// HALBORN

Stader Labs - Hedera Stader Protocol v2

Smart Contract Security Audit

Prepared by: **Halborn**

Date of Engagement: **June 9th, 2022 - June 10th, 2022**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	06/09/2022	Roberto Reigada
0.2	Document Updates	06/10/2022	Roberto Reigada
0.3	Draft Review	06/10/2022	Gabi Urrutia
1.0	Remediation Plan	06/10/2022	Roberto Reigada
1.1	Remediation Plan Review	06/10/2022	Gabi Urrutia

CONTACTS

CONTACT	COMPANY	EMAIL
Rob Behnke	Halborn	Rob.Behnke@halborn.com
Steven Walbroehl	Halborn	Steven.Walbroehl@halborn.com
Gabi Urrutia	Halborn	Gabi.Urrutia@halborn.com
Roberto Reigada	Halborn	Roberto.Reigada@halborn.com

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

Stader Labs engaged Halborn to conduct a security audit on their smart contracts beginning on June 9th, 2022 and ending on June 10th, 2022. The security assessment was scoped to the smart contracts provided in the GitHub repositories stader-labs/hedera-stader-protocol-v1

1.2 AUDIT SUMMARY

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

The purpose of this audit is to:

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

In summary, Halborn identified some security risks that were addressed by the Stader Labs team.

1.3 TEST APPROACH & METHODOLOGY

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

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

RISK METHODOLOGY:

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

RISK SCALE - LIKELIHOOD

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

RISK SCALE - IMPACT

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

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

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

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

1.4 SCOPE

IN-SCOPE:

The security assessment was scoped to the following smart contracts:

- Rewards.sol
- Staking.sol
- Undelegation.sol

Initial Commit ID:

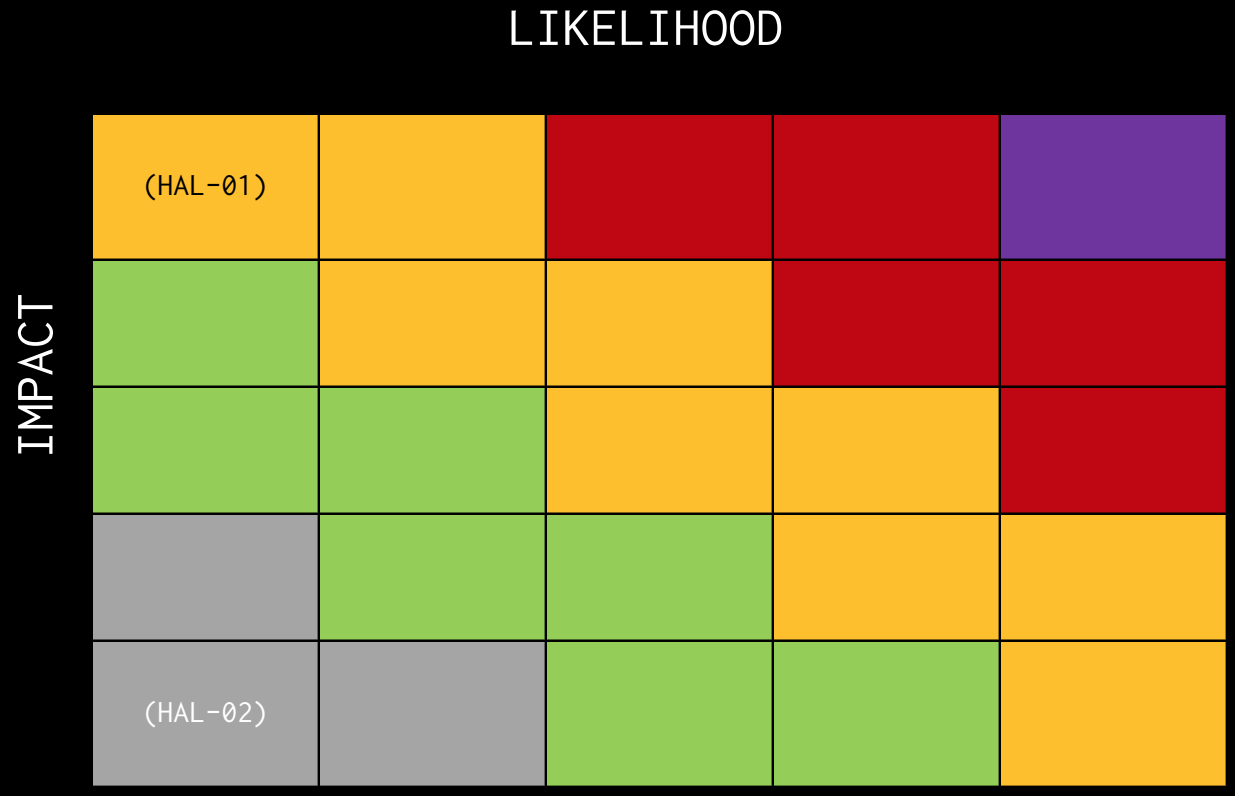
- 2ec1645e8acfc302be8f064c3503e708dc0f9247

Fixed Commit ID:

- 734df45ea5db9199c9e0d3bdad76aac1fe5a6b45

2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	0	1	0	1



EXECUTIVE OVERVIEW

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
HAL01 - USE OF TRANSFER INSTEAD OF CALL TO SEND NATIVE ASSETS	Medium	SOLVED - 06/10/2022
HAL02 - STAKING.GETEXCHANGERATE CAN BE FORCED TO RETURN ZERO	Informational	SOLVED - 06/10/2022

FINDINGS & TECH DETAILS

3.1 (HAL-01) USE OF TRANSFER INSTEAD OF CALL TO SEND NATIVE ASSETS - MEDIUM

Description:

The use of `transfer()` in multiple contracts may have unintended outcomes on the native asset being sent to the receiver. The transaction will fail when:

- The receiver address is a smart contract that does not implement a payable function.
- The receiver address is a smart contract that implements a payable fallback function which uses more than 2300 gas units.
- The receiver address is a smart contract that implements a payable fallback function that needs less than 2300 gas units but is called through proxy, raising the call's gas usage above 2300.
- Additionally, using higher than 2300 gas might be mandatory for some multisig wallets.

