// HALBORN

StakeWise - osETH

Smart Contract Security Assessment

Prepared by: **Halborn**

Date of Engagement: **June 26th, 2023 - July 27th, 2023**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	06/26/2023	Grzegorz Trawinski
0.2	Document Updates	07/18/2023	Grzegorz Trawinski
0.3	Draft Version	07/27/2023	Grzegorz Trawinski
0.4	Draft Review	07/27/2023	Piotr Cielas
0.5	Draft Review	07/27/2023	Gabi Urrutia
1.0	Remediation Plan	08/09/2023	Grzegorz Trawinski
1.1	Remediation Plan Review	08/09/2023	Piotr Cielas
1.2	Remediation Plan Review	08/09/2023	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
Grzegorz Trawinski	Halborn	Grzegorz.Trawinski@halborn.com

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

StakeWise makes ETH staking effortless by providing resilient infrastructure and yield strategies. StakeWise clients can participate in Ethereum's Proof-of-Stake consensus mechanism (staking) and receive ETH rewards in return. By introducing osETH StakeWise enables liquidity staking in the proposed solution.

StakeWise engaged Halborn to conduct a security assessment on their smart contracts beginning on June 26th, 2023 and ending on July 27th, 2023. The security assessment was scoped to the smart contracts provided in the v3-core GitHub repository. Commit hashes and further details can be found in the Scope section of this report.

Between April 3rd, 2023 and April 28th, 2023 the StakeWise team engaged Halborn to conduct a security assessment of Protocol V3. The findings identified in the previous assessment are not included in this report.

1.2 ASSESSMENT SUMMARY

Halborn was provided 5 weeks for the engagement and assigned 1 full-time security engineer to assess the security of the smart contracts in scope. The security team consists of a blockchain and smart contract security experts with advanced penetration testing and smart contract hacking skills, and deep knowledge of multiple blockchain protocols.

The purpose of the assessment is to:

- Identify potential security issues within the smart contracts.
- Ensure that smart contract functionality operates as intended.

In summary, Halborn identified some security risks that were accepted by the StakeWise 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 assessment. While manual testing is recommended to uncover flaws in logic, process, and implementation; automated testing techniques help enhance coverage of the code and can quickly identify items that do not follow the security best practices. The following phases and associated tools were used during the assessment:

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

2. RISK METHODOLOGY

Every vulnerability and issue observed by Halborn is ranked based on **two sets** of **Metrics** and a **Severity Coefficient**. This system is inspired by the industry standard Common Vulnerability Scoring System.

The two **Metric sets** are: **Exploitability** and **Impact**. **Exploitability** captures the ease and technical means by which vulnerabilities can be exploited and **Impact** describes the consequences of a successful exploit.

The **Severity Coefficients** is designed to further refine the accuracy of the ranking with two factors: **Reversibility** and **Scope**. These capture the impact of the vulnerability on the environment as well as the number of users and smart contracts affected.

The final score is a value between 0-10 rounded up to 1 decimal place and 10 corresponding to the highest security risk. This provides an objective and accurate rating of the severity of security vulnerabilities in smart contracts.

The system is designed to assist in identifying and prioritizing vulnerabilities based on their level of risk to address the most critical issues in a timely manner.

2.1 EXPLOITABILITY

Attack Origin (AO):

Captures whether the attack requires compromising a specific account.

Attack Cost (AC):

Captures the cost of exploiting the vulnerability incurred by the attacker relative to sending a single transaction on the relevant blockchain. Includes but is not limited to financial and computational cost.

Attack Complexity (AX):

Describes the conditions beyond the attacker's control that must exist in order to exploit the vulnerability. Includes but is not limited to macro situation, available third-party liquidity and regulatory challenges.

Metrics:

Exploitability Metric (m_E)	Metric Value	Numerical Value
Attack Origin (AO)	Arbitrary (AO:A) Specific (AO:S)	1 0.2
Attack Cost (AC)	Low (AC:L) Medium (AC:M) High (AC:H)	1 0.67 0.33
Attack Complexity (AX)	Low (AX:L) Medium (AX:M) High (AX:H)	1 0.67 0.33

Exploitability E is calculated using the following formula:

$$E = \prod m_e$$

2.2 IMPACT

Confidentiality (C):

Measures the impact to the confidentiality of the information resources managed by the contract due to a successfully exploited vulnerability. Confidentiality refers to limiting access to authorized users only.

Integrity (I):

Measures the impact to integrity of a successfully exploited vulnerability. Integrity refers to the trustworthiness and veracity of data stored and/or processed on-chain. Integrity impact directly affecting Deposit or Yield records is excluded.

Availability (A):

Measures the impact to the availability of the impacted component resulting from a successfully exploited vulnerability. This metric refers to smart contract features and functionality, not state. Availability impact directly affecting Deposit or Yield is excluded.

Deposit (D):

Measures the impact to the deposits made to the contract by either users or owners.

Yield (Y):

Measures the impact to the yield generated by the contract for either users or owners.

Metrics:

Impact Metric (m_I)	Metric Value	Numerical Value
Confidentiality (C)	None (I:N)	0
	Low (I:L)	0.25
	Medium (I:M)	0.5
	High (I:H)	0.75
	Critical (I:C)	1
Integrity (I)	None (I:N)	0
	Low (I:L)	0.25
	Medium (I:M)	0.5
	High (I:H)	0.75
	Critical (I:C)	1
Availability (A)	None (A:N)	0
	Low (A:L)	0.25
	Medium (A:M)	0.5
	High (A:H)	0.75
	Critical	1
Deposit (D)	None (D:N)	0
	Low (D:L)	0.25
	Medium (D:M)	0.5
	High (D:H)	0.75
	Critical (D:C)	1
Yield (Y)	None (Y:N)	0
	Low (Y:L)	0.25
	Medium: (Y:M)	0.5
	High: (Y:H)	0.75
	Critical (Y:H)	1

Impact I is calculated using the following formula:

$$I = max(m_I) + \frac{\sum m_I - max(m_I)}{4}$$

2.3 SEVERITY COEFFICIENT

Reversibility (R):

Describes the share of the exploited vulnerability effects that can be reversed. For upgradeable contracts, assume the contract private key is available.

Scope (S):

Captures whether a vulnerability in one vulnerable contract impacts resources in other contracts.

Coefficient (C)	Coefficient Value	Numerical Value
Reversibility (r)	None (R:N) Partial (R:P) Full (R:F)	1 0.5 0.25
Scope (s)	Changed (S:C) Unchanged (S:U)	1.25 1

Severity Coefficient C is obtained by the following product:

$$C = rs$$

The Vulnerability Severity Score S is obtained by:

$$S = min(10, EIC * 10)$$

The score is rounded up to 1 decimal places.

Severity	Score Value Range
Critical	9 - 10
High	7 - 8.9
Medium	4.5 - 6.9
Low	2 - 4.4
Informational	0 - 1.9

2.4 SCOPE

Code repositories:

1. osEth

- Repository: v3-core
- Commit ID: ab9c809555ca5e07f59fb4f639232cbafc83d30a
- Smart contracts in scope:
  - libraries/ExitQueue.sol
  - libraries/Errors.sol
  - osToken/OsToken.sol
  - osToken/OsTokenConfig.sol
  - osToken/PriceOracle.sol
  - vaults/modules/VaultMev.sol
  - vaults/modules/VaultValidators.sol
  - vaults/modules/VaultWhitelist.sol
  - vaults/modules/VaultImmutables.sol
  - vaults/modules/VaultToken.sol
  - vaults/modules/VaultAdmin.sol
  - vaults/modules/VaultEthStaking.sol
  - vaults/modules/VaultFee.sol
  - vaults/modules/VaultVersion.sol
  - vaults/modules/VaultOsToken.sol
  - vaults/modules/VaultToken.sol
  - vaults/modules/VaultEnterExit.sol
  - vaults/modules/VaultState.sol
  - vaults/VaultsRegistry.sol
  - vaults/ethereum/EthErc20Vault.sol
  - vaults/ethereum/EthGenesisVault.sol
  - vaults/ethereum/EthPrivErc20Vault.sol
  - vaults/ethereum/EthPrivVault.sol
  - vaults/ethereum/EthVaultFactory.sol
  - vaults/ethereum/EthVault.sol
  - vaults/ethereum/mev/SharedMevEscrow.sol
  - vaults/ethereum/mev/OwnMevEscrow.sol

- keeper/KeeperValidators.sol
- keeper/KeeperRewards.sol
- keeper/KeeperOracles.sol
- keeper/Keeper.sol
- base/ERC20Upgradeable.sol
- base/ERC20.sol
- base/Multicall.sol
- interfaces/*.sol

On the 5th of July, the team at Halborn received an updated code base with an amendment related to the `claimExitedAssets` function in commit ID: a03e5c794d4cc7ab4f6933754da91d9123877983.

Out-of-scope

- Third-party libraries and dependencies.
- Economic attacks.

3. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	0	1	2	0

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
(HAL-01) SLIPPAGE CHECK IS MISSING IN THE MINTOSTOKEN FUNCTION	Medium (5.0)	RISK ACCEPTED
(HAL-02) TREASURY UPDATE DOES NOT TRANSFER PREVIOUS BALANCE	Low (2.6)	RISK ACCEPTED
(HAL-03) OSTOKEN LACKS EMERGENCY-STOP PATTERN	Low (2.6)	RISK ACCEPTED

FINDINGS & TECH DETAILS

4.1 (HAL-01) SLIPPAGE CHECK IS MISSING IN THE MINTOSTOKEN FUNCTION - MEDIUM (5.0)

Description:

The mintOsToken function from the VaultOsToken module allows to mint osETH tokens up to a preset Loan to Value (LTV) limit. The amount of osETH tokens minted depends on a couple of factors, including the profit accrued over time in the OsToken contract and the health condition of other vaults participating in liquidity staking. However, the mintOsToken function does not support slippage check functionality. Thus, a user cannot define the expected bottom limit of shares issued with this functionality.

Listing 1: VaultOsToken.sol

```
56 function mintOsToken(
57     address receiver,
58     uint256 assets,
59     address referrer
60   ) external override returns (uint256 osTokenShares) {
61     _checkCollateralized();
62     _checkHarvested();
63
64     // mint osToken shares to the receiver
65     osTokenShares = _osToken.mintShares(receiver, assets);
66
67     // fetch user position
68     OsTokenPosition memory position = _positions[msg.sender];
69     if (position.shares > 0) {
70       _syncPositionFee(position);
71     } else {
72       position.cumulativeFeePerShare = SafeCast.toUint128(_osToken
↳ .cumulativeFeePerShare());
73     }
74
75     // add minted shares to the position
76     position.shares += SafeCast.toUint128(osTokenShares);
77
78     // calculate and validate LTV
```

```
79      if (
80        Math.mulDiv(
81          convertToAssets(_balances[msg.sender]),
82          _osTokenConfig.ltvPercent(),
83          _maxPercent
84        ) < _osToken.convertToAssets(position.shares)
85      ) {
86        revert Errors.LowLtv();
87      }
88
89      // update state
90      _positions[msg.sender] = position;
91
92      // emit event
93      emit OsTokenMinted(msg.sender, receiver, assets, osTokenShares
↳ , referrer);
94    }
```

BVSS:

AO:A/AC:L/AX:L/C:N/I:N/A:N/D:N/Y:M/R:N/S:U (5.0)

Recommendation:

It is recommended to include slippage in the input parameters of the mintOsToken function, so users can control the number of expected shares to be issued.

Remediation Plan:

RISK ACCEPTED: The StakeWise team accepted the risk of this finding. osToken slippage is predictable, as the rewards increase by a constant amount every second. To avoid slippage, a user can deduct the number of shares equal to 1 hour of rewards from the desired osToken shares. This approach is more gas efficient than slippage checks in the osToken functionality.

4.2 (HAL-02) TREASURY UPDATE DOES NOT TRANSFER PREVIOUS BALANCE - LOW (2.6)

Description:

The current implementation of the setTreasury function does not transfer shares balance from the previous treasury to the new one. Currently, treasury shares are accumulated in the updateState function, representing the fee collected by the solution owner. In case of an emergency, e.g., treasury private key loss or disclosure, the shares accumulated over time are permanently lost.

Listing 2: OsToken.sol

```
168   function setTreasury(address _treasury) public override
 ↳ onlyOwner {
169     if (_treasury == address(0)) revert Errors.ZeroAddress();
170
171     // update DAO treasury address
172     treasury = _treasury;
173     emit TreasuryUpdated(_treasury);
174   }
```

BVSS:

AO:A/AC:L/AX:H/C:N/I:M/A:N/D:M/Y:N/R:N/S:C (2.6)

Recommendation:

It is recommended to transfer shares to the new treasury when setting the new treasure with the setTreasury function.

Remediation Plan:

RISK ACCEPTED: The StakeWise team accepted the risk of this finding.

4.3 (HAL-03) OSTOKEN LACKS EMERGENCY-STOP PATTERN - LOW (2.6)

Description:

The current implementation of the OsToken contract does not implement any kind of emergency stop pattern. Such a pattern allows the project team to pause crucial functionalities, while being in the state of emergency, e.g., being under adversary attack. The most prevalent application of the emergency stop pattern is the the OpenZeppelin's Pausable library (or alternatively PausableUpgradeable.

In this case, if the emergency stop pattern is not implemented, then functions such as mintShares() or burnShares() cannot be temporarily disabled, while some vault is under adversary attack.

BVSS:

AO:A/AC:L/AX:H/C:N/I:M/A:N/D:M/Y:N/R:N/S:C (2.6)

Recommendation:

It is recommended to implement the emergency stop pattern within the OsToken contract.

Remediation Plan:

RISK ACCEPTED: The StakeWise team accepted the risk of this finding. Introduction of an emergency-stop mechanism may affect the decentralization of the protocol.

