StaderLabs - TokenERC20

Smart Contract Security Audit

Prepared by: **Halborn**

Date of Engagement: **January 7th, 2022 - January 8th, 2022**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	01/07/2022	Roberto Reigada
0.2	Document Updates	01/08/2022	Roberto Reigada
0.3	Draft Review	01/08/2022	Gabi Urrutia
1.0	Remediation Plan	01/17/2022	Roberto Reigada
1.1	Remediation Plan Review	01/17/2022	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

StaderLabs engaged Halborn to conduct a security audit on their smart contracts beginning on January 7th, 2022 and ending on January 8th, 2022. The security assessment was scoped to the smart contracts provided in the Github repository stader-labs/stader-token-erc20.

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 an informational finding that was addressed by StaderLabs team.

1.3 TEST APPROACH & METHODOLOGY

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

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

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 contract:

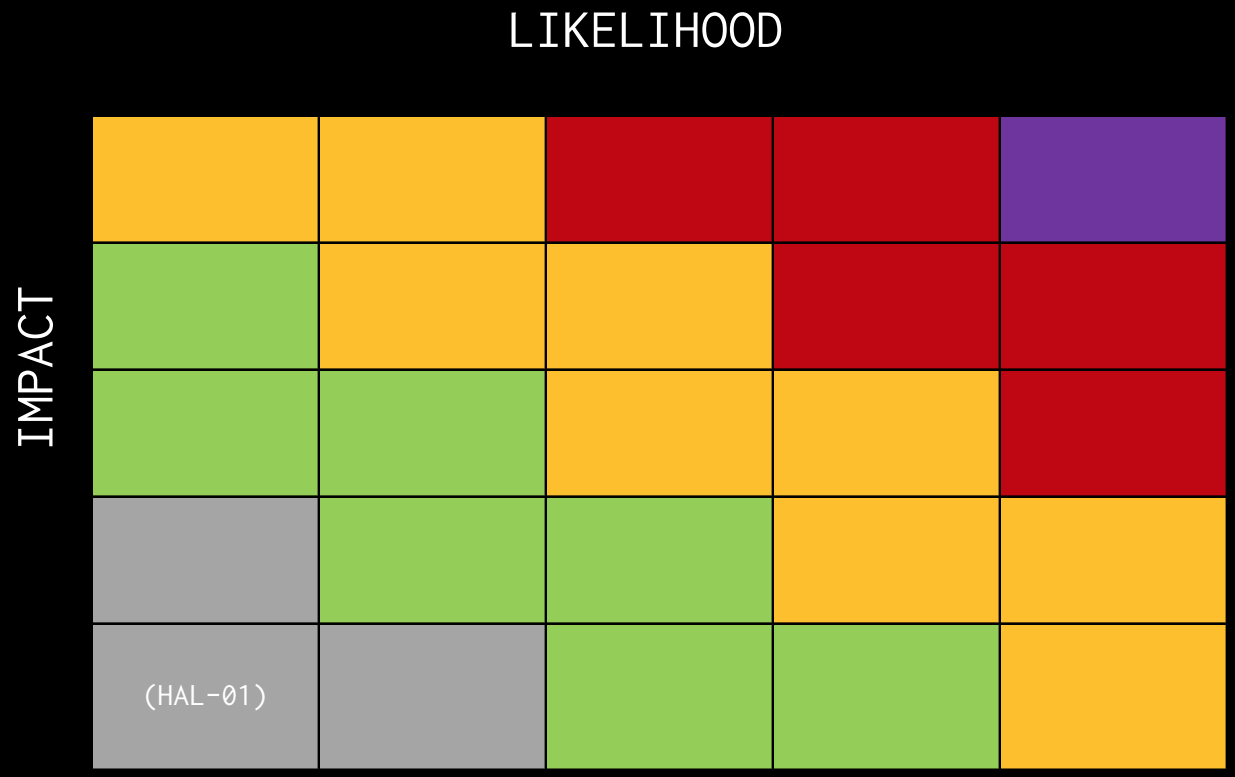
- Stader.sol

Commit ID: 9137e229ae01f05d472fee1881f1e8fe862f452c
Fixed Commit ID: 904a150a5458af1d5d6ca043ed05521a2373f587

