn(msg.sender,shares) (src/vaults/sectorVaults/AggregatorVault.sol#211)
                - _totalSupply -= amount (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#290)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1
```

```
AggregatorVault.requestRedeemFromStrategies(RedeemParams[]).i (src/vaults/sectorVaults/AggregatorVault.sol#142) is a local variable never initialized
AggregatorVault._getStrategyHoldings().i (src/vaults/sectorVaults/AggregatorVault.sol#218) is a local variable never initialized
AggregatorVault.withdrawFromStrategies(RedeemParams[]).i (src/vaults/sectorVaults/AggregatorVault.sol#151) is a local variable never initialized
AggregatorVault.depositIntoStrategies(DepositParams[]).i (src/vaults/sectorVaults/AggregatorVault.sol#111) is a local variable never initialized
AggregatorVault.removeStrategy(IVaultStrategy).i (src/vaults/sectorVaults/AggregatorVault.sol#84) is a local variable never initialized
AggregatorVault.emergencyRedeem().i (src/vaults/sectorVaults/AggregatorVault.sol#203) is a local variable never initialized
AggregatorVault.withdrawFromStrategies(RedeemParams[]).amountOut (src/vaults/sectorVaults/AggregatorVault.sol#158) is a local variable never initialized
SectorBase._harvest(uint256).feeShares (src/vaults/ERC4626/SectorBase.sol#57) is a local variable never initialized
AggregatorVault.getTvl().i (src/vaults/sectorVaults/AggregatorVault.sol#228) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
```

```
AggregatorVault.depositIntoStrategies(DepositParams[]) (src/vaults/sectorVaults/AggregatorVault.sol#109-138) ignores return value by strategy.deposit(address(this),addres
s(asset),0,param.minSharesOut) (src/vaults/sectorVaults/AggregatorVault.sol#129)
AggregatorVault.depositIntoStrategies(DepositParams[]) (src/vaults/sectorVaults/AggregatorVault.sol#109-138) ignores return value by strategy.deposit(amountIn,address(thi
s)) (src/vaults/sectorVaults/AggregatorVault.sol#132)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
```

```
SectorBase.emergencyAction(EAction[]) (src/vaults/ERC4626/SectorBase.sol#74-83) has external calls inside a loop: (success) = target.call{value: actions[i].value}(data) (
src/vaults/ERC4626/SectorBase.sol#79)
AggregatorVault.depositIntoStrategies(DepositParams[]) (src/vaults/sectorVaults/AggregatorVault.sol#109-138) has external calls inside a loop: strategy.sendERC20ToStrateg
y() == true (src/vaults/sectorVaults/AggregatorVault.sol#124)
AggregatorVault.depositIntoStrategies(DepositParams[]) (src/vaults/sectorVaults/AggregatorVault.sol#109-138) has external calls inside a loop: asset.safeTransfer(strategy
.strategy(),amountIn) (src/vaults/sectorVaults/AggregatorVault.sol#125)
AggregatorVault.depositIntoStrategies(DepositParams[]) (src/vaults/sectorVaults/AggregatorVault.sol#109-138) has external calls inside a loop: strategy.deposit(address(th
is),address(asset),0,param.minSharesOut) (src/vaults/sectorVaults/AggregatorVault.sol#129)
AggregatorVault.depositIntoStrategies(DepositParams[]) (src/vaults/sectorVaults/AggregatorVault.sol#109-138) has external calls inside a loop: strategy.deposit(amountIn,a
ddress(this)) (src/vaults/sectorVaults/AggregatorVault.sol#132)
AggregatorVault.requestRedeemFromStrategies(RedeemParams[]) (src/vaults/sectorVaults/AggregatorVault.sol#140-146) has external calls inside a loop: params[i].strategy.req
uestRedeem(params[i].shares) (src/vaults/sectorVaults/AggregatorVault.sol#144)
AggregatorVault.withdrawFromStrategies(RedeemParams[]) (src/vaults/sectorVaults/AggregatorVault.sol#149-178) has external calls inside a loop: amountOut = strategy.redeem
(address(this),shares,address(asset),param.minTokenOut) (src/vaults/sectorVaults/AggregatorVault.sol#161-166)
AggregatorVault.withdrawFromStrategies(RedeemParams[]) (src/vaults/sectorVaults/AggregatorVault.sol#149-178) has external calls inside a loop: amountOut = strategy.redeem
() (src/vaults/sectorVaults/AggregatorVault.sol#167)
AggregatorVault.emergencyRedeem() (src/vaults/sectorVaults/AggregatorVault.sol#183-212) has external calls inside a loop: balance = stratToken.balanceOf(address(this)) (s
rc/vaults/sectorVaults/AggregatorVault.sol#205)
AggregatorVault.getTvl() (src/vaults/sectorVaults/AggregatorVault.sol#225-233) has external calls inside a loop: tvl += strategy.underlyingBalance(address(this)) (src/vau
lts/sectorVaults/AggregatorVault.sol#230)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-a-loop
```