Code Location:

Rewards.sol
- Line 100: `payable(stakerAddress).transfer(epochRewards - daoFees);`
- Line 107: `payable(daoAddress).transfer(daoFees);`

Timelock.sol
- Line 108: `payable(to).transfer(amount);`

Undelegation.sol
- Line 72: `payable(msg.sender).transfer(amount);`

Risk Level:

Likelihood - 1

Impact - 5

Recommendation:

To transfer a native asset is recommended to either use:

- <receiver>.call.value(amount)
- OpenZeppelin Address.sendValue()

In both cases, care must be taken not to create reentrancy vulnerabilities. Consider using ReentrancyGuard or the Check-Effects-Interactions pattern.

Remediation Plan:

SOLVED: The Stader Labs team now uses OpenZeppelin Address.sendValue() to transfer the native assets.

3.2 (HAL-02) STAKING.GETEXCHANGERATE CAN BE FORCED TO RETURN ZERO - INFORMATIONAL

Description:

The Staking contract contains the view function getExchangeRate() which returns the tinybarValue for 1 Hbarx:

Listing 1: Staking.sol (Line 195)

```
189 /// @notice Calculation of exchange rate
190 /// @return exchangeRate i.e tinybarValue value for 1 hbarx
191 function getExchangeRate() external view returns (uint256) {
192     ///@dev 1HBar = 100_000_000 tinybar
193     uint256 exchangeRate = 1 * decimals;
194     if ((address(this).balance) != 0 && totalSupply != 0) {
195         exchangeRate = ((address(this).balance) * decimals) /
 ↳ totalSupply;
196     }
197     return exchangeRate;
198 }
```

In the event that the denominator (totalSupply) is greater than the totalSupply of Hbarx x 100_000_000 exchangeRate will be equal to zero.

We cannot estimate the exact impact of this, as it is a view function not used in any of the smart contracts, but it can definitely cause some problems in future integrations.

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

It is recommended to address this edge case where the exchangeRate returned is zero.

Remediation Plan:

SOLVED: The Stader Labs team claims that totalSupply will never be greater than the balance, as they add extra tokens to the balance, so the Hbar/Hbarx ratio is always > 1, plus the difference between them will never be more than 10**8 (decimals). Moreover, there is no minting done outside the contract that can change this ratio.

AUTOMATED TESTING

4.1 STATIC ANALYSIS REPORT

Description:

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

Slither results:

Rewards.sol

```
Rewards.setDaoFeesPercentage(uint256) (contracts/Rewards.sol#146-155) should emit an event for:
        - daoFeesPercentage = _daoFeesPercentage (contracts/Rewards.sol#154)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic

Reentrancy in Rewards.distributeStakingRewards() (contracts/Rewards.sol#81-111):
        External calls:
        - Address.sendValue(address(stakerAddress),epochRewards - daoFees) (contracts/Rewards.sol#102)
        Event emitted after the call(s):
        - DistributedRewards(stakerAddress,epochRewards - daoFees,currentTimestamp) (contracts/Rewards.sol#103-107)
Reentrancy in Rewards.distributeStakingRewards() (contracts/Rewards.sol#81-111):
        External calls:
        - Address.sendValue(address(stakerAddress),epochRewards - daoFees) (contracts/Rewards.sol#102)
        - Address.sendValue(address(daoAddress),daoFees) (contracts/Rewards.sol#109)
        Event emitted after the call(s):
        - DaoTransfer(daoAddress,daoFees,currentTimestamp) (contracts/Rewards.sol#110)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Rewards.distributeStakingRewards() (contracts/Rewards.sol#81-111) uses timestamp for comparisons
        Dangerous comparisons:
        - epochRewards > totalRewards (contracts/Rewards.sol#97)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp

Address.isContract(address) (node_modules/@openzeppelin/contracts/utils/Address.sol#27-37) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/Address.sol#33-35)
Address.verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#196-216) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/Address.sol#208-211)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['^0.8.0', '^0.8.9']
        - ^0.8.0 (node_modules/@openzeppelin/contracts/security/Pausable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/security/ReentrancyGuard.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Address.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol#4)
        - ^0.8.0 (contracts/Ownable.sol#4)
        - ^0.8.9 (contracts/Rewards.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#80-82) is never used and should be removed
Address.functionCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#90-96) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#109-115) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#123-134) is never used and should be removed
Address.functionDelegateCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#169-171) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#179-188) is never used and should be removed
Address.functionStaticCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#142-144) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#152-161) is never used and should be removed
Address.isContract(address) (node_modules/@openzeppelin/contracts/utils/Address.sol#27-37) is never used and should be removed
Address.verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#196-216) is never used and should be removed
Context._msgData() (node_modules/@openzeppelin/contracts/utils/Context.sol#21-23) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/security/Pausable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/security/ReentrancyGuard.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Address.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol#4) allows old versions
Pragma version^0.8.0 (contracts/Ownable.sol#4) allows old versions
Pragma version^0.8.9 (contracts/Rewards.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.13 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#55-60):
        - (success) = recipient.call{value: amount}() (node_modules/@openzeppelin/contracts/utils/Address.sol#58)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#123-134):
        - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#132)
Low level call in Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#152-161):
        - (success,returndata) = target.staticcall(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#159)
Low level call in Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#179-188):
        - (success,returndata) = target.delegatecall(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#186)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter Rewards.setEmissionRate(uint256)._emissionRate (contracts/Rewards.sol#119) is not in mixedCase
Parameter Rewards.setStakerAddress(address)._stakerAddress (contracts/Rewards.sol#126) is not in mixedCase
Parameter Rewards.setDaoAddress(address)._daoAddress (contracts/Rewards.sol#136) is not in mixedCase
Parameter Rewards.setDaoFeesPercentage(uint256)._daoFeesPercentage (contracts/Rewards.sol#146) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

Staking.sol

```
Staking.getExchangeRate() (contracts/Staking.sol#191-200) uses a dangerous strict equality:
        - totalSupply_ == 0 || address(this).balance == 0 (contracts/Staking.sol#196)
Timelock.withdraw(uint256) (contracts/Timelock.sol#100-114) uses a dangerous strict equality:
        - address(this).balance == 0 (contracts/Timelock.sol#101)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities

Reentrancy in Staking.stake() (contracts/Staking.sol#87-138):
        External calls:
        - HederaTokenService.associateToken(msg.sender,hbarxAddress) (contracts/Staking.sol#116)
                - (success,result) = precompileAddress.call(abi.encodeWithSelector(IHederaTokenService.associateToken.selector,account,token)) (contracts/HederaTokenService.sol#128-134)
        - (mintTokenResponse,newTotalSupply) = HederaTokenService.mintToken(hbarxAddress,hbarxMint,new bytes[](0)) (contracts/Staking.sol#118-120)
                - (success,result) = precompileAddress.call(abi.encodeWithSelector(IHederaTokenService.mintToken.selector,token,amount,metadata)) (contracts/HederaTokenService.sol#51-58)
        State variables written after the call(s):
        - totalSupply = uint256(newTotalSupply) (contracts/Staking.sol#121)
Reentrancy in Staking.unStake(uint256) (contracts/Staking.sol#142-183):
        External calls:
        - transferTokenResponse = HederaTokenService.transferToken(hbarxAddress,msg.sender,address(this),hbarxToBurn.toInt256().toInt64()) (contracts/Staking.sol#154-159)
                - (success,result) = precompileAddress.call(abi.encodeWithSelector(IHederaTokenService.transferToken.selector,token,sender,receiver,amount)) (contracts/HederaTokenService.sol#254-262)
        - (burnTokenResponse,newTotalSupply) = HederaTokenService.burnToken(hbarxAddress,hbarxToBurn.toUint64(),new int64[](0)) (contracts/Staking.sol#166-167)
                - (success,result) = precompileAddress.call(abi.encodeWithSelector(IHederaTokenService.burnToken.selector,token,amount,serialNumbers)) (contracts/HederaTokenService.sol#78-85)
        State variables written after the call(s):
        - totalSupply = uint256(newTotalSupply) (contracts/Staking.sol#168)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1

Timelock.setTimeLockOwner(address) (contracts/Timelock.sol#136-142) should emit an event for:
        - timelockOwner = _timelockOwner (contracts/Timelock.sol#141)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control

Reentrancy in Staking.stake() (contracts/Staking.sol#87-138):
        External calls:
        - HederaTokenService.associateToken(msg.sender,hbarxAddress) (contracts/Staking.sol#116)
                - (success,result) = precompileAddress.call(abi.encodeWithSelector(IHederaTokenService.associateToken.selector,account,token)) (contracts/HederaTokenService.sol#128-134)
        - (mintTokenResponse,newTotalSupply) = HederaTokenService.mintToken(hbarxAddress,hbarxMint,new bytes[](0)) (contracts/Staking.sol#118-120)
                - (success,result) = precompileAddress.call(abi.encodeWithSelector(IHederaTokenService.mintToken.selector,token,amount,metadata)) (contracts/HederaTokenService.sol#51-58)
        - transferTokenResponse = HederaTokenService.transferToken(hbarxAddress,address(this),msg.sender,hbarxToSend) (contracts/Staking.sol#126-131)
                - (success,result) = precompileAddress.call(abi.encodeWithSelector(IHederaTokenService.transferToken.selector,token,sender,receiver,amount)) (contracts/HederaTokenService.sol#254-262)
        Event emitted after the call(s):
        - Staked(msg.sender,hbarReceived,hbarxToSend) (contracts/Staking.sol#137)
Reentrancy in Staking.unStake(uint256) (contracts/Staking.sol#142-183):
        External calls:
        - transferTokenResponse = HederaTokenService.transferToken(hbarxAddress,msg.sender,address(this),hbarxToBurn.toInt256().toInt64()) (contracts/Staking.sol#154-159)
                - (success,result) = precompileAddress.call(abi.encodeWithSelector(IHederaTokenService.transferToken.selector,token,sender,receiver,amount)) (contracts/HederaTokenService.sol#254-262)
        - (burnTokenResponse,newTotalSupply) = HederaTokenService.burnToken(hbarxAddress,hbarxToBurn.toUint64(),new int64[](0)) (contracts/Staking.sol#166-167)
                - (success,result) = precompileAddress.call(abi.encodeWithSelector(IHederaTokenService.burnToken.selector,token,amount,serialNumbers)) (contracts/HederaTokenService.sol#78-85)
        - (success) = address(undelegationContractAddress).call{value: hbarToTransfer}(abi.encodeWithSignature(undelegate(address),msg.sender)) (contracts/Staking.sol#174-176)
        External calls sending eth:
        - (success) = address(undelegationContractAddress).call{value: hbarToTransfer}(abi.encodeWithSignature(undelegate(address),msg.sender)) (contracts/Staking.sol#174-176)
        Event emitted after the call(s):
        - UnStaked(msg.sender,hbarToTransfer,hbarxToBurn) (contracts/Staking.sol#180)
Reentrancy in Timelock.withdraw(uint256) (contracts/Timelock.sol#100-114):
        External calls:
        - Address.sendValue(address(to),amount) (contracts/Timelock.sol#111)
        Event emitted after the call(s):
        - Transferred(index,amount,to) (contracts/Timelock.sol#112)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Timelock.withdraw(uint256) (contracts/Timelock.sol#100-114) uses timestamp for comparisons
        Dangerous comparisons:
        - index >= withdrawQueue.length (contracts/Timelock.sol#102)
        - withdrawData.timestamp + lockedPeriod >= block.timestamp (contracts/Timelock.sol#106)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp

Address.isContract(address) (node_modules/@openzeppelin/contracts/utils/Address.sol#27-37) uses assembly