MANUAL TESTING

In the manual testing phase, multiple unit tests were written in the Foundry framework. Various scenarios were simulated depending on the context and the tested smart contract.

The purpose of the OwnMevEscrow and SharedMevEscrow contracts unit testing was to verify if the contract's business logic works as expected along with the implemented authorization.

```
Running 11 tests for test/HalbornEscrowTest.t.sol:HalbornEscrowTest
[PASS] test_OwnMevEscrow_harvest_as_non_vault_reverts() (gas: 147591)
[PASS] test_OwnMevEscrow_harvest_balance() (gas: 329667)
[PASS] test_OwnMevEscrow_harvest_balance_as_non_IVaultEthStaking_reverts() (gas: 155787)
[PASS] test_OwnMevEscrow_harvest_zero() (gas: 145239)
[PASS] test_OwnMevEscrow_transfer_ether_to_mev() (gas: 154297)
[PASS] test_SharedMevEscrow_harvest_above_balance_reverts() (gas: 340513)
[PASS] test_SharedMevEscrow_harvest_as_non_vault_reverts() (gas: 157173)
[PASS] test_SharedMevEscrow_harvest_balance() (gas: 340242)
[PASS] test_SharedMevEscrow_harvest_half_of_balance() (gas: 340232)
[PASS] test_SharedMevEscrow_harvest_zero() (gas: 324195)
[PASS] test_SharedMevEscrow_transfer_ether_to_mev() (gas: 163898)
Test result: ok. 11 passed; 0 failed; finished in 15.62ms
```

The purpose of the EthVault contract unit testing was to verify if contract business logic works as expected along with the implemented authorization. This part of the unit test targeted multiple functionalities in vault's modules, that were excluded in other tests. The upgradability functionality was also included.

```
Running 19 tests for test/HalbornEthVaultTest.t.sol:HalbornEthVaultTest
[PASS] test_EthVault_attempt_to_initilize_twice_reverts() (gas: 26335)
[PASS] test_EthVault_receiveFromMevEscrow_as_non_mev_reverts() (gas: 25723)
[PASS] test_EthVault_setFeeRecipient_as_non_owner_reverts() (gas: 18580)
[PASS] test_EthVault_setFeeRecipient_as_zero_address_reverts() (gas: 24227)
[PASS] test_EthVault_setFeeRecipient_valid_scenario() (gas: 32459)
[PASS] test_EthVault_setMetadata_as_admin() (gas: 20445)
[PASS] test_EthVault_setMetadata_as_non_admin_reverts() (gas: 18724)
[PASS] test_EthVault_setValidatorsRoot_as_not_an_owner_reverts() (gas: 20652)
[PASS] test_EthVault_setValidatorsRoot_twice_valid_scenario() (gas: 29501)
[PASS] test_EthVault_setValidatorsRoot_valid_scenario() (gas: 26166)
[PASS] test_EthVault_upgradeToAndCall_incorrect_vaultId_reverts() (gas: 78347)
[PASS] test_EthVault_upgradeToAndCall_initialize_twice_reverts() (gas: 4584372)
[PASS] test_EthVault_upgradeToAndCall_no_implementation_added_to_registry_reverts() (gas: 4496356)
[PASS] test_EthVault_upgradeToAndCall_the_same_implementation_reverts() (gas: 25328)
[PASS] test_EthVault_upgradeToAndCall_valid_scenario() (gas: 4601707)
[PASS] test_EthVault_upgradeToAndCall_zero_address_not_as_admin_reverts() (gas: 19346)
[PASS] test_EthVault_upgradeToAndCall_zero_address_reverts() (gas: 19357)
[PASS] test_EthVault_upgradeTo_zero_address_not_as_admin_reverts() (gas: 16614)
[PASS] test_EthVault_upgradeTo_zero_address_reverts() (gas: 16599)
Test result: ok. 19 passed; 0 failed; finished in 15.69ms
```

The purpose of the EthPrivateVault contract unit testing was to verify if the business logic added on top of the EthVault contract works as expected along with the implemented authorization. An attempt to bypass the white-list was performed.

```
Running 15 tests for test/HalbornEthPrivateVaultTest.t.sol:HalbornEthPrivateVaultTest
[PASS] test_EthPrivateVault_attempt_to_initilize_twice_reverts() (gas: 26268)
[PASS] test_EthPrivateVault_deposit_as_non_whitelisted() (gas: 143138)
[PASS] test_EthPrivateVault_deposit_as_owner_valid_scenario() (gas: 182174)
[PASS] test_EthPrivateVault_deposit_as_owner_with_empty_receiver_reverts() (gas: 145341)
[PASS] test_EthPrivateVault_deposit_as_owner_with_empty_referrer_valid_scenario() (gas: 182106)
[PASS] test_EthPrivateVault_deposit_valid_scenario() (gas: 65120)
[PASS] test_EthPrivateVault_deposit_via_receive_as_non_whitelisted_valid_scenario_to_report_now_call_fails() (gas: 155011)
[PASS] test_EthPrivateVault_deposit_via_receive_as_owner_valid_scenario() (gas: 181080)
[PASS] test_EthPrivateVault_setWhitelister_as_non_admin() (gas: 24962)
[PASS] test_EthPrivateVault_setWhitelister_twice_valid_scenario() (gas: 39194)
[PASS] test_EthPrivateVault_setWhitelister_valid_scenario() (gas: 34899)
[PASS] test_EthPrivateVault_updateWhitelist_as_non_admin() (gas: 25636)
[PASS] test_EthPrivateVault_updateWhitelist_non_existing_to_false_reverts() (gas: 29732)
[PASS] test_EthPrivateVault_updateWhitelist_twice_reverts() (gas: 55403)
[PASS] test_EthPrivateVault_updateWhitelist_valid_scenario() (gas: 42796)
Test result: ok. 15 passed; 0 failed; finished in 18.05ms
```

The purpose of the EthPrivErc20Vault contract unit testing was to verify if contract business logic added on top of the EthVault and EthErc20Vault contracts works as expected along with the implemented authorization.

```
Running 35 tests for test/HalbornEthPrivErc20VaultTest.t.sol:HalbornEthPrivErc20VaultTest
[PASS] test_EthPrivErc20Vault_attempt_to_initilize_twice_reverts() (gas: 26413)
[PASS] test_EthPrivErc20Vault_deposit_as_non_whitelisted_reverts() (gas: 152262)
[PASS] test_EthPrivErc20Vault_deposit_as_owner_valid_scenario() (gas: 193151)
[PASS] test_EthPrivErc20Vault_deposit_as_owner_with_empty_receiver_reverts() (gas: 154374)
[PASS] test_EthPrivErc20Vault_deposit_as_owner_with_empty_referrer_valid_scenario() (gas: 193150)
[PASS] test_EthPrivErc20Vault_deposit_registerValidator_mintOsToken_then_transfer() (gas: 901506)
[PASS] test_EthPrivErc20Vault_deposit_registerValidator_mintOsToken_then_transfer_above_ltv_reverts() (gas: 913269)
[PASS] test_EthPrivErc20Vault_deposit_then_transferFrom_without_approve_reverts() (gas: 412567)
[PASS] test_EthPrivErc20Vault_deposit_then_transfer_approve_transferFrom_valid_scenario() (gas: 423500)
[PASS] test_EthPrivErc20Vault_deposit_valid_scenario() (gas: 67120)
[PASS] test_EthPrivErc20Vault_deposit_via_receive_as_non_whitelisted_valid_scenario_to_report_now_call_fails() (gas: 168557)
[PASS] test_EthPrivErc20Vault_deposit_via_receive_as_owner_valid_scenario() (gas: 192037)
[PASS] test_EthPrivErc20Vault_receiveFromMevEscrow_as_non_mev_reverts() (gas: 25801)
[PASS] test_EthPrivErc20Vault_setFeeRecipient_as_non_owner_reverts() (gas: 18571)
[PASS] test_EthPrivErc20Vault_setFeeRecipient_as_zero_address_reverts() (gas: 24368)
[PASS] test_EthPrivErc20Vault_setFeeRecipient_valid_scenario() (gas: 32481)
[PASS] test_EthPrivErc20Vault_setMetadata_as_admin() (gas: 20423)
[PASS] test_EthPrivErc20Vault_setMetadata_as_non_admin_reverts() (gas: 18791)
[PASS] test_EthPrivErc20Vault_setValidatorsRoot_as_not_an_owner_reverts() (gas: 20653)
[PASS] test_EthPrivErc20Vault_setValidatorsRoot_twice_valid_scenario() (gas: 29445)
[PASS] test_EthPrivErc20Vault_setValidatorsRoot_valid_scenario() (gas: 26218)
[PASS] test_EthPrivErc20Vault_setWhitelister_as_non_admin() (gas: 25026)
[PASS] test_EthPrivErc20Vault_setWhitelister_twice_valid_scenario() (gas: 39208)
[PASS] test_EthPrivErc20Vault_setWhitelister_valid_scenario() (gas: 34876)
[PASS] test_EthPrivErc20Vault_updateWhitelist_as_non_admin() (gas: 25652)
[PASS] test_EthPrivErc20Vault_updateWhitelist_non_existing_to_false_reverts() (gas: 29623)
[PASS] test_EthPrivErc20Vault_updateWhitelist_twice_reverts() (gas: 55392)
[PASS] test_EthPrivErc20Vault_updateWhitelist_valid_scenario() (gas: 42805)
[PASS] test_EthPrivErc20Vault_upgradeToAndCall_incorrect_vaultId_reverts() (gas: 78414)
[PASS] test_EthPrivErc20Vault_upgradeToAndCall_no_implementation_added_to_registry_reverts() (gas: 4490024)
[PASS] test_EthPrivErc20Vault_upgradeToAndCall_the_same_implementation_reverts() (gas: 25308)
[PASS] test_EthPrivErc20Vault_upgradeToAndCall_zero_address_not_as_admin_reverts() (gas: 19290)
[PASS] test_EthPrivErc20Vault_upgradeToAndCall_zero_address_reverts() (gas: 19392)
[PASS] test_EthPrivErc20Vault_upgradeTo_zero_address_not_as_admin_reverts() (gas: 16681)
[PASS] test_EthPrivErc20Vault_upgradeTo_zero_address_reverts() (gas: 16553)
Test result: ok. 35 passed; 0 failed; finished in 80.92s
```

The purpose of the EthGenesisVault contract unit testing was to verify if the business logic added on top of the EthVault contract works as expected along with the implemented authorization.

```
Running 5 tests for test/HalbornEthGenesisVaultTest.t.sol:EthGenesisVaultTest
[PASS] test_ethGenesisVault_acceptPoolEscrowOwnership_as_not_future_owner_reverts() (gas: 23715)
[PASS] test_ethGenesisVault_attempt_to_initilize_twice_reverts() (gas: 26291)
[PASS] test_ethGenesisVault_commitOwnershipTransfer_then_acceptPoolEscrowOwnership_as_ethGenesisVault() (gas: 39587)
[PASS] test_ethGenesisVault_migrate_as_not_stakedEthToken() (gas: 43430)
[PASS] test_ethGenesisVault_migrate_as_stakedEthToken() (gas: 77592)
Test result: ok. 5 passed; 0 failed; finished in 14.73ms
```

In the fallowing unit testing exercise, multiple scenarios were simulated to verify if the deposit and withdraw/redeem functionalities work as

expected. The accounting of shares and assets was checked manually.

```
Running 14 tests for test/HalbornDepositTest.t.sol:HalbornDepositTest
[PASS] test_deposit_max() (gas: 320710)
[PASS] test_deposit_max_above_max_reverts() (gas: 35617)
[PASS] test_deposit_multiple_deposits() (gas: 382629)
[PASS] test_deposit_one() (gas: 318083)
[PASS] test_deposit_receiver_is_zero_address() (gas: 302477)
[PASS] test_deposit_referrer_as_depositor() (gas: 318126)
[PASS] test_deposit_then_redeem() (gas: 488657)
[PASS] test_deposit_then_redeem_balance_minus_one() (gas: 512941)
[PASS] test_deposit_then_redeem_one() (gas: 511347)
[PASS] test_deposit_then_redeem_to_zero_address_now_reverts() (gas: 497098)
[PASS] test_deposit_then_redeem_too_much_reverts() (gas: 499946)
[PASS] test_deposit_then_redeem_twice() (gas: 504054)
[PASS] test_deposit_zero_deposit_is_possible_now_reverts() (gas: 295879)
[PASS] test_deposit_zero_reverts() (gas: 295857)
Test result: ok. 14 passed; 0 failed; finished in 21.83ms
```

The purpose of the EthVaultFactory contract unit testing was to verify if the business logic works as expected.

```
Running 8 tests for test/HalbornEthVaultFactoryTest.t.sol:HalbornEthVaultFactoryTest
[PASS] test_EthVaultFactory_deploy_EthErc20Vault_valid_scenario() (gas: 1077080)
[PASS] test_EthVaultFactory_deploy_EthPrivErc20Vault_valid_scenario() (gas: 1125356)
[PASS] test_EthVaultFactory_deploy_EthPrivVault_valid_scenario() (gas: 1056469)
[PASS] test_EthVaultFactory_deploy_EthVault_feePrecent_set_to_zero() (gas: 343754)
[PASS] test_EthVaultFactory_deploy_EthVault_too_high_feePrecent_reverts() (gas: 199689)
[PASS] test_EthVaultFactory_deploy_EthVault_too_low_security_deposit_reverts() (gas: 274938)
[PASS] test_EthVaultFactory_deploy_EthVault_too_small_capacity_reverts() (gas: 252062)
[PASS] test_EthVaultFactory_deploy_EthVault_valid_scenario() (gas: 343710)
Test result: ok. 8 passed; 0 failed; finished in 15.19ms
```