```
AggregatorVault.depositIntoStrategies(DepositParams[]) (src/vaults/sectorVaults/AggregatorVault.sol#109-138) compares to a boolean constant:
        -strategy.sendERC20ToStrategy() == true (src/vaults/sectorVaults/AggregatorVault.sol#124)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#boolean-equality
```

```
SectorBase.beforeWithdraw(uint256,uint256) (src/vaults/ERC4626/SectorBase.sol#130-136) has costly operations inside a loop:
        - floatAmnt -= assets (src/vaults/ERC4626/SectorBase.sol#135)
AggregatorVault.depositIntoStrategies(DepositParams[]) (src/vaults/sectorVaults/AggregatorVault.sol#109-138) has costly operations inside a loop:
        - totalChildHoldings += amountIn (src/vaults/sectorVaults/AggregatorVault.sol#134)
AggregatorVault.withdrawFromStrategies(RedeemParams[]) (src/vaults/sectorVaults/AggregatorVault.sol#149-178) has costly operations inside a loop:
        - totalChildHoldings = 0 (src/vaults/sectorVaults/AggregatorVault.sol#170-172)
SectorBase.afterDeposit(uint256,uint256) (src/vaults/ERC4626/SectorBase.sol#123-128) has costly operations inside a loop:
        - floatAmnt += assets (src/vaults/ERC4626/SectorBase.sol#127)
AggregatorVault.withdrawFromStrategies(RedeemParams[]) (src/vaults/sectorVaults/AggregatorVault.sol#149-178) has costly operations inside a loop:
        - totalChildHoldings = totalChildHoldings - amountOut (src/vaults/sectorVaults/AggregatorVault.sol#170-172)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-operations-inside-a-loop
```

```
AggregatorVault.MAX_STRATS (src/vaults/sectorVaults/AggregatorVault.sol#38) should be constant
AggregatorVault.totalStrategyHoldings (src/vaults/sectorVaults/AggregatorVault.sol#43) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
```

IMXLend.sol

```
IMXLend._stratGetAndUpdateTvl().underlyingBalance (src/vaults/strategyVaults/IMXLend.sol#49) shadows:
        - SCYVault.underlyingBalance(address) (src/vaults/ERC5115/SCYVault.sol#335-342) (function)
        - ISCYStrategy.underlyingBalance(address) (src/interfaces/ERC5115/ISCYStrategy.sol#34) (function)
IMXLend._strategyTvl().underlyingBalance (src/vaults/strategyVaults/IMXLend.sol#56) shadows:
        - SCYVault.underlyingBalance(address) (src/vaults/ERC5115/SCYVault.sol#335-342) (function)
        - ISCYStrategy.underlyingBalance(address) (src/interfaces/ERC5115/ISCYStrategy.sol#34) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
```

```
Pragma version0.8.16 (src/vaults/strategyVaults/IMXLend.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
```