        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/Address.sol#33-35)
Address.verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#196-216) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/Address.sol#208-211)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['^0.4.8<0.9.0', '>=0.5.0<0.9.0', '^0.8.0', '^0.8.9']
        - ^0.8.0 (node_modules/@openzeppelin/contracts/security/Pausable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/security/ReentrancyGuard.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Address.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#4)
        - >=0.4.9<0.9.0 (contracts/HederaResponseCodes.sol#2)
        - >=0.5.0<0.9.0 (contracts/HederaTokenService.sol#2)
        - ABIEncoderV2 (contracts/HederaTokenService.sol#3)
        - >=0.4.9<0.9.0 (contracts/IHederaTokenService.sol#2)
        - ABIEncoderV2 (contracts/IHederaTokenService.sol#3)
        - ^0.8.0 (contracts/Ownable.sol#4)
        - ^0.8.9 (contracts/Staking.sol#2)
        - ^0.8.9 (contracts/Timelock.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#80-82) is never used and should be removed
Address.functionCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#90-96) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#109-115) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#123-134) is never used and should be removed
Address.functionDelegateCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#169-171) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#179-188) is never used and should be removed
Address.functionStaticCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#142-144) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#152-161) is never used and should be removed
Address.isContract(address) (node_modules/@openzeppelin/contracts/utils/Address.sol#27-37) is never used and should be removed
Address.verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#196-216) is never used and should be removed
Context._msgData() (node_modules/@openzeppelin/contracts/utils/Context.sol#21-23) is never used and should be removed
HederaTokenService.associateTokens(address,address[]) (contracts/HederaTokenService.sol#108-122) is never used and should be removed
HederaTokenService.cryptoTransfer(IHederaTokenService.TokenTransferList[]) (contracts/HederaTokenService.sol#14-26) is never used and should be removed
HederaTokenService.dissociateToken(address,address) (contracts/HederaTokenService.sol#174-188) is never used and should be removed
HederaTokenService.dissociateTokens(address,address[]) (contracts/HederaTokenService.sol#158-172) is never used and should be removed
HederaTokenService.transferNFT(address,address,address,int64) (contracts/HederaTokenService.sol#275-293) is never used and should be removed
HederaTokenService.transferNFTs(address,address[],address[],int64[]) (contracts/HederaTokenService.sol#221-239) is never used and should be removed
HederaTokenService.transferTokens(address,address[],int64[]) (contracts/HederaTokenService.sol#198-214) is never used and should be removed
SafeCast.toInt128(int256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#152-155) is never used and should be removed
SafeCast.toInt16(int256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#206-209) is never used and should be removed
SafeCast.toInt32(int256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#188-191) is never used and should be removed
SafeCast.toInt8(int256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#224-227) is never used and should be removed
SafeCast.toUint128(uint256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#47-50) is never used and should be removed
SafeCast.toUint16(uint256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#107-110) is never used and should be removed
SafeCast.toUint224(uint256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#32-35) is never used and should be removed
SafeCast.toUint256(int256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#134-137) is never used and should be removed
SafeCast.toUint32(uint256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#92-95) is never used and should be removed
SafeCast.toUint8(uint256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#122-125) is never used and should be removed
SafeCast.toUint96(uint256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#62-65) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/security/Pausable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/security/ReentrancyGuard.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Address.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#4) allows old versions
Pragma version>=0.4.9<0.9.0 (contracts/HederaResponseCodes.sol#2) is too complex
Pragma version>=0.5.0<0.9.0 (contracts/HederaTokenService.sol#2) is too complex
Pragma version>=0.4.9<0.9.0 (contracts/IHederaTokenService.sol#2) is too complex
Pragma version^0.8.0 (contracts/Ownable.sol#4) allows old versions
Pragma version^0.8.9 (contracts/Staking.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version^0.8.9 (contracts/Timelock.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.13 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#55-60):
        - (success) = recipient.call{value: amount}() (node_modules/@openzeppelin/contracts/utils/Address.sol#58)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#123-134):
        - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#132)
Low level call in Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#152-161):
        - (success,returndata) = target.staticcall(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#159)
Low level call in Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#179-188):
        - (success,returndata) = target.delegatecall(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#186)
Low level call in HederaTokenService.cryptoTransfer(IHederaTokenService.TokenTransferList[]) (contracts/HederaTokenService.sol#14-26):
        - (success,result) = precompileAddress.call(abi.encodeWithSelector(IHederaTokenService.cryptoTransfer.selector,tokenTransfers)) (contracts/HederaTokenService.sol#17-22)
Low level call in HederaTokenService.mintToken(address,uint64,bytes[]) (contracts/HederaTokenService.sol#39-62):
        - (success,result) = precompileAddress.call(abi.encodeWithSelector(IHederaTokenService.mintToken.selector,token,amount,metadata)) (contracts/HederaTokenService.sol#51-58)
Low level call in HederaTokenService.burnToken(address,uint64,int64[]) (contracts/HederaTokenService.sol#73-93):
        - (success,result) = precompileAddress.call(abi.encodeWithSelector(IHederaTokenService.burnToken.selector,token,amount,serialNumbers)) (contracts/HederaTokenService.sol#78-85)
Low level call in HederaTokenService.associateTokens(address,address[]) (contracts/HederaTokenService.sol#108-122):
        - (success,result) = precompileAddress.call(abi.encodeWithSelector(IHederaTokenService.associateTokens.selector,account,tokens)) (contracts/HederaTokenService.sol#112-118)
Low level call in HederaTokenService.associateToken(address,address) (contracts/HederaTokenService.sol#124-138):
        - (success,result) = precompileAddress.call(abi.encodeWithSelector(IHederaTokenService.associateToken.selector,account,token)) (contracts/HederaTokenService.sol#128-134)
Low level call in HederaTokenService.dissociateTokens(address,address[]) (contracts/HederaTokenService.sol#158-172):