The purpose of the MultiCall contract testing was to verify if the business logic works as expected.

```
Running 3 tests for test/HalbornMultiCallTest.t.sol:HalbornMultiCallTest
[PASS] test_multicall_poc_1_approve_transfer() (gas: 1458791)
[PASS] test_multicall_poc_2_empty_deposit_possible_now_reverts() (gas: 1370016)
[PASS] test_multicall_poc_3_redeem_twice() (gas: 515519)
Test result: ok. 3 passed; 0 failed; finished in 20.46ms
```

The purpose of the VaultsRegistry contract unit testing was to verify if the business logic works as expected along with the implemented authorization.

```
Running 16 tests for test/HalbornVaultsRegistryTest.t.sol:HalbornVaultsRegistryTest
[PASS] test_addFactory_as_not_an_owner_reverts() (gas: 13589)
[PASS] test_addFactory_as_owner() (gas: 36274)
[PASS] test_addFactory_as_owner_twice_reverts() (gas: 37858)
[PASS] test_addFactory_then_addVault() (gas: 63901)
[PASS] test_addFactory_then_addVault_twice() (gas: 66648)
[PASS] test_addFactory_then_removeFactory_as_owner() (gas: 26905)
[PASS] test_addFactory_then_removeFactory_twice_as_owner_reverts() (gas: 28266)
[PASS] test_addVaultImpl_as_not_an_owner_reverts() (gas: 13542)
[PASS] test_addVaultImpl_as_owner() (gas: 36273)
[PASS] test_addVaultImpl_as_owner_twice_reverts() (gas: 37835)
[PASS] test_addVaultImpl_then_removeVaultImpl_as_owner() (gas: 26842)
[PASS] test_addVaultImpl_then_removeVaultImpl_twice_as_owner_reverts() (gas: 28184)
[PASS] test_addVault_as_not_owner_reverts() (gas: 15298)
[PASS] test_addVault_as_owner() (gas: 38622)
[PASS] test_removeFactory_as_not_an_owner_reverts() (gas: 13621)
[PASS] test_removeVaultImpl_as_not_an_owner_reverts() (gas: 13610)
Test result: ok. 16 passed; 0 failed; finished in 7.22ms
```

The purpose of the Keeper contract unit testing was to verify if the business logic works as expected along with the implemented authorization. Various scenarios related to signature handling were included.

```
Running 23 tests for test/HalbornKeeperTest.t.sol:HalbornKeeperTest
[PASS] test_KeeperOracles_addOracle_as_not_owner() (gas: 11675)
[PASS] test_KeeperOracles_addOracle_as_owner() (gas: 43717)
[PASS] test_KeeperOracles_addOracle_same_address_twice_as_owner_reverts() (gas: 45898)
[PASS] test_KeeperOracles_addOracle_then_removeOracle_as_owner() (gas: 34223)
[PASS] test_KeeperOracles_removeOracle_all_is_possible_as_owner() (gas: 103474)
[PASS] test_KeeperOracles_removeOracle_as_not_owner() (gas: 11708)
[PASS] test_KeeperOracles_removeOracle_for_more_than_validatorsMinOracles_is_possible_as_owner() (gas: 90080)
[PASS] test_KeeperOracles_removeOracle_for_ten_oracles_as_owner() (gas: 97451)
[PASS] test_KeeperOracles_removeOracle_non_existing_as_owner_reverts() (gas: 19874)
[PASS] test_KeeperOracles_setRewardsMinOracles_as_not_owner_reverts() (gas: 13953)
[PASS] test_KeeperOracles_setRewardsMinOracles_as_owner() (gas: 39826)
[PASS] test_KeeperOracles_setRewardsMinOracles_as_zero_owner_reverts() (gas: 16125)
[PASS] test_KeeperOracles_setRewardsMinOracles_for_more_than_totalOracles_as_owner_reverts() (gas: 19164)
[PASS] test_KeeperOracles_setRewardsMinOracles_for_totalOracles_as_owner() (gas: 41372)
[PASS] test_KeeperOracles_setValidatorsMinOracles_as_not_owner_reverts() (gas: 14022)
[PASS] test_KeeperOracles_setValidatorsMinOracles_as_owner() (gas: 39800)
[PASS] test_KeeperOracles_setValidatorsMinOracles_as_zero_owner_reverts() (gas: 16149)
[PASS] test_KeeperOracles_setValidatorsMinOracles_for_more_than_totalOracles_as_owner_reverts() (gas: 19233)
[PASS] test_KeeperOracles_setValidatorsMinOracles_for_totalOracles_as_owner() (gas: 41385)
[PASS] test_updateExitSignatures_twice_reverts() (gas: 625226)
[PASS] test_updateExitSignatures_twice_with_new_nonce() (gas: 681644)
[PASS] test_updateExitSignatures_valid_scenario() (gas: 612644)
[PASS] test_updateExitSignatures_without_validator_registered_reverts() (gas: 359961)
Test result: ok. 23 passed; 0 failed; finished in 81.24s
```

In the fallowing unit testing exercise, multiple scenarios were simulated to verify if:

- oracle signature verification works as expected,
- validator registration works as expected,
- multiple validator registration works as expected,
- set rewards root works as expected,
- state update, harvest and state update and deposit work as expected,
- the accounting related to ExitQueue works as expected,
- the accounting differences between standard withdrawal and claim assets from ExitQueue exists,

- the possibility to overwrite the ExitQueue records exists,
- the accounting in case of profit and loss works as expected.

Additionally, a scenario was identified in which harvest reverts due to integer underflow. Whenever a subsequent call to harvest sets a smaller MEV reward, this issue occurs. However, the StakeWise team confirmed that it is not considered as an issue as HarvestParams are strictly controlled off-chain and a subsequent call cannot introduce a smaller MEV reward compared to previous.

```
Running 39 tests for test/HalbornValidatorTest.t.sol:HalbornValidatorTest
[PASS] test_ValidatorsRegistry_get_deposit_root() (gas: 105315)
[PASS] test_keeper_verifyAllSignatures_all_duplicated_signatures_reverts() (gas: 259441)
[PASS] test_keeper_verifyAllSignatures_empty_exitSignaturesIpfsHash_reverts() (gas: 249061)
[PASS] test_keeper_verifyAllSignatures_incorrect_depositRoot_reverts() (gas: 249083)
[PASS] test_keeper_verifyAllSignatures_incorrect_validators_data_revert() (gas: 247538)
[PASS] test_keeper_verifyAllSignatures_incorrect_vault_address_reverts() (gas: 247239)
[PASS] test_keeper_verifyAllSignatures_last_signature_duplicated_reverts() (gas: 313595)
[PASS] test_keeper_verifyAllSignatures_only_one_signature_reverts() (gas: 239822)
[PASS] test_keeper_verifyAllSignatures_signatures_incorrect_order_reverts() (gas: 288343)
[PASS] test_keeper_verifyAllSignatures_signatures_not_long_enough_reverts() (gas: 242798)
[PASS] test_keeper_verifyAllSignatures_six_signatures_reverts() (gas: 241424)
[PASS] test_keeper_verifyAllSignatures_valid_scenario() (gas: 310742)
[PASS] test_registerValidator_attempt_to_register_twice_reverts() (gas: 933775)
[PASS] test_registerValidator_invalid_proof_reverts() (gas: 801643)
[PASS] test_registerValidator_not_enough_deposit_reverts() (gas: 799440)
[PASS] test_registerValidator_then_enterExitQueue_then_claimExitedAssets_valid_scenario() (gas: 1709149)
[PASS] test_registerValidator_then_enterExitQueue_then_updateState_100_ether_then_claimExitedAssets_valid_scenario() (gas: 2578509)
[PASS] test_registerValidator_then_enterExitQueue_then_updateState_then_claimExitedAssets_attempt_to_override_queue() (gas: 2840426)
[PASS] test_registerValidator_then_enterExitQueue_then_updateState_then_claimExitedAssets_but_reverse_order_valid_scenario() (gas: 2578465)
[PASS] test_registerValidator_then_enterExitQueue_then_updateState_then_claimExitedAssets_redeem_versus_claimExited_but_withdraw_withdrawableAssets_before() (gas: 2843656)
[PASS] test_registerValidator_then_enterExitQueue_then_updateState_then_claimExitedAssets_then_deposit_valid_scenario() (gas: 2792948)
[PASS] test_registerValidator_then_enterExitQueue_then_updateState_then_claimExitedAssets_valid_scenario() (gas: 2569794)
[PASS] test_registerValidator_then_enterExitQueue_then_updateState_then_claimExitedAssets_withdraw_versus_claimExited() (gas: 2996791)
[PASS] test_registerValidator_then_enterExitQueue_then_updateState_with_loss_then_claimExitedAssets_valid_scenario() (gas: 2547148)
[PASS] test_registerValidator_then_enterExitQueue_valid_scenario() (gas: 1327346)
[PASS] test_registerValidator_then_updateRewards_quadrice_various_but_fourth_as_first() (gas: 2141171)
[PASS] test_registerValidator_then_updateRewards_then_deposit() (gas: 1447336)
[PASS] test_registerValidator_then_updateRewards_then_harvest_then_enterExitQueue_valid_scenario() (gas: 1136329)
[FAIL. Reason: Arithmetic over/underflow] test_registerValidator_then_updateRewards_then_harvest_then_updateRewards_then_harvest_integer_underflow_reverts() (gas: 1809139)
[PASS] test_registerValidator_then_updateRewards_then_harvest_twice() (gas: 1673569)
[PASS] test_registerValidator_then_updateRewards_then_harvest_valid_scenario() (gas: 1669789)
[PASS] test_registerValidator_then_updateRewards_then_updateStateAndDeposit_valid_scenario() (gas: 1703189)
[PASS] test_registerValidator_then_updateRewards_twice_the_same_rewardTreeRoot_reverts() (gas: 1277836)
[PASS] test_registerValidator_then_updateRewards_twice_various() (gas: 1363321)
[PASS] test_registerValidator_then_updateRewards_twice_various_then_deposit_reverts() (gas: 1737207)
[PASS] test_registerValidator_then_updateRewards_valid_scenario() (gas: 1231867)
[PASS] test_registerValidator_then_updateState_with_loss_then_withdraw_valid_scenario() (gas: 2066792)
[PASS] test_registerValidator_valid_scenario() (gas: 889959)
[PASS] test_registerValidator_valid_scenario_then_updateRewards_too_early_reverts() (gas: 1173806)
Test result: FAILED. 38 passed; 1 failed; finished in 120.11s
```

The purpose of the OsToken contract unit testing was to verify if the business logic works as expected along with the implemented authorization, including minting, redeeming and burning of tokens. The accounting of positions, shares, and assets were checked manually.

```
Running 28 tests for test/HalbornOsTokenTest.t.sol:HalbornOsTokenTest
[PASS] test_OsToken_deposit_no_registerValidator_mintOsToken_reverts_NotCollateralized() (gas: 531724)
[PASS] test_OsToken_deposit_registerValidator_mintOsToken_above_LowLtv_reverts() (gas: 1001109)
[PASS] test_OsToken_deposit_registerValidator_mintOsToken_above_max_reverts_LowLtv() (gas: 989039)
[PASS] test_OsToken_deposit_registerValidator_mintOsToken_equal_to_LowLtv() (gas: 1023945)
[PASS] test_OsToken_deposit_registerValidator_mintOsToken_forward_time_burnOsToken() (gas: 1440466)
[PASS] test_OsToken_deposit_registerValidator_mintOsToken_forward_time_burnOsToken_forward_time_mintOsToken() (gas: 1861110)
[PASS] test_OsToken_deposit_registerValidator_mintOsToken_forward_time_simulateProfit_burnOsToken() (gas: 2237410)
[PASS] test_OsToken_deposit_registerValidator_mintOsToken_forward_time_simulateProfit_redeemTokens() (gas: 2056370)
[PASS] test_OsToken_deposit_registerValidator_mintOsToken_forward_time_updateState_burnOsToken() (gas: 1437919)
[PASS] test_OsToken_deposit_registerValidator_mintOsToken_liquidateOsToken_reverts_with_InvalidHealthFactor() (gas: 1241903)
[PASS] test_OsToken_deposit_registerValidator_mintOsToken_max_minus_one_reverts_LowLtv() (gas: 989062)
[PASS] test_OsToken_deposit_registerValidator_mintOsToken_max_reverts_LowLtv() (gas: 989082)
[PASS] test_OsToken_deposit_registerValidator_mintOsToken_possible() (gas: 1011995)
[PASS] test_OsToken_deposit_registerValidator_mintOsToken_simulateLoss_liquidateOsToken() (gas: 2205899)
[PASS] test_OsToken_deposit_registerValidator_mintOsToken_simulateLoss_redeemOsToken() (gas: 2208196)
[PASS] test_OsToken_deposit_registerValidator_mintOsToken_then_burnOsToken() (gas: 1159484)
[PASS] test_OsToken_deposit_registerValidator_mintOsToken_then_burnOsToken_less_than_max() (gas: 1220416)
[PASS] test_OsToken_deposit_registerValidator_mintOsToken_then_burnOsToken_too_manny_reverts() (gas: 1212970)
[PASS] test_OsToken_deposit_registerValidator_mintOsToken_u1_u2_simulateLoss_liquidateOsToken_u1() (gas: 2230586)
[PASS] test_OsToken_deposit_registerValidator_mintOsToken_u1_u2_simulateLoss_redeemOsToken_u3() (gas: 2233011)
[PASS] test_OsToken_deposit_registerValidator_mintOsToken_u1_u2_simulateLoss_redeemOsToken_u3_u2_reverts_RedemptionExceeded() (gas: 2259883)
[PASS] test_OsToken_deposit_registerValidator_updateRewards_mintOsToken_possible() (gas: 1355509)
[PASS] test_OsToken_setAvgRewardPerSecond_deposit_registerValidator_mintOsToken_updateState_forward_time_updateState() (gas: 1614183)
[PASS] test_OsToken_updateRewards_twice_mintOsToken_reverts_NotHarvested() (gas: 1405879)
[PASS] test_OsToken_updateRewards_updateState_mintOsToken_max() (gas: 1816842)
[PASS] test_OsToken_updateRewards_updateState_mintOsToken_possible() (gas: 1816817)
[PASS] test_OsToken_updateRewards_updateState_mintOsToken_twice() (gas: 1838669)
[PASS] test_OsToken_updateRewards_updateState_twice_mintOsToken_possible() (gas: 2562592)
Test result: ok. 28 passed; 0 failed; finished in 120.12s
```

The purpose of the OsTokenConfig unit testing was to verify if the business logic works as expected along with the implemented authorization.

```
Running 2 tests for test/HalbornOsTokenConfig.t.sol:HalbornOsTokenConfig
[PASS] test_OsTokenConfig_updateConfig() (gas: 38906)
[PASS] test_OsTokenConfig_updateConfig_as_not_owner_reverts() (gas: 36588)
Test result: ok. 2 passed; 0 failed; finished in 15.01ms
```

AUTOMATED TESTING

6.1 STATIC ANALYSIS REPORT

Description:

Halborn used automated testing techniques to enhance the coverage of certain areas of the smart contracts in scope. Among the tools used was Slither, a Solidity static analysis framework. After Halborn verified the smart contracts in the repository and was able to compile them correctly into their ABIs and binary format, Slither was run against the contracts. This tool can statically verify mathematical relationships between Solidity variables to detect invalid or inconsistent usage of the contracts' APIs across the entire code-base.
The security team assessed all findings identified by the Slither software, however, findings with severity Information and Optimization are not included in the below results for report readability.