```
Variable IMXLend._stratClosePosition(uint256).yeildTokenAmnt (src/vaults/strategyVaults/IMXLend.sol#61) is too similar to SCYVault.withdrawFromStrategy(uint256,uint256).yieldTokenAmnt (src/vaults/ERC5115/SCYVault.sol#265)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
```

IMXVault.sol

```
Pragma version0.8.16 (src/vaults/strategyVaults/IMXVault.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
```

```
Variable IMXVault._stratRedeem(address,uint256).yeildTokenAmnt (src/vaults/strategyVaults/IMXVault.sol#31) is too similar to SCYVault.withdrawFromStrategy(uint256,uint256).yieldTokenAmnt (src/vaults/ERC5115/SCYVault.sol#265)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
```

Stargate.sol

```
Pragma version0.8.16 (src/interfaces/stargate/IStargatePool.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.16 (src/interfaces/stargate/IStargateRouter.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
```

```
Pragma version0.8.16 (src/vaults/strategyVaults/Stargate.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.16 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
```

```
Parameter Stargate.redeemRemote(uint16,uint256,uint256,address,uint256,uint256,bytes,lzTxObj)._dstChainId (src/vaults/strategyVaults/Stargate.sol#98) is not in mixedCase
Parameter Stargate.redeemRemote(uint16,uint256,uint256,address,uint256,uint256,bytes,lzTxObj)._srcPoolId (src/vaults/strategyVaults/Stargate.sol#99) is not in mixedCase
Parameter Stargate.redeemRemote(uint16,uint256,uint256,address,uint256,uint256,bytes,lzTxObj)._dstPoolId (src/vaults/strategyVaults/Stargate.sol#100) is not in mixedCase
Parameter Stargate.redeemRemote(uint16,uint256,uint256,address,uint256,uint256,bytes,lzTxObj)._refundAddress (src/vaults/strategyVaults/Stargate.sol#101) is not in mixedCase
Parameter Stargate.redeemRemote(uint16,uint256,uint256,address,uint256,uint256,bytes,lzTxObj)._amountLP (src/vaults/strategyVaults/Stargate.sol#102) is not in mixedCase
Parameter Stargate.redeemRemote(uint16,uint256,uint256,address,uint256,uint256,bytes,lzTxObj)._minAmountLD (src/vaults/strategyVaults/Stargate.sol#103) is not in mixedCase
Parameter Stargate.redeemRemote(uint16,uint256,uint256,address,uint256,uint256,bytes,lzTxObj)._to (src/vaults/strategyVaults/Stargate.sol#104) is not in mixedCase
Parameter Stargate.redeemRemote(uint16,uint256,uint256,address,uint256,uint256,bytes,lzTxObj)._lzTxParams (src/vaults/strategyVaults/Stargate.sol#105) is not in mixedCase
Parameter Stargate.redeemLocal(uint16,uint256,uint256,address,uint256,bytes,lzTxObj)._dstChainId (src/vaults/strategyVaults/Stargate.sol#120) is not in mixedCase
Parameter Stargate.redeemLocal(uint16,uint256,uint256,address,uint256,bytes,lzTxObj)._srcPoolId (src/vaults/strategyVaults/Stargate.sol#121) is not in mixedCase
Parameter Stargate.redeemLocal(uint16,uint256,uint256,address,uint256,bytes,lzTxObj)._dstPoolId (src/vaults/strategyVaults/Stargate.sol#122) is not in mixedCase
Parameter Stargate.redeemLocal(uint16,uint256,uint256,address,uint256,bytes,lzTxObj)._refundAddress (src/vaults/strategyVaults/Stargate.sol#123) is not in mixedCase
Parameter Stargate.redeemLocal(uint16,uint256,uint256,address,uint256,bytes,lzTxObj)._amountLP (src/vaults/strategyVaults/Stargate.sol#124) is not in mixedCase
Parameter Stargate.redeemLocal(uint16,uint256,uint256,address,uint256,bytes,lzTxObj)._to (src/vaults/strategyVaults/Stargate.sol#125) is not in mixedCase
Parameter Stargate.redeemLocal(uint16,uint256,uint256,address,uint256,bytes,lzTxObj)._lzTxParams (src/vaults/strategyVaults/Stargate.sol#126) is not in mixedCase
Parameter Stargate.sendCredits(uint16,uint256,uint256,address)._dstChainId (src/vaults/strategyVaults/Stargate.sol#140) is not in mixedCase
Parameter Stargate.sendCredits(uint16,uint256,uint256,address)._srcPoolId (src/vaults/strategyVaults/Stargate.sol#141) is not in mixedCase
Parameter Stargate.sendCredits(uint16,uint256,uint256,address)._dstPoolId (src/vaults/strategyVaults/Stargate.sol#142) is not in mixedCase
Parameter Stargate.sendCredits(uint16,uint256,uint256,address)._refundAddress (src/vaults/strategyVaults/Stargate.sol#143) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