Stader Token Address: 0x30D20208d987713f46DFD34EF128Bb16C404D10f

2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	0	0	0	1



SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
HAL-01 - SOLC 0.8.2 COMPILER VERSION CONTAINS MULTIPLE BUGS	Informational	SOLVED - 01/17/2022

FINDINGS & TECH DETAILS

3.1 (HAL-01) SOLC 0.8.2 COMPILER VERSION CONTAINS MULTIPLE BUGS - INFORMATIONAL

Description:

Solidity compiler version 0.8.3, 0.8.4 and 0.8.9 fixed important bugs in the compiler. The version 0.8.2 set in the hardhat.config.js file is missing all these fixes:

- 0.8.3
- 0.8.4
- 0.8.9

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

It is recommended to use the most tested and stable versions, such as 0.6.12 or 0.7.6. Otherwise, if you still want to use ^0.8.0, because of the new functionality it provides, it is recommended to use 0.8.9 version.

Remediation Plan:

SOLVED: The StaderLabs team set the pragma to the 0.8.9 version.

MANUAL TESTING

Halborn performed several manual tests in the Stader.sol contract:

```
Deploying Stader (contract_Stader)...
Transaction sent: 0x93448f1b48b07001a8446831a0833553249737e450fd32434760ed80c7c20cd2
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 0
  Stader.constructor confirmed   Block: 13957576   Gas used: 1901673 (28.29%)
  Stader deployed at: 0x30212306b27024526c6094334EcAB78cCFa23a62

Stader (contract_Stader) deployed

contract_Stader.signatures:
{'DOMAIN_SEPARATOR': '0x3644e515', 'allowance': '0xdd62ed3e', 'approve': '0x095ea7b3', 'balanceOf': '0x70a08231', 'burn': '0x42966c68', 'burnFrom': '0x79cc6790', 'checkpoints': '0xf1127ed8', 'decimals': '0x313ce567', 'decreaseAllowance':
 '0xa457c2d7', 'delegate': '0x5c19a95c', 'delegateBySig': '0xc3cda520', 'delegates': '0x587cde1e', 'getPastTotalSupply': '0x8e539e8c', 'getPastVotes': '0x3a46b1a8', 'getVotes': '0x9ab24eb0', 'increaseAllowance': '0x39509351', 'name': '0x
06fdde03', 'nonces': '0x7ecebe00', 'numCheckpoints': '0x6fcfff45', 'permit': '0xd505accf', 'symbol': '0x95d89b41', 'totalSupply': '0x18160ddd', 'transfer': '0xa9059cbb', 'transferFrom': '0x23b872dd'}

contract_Stader.DOMAIN_SEPARATOR() -> 0x3fc6bbee4f9f5338b91e23e0b942deb1c592564934025dcd1667ddd25f7f4475
contract_Stader.name() -> Stader
contract_Stader.symbol() -> SD
contract_Stader.decimals() -> 18
contract_Stader.totalSupply() -> 150000000000000000000000000
contract_Stader.getPastTotalSupply(block.number - 10) -> 0

contract_Stader.allowance(user1.address, user2.address) -> 0

Calling -> contract_Stader.approve(user2.address, 100000_000000000000000000, {'from': user1})
Transaction sent: 0xfbc6cbf7109b2700394ac6d70f39972b7ef95e01c3fee1e84c306160a879e73c
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 0
  Stader.approve confirmed   Block: 13957577   Gas used: 44198 (0.66%)

contract_Stader.allowance(user1.address, user2.address) -> 100000000000000000000000

Calling -> contract_Stader.approve(user2.address, 0, {'from': user1})
Transaction sent: 0x962b10e24e296732613ddf576cbde05fb4a837ac5540cbc2af5689f40445a6aa
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 1
  Stader.approve confirmed   Block: 13957578   Gas used: 14551 (0.22%)

contract_Stader.allowance(user1.address, user2.address) -> 0

contract_Stader.balanceOf(owner.address) -> 150000000000000000000000000
contract_Stader.balanceOf(user1.address) -> 0

Calling -> contract_Stader.transfer(user1.address, 1, {'from': owner})
Transaction sent: 0x4677c7c736f5f937fdd1c9dc159ac2b0499c9d983ee437da5a24db2bd1b4cbbc
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 1
  Stader.transfer confirmed   Block: 13957579   Gas used: 52951 (0.79%)

contract_Stader.balanceOf(owner.address) -> 149999999999999999999999999
contract_Stader.balanceOf(user1.address) -> 1