Results:

Slither results for OsToken.sol	
Finding	**Impact**
OsToken.cumulativeFeePerShare() (contracts/osToken/OsToken.sol#216-244) uses a dangerous strict equality: - profitAccrued == 0 (contracts/osToken/OsToken.sol#222)	Medium
OsToken.cumulativeFeePerShare() (contracts/osToken/OsToken.sol#216-244) uses a dangerous strict equality: - treasuryAssets == 0 (contracts/osToken/OsToken.sol#226)	Medium
OsToken._convertToShares(uint256,uint256,uint256,Math.Rounding) (contracts/osToken/OsToken.sol#307-319) uses a dangerous strict equality: - (assets == 0 || totalShares == 0) (contracts/osToken/OsToken.sol#315-318)	Medium
OsToken.updateState() (contracts/osToken/OsToken.sol#247-302) uses a dangerous strict equality: - treasuryAssets == 0 (contracts/osToken/OsToken.sol#264)	Medium

Finding	Impact
OsToken._unclaimedAssets() (contracts/osToken/OsToken.sol#336-345) uses a dangerous strict equality: - timeElapsed == 0 (contracts/osToken/OsToken.sol#343)	Medium
OsToken.totalAssets() (contracts/osToken/OsToken.sol#84-90) uses a dangerous strict equality: - profitAccrued == 0 (contracts/osToken/OsToken.sol#86)	Medium
OsToken.updateState() (contracts/osToken/OsToken.sol#247-302) uses a dangerous strict equality: - profitAccrued == 0 (contracts/osToken/OsToken.sol#252)	Medium
OsToken.constructor(address,address,address,uint16,uint256,string,string)._keeper (contracts/osToken/OsToken.sol#61) lacks a zero-check on : - keeper = _keeper (contracts/osToken/OsToken.sol#69)	Low
OsToken._unclaimedAssets() (contracts/osToken/OsToken.sol#336-345) uses timestamp for comparisons Dangerous comparisons: - timeElapsed == 0 (contracts/osToken/OsToken.sol#343)	Low
OsToken._convertToShares(uint256,uint256,uint256,Math.Rounding) (contracts/osToken/OsToken.sol#307-319) uses timestamp for comparisons Dangerous comparisons: - (assets == 0 || totalShares == 0) (contracts/osToken/OsToken.sol#315-318)	Low
OsToken.updateState() (contracts/osToken/OsToken.sol#247-302) uses timestamp for comparisons Dangerous comparisons: - profitAccrued == 0 (contracts/osToken/OsToken.sol#252) - lastUpdateTimestamp != block.timestamp (contracts/osToken/OsToken.sol#255) - treasuryAssets == 0 (contracts/osToken/OsToken.sol#264)	Low
OsToken.cumulativeFeePerShare() (contracts/osToken/OsToken.sol#216-244) uses timestamp for comparisons Dangerous comparisons: - profitAccrued == 0 (contracts/osToken/OsToken.sol#222) - treasuryAssets == 0 (contracts/osToken/OsToken.sol#226)	Low
OsToken.totalAssets() (contracts/osToken/OsToken.sol#84-90) uses timestamp for comparisons Dangerous comparisons: - profitAccrued == 0 (contracts/osToken/OsToken.sol#86)	Low

Finding	Impact
ERC20.permit(address,address,uint256,uint256,uint8,bytes32,bytes32) (contracts/base/ERC20.sol#93-128) uses timestamp for comparisons Dangerous comparisons: - deadline < block.timestamp (contracts/base/ERC20.sol#103)	Low
End of table for OsToken.sol	

Slither results for PriceOracle.sol	
Finding	**Impact**
PriceOracle.constructor(address)._osToken (contracts/osToken/PriceOracle.sol#20) lacks a zero-check on : - osToken = _osToken (contracts/osToken/PriceOracle.sol#21)	Low
End of table for PriceOracle.sol	

Slither results for EthVaultFactory.sol	
Finding	**Impact**
OwnMevEscrow.harvest() (contracts/vaults/ethereum/mev/OwnMevEscrow.sol#23-32) uses a dangerous strict equality: - assets == 0 (contracts/vaults/ethereum/mev/OwnMevEscrow.sol#27)	Medium
EthVaultFactory.createVault(bytes,bool)._mevEscrow (contracts/vaults/ethereum/EthVaultFactory.sol#47) is a local variable never initialized	Medium
OwnMevEscrow.constructor(address)._vault (contracts/vaults/ethereum/mev/OwnMevEscrow.sol#18) lacks a zero-check on : - vault = address(_vault) (contracts/vaults/ethereum/mev/OwnMevEscrow.sol#19)	Low
EthVaultFactory.constructor(address,IVaultsRegistry)._implementation (contracts/vaults/ethereum/EthVaultFactory.sol#33) lacks a zero-check on : - implementation = _implementation (contracts/vaults/ethereum/EthVaultFactory.sol#34)	Low

Finding	Impact
Reentrancy in EthVaultFactory.createVault(bytes,bool) (contracts/vaults/ethereum/EthVaultFactory.sol#39-71):External calls: - vault = address(new ERC1967Proxy(implementation,)) (contracts/vaults/ethereum/EthVaultFactory.sol#44) - IEthVault(vault).initialize{value: msg.value}(params) (contracts/vaults/ethereum/EthVaultFactory.sol#58)External calls sending eth: - IEthVault(vault).initialize{value: msg.value}(params) (contracts/vaults/ethereum/EthVaultFactory.sol#58) State variables written after the call(s): - delete ownMevEscrow (contracts/vaults/ethereum/EthVaultFactory.sol#61) - delete vaultAdmin (contracts/vaults/ethereum/EthVaultFactory.sol#64)	Low
Reentrancy in EthVaultFactory.createVault(bytes,bool) (contracts/vaults/ethereum/EthVaultFactory.sol#39-71):External calls: - vault = address(new ERC1967Proxy(implementation,)) (contracts/vaults/ethereum/EthVaultFactory.sol#44) State variables written after the call(s): - ownMevEscrow = _mevEscrow (contracts/vaults/ethereum/EthVaultFactory.sol#51) - vaultAdmin = msg.sender (contracts/vaults/ethereum/EthVaultFactory.sol#55)	Low
Reentrancy in EthVaultFactory.createVault(bytes,bool) (contracts/vaults/ethereum/EthVaultFactory.sol#39-71):External calls: - vault = address(new ERC1967Proxy(implementation,)) (contracts/vaults/ethereum/EthVaultFactory.sol#44) - IEthVault(vault).initialize{value: msg.value}(params) (contracts/vaults/ethereum/EthVaultFactory.sol#58) - _vaultsRegistry.addVault(vault) (contracts/vaults/ethereum/EthVaultFactory.sol#67)External calls sending eth: - IEthVault(vault).initialize{value: msg.value}(params) (contracts/vaults/ethereum/EthVaultFactory.sol#58) Event emitted after the call(s): - VaultCreated(msg.sender,vault,_mevEscrow,params) (contracts/vaults/ethereum/EthVaultFactory.sol#70)	Low
End of table for EthVaultFactory.sol	

Slither results for EthPrivVault.sol	
Finding	**Impact**
VaultState.convertToAssets(uint256) (contracts/vaults/modules/VaultState.sol#51-54) uses a dangerous strict equality: - (totalShares == 0) (contracts/vaults/modules/VaultState.sol#53)	Medium
VaultState._processTotalAssetsDelta(int256) (contracts/vaults/modules/VaultState.sol#97-139) uses a dangerous strict equality: - totalShares == 0 (contracts/vaults/modules/VaultState.sol#125)	Medium
VaultState._updateExitQueue() (contracts/vaults/modules/VaultState.sol#145-177) uses a dangerous strict equality: - exitedAssets == 0 (contracts/vaults/modules/VaultState.sol#156)	Medium
VaultState._updateExitQueue() (contracts/vaults/modules/VaultState.sol#145-177) uses a dangerous strict equality: - burnedShares == 0 (contracts/vaults/modules/VaultState.sol#160)	Medium
VaultState._convertToShares(uint256,Math.Rounding) (contracts/vaults/modules/VaultState.sol#215-226) uses a dangerous strict equality: - (assets == 0 || totalShares == 0) (contracts/vaults/modules/VaultState.sol#222-225)	Medium
VaultState._updateExitQueue() (contracts/vaults/modules/VaultState.sol#145-177) uses a dangerous strict equality: - _queuedShares == 0 (contracts/vaults/modules/VaultState.sol#148)	Medium

Finding	Impact
Reentrancy in VaultEthStaking.updateStateAndDeposit(address,address,IKeeperRewards.HarvestParams) (contracts/vaults/modules/VaultEthStaking.sol#42-49):External calls: - updateState(harvestParams) (contracts/vaults/modules/VaultEthStaking.sol#47) - (totalAssetsDelta,unlockedMevDelta) = IKeeperRewards(_keeper).harvest(harvestParams) (contracts/vaults/modules/VaultMev.sol#44-46) - ISharedMevEscrow(_mevEscrow).harvest(unlockedMevDelta) (contracts/vaults/modules/VaultMev.sol#53) - totalAssetsDelta + int256(IOwnMevEscrow(_mevEscrow).harvest()) (contracts/vaults/modules/VaultMev.sol#59) State variables written after the call(s): - deposit(receiver,referrer) (contracts/vaults/modules/VaultEthStaking.sol#48) - _balances[owner] += shares (contracts/vaults/modules/VaultState.sol#192)VaultState._balances (contracts/vaults/modules/VaultState.sol#36) can be used in cross function reentrancies: - VaultState._burnShares(address,uint256) (contracts/vaults/modules/VaultState.sol#201-210) - VaultState._mintShares(address,uint256) (contracts/vaults/modules/VaultState.sol#184-194) - VaultEnterExit.enterExitQueue(uint256,address) (contracts/vaults/modules/VaultEnterExit.sol#57-83) - deposit(receiver,referrer) (contracts/vaults/modules/VaultEthStaking.sol#48) - _totalAssets = SafeCast.toUint128(totalAssetsAfter) (contracts/vaults/modules/VaultEnterExit.sol#150)VaultState._totalAssets (contracts/vaults/modules/VaultState.sol#26) can be used in cross function reentrancies: - VaultState._convertToShares(uint256,Math.Rounding) (contracts/vaults/modules/VaultState.sol#215-226) - VaultEnterExit._deposit(address,uint256,address) (contracts/vaults/modules/VaultEnterExit.sol#130-154) - VaultState._processTotalAssetsDelta(int256) (contracts/vaults/modules/VaultState.sol#97-139) - VaultState._updateExitQueue() (contracts/vaults/modules/VaultState.sol#145-177) - VaultState.convertToAssets(uint256) (contracts/vaults/modules/VaultState.sol#51-54) - VaultEnterExit.redeem(uint256,address) (contracts/vaults/modules/VaultEnterExit.sol#30-54) - VaultState.totalAssets()	Medium

Finding	Impact
Reentrancy in VaultOsToken._redeemOsToken(address,address,uint256,bool) (contracts/vaults/modules/VaultOsToken.sol#160-249):External calls: - _osToken.updateState() (contracts/vaults/modules/VaultOsToken.sol#170) - _osToken.burnShares(msg.sender,osTokenShares) (contracts/vaults/modules/VaultOsToken.sol#222) State variables written after the call(s): - _burnShares(owner,sharesToBurn) (contracts/vaults/modules/VaultOsToken.sol#236) - _balances[owner] -= shares (contracts/vaults/modules/VaultState.sol#203)VaultState._balances (contracts/vaults/modules/VaultState.sol#36) can be used in cross function reentrancies: - VaultState._burnShares(address,uint256) (contracts/vaults/modules/VaultState.sol#201-210) - VaultOsToken._checkOsTokenPosition(address) (contracts/vaults/modules/VaultOsToken.sol#273-291) - VaultState._mintShares(address,uint256) (contracts/vaults/modules/VaultState.sol#184-194) - VaultOsToken._redeemOsToken(address,address,uint256,bool) (contracts/vaults/modules/VaultOsToken.sol#160-249) - VaultEnterExit.enterExitQueue(uint256,address) (contracts/vaults/modules/VaultEnterExit.sol#57-83) - VaultOsToken.mintOsToken(address,uint256,address) (contracts/vaults/modules/VaultOsToken.sol#56-94) - _positions[owner] = position (contracts/vaults/modules/VaultOsToken.sol#226)VaultOsToken._positions (contracts/vaults/modules/VaultOsToken.sol#33) can be used in cross function reentrancies: - VaultOsToken._checkOsTokenPosition(address) (contracts/vaults/modules/VaultOsToken.sol#273-291) - VaultOsToken._redeemOsToken(address,address,uint256,bool) (contracts/vaults/modules/VaultOsToken.sol#160-249) - VaultOsToken.burnOsToken(uint128) (contracts/vaults/modules/VaultOsToken.sol#97-112) - VaultOsToken.mintOsToken(address,uint256,address) (contracts/vaults/modules/VaultOsToken.sol#56-94) - VaultOsToken.osTokenPositions(address) (contracts/vaults/modules/VaultOsToken.sol#49-53) - _totalAssets -= SafeCast.toUint128(receivedAssets) (contracts/vaults/modules/VaultOsToken.sol#232)VaultState._totalAssets (contracts/vaults/modules/VaultState.sol#26) can be used in cross function reentrancies:	Medium