        - (success,result) = precompileAddress.call(abi.encodeWithSelector(IHederaTokenService.dissociateTokens.selector,account,tokens)) (contracts/HederaTokenService.sol#162-168)
Low level call in HederaTokenService.dissociateToken(address,address) (contracts/HederaTokenService.sol#174-188):
        - (success,result) = precompileAddress.call(abi.encodeWithSelector(IHederaTokenService.dissociateToken.selector,account,token)) (contracts/HederaTokenService.sol#178-184)
Low level call in HederaTokenService.transferTokens(address,address[],address[],int64[]) (contracts/HederaTokenService.sol#198-214):
        - (success,result) = precompileAddress.call(abi.encodeWithSelector(IHederaTokenService.transferTokens.selector,token,accountIds,amounts)) (contracts/HederaTokenService.sol#203-210)
Low level call in HederaTokenService.transferNFTs(address,address[],address[],int64[]) (contracts/HederaTokenService.sol#221-239):
        - (success,result) = precompileAddress.call(abi.encodeWithSelector(IHederaTokenService.transferNFTs.selector,token,sender,receiver,serialNumber)) (contracts/HederaTokenService.sol#227-235)
Low level call in HederaTokenService.transferToken(address,address,address,int64) (contracts/HederaTokenService.sol#248-266):
        - (success,result) = precompileAddress.call(abi.encodeWithSelector(IHederaTokenService.transferToken.selector,token,sender,receiver,amount)) (contracts/HederaTokenService.sol#254-262)
Low level call in HederaTokenService.transferNFT(address,address,address,int64) (contracts/HederaTokenService.sol#275-293):
        - (success,result) = precompileAddress.call(abi.encodeWithSelector(IHederaTokenService.transferNFT.selector,token,sender,receiver,serialNumber)) (contracts/HederaTokenService.sol#281-289)
Low level call in Staking.unStake(uint256) (contracts/Staking.sol#142-183):
        - (success) = address(undelegationContractAddress).call{value: hbarToTransfer}(abi.encodeWithSignature(undelegate(address),msg.sender)) (contracts/Staking.sol#174-176)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Constant HederaTokenService.precompileAddress (contracts/HederaTokenService.sol#9) is not in UPPER_CASE_WITH_UNDERSCORES
Parameter Staking.updateMinDeposit(uint256)._newMinDeposit (contracts/Staking.sol#218) is not in mixedCase
Parameter Staking.updateMaxDeposit(uint256)._newMaxDeposit (contracts/Staking.sol#224) is not in mixedCase
Parameter Staking.setRewardsContractAddress(address)._rewardsContractAddress (contracts/Staking.sol#230) is not in mixedCase
Parameter Staking.setUndelegationContractAddress(address)._undelegationContractAddress (contracts/Staking.sol#241) is not in mixedCase
Parameter Timelock.setTimeLockOwner(address)._timelockOwner (contracts/Timelock.sol#136) is not in mixedCase
Parameter Timelock.setlockedPeriod(uint256)._lockedPeriod (contracts/Timelock.sol#146) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

```
HederaResponseCodes.OK (contracts/HederaResponseCodes.sol#6) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_TRANSACTION (contracts/HederaResponseCodes.sol#7) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.PAYER_ACCOUNT_NOT_FOUND (contracts/HederaResponseCodes.sol#8) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_NODE_ACCOUNT (contracts/HederaResponseCodes.sol#9) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TRANSACTION_EXPIRED (contracts/HederaResponseCodes.sol#10) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_TRANSACTION_START (contracts/HederaResponseCodes.sol#11) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_TRANSACTION_DURATION (contracts/HederaResponseCodes.sol#12) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_SIGNATURE (contracts/HederaResponseCodes.sol#13) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.MEMO_TOO_LONG (contracts/HederaResponseCodes.sol#14) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INSUFFICIENT_TX_FEE (contracts/HederaResponseCodes.sol#15) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INSUFFICIENT_PAYER_BALANCE (contracts/HederaResponseCodes.sol#16) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.DUPLICATE_TRANSACTION (contracts/HederaResponseCodes.sol#17) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.BUSY (contracts/HederaResponseCodes.sol#18) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.NOT_SUPPORTED (contracts/HederaResponseCodes.sol#19) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_FILE_ID (contracts/HederaResponseCodes.sol#21) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_ACCOUNT_ID (contracts/HederaResponseCodes.sol#22) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_CONTRACT_ID (contracts/HederaResponseCodes.sol#23) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_TRANSACTION_ID (contracts/HederaResponseCodes.sol#24) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.RECEIPT_NOT_FOUND (contracts/HederaResponseCodes.sol#25) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.RECORD_NOT_FOUND (contracts/HederaResponseCodes.sol#26) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_SOLIDITY_ID (contracts/HederaResponseCodes.sol#27) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.UNKNOWN (contracts/HederaResponseCodes.sol#29) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.SUCCESS (contracts/HederaResponseCodes.sol#30) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.FAIL_INVALID (contracts/HederaResponseCodes.sol#31) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.FAIL_FEE (contracts/HederaResponseCodes.sol#32) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.FAIL_BALANCE (contracts/HederaResponseCodes.sol#33) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.KEY_REQUIRED (contracts/HederaResponseCodes.sol#35) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.BAD_ENCODING (contracts/HederaResponseCodes.sol#36) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INSUFFICIENT_ACCOUNT_BALANCE (contracts/HederaResponseCodes.sol#37) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_SOLIDITY_ADDRESS (contracts/HederaResponseCodes.sol#38) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INSUFFICIENT_GAS (contracts/HederaResponseCodes.sol#40) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.CONTRACT_SIZE_LIMIT_EXCEEDED (contracts/HederaResponseCodes.sol#41) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.LOCAL_CALL_MODIFICATION_EXCEPTION (contracts/HederaResponseCodes.sol#42) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.CONTRACT_REVERT_EXECUTED (contracts/HederaResponseCodes.sol#43) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.CONTRACT_EXECUTION_EXCEPTION (contracts/HederaResponseCodes.sol#44) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_RECEIVING_NODE_ACCOUNT (contracts/HederaResponseCodes.sol#45) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.MISSING_QUERY_HEADER (contracts/HederaResponseCodes.sol#46) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.ACCOUNT_UPDATE_FAILED (contracts/HederaResponseCodes.sol#48) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_KEY_ENCODING (contracts/HederaResponseCodes.sol#49) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.NULL_SOLIDITY_ADDRESS (contracts/HederaResponseCodes.sol#50) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.CONTRACT_UPDATE_FAILED (contracts/HederaResponseCodes.sol#52) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_QUERY_HEADER (contracts/HederaResponseCodes.sol#53) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_FEE_SUBMITTED (contracts/HederaResponseCodes.sol#55) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_PAYER_SIGNATURE (contracts/HederaResponseCodes.sol#56) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.KEY_NOT_PROVIDED (contracts/HederaResponseCodes.sol#58) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_EXPIRATION_TIME (contracts/HederaResponseCodes.sol#59) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.NO_WACL_KEY (contracts/HederaResponseCodes.sol#60) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.FILE_CONTENT_EMPTY (contracts/HederaResponseCodes.sol#61) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_ACCOUNT_AMOUNTS (contracts/HederaResponseCodes.sol#62) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.EMPTY_TRANSACTION_BODY (contracts/HederaResponseCodes.sol#63) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_TRANSACTION_BODY (contracts/HederaResponseCodes.sol#64) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_SIGNATURE_TYPE_MISMATCHING_KEY (contracts/HederaResponseCodes.sol#66) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_SIGNATURE_COUNT_MISMATCHING_KEY (contracts/HederaResponseCodes.sol#67) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.EMPTY_LIVE_HASH_BODY (contracts/HederaResponseCodes.sol#69) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.EMPTY_LIVE_HASH (contracts/HederaResponseCodes.sol#70) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.EMPTY_LIVE_HASH_KEYS (contracts/HederaResponseCodes.sol#71) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_LIVE_HASH_SIZE (contracts/HederaResponseCodes.sol#72) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.EMPTY_QUERY_BODY (contracts/HederaResponseCodes.sol#74) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.EMPTY_LIVE_HASH_QUERY (contracts/HederaResponseCodes.sol#75) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.LIVE_HASH_NOT_FOUND (contracts/HederaResponseCodes.sol#76) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.ACCOUNT_ID_DOES_NOT_EXIST (contracts/HederaResponseCodes.sol#77) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.LIVE_HASH_ALREADY_EXISTS (contracts/HederaResponseCodes.sol#78) is never used in Staking (contracts/Staking.sol#18-278)

HederaResponseCodes.INVALID_FILE_WACL (contracts/HederaResponseCodes.sol#80) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.SERIALIZATION_FAILED (contracts/HederaResponseCodes.sol#81) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TRANSACTION_OVERSIZE (contracts/HederaResponseCodes.sol#82) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TRANSACTION_TOO_MANY_LAYERS (contracts/HederaResponseCodes.sol#83) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.CONTRACT_DELETED (contracts/HederaResponseCodes.sol#84) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.PLATFORM_NOT_ACTIVE (contracts/HederaResponseCodes.sol#86) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.KEY_PREFIX_MISMATCH (contracts/HederaResponseCodes.sol#87) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.PLATFORM_TRANSACTION_NOT_CREATED (contracts/HederaResponseCodes.sol#88) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_RENEWAL_PERIOD (contracts/HederaResponseCodes.sol#89) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_PAYER_ACCOUNT_ID (contracts/HederaResponseCodes.sol#90) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.ACCOUNT_DELETED (contracts/HederaResponseCodes.sol#91) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.FILE_DELETED (contracts/HederaResponseCodes.sol#92) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.ACCOUNT_REPEATED_IN_ACCOUNT_AMOUNTS (contracts/HederaResponseCodes.sol#93) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.SETTING_NEGATIVE_ACCOUNT_BALANCE (contracts/HederaResponseCodes.sol#94) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.OBTAINER_REQUIRED (contracts/HederaResponseCodes.sol#95) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.OBTAINER_SAME_CONTRACT_ID (contracts/HederaResponseCodes.sol#96) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.OBTAINER_DOES_NOT_EXIST (contracts/HederaResponseCodes.sol#97) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.MODIFYING_IMMUTABLE_CONTRACT (contracts/HederaResponseCodes.sol#98) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.FILE_SYSTEM_EXCEPTION (contracts/HederaResponseCodes.sol#99) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.AUTORENEW_DURATION_NOT_IN_RANGE (contracts/HederaResponseCodes.sol#100) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.ERROR_DECODING_BYTESTRING (contracts/HederaResponseCodes.sol#101) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.CONTRACT_FILE_EMPTY (contracts/HederaResponseCodes.sol#102) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.CONTRACT_BYTECODE_EMPTY (contracts/HederaResponseCodes.sol#103) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_INITIAL_BALANCE (contracts/HederaResponseCodes.sol#104) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_RECEIVE_RECORD_THRESHOLD (contracts/HederaResponseCodes.sol#105) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_SEND_RECORD_THRESHOLD (contracts/HederaResponseCodes.sol#106) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.ACCOUNT_IS_NOT_GENESIS_ACCOUNT (contracts/HederaResponseCodes.sol#107) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.PAYER_ACCOUNT_UNAUTHORIZED (contracts/HederaResponseCodes.sol#108) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_FREEZE_TRANSACTION_BODY (contracts/HederaResponseCodes.sol#109) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.FREEZE_TRANSACTION_BODY_NOT_FOUND (contracts/HederaResponseCodes.sol#110) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TRANSFER_LIST_SIZE_LIMIT_EXCEEDED (contracts/HederaResponseCodes.sol#111) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.RESULT_SIZE_LIMIT_EXCEEDED (contracts/HederaResponseCodes.sol#112) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.NOT_SPECIAL_ACCOUNT (contracts/HederaResponseCodes.sol#113) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.CONTRACT_NEGATIVE_GAS (contracts/HederaResponseCodes.sol#114) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.CONTRACT_NEGATIVE_VALUE (contracts/HederaResponseCodes.sol#115) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_FEE_FILE (contracts/HederaResponseCodes.sol#116) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_EXCHANGE_RATE_FILE (contracts/HederaResponseCodes.sol#117) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INSUFFICIENT_LOCAL_CALL_GAS (contracts/HederaResponseCodes.sol#118) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.ENTITY_NOT_ALLOWED_TO_DELETE (contracts/HederaResponseCodes.sol#119) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.AUTHORIZATION_FAILED (contracts/HederaResponseCodes.sol#120) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.FILE_UPLOADED_PROTO_INVALID (contracts/HederaResponseCodes.sol#121) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.FILE_UPLOADED_PROTO_NOT_SAVED_TO_DISK (contracts/HederaResponseCodes.sol#122) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.FEE_SCHEDULE_FILE_PART_UPLOADED (contracts/HederaResponseCodes.sol#123) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.EXCHANGE_RATE_CHANGE_LIMIT_EXCEEDED (contracts/HederaResponseCodes.sol#124) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.MAX_CONTRACT_STORAGE_EXCEEDED (contracts/HederaResponseCodes.sol#125) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TRANSFER_ACCOUNT_SAME_AS_DELETE_ACCOUNT (contracts/HederaResponseCodes.sol#126) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TOTAL_LEDGER_BALANCE_INVALID (contracts/HederaResponseCodes.sol#127) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.EXPIRATION_REDUCTION_NOT_ALLOWED (contracts/HederaResponseCodes.sol#128) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.MAX_GAS_LIMIT_EXCEEDED (contracts/HederaResponseCodes.sol#129) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.MAX_FILE_SIZE_EXCEEDED (contracts/HederaResponseCodes.sol#130) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_TOPIC_ID (contracts/HederaResponseCodes.sol#132) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_ADMIN_KEY (contracts/HederaResponseCodes.sol#133) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_SUBMIT_KEY (contracts/HederaResponseCodes.sol#134) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.UNAUTHORIZED (contracts/HederaResponseCodes.sol#135) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_TOPIC_MESSAGE (contracts/HederaResponseCodes.sol#136) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_AUTORENEW_ACCOUNT (contracts/HederaResponseCodes.sol#137) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.AUTORENEW_ACCOUNT_NOT_ALLOWED (contracts/HederaResponseCodes.sol#138) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TOPIC_EXPIRED (contracts/HederaResponseCodes.sol#142) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_CHUNK_NUMBER (contracts/HederaResponseCodes.sol#143) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_CHUNK_TRANSACTION_ID (contracts/HederaResponseCodes.sol#144) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.ACCOUNT_FROZEN_FOR_TOKEN (contracts/HederaResponseCodes.sol#145) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TOKENS_PER_ACCOUNT_LIMIT_EXCEEDED (contracts/HederaResponseCodes.sol#146) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_TOKEN_ID (contracts/HederaResponseCodes.sol#147) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_TOKEN_DECIMALS (contracts/HederaResponseCodes.sol#148) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_TOKEN_INITIAL_SUPPLY (contracts/HederaResponseCodes.sol#149) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_TREASURY_ACCOUNT_FOR_TOKEN (contracts/HederaResponseCodes.sol#150) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_TOKEN_SYMBOL (contracts/HederaResponseCodes.sol#151) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TOKEN_HAS_NO_FREEZE_KEY (contracts/HederaResponseCodes.sol#152) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TRANSFERS_NOT_ZERO_SUM_FOR_TOKEN (contracts/HederaResponseCodes.sol#153) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.MISSING_TOKEN_SYMBOL (contracts/HederaResponseCodes.sol#154) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TOKEN_SYMBOL_TOO_LONG (contracts/HederaResponseCodes.sol#155) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.ACCOUNT_KYC_NOT_GRANTED_FOR_TOKEN (contracts/HederaResponseCodes.sol#156) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TOKEN_HAS_NO_KYC_KEY (contracts/HederaResponseCodes.sol#157) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INSUFFICIENT_TOKEN_BALANCE (contracts/HederaResponseCodes.sol#158) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TOKEN_WAS_DELETED (contracts/HederaResponseCodes.sol#159) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TOKEN_HAS_NO_SUPPLY_KEY (contracts/HederaResponseCodes.sol#160) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TOKEN_HAS_NO_WIPE_KEY (contracts/HederaResponseCodes.sol#161) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_TOKEN_MINT_AMOUNT (contracts/HederaResponseCodes.sol#162) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_TOKEN_BURN_AMOUNT (contracts/HederaResponseCodes.sol#163) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TOKEN_NOT_ASSOCIATED_TO_ACCOUNT (contracts/HederaResponseCodes.sol#164) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.CANNOT_WIPE_TOKEN_TREASURY_ACCOUNT (contracts/HederaResponseCodes.sol#165) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_KYC_KEY (contracts/HederaResponseCodes.sol#166) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_WIPE_KEY (contracts/HederaResponseCodes.sol#167) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_FREEZE_KEY (contracts/HederaResponseCodes.sol#168) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_SUPPLY_KEY (contracts/HederaResponseCodes.sol#169) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.MISSING_TOKEN_NAME (contracts/HederaResponseCodes.sol#170) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TOKEN_NAME_TOO_LONG (contracts/HederaResponseCodes.sol#171) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_WIPING_AMOUNT (contracts/HederaResponseCodes.sol#172) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TOKEN_IS_IMMUTABLE (contracts/HederaResponseCodes.sol#173) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TOKEN_ALREADY_ASSOCIATED_TO_ACCOUNT (contracts/HederaResponseCodes.sol#174) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TRANSACTION_REQUIRES_ZERO_TOKEN_BALANCES (contracts/HederaResponseCodes.sol#175) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.ACCOUNT_IS_TREASURY (contracts/HederaResponseCodes.sol#176) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TOKEN_ID_REPEATED_IN_TOKEN_LIST (contracts/HederaResponseCodes.sol#177) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TOKEN_TRANSFER_LIST_SIZE_LIMIT_EXCEEDED (contracts/HederaResponseCodes.sol#178) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.EMPTY_TOKEN_TRANSFER_BODY (contracts/HederaResponseCodes.sol#179) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.EMPTY_TOKEN_TRANSFER_ACCOUNT_AMOUNTS (contracts/HederaResponseCodes.sol#180) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_SCHEDULE_ID (contracts/HederaResponseCodes.sol#182) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.SCHEDULE_IS_IMMUTABLE (contracts/HederaResponseCodes.sol#183) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_SCHEDULE_PAYER_ID (contracts/HederaResponseCodes.sol#184) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_SCHEDULE_ACCOUNT_ID (contracts/HederaResponseCodes.sol#185) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.NO_NEW_VALID_SIGNATURES (contracts/HederaResponseCodes.sol#186) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.UNRESOLVABLE_REQUIRED_SIGNERS (contracts/HederaResponseCodes.sol#187) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.SCHEDULED_TRANSACTION_NOT_IN_WHITELIST (contracts/HederaResponseCodes.sol#188) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.SOME_SIGNATURES_WERE_INVALID (contracts/HederaResponseCodes.sol#189) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.TRANSACTION_ID_FIELD_NOT_ALLOWED (contracts/HederaResponseCodes.sol#190) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.IDENTICAL_SCHEDULE_ALREADY_CREATED (contracts/HederaResponseCodes.sol#191) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.INVALID_ZERO_BYTE_IN_STRING (contracts/HederaResponseCodes.sol#192) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.SCHEDULE_ALREADY_DELETED (contracts/HederaResponseCodes.sol#193) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.SCHEDULE_ALREADY_EXECUTED (contracts/HederaResponseCodes.sol#194) is never used in Staking (contracts/Staking.sol#18-278)
HederaResponseCodes.MESSAGE_SIZE_TOO_LARGE (contracts/HederaResponseCodes.sol#195) is never used in Staking (contracts/Staking.sol#18-278)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable
```