Synapse.sol

```
Pragma version0.8.16 (src/vaults/strategyVaults/Synapse.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.16 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
```

```
Variable Synapse._nTokens (src/vaults/strategyVaults/Synapse.sol#21) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

SafeETH.sol

```
SafeETH.safeTransferETH(address,uint256) (src/libraries/SafeETH.sol#5-14) uses assembly
        - INLINE ASM (src/libraries/SafeETH.sol#8-11)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

SafeETH.safeTransferETH(address,uint256) (src/libraries/SafeETH.sol#5-14) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version0.8.16 (src/libraries/SafeETH.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.16 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
```

IMX.sol

```
Pragma version0.8.16 (src/strategies/imx/IMX.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
```

UniUtils.sol

```
UniUtils._getAmountIn(IUniswapV2Pair,uint256,address,address) (src/libraries/UniUtils.sol#60-71) is never used and should be removed
UniUtils._getAmountOut(IUniswapV2Pair,uint256,address,address) (src/libraries/UniUtils.sol#46-58) is never used and should be removed
UniUtils._getPairReserves(IUniswapV2Pair,address,address) (src/libraries/UniUtils.sol#15-23) is never used and should be removed
UniUtils._getPairTokens(IUniswapV2Pair) (src/libraries/UniUtils.sol#11-13) is never used and should be removed
UniUtils._quote(uint256,uint256,uint256) (src/libraries/UniUtils.sol#26-34) is never used and should be removed
UniUtils._sortTokens(address,address) (src/libraries/UniUtils.sol#36-44) is never used and should be removed
UniUtils._swapExactTokensForTokens(IUniswapV2Pair,uint256,address,address) (src/libraries/UniUtils.sol#73-89) is never used and should be removed
UniUtils._swapTokensForExactTokens(IUniswapV2Pair,uint256,address,address) (src/libraries/UniUtils.sol#91-106) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version0.8.16 (src/libraries/UniUtils.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.16 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
```

SectorTimelock.sol

```
SectorTimelock._call(bytes32,uint256,address,uint256,bytes) (src/SectorTimelock.sol#338-349) sends eth to arbitrary user
        Dangerous calls:
        - (success) = target.call{value: value}(data) (src/SectorTimelock.sol#345)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#functions-that-send-ether-to-arbitrary-destinations

SectorTimelock.isOperationDone(bytes32) (src/SectorTimelock.sol#141-143) uses a dangerous strict equality:
        - getTimestamp(id) == _DONE_TIMESTAMP (src/SectorTimelock.sol#142)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities

SectorTimelock._call(bytes32,uint256,address,uint256,bytes) (src/SectorTimelock.sol#338-349) has external calls inside a loop: (success) = target.call{value: value}(data)
 (src/SectorTimelock.sol#345)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-a-loop