AUTOMATED TESTING

Finding	Impact
ExitQueue.getCheckpointIndex(ExitQueue.History,uint256).low (contracts/libraries/ExitQueue.sol#57) is a local variable never initialized	Medium
VaultOsToken._redeemOsToken(address,address,uint256,bool) (contracts/vaults/modules/VaultOsToken.sol#160-249) ignores return value by _osToken.burnShares(msg.sender,osTokenShares) (contracts/vaults/modules/VaultOsToken.sol#222)	Medium
EthPrivVault.constructor(address,address,address,address,address,address)._keeper (contracts/vaults/ethereum/EthPrivVault.sol#36) shadows: - VaultImmutables._keeper (contracts/vaults/modules/VaultImmutables.sol#15) (state variable)	Low
EthVault.constructor(address,address,address,address,address,address)._validatorsRegistry (contracts/vaults/ethereum/EthVault.sol#57) shadows: - VaultImmutables._validatorsRegistry (contracts/vaults/modules/VaultImmutables.sol#21) (state variable)	Low
EthPrivVault.constructor(address,address,address,address,address,address)._vaultsRegistry (contracts/vaults/ethereum/EthPrivVault.sol#37) shadows: - VaultImmutables._vaultsRegistry (contracts/vaults/modules/VaultImmutables.sol#18) (state variable)	Low
EthVault.constructor(address,address,address,address,address,address)._vaultsRegistry (contracts/vaults/ethereum/EthVault.sol#56) shadows: - VaultImmutables._vaultsRegistry (contracts/vaults/modules/VaultImmutables.sol#18) (state variable)	Low
EthPrivVault.constructor(address,address,address,address,address,address)._validatorsRegistry (contracts/vaults/ethereum/EthPrivVault.sol#38) shadows: - VaultImmutables._validatorsRegistry (contracts/vaults/modules/VaultImmutables.sol#21) (state variable)	Low
EthVault.constructor(address,address,address,address,address,address)._keeper (contracts/vaults/ethereum/EthVault.sol#55) shadows: - VaultImmutables._keeper (contracts/vaults/modules/VaultImmutables.sol#15) (state variable)	Low

Finding	Impact
EthVault.__EthVault_init(address,address,IEthVault.EthVaultInitParams).admin (contracts/vaults/ethereum/EthVault.sol#131) shadows: - VaultAdmin.admin (contracts/vaults/modules/VaultAdmin.sol#16) (state variable) - IVaultAdmin.admin() (contracts/interfaces/IVaultAdmin.sol#22) (function)	Low
EthPrivVault.initialize(bytes).admin (contracts/vaults/ethereum/EthPrivVault.sol#50) shadows: - VaultAdmin.admin (contracts/vaults/modules/VaultAdmin.sol#16) (state variable) - IVaultAdmin.admin() (contracts/interfaces/IVaultAdmin.sol#22) (function)	Low
Multicall.multicall(bytes[]) (contracts/base/Multicall.sol#15-31) has external calls inside a loop: (success,result) = address(this).delegatecall(data[i]) (contracts/base/Multicall.sol#18)	Low
Reentrancy in VaultOsToken.burnOsToken(uint128) (contracts/vaults/modules/VaultOsToken.sol#97-112):External calls: - assets = _osToken.burnShares(msg.sender,osTokenShares) (contracts/vaults/modules/VaultOsToken.sol#99) State variables written after the call(s): - _positions[msg.sender] = position (contracts/vaults/modules/VaultOsToken.sol#108)	Low
Reentrancy in VaultValidators.registerValidator(IKeeperValidators.ApprovalParams,bytes32[]) (contracts/vaults/modules/VaultValidators.sol#45-82):External calls: - IKeeperValidators(_keeper).approveValidators(keeperParams) (contracts/vaults/modules/VaultValidators.sol#52) State variables written after the call(s): - validatorIndex = currentIndex + 1 (contracts/vaults/modules/VaultValidators.sol#80)	Low

Finding	Impact
Reentrancy in VaultValidators.registerValidators(IKeeperValidators.ApprovalParams,uint256[],bool[],bytes32[]) (contracts/vaults/modules/VaultValidators.sol#85-128):External calls: - IKeeperValidators(_keeper).approveValidators(keeperParams) (contracts/vaults/modules/VaultValidators.sol#94) State variables written after the call(s): - validatorIndex += validatorsCount (contracts/vaults/modules/VaultValidators.sol#126)	Low
Reentrancy in VaultOsToken.mintOsToken(address,uint256,address) (contracts/vaults/modules/VaultOsToken.sol#56-94):External calls: - osTokenShares = _osToken.mintShares(receiver,assets) (contracts/vaults/modules/VaultOsToken.sol#65) State variables written after the call(s): - _positions[msg.sender] = position (contracts/vaults/modules/VaultOsToken.sol#90)	Low
Reentrancy in VaultEthStaking.updateStateAndDeposit(address,address,IKeeperRewards.HarvestParams) (contracts/vaults/modules/VaultEthStaking.sol#42-49):External calls: - updateState(harvestParams) (contracts/vaults/modules/VaultEthStaking.sol#47) - (totalAssetsDelta,unlockedMevDelta) = IKeeperRewards(_keeper).harvest(harvestParams) (contracts/vaults/modules/VaultMev.sol#44-46) - ISharedMevEscrow(_mevEscrow).harvest(unlockedMevDelta) (contracts/vaults/modules/VaultMev.sol#53) - totalAssetsDelta + int256(IOwnMevEscrow(_mevEscrow).harvest()) (contracts/vaults/modules/VaultMev.sol#59) Event emitted after the call(s): - Deposited(msg.sender,to,assets,shares,referrer) (contracts/vaults/modules/VaultEnterExit.sol#153) - deposit(receiver,referrer) (contracts/vaults/modules/VaultEthStaking.sol#48)	Low

Finding	Impact
Reentrancy in VaultEthStaking._registerMultipleValidators(bytes,uint256[]) (contracts/vaults/modules/VaultEthStaking.sol#77-116):External calls: - IEthValidatorsRegistry(_validatorsRegistry).deposit{value: validatorDeposit}(publicKey,withdrawalCreds,validator,bytes32(validator)) (contracts/vaults/modules/VaultEthStaking.sol#102-107)Event emitted after the call(s): - ValidatorRegistered(publicKey) (contracts/vaults/modules/VaultEthStaking.sol#114)	Low
Reentrancy in VaultOsToken.burnOsToken(uint128) (contracts/vaults/modules/VaultOsToken.sol#97-112):External calls: - assets = _osToken.burnShares(msg.sender,osTokenShares) (contracts/vaults/modules/VaultOsToken.sol#99) Event emitted after the call(s): - OsTokenBurned(msg.sender,assets,osTokenShares) (contracts/vaults/modules/VaultOsToken.sol#111)	Low
Reentrancy in VaultEthStaking._registerSingleValidator(bytes) (contracts/vaults/modules/VaultEthStaking.sol#64-74):External calls: - IEthValidatorsRegistry(_validatorsRegistry).deposit{value: _validatorDeposit()}(publicKey,_withdrawalCredentials(),validator,bytes32(validator)) (contracts/vaults/modules/VaultEthStaking.sol#66-71) Event emitted after the call(s): - ValidatorRegistered(publicKey) (contracts/vaults/modules/VaultEthStaking.sol#73)	Low
Reentrancy in VaultOsToken.redeemOsToken(uint256,address,address) (contracts/vaults/modules/VaultOsToken.sol#125-132):External calls: - receivedAssets = _redeemOsToken(owner,receiver,osTokenShares,false) (contracts/vaults/modules/VaultOsToken.sol#130) - _osToken.updateState() (contracts/vaults/modules/VaultOsToken.sol#170) - _osToken.burnShares(msg.sender,osTokenShares) (contracts/vaults/modules/VaultOsToken.sol#222) Event emitted after the call(s): - OsTokenRedeemed(msg.sender,owner,receiver,osTokenShares,receivedAssets) (contracts/vaults/modules/VaultOsToken.sol#131)	Low

Finding	Impact
Reentrancy in VaultOsToken.liquidateOsToken(uint256,address,address) (contracts/vaults/modules/VaultOsToken.sol#115-122):External calls: - receivedAssets = _redeemOsToken(owner,receiver,osTokenShares,true) (contracts/vaults/modules/VaultOsToken.sol#120) - _osToken.updateState() (contracts/vaults/modules/VaultOsToken.sol#170) - _osToken.burnShares(msg.sender,osTokenShares) (contracts/vaults/modules/VaultOsToken.sol#222) Event emitted after the call(s): - OsTokenLiquidated(msg.sender,owner,receiver,osTokenShares,receivedAssets) (contracts/vaults/modules/VaultOsToken.sol#121)	Low
Reentrancy in VaultOsToken.mintOsToken(address,uint256,address) (contracts/vaults/modules/VaultOsToken.sol#56-94):External calls: - osTokenShares = _osToken.mintShares(receiver,assets) (contracts/vaults/modules/VaultOsToken.sol#65) Event emitted after the call(s): - OsTokenMinted(msg.sender,receiver,assets,osTokenShares,referrer) (contracts/vaults/modules/VaultOsToken.sol#93)	Low
VaultState.canUpdateExitQueue() (contracts/vaults/modules/VaultState.sol#77-79) uses timestamp for comparisons Dangerous comparisons: - block.timestamp >= _exitQueueNextUpdate (contracts/vaults/modules/VaultState.sol#78)	Low
End of table for EthPrivVault.sol	

Slither results for EthPrivErc20Vault.sol	
Finding	**Impact**
VaultState.convertToAssets(uint256) (contracts/vaults/modules/VaultState.sol#51-54) uses a dangerous strict equality: - (totalShares == 0) (contracts/vaults/modules/VaultState.sol#53)	Medium
VaultState._processTotalAssetsDelta(int256) (contracts/vaults/modules/VaultState.sol#97-139) uses a dangerous strict equality: - totalShares == 0 (contracts/vaults/modules/VaultState.sol#125)	Medium

Finding	Impact
VaultState._updateExitQueue() (contracts/vaults/modules/VaultState.sol#145-177) uses a dangerous strict equality: - exitedAssets == 0 (contracts/vaults/modules/VaultState.sol#156)	Medium
VaultState._updateExitQueue() (contracts/vaults/modules/VaultState.sol#145-177) uses a dangerous strict equality: - burnedShares == 0 (contracts/vaults/modules/VaultState.sol#160)	Medium
VaultState._convertToShares(uint256,Math.Rounding) (contracts/vaults/modules/VaultState.sol#215-226) uses a dangerous strict equality: - (assets == 0 || totalShares == 0) (contracts/vaults/modules/VaultState.sol#222-225)	Medium
VaultState._updateExitQueue() (contracts/vaults/modules/VaultState.sol#145-177) uses a dangerous strict equality: - _queuedShares == 0 (contracts/vaults/modules/VaultState.sol#148)	Medium

Finding	Impact
Reentrancy in VaultEthStaking.updateStateAndDeposit(address,address,IKeeperRewards.HarvestParams) (contracts/vaults/modules/VaultEthStaking.sol#42-49):External calls: - updateState(harvestParams) (contracts/vaults/modules/VaultEthStaking.sol#47) - (totalAssetsDelta,unlockedMevDelta) = IKeeperRewards(_keeper).harvest(harvestParams) (contracts/vaults/modules/VaultMev.sol#44-46) - ISharedMevEscrow(_mevEscrow).harvest(unlockedMevDelta) (contracts/vaults/modules/VaultMev.sol#53) - totalAssetsDelta + int256(IOwnMevEscrow(_mevEscrow).harvest()) (contracts/vaults/modules/VaultMev.sol#59) State variables written after the call(s): - deposit(receiver,referrer) (contracts/vaults/modules/VaultEthStaking.sol#48) - _balances[owner] += shares (contracts/vaults/modules/VaultState.sol#192)VaultState._balances (contracts/vaults/modules/VaultState.sol#36) can be used in cross function reentrancies: - VaultState._burnShares(address,uint256) (contracts/vaults/modules/VaultState.sol#201-210) - VaultState._mintShares(address,uint256) (contracts/vaults/modules/VaultState.sol#184-194) - VaultEnterExit.enterExitQueue(uint256,address) (contracts/vaults/modules/VaultEnterExit.sol#57-83) - deposit(receiver,referrer) (contracts/vaults/modules/VaultEthStaking.sol#48) - _totalAssets = SafeCast.toUint128(totalAssetsAfter) (contracts/vaults/modules/VaultEnterExit.sol#150)VaultState._totalAssets (contracts/vaults/modules/VaultState.sol#26) can be used in cross function reentrancies: - VaultState._convertToShares(uint256,Math.Rounding) (contracts/vaults/modules/VaultState.sol#215-226) - VaultEnterExit._deposit(address,uint256,address) (contracts/vaults/modules/VaultEnterExit.sol#130-154) - VaultState._processTotalAssetsDelta(int256) (contracts/vaults/modules/VaultState.sol#97-139) - VaultState._updateExitQueue() (contracts/vaults/modules/VaultState.sol#145-177) - VaultState.convertToAssets(uint256) (contracts/vaults/modules/VaultState.sol#51-54) - VaultEnterExit.redeem(uint256,address) (contracts/vaults/modules/VaultEnterExit.sol#30-54) - VaultState.totalAssets()	Medium