AUTOMATED TESTING

Undelegation.sol

```
Reentrancy in Undelegation.withdraw(uint256) (contracts/Undelegation.sol#60-76):
        External calls:
        - Address.sendValue(address(msg.sender),amount) (contracts/Undelegation.sol#74)
        Event emitted after the call(s):
        - Withdrawn(msg.sender,amount) (contracts/Undelegation.sol#75)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Undelegation.withdraw(uint256) (contracts/Undelegation.sol#60-76) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(undelegateData.timestamp + unbondingTime <= block.timestamp,Release time not reached) (contracts/Undelegation.sol#66-69)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp

Address.isContract(address) (node_modules/@openzeppelin/contracts/utils/Address.sol#27-37) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/Address.sol#33-35)
Address.verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#196-216) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/Address.sol#208-211)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['^0.8.0', '^0.8.9']
        - ^0.8.0 (node_modules/@openzeppelin/contracts/security/Pausable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/security/ReentrancyGuard.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Address.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#4)
        - ^0.8.0 (contracts/Ownable.sol#4)
        - ^0.8.9 (contracts/Undelegation.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#80-82) is never used and should be removed
Address.functionCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#90-96) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#109-115) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#123-134) is never used and should be removed
Address.functionDelegateCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#169-171) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#179-188) is never used and should be removed
Address.functionStaticCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#142-144) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#152-161) is never used and should be removed
Address.isContract(address) (node_modules/@openzeppelin/contracts/utils/Address.sol#27-37) is never used and should be removed
Address.verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#196-216) is never used and should be removed
Context._msgData() (node_modules/@openzeppelin/contracts/utils/Context.sol#21-23) is never used and should be removed
SafeCast.toInt128(int256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#152-155) is never used and should be removed
SafeCast.toInt16(int256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#206-209) is never used and should be removed
SafeCast.toInt256(uint256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#236-240) is never used and should be removed
SafeCast.toInt32(int256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#188-191) is never used and should be removed
SafeCast.toInt64(int256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#170-173) is never used and should be removed
SafeCast.toInt8(int256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#224-227) is never used and should be removed
SafeCast.toUint128(uint256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#47-50) is never used and should be removed
SafeCast.toUint16(uint256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#107-110) is never used and should be removed
SafeCast.toUint224(uint256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#32-35) is never used and should be removed
SafeCast.toUint256(int256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#134-137) is never used and should be removed
SafeCast.toUint32(uint256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#92-95) is never used and should be removed
SafeCast.toUint64(uint256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#77-80) is never used and should be removed
SafeCast.toUint8(uint256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#122-125) is never used and should be removed
SafeCast.toUint96(uint256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#62-65) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/security/Pausable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/security/ReentrancyGuard.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Address.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#4) allows old versions
Pragma version^0.8.0 (contracts/Ownable.sol#4) allows old versions
Pragma version^0.8.9 (contracts/Undelegation.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.13 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#55-60):
        - (success) = recipient.call{value: amount}() (node_modules/@openzeppelin/contracts/utils/Address.sol#58)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#123-134):
        - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#132)
Low level call in Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#152-161):
        - (success,returndata) = target.staticcall(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#159)
Low level call in Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#179-188):
        - (success,returndata) = target.delegatecall(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#186)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter Undelegation.setStakingContractAddress(address)._stakingContractAddress (contracts/Undelegation.sol#84) is not in mixedCase
Parameter Undelegation.setUnbondingTime(uint256)._unbondingTime (contracts/Undelegation.sol#97) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

- The reentrancies flagged are all protected with the `nonReentrant` modifier.
- No major issues found by Slither.

4.2 AUTOMATED SECURITY SCAN

Description:

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

MythX results:

Rewards.sol

Report for contracts/Rewards.sol
https://dashboard.mythx.io/#/console/analyses/9bad902c-a670-4947-a5e1-2c50f6af39b0

Line	SWC Title	Severity	Short Description
2	(SWC-103) Floating Pragma	Low	A floating pragma is set.
25	(SWC-108) State Variable Default Visibility	Low	State variable visibility is not set.
27	(SWC-108) State Variable Default Visibility	Low	State variable visibility is not set.
91	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
93	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered
94	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
99	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
99	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
102	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
105	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered

Staking.sol

Report for contracts/Staking.sol
https://dashboard.mythx.io/#/console/analyses/e3e0a616-7167-4c67-ac9c-0d7c675dfa78

Line	SWC Title	Severity	Short Description
2	(SWC-103) Floating Pragma	Low	A floating pragma is set.
31	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "**" discovered
36	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
36	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "**" discovered
104	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
106	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
106	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
107	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
149	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered

149	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
193	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
197	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
197	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered

Undelegation.sol

Report for contracts/Undelegation.sol
https://dashboard.mythx.io/#/console/analyses/aead3ce4-e9f0-4fa1-a5ee-279fe0cfdfe7

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

- No major issues found by MythX. MythX correctly flagged that some state variables are missing the public/private keyword, so all of them will be declared as private by default.

THANK YOU FOR CHOOSING

// HALBORN