Reentrancy in SectorTimelock._call(bytes32,uint256,address,uint256,bytes) (src/SectorTimelock.sol#338-349):
        External calls:
        - (success) = target.call{value: value}(data) (src/SectorTimelock.sol#345)
        Event emitted after the call(s):
        - CallExecuted(id,index,target,value,data) (src/SectorTimelock.sol#348)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

SectorTimelock.isOperation(bytes32) (src/SectorTimelock.sol#119-121) uses timestamp for comparisons
        Dangerous comparisons:
        - getTimestamp(id) > 0 (src/SectorTimelock.sol#120)
SectorTimelock.isOperationPending(bytes32) (src/SectorTimelock.sol#126-128) uses timestamp for comparisons
        Dangerous comparisons:
        - getTimestamp(id) > _DONE_TIMESTAMP (src/SectorTimelock.sol#127)
SectorTimelock.isOperationReady(bytes32) (src/SectorTimelock.sol#133-136) uses timestamp for comparisons
        Dangerous comparisons:
        - timestamp > _DONE_TIMESTAMP && timestamp <= block.timestamp (src/SectorTimelock.sol#135)
SectorTimelock.isOperationDone(bytes32) (src/SectorTimelock.sol#141-143) uses timestamp for comparisons
        Dangerous comparisons:
        - getTimestamp(id) == _DONE_TIMESTAMP (src/SectorTimelock.sol#142)
SectorTimelock._beforeCall(bytes32,bytes32) (src/SectorTimelock.sol#317-323) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(predecessor == bytes32(0) || isOperationDone(predecessor),TimelockController: missing dependency) (src/SectorTimelock.sol#319-322)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
```

```
Different versions of Solidity are used:
        - Version used: ['0.8.16', '^0.8.0']
        - ^0.8.0 (node_modules/@openzeppelin/contracts/access/AccessControl.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/access/IAccessControl.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Strings.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/introspection/ERC165.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/introspection/IERC165.sol#4)
        - 0.8.16 (src/SectorTimelock.sol#4)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Context._msgData() (node_modules/@openzeppelin/contracts/utils/Context.sol#21-23) is never used and should be removed
Strings.toHexString(address) (node_modules/@openzeppelin/contracts/utils/Strings.sol#72-74) is never used and should be removed
Strings.toHexString(uint256) (node_modules/@openzeppelin/contracts/utils/Strings.sol#41-52) is never used and should be removed
Strings.toString(uint256) (node_modules/@openzeppelin/contracts/utils/Strings.sol#16-36) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/access/AccessControl.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/access/IAccessControl.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Strings.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/introspection/ERC165.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/introspection/IERC165.sol#4) allows old versions
Pragma version0.8.16 (src/SectorTimelock.sol#4) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.16 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in SectorTimelock._call(bytes32,uint256,address,uint256,bytes) (src/SectorTimelock.sol#338-349):
        - (success) = target.call{value: value}(data) (src/SectorTimelock.sol#345)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
```


```
Low level call in SectorTimelock._call(bytes32,uint256,address,uint256,bytes) (src/SectorTimelock.sol#338-349):
        - (success) = target.call{value: value}(data) (src/SectorTimelock.sol#345)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
```


```
grantRole(bytes32,address) should be declared external:
        - AccessControl.grantRole(bytes32,address) (node_modules/@openzeppelin/contracts/access/AccessControl.sol#144-146)
revokeRole(bytes32,address) should be declared external:
        - AccessControl.revokeRole(bytes32,address) (node_modules/@openzeppelin/contracts/access/AccessControl.sol#159-161)
renounceRole(bytes32,address) should be declared external:
        - AccessControl.renounceRole(bytes32,address) (node_modules/@openzeppelin/contracts/access/AccessControl.sol#179-183)
schedule(address,uint256,bytes,bytes32,bytes32,uint256) should be declared external:
        - SectorTimelock.schedule(address,uint256,bytes,bytes32,bytes32,uint256) (src/SectorTimelock.sol#199-210)
scheduleBatch(address[],uint256[],bytes[],bytes32,bytes32,uint256) should be declared external:
        - SectorTimelock.scheduleBatch(address[],uint256[],bytes[],bytes32,bytes32,uint256) (src/SectorTimelock.sol#221-237)
cancel(bytes32) should be declared external:
        - SectorTimelock.cancel(bytes32) (src/SectorTimelock.sol#255-260)
execute(address,uint256,bytes,bytes32,bytes32) should be declared external:
        - SectorTimelock.execute(address,uint256,bytes,bytes32,bytes32) (src/SectorTimelock.sol#274-285)
executeBatch(address[],uint256[],bytes[],bytes32,bytes32) should be declared external:
        - SectorTimelock.executeBatch(address[],uint256[],bytes[],bytes32,bytes32) (src/SectorTimelock.sol#296-312)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