Finding	Impact
Reentrancy in VaultOsToken._redeemOsToken(address,address,uint256,bool) (contracts/vaults/modules/VaultOsToken.sol#160-249):External calls: - _osToken.updateState() (contracts/vaults/modules/VaultOsToken.sol#170) - _osToken.burnShares(msg.sender,osTokenShares) (contracts/vaults/modules/VaultOsToken.sol#222) State variables written after the call(s): - _burnShares(owner,sharesToBurn) (contracts/vaults/modules/VaultOsToken.sol#236) - _balances[owner] -= shares (contracts/vaults/modules/VaultState.sol#203)VaultState._balances (contracts/vaults/modules/VaultState.sol#36) can be used in cross function reentrancies: - VaultState._burnShares(address,uint256) (contracts/vaults/modules/VaultState.sol#201-210) - VaultOsToken._checkOsTokenPosition(address) (contracts/vaults/modules/VaultOsToken.sol#273-291) - VaultState._mintShares(address,uint256) (contracts/vaults/modules/VaultState.sol#184-194) - VaultOsToken._redeemOsToken(address,address,uint256,bool) (contracts/vaults/modules/VaultOsToken.sol#160-249) - VaultEnterExit.enterExitQueue(uint256,address) (contracts/vaults/modules/VaultEnterExit.sol#57-83) - VaultOsToken.mintOsToken(address,uint256,address) (contracts/vaults/modules/VaultOsToken.sol#56-94) - _positions[owner] = position (contracts/vaults/modules/VaultOsToken.sol#226)VaultOsToken._positions (contracts/vaults/modules/VaultOsToken.sol#33) can be used in cross function reentrancies: - VaultOsToken._checkOsTokenPosition(address) (contracts/vaults/modules/VaultOsToken.sol#273-291) - VaultOsToken._redeemOsToken(address,address,uint256,bool) (contracts/vaults/modules/VaultOsToken.sol#160-249) - VaultOsToken.burnOsToken(uint128) (contracts/vaults/modules/VaultOsToken.sol#97-112) - VaultOsToken.mintOsToken(address,uint256,address) (contracts/vaults/modules/VaultOsToken.sol#56-94) - VaultOsToken.osTokenPositions(address) (contracts/vaults/modules/VaultOsToken.sol#49-53) - _totalAssets -= SafeCast.toUint128(receivedAssets) (contracts/vaults/modules/VaultOsToken.sol#232)VaultState._totalAssets (contracts/vaults/modules/VaultState.sol#26) can be used in cross function reentrancies:	Medium

Finding	Impact
ExitQueue.getCheckpointIndex(ExitQueue.History,uint256).low (contracts/libraries/ExitQueue.sol#57) is a local variable never initialized	Medium
VaultOsToken._redeemOsToken(address,address,uint256,bool) (contracts/vaults/modules/VaultOsToken.sol#160-249) ignores return value by _osToken.burnShares(msg.sender,osTokenShares) (contracts/vaults/modules/VaultOsToken.sol#222)	Medium
EthErc20Vault.constructor(address,address,address,address,address,address)._vaultsRegistry (contracts/vaults/ethereum/EthErc20Vault.sol#61) shadows: - VaultImmutables._vaultsRegistry (contracts/vaults/modules/VaultImmutables.sol#18) (state variable)	Low
EthErc20Vault.__EthErc20Vault_init(address,address,IEthErc20Vault.EthErc20VaultInitParams).admin (contracts/vaults/ethereum/EthErc20Vault.sol#174) shadows: - VaultAdmin.admin (contracts/vaults/modules/VaultAdmin.sol#16) (state variable) - IVaultAdmin.admin() (contracts/interfaces/IVaultAdmin.sol#22) (function)	Low
EthErc20Vault.constructor(address,address,address,address,address,address)._validatorsRegistry (contracts/vaults/ethereum/EthErc20Vault.sol#62) shadows: - VaultImmutables._validatorsRegistry (contracts/vaults/modules/VaultImmutables.sol#21) (state variable)	Low
EthPrivErc20Vault.initialize(bytes).admin (contracts/vaults/ethereum/EthPrivErc20Vault.sol#57) shadows: - VaultAdmin.admin (contracts/vaults/modules/VaultAdmin.sol#16) (state variable) - IVaultAdmin.admin() (contracts/interfaces/IVaultAdmin.sol#22) (function)	Low
EthPrivErc20Vault.constructor(address,address,address,address,address,address)._keeper (contracts/vaults/ethereum/EthPrivErc20Vault.sol#36) shadows: - VaultImmutables._keeper (contracts/vaults/modules/VaultImmutables.sol#15) (state variable)	Low

Finding	Impact
EthPrivErc20Vault.constructor(address,address,address,address,address,address)._validatorsRegistry (contracts/vaults/ethereum/EthPrivErc20Vault.sol#38) shadows: - VaultImmutables._validatorsRegistry (contracts/vaults/modules/VaultImmutables.sol#21) (state variable)	Low
EthErc20Vault.constructor(address,address,address,address,address,address)._keeper (contracts/vaults/ethereum/EthErc20Vault.sol#60) shadows: - VaultImmutables._keeper (contracts/vaults/modules/VaultImmutables.sol#15) (state variable)	Low
EthPrivErc20Vault.constructor(address,address,address,address,address,address)._vaultsRegistry (contracts/vaults/ethereum/EthPrivErc20Vault.sol#37) shadows: - VaultImmutables._vaultsRegistry (contracts/vaults/modules/VaultImmutables.sol#18) (state variable)	Low
Multicall.multicall(bytes[]) (contracts/base/Multicall.sol#15-31) has external calls inside a loop: (success,result) = address(this).delegatecall(data[i]) (contracts/base/Multicall.sol#18)	Low
Reentrancy in VaultOsToken.burnOsToken(uint128) (contracts/vaults/modules/VaultOsToken.sol#97-112):External calls: - assets = _osToken.burnShares(msg.sender,osTokenShares) (contracts/vaults/modules/VaultOsToken.sol#99) State variables written after the call(s): - _positions[msg.sender] = position (contracts/vaults/modules/VaultOsToken.sol#108)	Low
Reentrancy in VaultValidators.registerValidator(IKeeperValidators.ApprovalParams,bytes32[]) (contracts/vaults/modules/VaultValidators.sol#45-82):External calls: - IKeeperValidators(_keeper).approveValidators(keeperParams) (contracts/vaults/modules/VaultValidators.sol#52) State variables written after the call(s): - validatorIndex = currentIndex + 1 (contracts/vaults/modules/VaultValidators.sol#80)	Low

Finding	Impact
Reentrancy in VaultValidators.registerValidators(IKeeperValidators.ApprovalParams,uint256[],bool[],bytes32[]) (contracts/vaults/modules/VaultValidators.sol#85-128):External calls: - IKeeperValidators(_keeper).approveValidators(keeperParams) (contracts/vaults/modules/VaultValidators.sol#94) State variables written after the call(s): - validatorIndex += validatorsCount (contracts/vaults/modules/VaultValidators.sol#126)	Low
Reentrancy in VaultOsToken.mintOsToken(address,uint256,address) (contracts/vaults/modules/VaultOsToken.sol#56-94):External calls: - osTokenShares = _osToken.mintShares(receiver,assets) (contracts/vaults/modules/VaultOsToken.sol#65) State variables written after the call(s): - _positions[msg.sender] = position (contracts/vaults/modules/VaultOsToken.sol#90)	Low
Reentrancy in VaultEthStaking.updateStateAndDeposit(address,address,IKeeperRewards.HarvestParams) (contracts/vaults/modules/VaultEthStaking.sol#42-49):External calls: - updateState(harvestParams) (contracts/vaults/modules/VaultEthStaking.sol#47) - (totalAssetsDelta,unlockedMevDelta) = IKeeperRewards(_keeper).harvest(harvestParams) (contracts/vaults/modules/VaultMev.sol#44-46) - ISharedMevEscrow(_mevEscrow).harvest(unlockedMevDelta) (contracts/vaults/modules/VaultMev.sol#53) - totalAssetsDelta + int256(IOwnMevEscrow(_mevEscrow).harvest()) (contracts/vaults/modules/VaultMev.sol#59) Event emitted after the call(s): - Deposited(msg.sender,to,assets,shares,referrer) (contracts/vaults/modules/VaultEnterExit.sol#153) - deposit(receiver,referrer) (contracts/vaults/modules/VaultEthStaking.sol#48)	Low

Finding	Impact
Reentrancy in VaultEthStaking._registerMultipleValidators(bytes,uint256[]) (contracts/vaults/modules/VaultEthStaking.sol#77-116):External calls: - IEthValidatorsRegistry(_validatorsRegistry).deposit{value: validatorDeposit}(publicKey,withdrawalCreds,validator,bytes32(validator)) (contracts/vaults/modules/VaultEthStaking.sol#102-107)Event emitted after the call(s): - ValidatorRegistered(publicKey) (contracts/vaults/modules/VaultEthStaking.sol#114)	Low
Reentrancy in VaultOsToken.burnOsToken(uint128) (contracts/vaults/modules/VaultOsToken.sol#97-112):External calls: - assets = _osToken.burnShares(msg.sender,osTokenShares) (contracts/vaults/modules/VaultOsToken.sol#99) Event emitted after the call(s): - OsTokenBurned(msg.sender,assets,osTokenShares) (contracts/vaults/modules/VaultOsToken.sol#111)	Low
Reentrancy in VaultEthStaking._registerSingleValidator(bytes) (contracts/vaults/modules/VaultEthStaking.sol#64-74):External calls: - IEthValidatorsRegistry(_validatorsRegistry).deposit{value: _validatorDeposit()}(publicKey,_withdrawalCredentials(),validator,bytes32(validator)) (contracts/vaults/modules/VaultEthStaking.sol#66-71) Event emitted after the call(s): - ValidatorRegistered(publicKey) (contracts/vaults/modules/VaultEthStaking.sol#73)	Low
Reentrancy in VaultOsToken.redeemOsToken(uint256,address,address) (contracts/vaults/modules/VaultOsToken.sol#125-132):External calls: - receivedAssets = _redeemOsToken(owner,receiver,osTokenShares,false) (contracts/vaults/modules/VaultOsToken.sol#130) - _osToken.updateState() (contracts/vaults/modules/VaultOsToken.sol#170) - _osToken.burnShares(msg.sender,osTokenShares) (contracts/vaults/modules/VaultOsToken.sol#222) Event emitted after the call(s): - OsTokenRedeemed(msg.sender,owner,receiver,osTokenShares,receivedAssets) (contracts/vaults/modules/VaultOsToken.sol#131)	Low

Finding	Impact
Reentrancy in VaultOsToken.liquidateOsToken(uint256,address,address) (contracts/vaults/modules/VaultOsToken.sol#115-122):External calls: - receivedAssets = _redeemOsToken(owner,receiver,osTokenShares,true) (contracts/vaults/modules/VaultOsToken.sol#120) - _osToken.updateState() (contracts/vaults/modules/VaultOsToken.sol#170) - _osToken.burnShares(msg.sender,osTokenShares) (contracts/vaults/modules/VaultOsToken.sol#222) Event emitted after the call(s): - OsTokenLiquidated(msg.sender,owner,receiver,osTokenShares,receivedAssets) (contracts/vaults/modules/VaultOsToken.sol#121)	Low
Reentrancy in VaultOsToken.mintOsToken(address,uint256,address) (contracts/vaults/modules/VaultOsToken.sol#56-94):External calls: - osTokenShares = _osToken.mintShares(receiver,assets) (contracts/vaults/modules/VaultOsToken.sol#65) Event emitted after the call(s): - OsTokenMinted(msg.sender,receiver,assets,osTokenShares,referrer) (contracts/vaults/modules/VaultOsToken.sol#93)	Low
ERC20Upgradeable.permit(address,address,uint256,uint256,uint8,bytes32,bytes32) (contracts/base/ERC20Upgradeable.sol#98-133) uses timestamp for comparisons Dangerous comparisons: - deadline < block.timestamp (contracts/base/ERC20Upgradeable.sol#108)	Low
VaultState.canUpdateExitQueue() (contracts/vaults/modules/VaultState.sol#77-79) uses timestamp for comparisons Dangerous comparisons: - block.timestamp >= _exitQueueNextUpdate (contracts/vaults/modules/VaultState.sol#78)	Low
End of table for EthPrivErc20Vault.sol	

Slither results for EthGenesisVault.sol	
Finding	**Impact**
VaultState.convertToAssets(uint256) (contracts/vaults/modules/VaultState.sol#51-54) uses a dangerous strict equality: - (totalShares == 0) (contracts/vaults/modules/VaultState.sol#53)	Medium

Finding	Impact
VaultState._processTotalAssetsDelta(int256) (contracts/vaults/modules/VaultState.sol#97-139) uses a dangerous strict equality: - totalShares == 0 (contracts/vaults/modules/VaultState.sol#125)	Medium
VaultState._updateExitQueue() (contracts/vaults/modules/VaultState.sol#145-177) uses a dangerous strict equality: - exitedAssets == 0 (contracts/vaults/modules/VaultState.sol#156)	Medium
VaultState._updateExitQueue() (contracts/vaults/modules/VaultState.sol#145-177) uses a dangerous strict equality: - burnedShares == 0 (contracts/vaults/modules/VaultState.sol#160)	Medium
VaultState._convertToShares(uint256,Math.Rounding) (contracts/vaults/modules/VaultState.sol#215-226) uses a dangerous strict equality: - (assets == 0 || totalShares == 0) (contracts/vaults/modules/VaultState.sol#222-225)	Medium
VaultState._updateExitQueue() (contracts/vaults/modules/VaultState.sol#145-177) uses a dangerous strict equality: - _queuedShares == 0 (contracts/vaults/modules/VaultState.sol#148)	Medium

Finding	Impact
Reentrancy in VaultEthStaking.updateStateAndDeposit(address,address,IKeeperRewards.HarvestParams) (contracts/vaults/modules/VaultEthStaking.sol#42-49):External calls: - updateState(harvestParams) (contracts/vaults/modules/VaultEthStaking.sol#47) - (totalAssetsDelta,unlockedMevDelta) = IKeeperRewards(_keeper).harvest(harvestParams) (contracts/vaults/modules/VaultMev.sol#44-46) - ISharedMevEscrow(_mevEscrow).harvest(unlockedMevDelta) (contracts/vaults/modules/VaultMev.sol#53) - totalAssetsDelta + int256(IOwnMevEscrow(_mevEscrow).harvest()) (contracts/vaults/modules/VaultMev.sol#59) State variables written after the call(s): - deposit(receiver,referrer) (contracts/vaults/modules/VaultEthStaking.sol#48) - _balances[owner] += shares (contracts/vaults/modules/VaultState.sol#192)VaultState._balances (contracts/vaults/modules/VaultState.sol#36) can be used in cross function reentrancies: - VaultState._burnShares(address,uint256) (contracts/vaults/modules/VaultState.sol#201-210) - VaultState._mintShares(address,uint256) (contracts/vaults/modules/VaultState.sol#184-194) - VaultEnterExit.enterExitQueue(uint256,address) (contracts/vaults/modules/VaultEnterExit.sol#57-83) - deposit(receiver,referrer) (contracts/vaults/modules/VaultEthStaking.sol#48) - _totalAssets = SafeCast.toUint128(totalAssetsAfter) (contracts/vaults/modules/VaultEnterExit.sol#150)VaultState._totalAssets (contracts/vaults/modules/VaultState.sol#26) can be used in cross function reentrancies: - VaultState._convertToShares(uint256,Math.Rounding) (contracts/vaults/modules/VaultState.sol#215-226) - VaultEnterExit._deposit(address,uint256,address) (contracts/vaults/modules/VaultEnterExit.sol#130-154) - VaultState._processTotalAssetsDelta(int256) (contracts/vaults/modules/VaultState.sol#97-139) - VaultState._updateExitQueue() (contracts/vaults/modules/VaultState.sol#145-177) - VaultState.convertToAssets(uint256) (contracts/vaults/modules/VaultState.sol#51-54) - VaultEnterExit.redeem(uint256,address) (contracts/vaults/modules/VaultEnterExit.sol#30-54) - VaultState.totalAssets()	Medium

Finding	Impact
Reentrancy in VaultOsToken._redeemOsToken(address,address,uint256,bool) (contracts/vaults/modules/VaultOsToken.sol#160-249):External calls: - _osToken.updateState() (contracts/vaults/modules/VaultOsToken.sol#170) - _osToken.burnShares(msg.sender,osTokenShares) (contracts/vaults/modules/VaultOsToken.sol#222) State variables written after the call(s): - _burnShares(owner,sharesToBurn) (contracts/vaults/modules/VaultOsToken.sol#236) - _balances[owner] -= shares (contracts/vaults/modules/VaultState.sol#203)VaultState._balances (contracts/vaults/modules/VaultState.sol#36) can be used in cross function reentrancies: - VaultState._burnShares(address,uint256) (contracts/vaults/modules/VaultState.sol#201-210) - VaultOsToken._checkOsTokenPosition(address) (contracts/vaults/modules/VaultOsToken.sol#273-291) - VaultState._mintShares(address,uint256) (contracts/vaults/modules/VaultState.sol#184-194) - VaultOsToken._redeemOsToken(address,address,uint256,bool) (contracts/vaults/modules/VaultOsToken.sol#160-249) - VaultEnterExit.enterExitQueue(uint256,address) (contracts/vaults/modules/VaultEnterExit.sol#57-83) - VaultOsToken.mintOsToken(address,uint256,address) (contracts/vaults/modules/VaultOsToken.sol#56-94) - _positions[owner] = position (contracts/vaults/modules/VaultOsToken.sol#226)VaultOsToken._positions (contracts/vaults/modules/VaultOsToken.sol#33) can be used in cross function reentrancies: - VaultOsToken._checkOsTokenPosition(address) (contracts/vaults/modules/VaultOsToken.sol#273-291) - VaultOsToken._redeemOsToken(address,address,uint256,bool) (contracts/vaults/modules/VaultOsToken.sol#160-249) - VaultOsToken.burnOsToken(uint128) (contracts/vaults/modules/VaultOsToken.sol#97-112) - VaultOsToken.mintOsToken(address,uint256,address) (contracts/vaults/modules/VaultOsToken.sol#56-94) - VaultOsToken.osTokenPositions(address) (contracts/vaults/modules/VaultOsToken.sol#49-53) - _totalAssets -= SafeCast.toUint128(receivedAssets) (contracts/vaults/modules/VaultOsToken.sol#232)VaultState._totalAssets (contracts/vaults/modules/VaultState.sol#26) can be used in cross function reentrancies:	Medium

Finding	Impact
ExitQueue.getCheckpointIndex(ExitQueue.History,uint256).low (contracts/libraries/ExitQueue.sol#57) is a local variable never initialized	Medium
VaultOsToken._redeemOsToken(address,address,uint256,bool) (contracts/vaults/modules/VaultOsToken.sol#160-249) ignores return value by _osToken.burnShares(msg.sender,osTokenShares) (contracts/vaults/modules/VaultOsToken.sol#222)	Medium
EthVault.constructor(address,address,address,address,address,address)._validatorsRegistry (contracts/vaults/ethereum/EthVault.sol#57) shadows: - VaultImmutables._validatorsRegistry (contracts/vaults/modules/VaultImmutables.sol#21) (state variable)	Low
EthGenesisVault.constructor(address,address,address,address,address,address,address,address)._keeper (contracts/vaults/ethereum/EthGenesisVault.sol#45) shadows: - VaultImmutables._keeper (contracts/vaults/modules/VaultImmutables.sol#15) (state variable)	Low
EthVault.constructor(address,address,address,address,address,address)._vaultsRegistry (contracts/vaults/ethereum/EthVault.sol#56) shadows: - VaultImmutables._vaultsRegistry (contracts/vaults/modules/VaultImmutables.sol#18) (state variable)	Low
EthGenesisVault.initialize(bytes).admin (contracts/vaults/ethereum/EthGenesisVault.sol#64) shadows: - VaultAdmin.admin (contracts/vaults/modules/VaultAdmin.sol#16) (state variable) - IVaultAdmin.admin() (contracts/interfaces/IVaultAdmin.sol#22) (function)	Low
EthVault.constructor(address,address,address,address,address,address)._keeper (contracts/vaults/ethereum/EthVault.sol#55) shadows: - VaultImmutables._keeper (contracts/vaults/modules/VaultImmutables.sol#15) (state variable)	Low
EthGenesisVault.constructor(address,address,address,address,address,address,address,address)._validatorsRegistry (contracts/vaults/ethereum/EthGenesisVault.sol#47) shadows: - VaultImmutables._validatorsRegistry (contracts/vaults/modules/VaultImmutables.sol#21) (state variable)	Low

Finding	Impact
EthVault.__EthVault_init(address,address,IEthVault.EthVaultInitParams).admin (contracts/vaults/ethereum/EthVault.sol#131) shadows: - VaultAdmin.admin (contracts/vaults/modules/VaultAdmin.sol#16) (state variable) - IVaultAdmin.admin() (contracts/interfaces/IVaultAdmin.sol#22) (function)	Low
EthGenesisVault.constructor(address,address,address,address,address,address,address,address)._vaultsRegistry (contracts/vaults/ethereum/EthGenesisVault.sol#46) shadows: - VaultImmutables._vaultsRegistry (contracts/vaults/modules/VaultImmutables.sol#18) (state variable)	Low
EthGenesisVault.constructor(address,address,address,address,address,address,address,address).stakedEthToken (contracts/vaults/ethereum/EthGenesisVault.sol#52) lacks a zero-check on : - _stakedEthToken = stakedEthToken (contracts/vaults/ethereum/EthGenesisVault.sol#57)	Low
Multicall.multicall(bytes[]) (contracts/base/Multicall.sol#15-31) has external calls inside a loop: (success,result) = address(this).delegatecall(data[i]) (contracts/base/Multicall.sol#18)	Low
Reentrancy in VaultOsToken.burnOsToken(uint128) (contracts/vaults/modules/VaultOsToken.sol#97-112):External calls: - assets = _osToken.burnShares(msg.sender,osTokenShares) (contracts/vaults/modules/VaultOsToken.sol#99) State variables written after the call(s): - _positions[msg.sender] = position (contracts/vaults/modules/VaultOsToken.sol#108)	Low
Reentrancy in VaultValidators.registerValidator(IKeeperValidators.ApprovalParams,bytes32[]) (contracts/vaults/modules/VaultValidators.sol#45-82):External calls: - IKeeperValidators(_keeper).approveValidators(keeperParams) (contracts/vaults/modules/VaultValidators.sol#52) State variables written after the call(s): - validatorIndex = currentIndex + 1 (contracts/vaults/modules/VaultValidators.sol#80)	Low

Finding	Impact
Reentrancy in VaultValidators.registerValidators(IKeeperValidators.ApprovalParams,uint256[],bool[],bytes32[]) (contracts/vaults/modules/VaultValidators.sol#85-128):External calls: - IKeeperValidators(_keeper).approveValidators(keeperParams) (contracts/vaults/modules/VaultValidators.sol#94) State variables written after the call(s): - validatorIndex += validatorsCount (contracts/vaults/modules/VaultValidators.sol#126)	Low
Reentrancy in VaultOsToken.mintOsToken(address,uint256,address) (contracts/vaults/modules/VaultOsToken.sol#56-94):External calls: - osTokenShares = _osToken.mintShares(receiver,assets) (contracts/vaults/modules/VaultOsToken.sol#65) State variables written after the call(s): - _positions[msg.sender] = position (contracts/vaults/modules/VaultOsToken.sol#90)	Low
Reentrancy in VaultEthStaking.updateStateAndDeposit(address,address,IKeeperRewards.HarvestParams) (contracts/vaults/modules/VaultEthStaking.sol#42-49):External calls: - updateState(harvestParams) (contracts/vaults/modules/VaultEthStaking.sol#47) - (totalAssetsDelta,unlockedMevDelta) = IKeeperRewards(_keeper).harvest(harvestParams) (contracts/vaults/modules/VaultMev.sol#44-46) - ISharedMevEscrow(_mevEscrow).harvest(unlockedMevDelta) (contracts/vaults/modules/VaultMev.sol#53) - totalAssetsDelta + int256(IOwnMevEscrow(_mevEscrow).harvest()) (contracts/vaults/modules/VaultMev.sol#59) Event emitted after the call(s): - Deposited(msg.sender,to,assets,shares,referrer) (contracts/vaults/modules/VaultEnterExit.sol#153) - deposit(receiver,referrer) (contracts/vaults/modules/VaultEthStaking.sol#48)	Low

Finding	Impact
Reentrancy in VaultEthStaking._registerMultipleValidators(bytes,uint256[]) (contracts/vaults/modules/VaultEthStaking.sol#77-116):External calls: - IEthValidatorsRegistry(_validatorsRegistry).deposit{value: validatorDeposit}(publicKey,withdrawalCreds,validator,bytes32(validator)) (contracts/vaults/modules/VaultEthStaking.sol#102-107)Event emitted after the call(s): - ValidatorRegistered(publicKey) (contracts/vaults/modules/VaultEthStaking.sol#114)	Low
Reentrancy in VaultOsToken.burnOsToken(uint128) (contracts/vaults/modules/VaultOsToken.sol#97-112):External calls: - assets = _osToken.burnShares(msg.sender,osTokenShares) (contracts/vaults/modules/VaultOsToken.sol#99) Event emitted after the call(s): - OsTokenBurned(msg.sender,assets,osTokenShares) (contracts/vaults/modules/VaultOsToken.sol#111)	Low
Reentrancy in EthGenesisVault._registerSingleValidator(bytes) (contracts/vaults/ethereum/EthGenesisVault.sol#130-135):External calls: - _pullAssets() (contracts/vaults/ethereum/EthGenesisVault.sol#133) - _poolEscrow.withdraw(address(this),escrowBalance) (contracts/vaults/ethereum/EthGenesisVault.sol#151) - super._registerSingleValidator(validator) (contracts/vaults/ethereum/EthGenesisVault.sol#134) - IEthValidatorsRegistry(_validatorsRegistry).deposit{value: _validatorDeposit()}(publicKey,_withdrawalCredentials(),validator,bytes32(validator)) (contracts/vaults/modules/VaultEthStaking.sol#66-71)External calls sending eth: - super._registerSingleValidator(validator) (contracts/vaults/ethereum/EthGenesisVault.sol#134) - IEthValidatorsRegistry(_validatorsRegistry).deposit{value: _validatorDeposit()}(publicKey,_withdrawalCredentials(),validator,bytes32(validator)) (contracts/vaults/modules/VaultEthStaking.sol#66-71) Event emitted after the call(s): - ValidatorRegistered(publicKey) (contracts/vaults/modules/VaultEthStaking.sol#73) - super._registerSingleValidator(validator) (contracts/vaults/ethereum/EthGenesisVault.sol#134)	Low