- Reentrancy issues are false positives.
- Usage of timestamp for comparisons is false positive.
- Vast number of findings are false positives.
- Some functions might be marked as external, however, declaring a function as external over public does not save gas if the Solidity version is ^0.8.0.
- Multiple informational issues related to solidity naming convention were identified.
- No major issues were found by Slither.

4.2 AUTOMATED SECURITY SCAN

Description:

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

MythX results:

HLPVault.sol

Report for src/common/Accounting.sol
https://dashboard.mythx.io/#/console/analyses/edde052e-7444-4e1a-8fe9-1be0fd0004e1

Line	SWC Title	Severity	Short Description
18	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered

Report for src/common/StratAuth.sol
https://dashboard.mythx.io/#/console/analyses/edde052e-7444-4e1a-8fe9-1be0fd0004e1

Line	SWC Title	Severity	Short Description
22	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered
23	(SWC-110) Assert Violation	Unknown	Out of bounds array access
24	(SWC-110) Assert Violation	Unknown	Out of bounds array access
25	(SWC-110) Assert Violation	Unknown	Out of bounds array access

Report for src/libraries/UniUtils.sol
https://dashboard.mythx.io/#/console/analyses/edde052e-7444-4e1a-8fe9-1be0fd0004e1

Line	SWC Title	Severity	Short Description
33	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
33	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
54	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
55	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
56	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
56	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
57	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
68	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
69	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
69	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
70	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
70	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered

Report for src/strategies/hlp/HLPCore.sol
https://dashboard.mythx.io/#/console/analyses/edde052e-7444-4e1a-8fe9-1be0fd0004e1

Line	SWC Title	Severity	Short Description
187	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
187	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
188	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
188	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
188	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
189	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
189	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
189	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
193	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
216	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
270	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
272	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered

279	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
298	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
307	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
307	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
308	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
308	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
309	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
309	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
314	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
354	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
354	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
354	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
408	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
414	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
431	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
431	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
432	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
434	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
460	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
464	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
483	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
493	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
493	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
504	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
524	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
525	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
527	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
535	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
546	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
546	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
547	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
580	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
584	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
585	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
599	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
607	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
607	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
607	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
608	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered

608	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
608	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
608	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
615	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
615	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
615	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
623	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
623	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
630	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
630	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
630	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered

Report for src/strategies/imx/IMXCore.sol
https://dashboard.mythx.io/#/console/analyses/edde052e-7444-4e1a-8fe9-1be0fd0004e1

Line	SWC Title	Severity	Short Description
54	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
55	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
56	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
56	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
56	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
62	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
63	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
147	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
194	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
209	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
209	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
210	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
216	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
216	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
217	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
217	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
218	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
218	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
235	(SWC-110) Assert Violation	Unknown	Out of bounds array access
258	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
258	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
259	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
269	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered

269	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
270	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
273	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
273	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
273	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
276	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
277	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
279	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
308	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
314	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
314	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
314	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
322	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
329	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
329	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
330	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
351	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
357	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
359	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
360	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
366	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
373	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
373	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
373	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
374	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
374	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
374	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
396	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
396	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
396	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered

Report for src/strategies/imx/IMXFarm.sol
https://dashboard.mythx.io/#/console/analyses/edde052e-7444-4e1a-8fe9-1be0fd0004e1

Line	SWC Title	Severity	Short Description
143	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
144	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
149	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered
150	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
168	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
188	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
188	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
188	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
223	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
232	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
252	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
252	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
252	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
268	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
284	(SWC-110) Assert Violation	Unknown	Out of bounds array access
285	(SWC-110) Assert Violation	Unknown	Out of bounds array access
303	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
303	(SWC-101) Integer Overflow and Underflow	Unknown	Compiler-rewritable "<uint> - 1" discovered
303	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
303	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
330	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
330	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
331	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
331	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
331	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
331	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
331	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
331	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
331	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
337	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
337	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
339	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
339	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
339	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
339	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered

345	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
345	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
347	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
347	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
347	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
347	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
352	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
352	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered

Report for src/strategies/mixins/IFarmable.sol
https://dashboard.mythx.io/#/console/analyses/edde052e-7444-4e1a-8fe9-1be0fd0004e1

Line	SWC Title	Severity	Short Description
19	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
19	(SWC-101) Integer Overflow and Underflow	Unknown	Compiler-rewritable "<uint> - 1" discovered
19	(SWC-110) Assert Violation	Unknown	Out of bounds array access
23	(SWC-110) Assert Violation	Unknown	Out of bounds array access

Report for src/strategies/mixins/ILending.sol
https://dashboard.mythx.io/#/console/analyses/edde052e-7444-4e1a-8fe9-1be0fd0004e1

Line	SWC Title	Severity	Short Description
44	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
44	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
52	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
52	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
53	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
53	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
63	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
64	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
77	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
77	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
79	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered

Report for src/strategies/mixins/IUniLp.sol
https://dashboard.mythx.io/#/console/analyses/edde052e-7444-4e1a-8fe9-1be0fd0004e1

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

Report for src/vaults/ERC5115/SCYBase.sol
https://dashboard.mythx.io/#/console/analyses/edde052e-7444-4e1a-8fe9-1be0fd0004e1

Line	SWC Title	Severity	Short Description
66	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered

Report for src/vaults/ERC5115/SCYVault.sol
https://dashboard.mythx.io/#/console/analyses/edde052e-7444-4e1a-8fe9-1be0fd0004e1

Line	SWC Title	Severity	Short Description
114	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
127	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
127	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
127	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
133	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
133	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
136	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered
144	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
145	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
150	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
165	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
171	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
175	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
178	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
178	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
181	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
181	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
181	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
183	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
188	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
197	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
203	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
204	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
206	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
209	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
209	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
209	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
224	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
232	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
233	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
233	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
233	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
244	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
245	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
255	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered

268	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
275	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
284	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered
285	(SWC-110) Assert Violation	Unknown	Out of bounds array access
286	(SWC-110) Assert Violation	Unknown	Out of bounds array access
287	(SWC-110) Assert Violation	Unknown	Out of bounds array access
302	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
308	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
323	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
331	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
332	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
332	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
339	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
340	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
340	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
340	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
341	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
341	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
352	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
352	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
353	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
353	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
353	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
390	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
404	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
404	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
418	(SWC-110) Assert Violation	Unknown	Out of bounds array access
420	(SWC-110) Assert Violation	Unknown	Out of bounds array access

IMXLend.sol

Report for src/vaults/strategyVaults/IMXLend.sol
https://dashboard.mythx.io/#/console/analyses/45015e6a-ca25-453e-aca0-e9947a989a39

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

Stargate.sol

Report for src/vaults/strategyVaults/Stargate.sol
https://dashboard.mythx.io/#/console/analyses/2a2c0cf8-8603-4001-8f19-32227784a11c

Line	SWC Title	Severity	Short Description
40	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
40	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
72	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
89	(SWC-110) Assert Violation	Unknown	Out of bounds array access
92	(SWC-110) Assert Violation	Unknown	Out of bounds array access