AUTOMATED TESTING

Finding	Impact
Reentrancy in VaultEthStaking._registerSingleValidator(bytes) (contracts/vaults/modules/VaultEthStaking.sol#64-74):External calls: - IEthValidatorsRegistry(_validatorsRegistry).deposit{value: _validatorDeposit()}(publicKey,_withdrawalCredentials(),validator,bytes32(validator)) (contracts/vaults/modules/VaultEthStaking.sol#66-71) Event emitted after the call(s): - ValidatorRegistered(publicKey) (contracts/vaults/modules/VaultEthStaking.sol#73)	Low
Reentrancy in VaultOsToken.redeemOsToken(uint256,address,address) (contracts/vaults/modules/VaultOsToken.sol#125-132):External calls: - receivedAssets = _redeemOsToken(owner,receiver,osTokenShares,false) (contracts/vaults/modules/VaultOsToken.sol#130) - _osToken.updateState() (contracts/vaults/modules/VaultOsToken.sol#170) - _osToken.burnShares(msg.sender,osTokenShares) (contracts/vaults/modules/VaultOsToken.sol#222) Event emitted after the call(s): - OsTokenRedeemed(msg.sender,owner,receiver,osTokenShares,receivedAssets) (contracts/vaults/modules/VaultOsToken.sol#131)	Low
Reentrancy in VaultOsToken.liquidateOsToken(uint256,address,address) (contracts/vaults/modules/VaultOsToken.sol#115-122):External calls: - receivedAssets = _redeemOsToken(owner,receiver,osTokenShares,true) (contracts/vaults/modules/VaultOsToken.sol#120) - _osToken.updateState() (contracts/vaults/modules/VaultOsToken.sol#170) - _osToken.burnShares(msg.sender,osTokenShares) (contracts/vaults/modules/VaultOsToken.sol#222) Event emitted after the call(s): - OsTokenLiquidated(msg.sender,owner,receiver,osTokenShares,receivedAssets) (contracts/vaults/modules/VaultOsToken.sol#121)	Low

Finding	Impact
Reentrancy in EthGenesisVault._registerMultipleValidators(bytes,uint256[]) (contracts/vaults/ethereum/EthGenesisVault.sol#138-144):External calls: - _pullAssets() (contracts/vaults/ethereum/EthGenesisVault.sol#142) - _poolEscrow.withdraw(address(this),escrowBalance) (contracts/vaults/ethereum/EthGenesisVault.sol#151) - super._registerMultipleValidators(validators,indexes) (contracts/vaults/ethereum/EthGenesisVault.sol#143) - IEthValidatorsRegistry(_validatorsRegistry).deposit{value: validatorDeposit}(publicKey,withdrawalCreds,validator,bytes32(validator)) (contracts/vaults/modules/VaultEthStaking.sol#102-107)External calls sending eth: - super._registerMultipleValidators(validators,indexes) (contracts/vaults/ethereum/EthGenesisVault.sol#143) - IEthValidatorsRegistry(_validatorsRegistry).deposit{value: validatorDeposit}(publicKey,withdrawalCreds,validator,bytes32(validator)) (contracts/vaults/modules/VaultEthStaking.sol#102-107)Event emitted after the call(s): - ValidatorRegistered(publicKey) (contracts/vaults/modules/VaultEthStaking.sol#114) - super._registerMultipleValidators(validators,indexes) (contracts/vaults/ethereum/EthGenesisVault.sol#143)	Low
Reentrancy in VaultOsToken.mintOsToken(address,uint256,address) (contracts/vaults/modules/VaultOsToken.sol#56-94):External calls: - osTokenShares = _osToken.mintShares(receiver,assets) (contracts/vaults/modules/VaultOsToken.sol#65) Event emitted after the call(s): - OsTokenMinted(msg.sender,receiver,assets,osTokenShares,referrer) (contracts/vaults/modules/VaultOsToken.sol#93)	Low
VaultState.canUpdateExitQueue() (contracts/vaults/modules/VaultState.sol#77-79) uses timestamp for comparisons Dangerous comparisons: - block.timestamp >= _exitQueueNextUpdate (contracts/vaults/modules/VaultState.sol#78)	Low
End of table for EthGenesisVault.sol	

Slither results for OwnMevEscrow.sol	
Finding	**Impact**

Finding	Impact
OwnMevEscrow.harvest() (contracts/vaults/ethereum/mev/OwnMevEscrow.sol#23-32) uses a dangerous strict equality: - assets == 0 (contracts/vaults/ethereum/mev/OwnMevEscrow.sol#27)	Medium
OwnMevEscrow.constructor(address)._vault (contracts/vaults/ethereum/mev/OwnMevEscrow.sol#18) lacks a zero-check on : - vault = address(_vault) (contracts/vaults/ethereum/mev/OwnMevEscrow.sol#19)	Low
End of table for OwnMevEscrow.sol	

Slither results for Keeper.sol	
Finding	**Impact**
Reentrancy in KeeperRewards.updateRewards(IKeeperRewards.RewardsUpdateParams) (contracts/keeper/KeeperRewards.sol#83-130): External calls: - _osToken.setAvgRewardPerSecond(params.avgRewardPerSecond) (contracts/keeper/KeeperRewards.sol#120) Event emitted after the call(s): - RewardsUpdated(msg.sender,params.rewardsRoot,params.avgRewardPerSecond,params.updateTimestamp,nonce,params.rewardsIpfsHash) (contracts/keeper/KeeperRewards.sol#122-129)	Low
KeeperRewards.canUpdateRewards() (contracts/keeper/KeeperRewards.sol#133-140) uses timestamp for comparisons Dangerous comparisons: - _lastRewardsTimestamp + rewardsDelay < block.timestamp (contracts/keeper/KeeperRewards.sol#138)	Low
End of table for Keeper.sol	

Slither results for Multicall.sol	
Finding	**Impact**
Multicall.multicall(bytes[]) (contracts/base/Multicall.sol#15-31) has external calls inside a loop: (success,result) = address(this).delegatecall(data[i]) (contracts/base/Multicall.sol#18)	Low
End of table for Multicall.sol	

Slither results for ERC20Upgradeable.sol	
Finding	**Impact**
ERC20Upgradeable.permit(address,address,uint256,uint256,uint8,bytes32,bytes32) (contracts/base/ERC20Upgradeable.sol#98-133) uses timestamp for comparisons Dangerous comparisons: - deadline < block.timestamp (contracts/base/ERC20Upgradeable.sol#108)	Low
End of table for ERC20Upgradeable.sol	

Slither results for ERC20.sol	
Finding	**Impact**
ERC20.permit(address,address,uint256,uint256,uint8,bytes32,bytes32) (contracts/base/ERC20.sol#93-128) uses timestamp for comparisons Dangerous comparisons: - deadline < block.timestamp (contracts/base/ERC20.sol#103)	Low
End of table for ERC20.sol	

KeeperOracles.sol

Slither did not identify any vulnerabilities in the contract.

SharedMevEscrow.sol

Slither did not identify any vulnerabilities in the contract.

OsTokenConfig.sol

Slither did not identify any vulnerabilities in the contract.

VaultsRegistry.sol

Slither did not identify any vulnerabilities in the contract.

The results of the scans of the KeeperValidators.sol, KeeperRewards.sol and KeeperOracles.sol files were included in the Keeper.sol scan results.

The results of the scan of the EthVault.sol file were included in the EthPrivVault.sol file scan results.

The results of the scan of the EthErc20Vault.sol were included in the EthPrivErc20Vault.sol file scan results.

The findings obtained as a result of the Slither scan were reviewed, and they

were not included in the report as they were determined false positives.

6.2 AUTOMATED SECURITY SCAN

Description:

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

Results:

ERC20.sol

```
Report for base/ERC20.sol
https://dashboard.mythx.io/#/console/analyses/b27f7783-7208-4ea1-abf9-39693756888f
```

Line	SWC Title	Severity	Short Description
65	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
74	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
113	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered
160	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered
165	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
180	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered

ERC20Upgradeable.sol

```
Report for contracts/base/ERC20Upgradeable.sol
https://dashboard.mythx.io/#/console/analyses/205e584d-f83f-4cfd-8386-4f35e89f8f90
```

Line	SWC Title	Severity	Short Description
63	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
72	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
90	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
118	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered

PriceOracle.sol

```
Report for osToken/PriceOracle.sol
https://dashboard.mythx.io/#/console/analyses/1fc26641-5b46-4b1b-8972-a91d7a914934
```

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

OsToken.sol

Report for contracts/osToken/OsToken.sol
https://dashboard.mythx.io/#/console/analyses/37692e54-3b4f-467f-ba26-f7e2fe3532c8

Line	SWC Title	Severity	Short Description
89	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
89	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
117	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
121	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
127	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
146	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered
151	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered
153	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered
238	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
238	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
243	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
262	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
281	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
292	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
297	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
300	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
341	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
344	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered

Keeper.sol

Report for contracts/keeper/KeeperOracles.sol
https://dashboard.mythx.io/#/console/analyses/915d59eb-29b8-4473-857a-34601f20eb49

Line	SWC Title	Severity	Short Description
37	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
50	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
50	(SWC-101) Integer Overflow and Underflow	Unknown	Compiler-rewritable "<uint> - 1" discovered
82	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
93	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
103	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered
104	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered

Report for contracts/keeper/KeeperRewards.sol
https://dashboard.mythx.io/#/console/analyses/915d59eb-29b8-4473-857a-34601f20eb49

Line	SWC Title	Severity	Short Description
118	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
138	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
149	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
178	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered
201	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
209	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered

Report for contracts/keeper/KeeperValidators.sol
https://dashboard.mythx.io/#/console/analyses/915d59eb-29b8-4473-857a-34601f20eb49

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

EthPrivErc20Vault.sol

Report for contracts/base/Multicall.sol
https://dashboard.mythx.io/#/console/analyses/54f8b1fd-cc3e-4af5-b4c3-312f7fec77c2

Line	SWC Title	Severity	Short Description
15	(SWC-118) Incorrect Constructor Name	Medium	Potential incorrect constructor name "multicall".
17	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered
18	(SWC-110) Assert Violation	Unknown	Out of bounds array access
29	(SWC-110) Assert Violation	Unknown	Out of bounds array access

Report for contracts/libraries/ExitQueue.sol
https://dashboard.mythx.io/#/console/analyses/54f8b1fd-cc3e-4af5-b4c3-312f7fec77c2

Line	SWC Title	Severity	Short Description
42	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
42	(SWC-101) Integer Overflow and Underflow	Unknown	Compiler-rewritable "<uint> - 1" discovered
63	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
94	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
94	(SWC-101) Integer Overflow and Underflow	Unknown	Compiler-rewritable "<uint> - 1" discovered
111	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
120	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
122	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
125	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
126	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
128	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered
144	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
158	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered

Report for contracts/vaults/modules/VaultEnterExit.sol
https://dashboard.mythx.io/#/console/analyses/54f8b1fd-cc3e-4af5-b4c3-312f7fec77c2

Line	SWC Title	Severity	Short Description
45	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered
69	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
75	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered
79	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
109	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
113	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
118	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered
142	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered

Report for contracts/vaults/modules/VaultEthStaking.sol
https://dashboard.mythx.io/#/console/analyses/54f8b1fd-cc3e-4af5-b4c3-312f7fec77c2

Line	SWC Title	Severity	Short Description
94	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
97	(SWC-110) Assert Violation	Unknown	Out of bounds array access
111	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered
112	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered

Report for contracts/vaults/modules/VaultMev.sol
https://dashboard.mythx.io/#/console/analyses/54f8b1fd-cc3e-4af5-b4c3-312f7fec77c2

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

Report for contracts/vaults/modules/VaultOsToken.sol
https://dashboard.mythx.io/#/console/analyses/54f8b1fd-cc3e-4af5-b4c3-312f7fec77c2

Line	SWC Title	Severity	Short Description
82	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
113	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered
216	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
224	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
245	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered
252	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered
269	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
289	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
290	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered

Report for contracts/vaults/modules/VaultState.sol
https://dashboard.mythx.io/#/console/analyses/54f8b1fd-cc3e-4af5-b4c3-312f7fec77c2

Line	SWC Title	Severity	Short Description
71	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
72	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
102	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered
111	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
132	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
153	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
162	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
163	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
167	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
175	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered
176	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered
186	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
192	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
203	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered
208	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered

Report for contracts/vaults/modules/VaultToken.sol
https://dashboard.mythx.io/#/console/analyses/54f8b1fd-cc3e-4af5-b4c3-312f7fec77c2

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

Report for contracts/vaults/modules/VaultValidators.sol
https://dashboard.mythx.io/#/console/analyses/54f8b1fd-cc3e-4af5-b4c3-312f7fec77c2

Line	SWC Title	Severity	Short Description
80	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
98	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
105	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
126	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered

Report for contracts/vaults/modules/VaultVersion.sol
https://dashboard.mythx.io/#/console/analyses/54f8b1fd-cc3e-4af5-b4c3-312f7fec77c2

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

EthGenesisVault.sol

Report for contracts/vaults/ethereum/EthGenesisVault.sol
https://dashboard.mythx.io/#/console/analyses/9bdb8804-bbae-469a-bfd1-391e514c0770

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

VaultsRegistry.sol

MythX did not identify any vulnerabilities in the file.

EthVaultFactory.sol

MythX did not identify any vulnerabilities in the file.

OsTokenConfig.sol

MythX did not identify any vulnerabilities in the file.

The results of the scans of the EthVault.sol, EthPrivVault.sol, EthErc20Vault , and EthGenesisVault.sol files were included in the results of the EthPrivErc20Vault.sol file scan results.

The results of the scans of the KeeperValidators.sol, KeeperRewards.sol, KeeperOracles.sol files were included in the Keeper.sol file scan results.

The findings obtained as a result of the MythX scan were examined, and they were not included in the report, as they were determined false positives.

THANK YOU FOR CHOOSING

// HALBORN