Report for src/strategies/mixins/IFarmable.sol
https://dashboard.mythx.io/#/console/analyses/2a2c0cf8-8603-4001-8f19-32227784a11c

Line	SWC Title	Severity	Short Description
19	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
19	(SWC-101) Integer Overflow and Underflow	Unknown	Compiler-rewritable "<uint> - 1" discovered
19	(SWC-110) Assert Violation	Unknown	Out of bounds array access
23	(SWC-110) Assert Violation	Unknown	Out of bounds array access

Report for src/strategies/mixins/IUniLp.sol
https://dashboard.mythx.io/#/console/analyses/2a2c0cf8-8603-4001-8f19-32227784a11c

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

Synapse.sol

Report for src/vaults/strategyVaults/Synapse.sol
https://dashboard.mythx.io/#/console/analyses/d0456f4c-de3c-4782-9974-7705b24e2202

Line	SWC Title	Severity	Short Description
43	(SWC-110) Assert Violation	Unknown	Out of bounds array access
88	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
105	(SWC-110) Assert Violation	Unknown	Out of bounds array access
108	(SWC-110) Assert Violation	Unknown	Out of bounds array access

MasterChefCompMulti.sol

Report for src/strategies/adapters/MasterChefFarm.sol
https://dashboard.mythx.io/#/console/analyses/9a130b3f-5c79-45c2-8ce9-daca4d13e6fb

Line	SWC Title	Severity	Short Description
67	(SWC-110) Assert Violation	Unknown	Out of bounds array access
68	(SWC-110) Assert Violation	Unknown	Out of bounds array access
70	(SWC-110) Assert Violation	Unknown	Out of bounds array access
71	(SWC-110) Assert Violation	Unknown	Out of bounds array access
88	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
94	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered

Report for src/strategies/adapters/CompoundFarm.sol
https://dashboard.mythx.io/#/console/analyses/9a130b3f-5c79-45c2-8ce9-daca4d13e6fb

Line	SWC Title	Severity	Short Description
35	(SWC-110) Assert Violation	Unknown	Out of bounds array access
36	(SWC-110) Assert Violation	Unknown	Out of bounds array access
40	(SWC-110) Assert Violation	Unknown	Out of bounds array access
41	(SWC-110) Assert Violation	Unknown	Out of bounds array access
44	(SWC-110) Assert Violation	Unknown	Out of bounds array access
45	(SWC-110) Assert Violation	Unknown	Out of bounds array access

Report for src/strategies/adapters/CompMultiFarm.sol
https://dashboard.mythx.io/#/console/analyses/9a130b3f-5c79-45c2-8ce9-daca4d13e6fb

Line	SWC Title	Severity	Short Description
21	(SWC-110) Assert Violation	Unknown	Out of bounds array access
23	(SWC-110) Assert Violation	Unknown	Out of bounds array access
24	(SWC-110) Assert Violation	Unknown	Out of bounds array access
25	(SWC-110) Assert Violation	Unknown	Out of bounds array access

Report for src/strategies/mixins/ILending.sol
https://dashboard.mythx.io/#/console/analyses/9a130b3f-5c79-45c2-8ce9-daca4d13e6fb

Line	SWC Title	Severity	Short Description
44	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
44	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
52	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
52	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
53	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
53	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
63	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
64	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
77	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
77	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
79	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered

- Majority of identified issues are related to arithmetic operations and out of bounds array access.
- The Integer Overflow and Underflow findings are false positives. The actual instances of Integer Overflow and Underflow were identified and reported during dynamic testing.
- IMXVault.sol, AggregatorVault.sol, IMX.sol, UniUtils.sol, SafeETH.sol, SectorTimelock.sol yielded no result.
- Findings related to the OpenZepplin libraries were omitted.
- Duplicate contract scans were omitted.
- No major issues were discovered by Mythx software.

THANK YOU FOR CHOOSING

// HALBORN