Calling -> contract_Stader.approve(user1.address, 1000, {'from': owner})
Transaction sent: 0xd18f0955c47ff8fa0027212ee65272465393a8eb5d42470e94cbf178183d8b89
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 2
  Stader.approve confirmed   Block: 13957580   Gas used: 44126 (0.66%)

Calling -> contract_Stader.transferFrom(owner.address, user1.address, 1000, {'from': user1})
Transaction sent: 0xeb54c68f89446368bfb7ee41a89580d23d0228e7df11614289a7e0a1175c52f0
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 2
  Stader.transferFrom confirmed   Block: 13957581   Gas used: 31568 (0.47%)

contract_Stader.balanceOf(owner.address) -> 149999999999999999999998999
contract_Stader.balanceOf(user1.address) -> 1001

contract_Stader.balanceOf(owner.address) -> 149999999999999999999998999

Calling -> contract_Stader.burn(8999, {'from': owner})
Transaction sent: 0xe9fddcdcb28a116d2f40e25d62ae3cfc78030216e06104a3671d1281eed1f2ce
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 3
  Stader.burn confirmed   Block: 13957582   Gas used: 68168 (1.01%)

contract_Stader.balanceOf(owner.address) -> 149999999999999999999990000

Calling -> contract_Stader.increaseAllowance(user1.address, 1000, {'from': owner})
Transaction sent: 0x5bd1c9d6749fb89660ab685cd6b6caff90551613805e2f9ca8b75715732246 1d
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 4
  Stader.increaseAllowance confirmed   Block: 13957583   Gas used: 30141 (0.45%)

Calling -> contract_Stader.burnFrom(owner.address, 1000, {'from': user1})
Transaction sent: 0xf0f0f6fbf480fb9d603c55961391709ad4fccfaabe5d4540f6617073cb6bf316
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 3
  Stader.burnFrom confirmed   Block: 13957584   Gas used: 61759 (0.92%)

contract_Stader.balanceOf(owner.address) -> 149999999999999999999989000

contract_Stader.allowance(user1.address, user3.address) -> 0

Calling -> contract_Stader.increaseAllowance(user3.address, 100000_000000000000000000, {'from': user1})
Transaction sent: 0x8ed5dbcd788ef9e0cbdf6b73cc11029c93f21cbcc6f33a257913469ac056cc57
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 4
  Stader.increaseAllowance confirmed   Block: 13957585   Gas used: 45213 (0.67%)

contract_Stader.allowance(user1.address, user3.address) -> 100000000000000000000000

Calling -> contract_Stader.decreaseAllowance(user3.address, 1, {'from': user1})
Transaction sent: 0xd79ad263aabcb63ca5ebc1f35ccb43ad4d7dce0694d08acef979df8379831cb3
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 5
  Stader.decreaseAllowance confirmed   Block: 13957586   Gas used: 30082 (0.45%)

contract_Stader.allowance(user1.address, user3.address) -> 99999999999999999999999

contract_Stader.numCheckpoints(owner) -> 0
contract_Stader.getVotes(owner) -> 0
contract_Stader.delegates(owner) -> 0x0000000000000000000000000000000000000000

Calling -> contract_Stader.delegate(owner.address, {'from': owner})
Transaction sent: 0x4db8340ff513ed0846e26389c90fe1942a47d21cddbcfa49b87be4420ac07818
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 5
  Stader.delegate confirmed   Block: 13957587   Gas used: 89944 (1.34%)

owner.address -> 0x995354642c81F4912756e4319Fc59B8871827729
contract_Stader.delegates(owner) -> 0x995354642c81F4912756e4319Fc59B8871827729
contract_Stader.checkpoints(owner, 0) -> (13957587, 149999999999999999999989000)
contract_Stader.numCheckpoints(owner) -> 1
contract_Stader.getVotes(owner) -> 149999999999999999999989000
contract_Stader.getPastVotes(owner, block.number - 10) -> 0
```

As the contract just follows the standard OpenZeppelin ERC20 contracts ERC20, ERC20Burnable, ERC20Permit and ERC20Votes, the manual tests were focused on testing the main functions of these contracts:

- approve()
- balanceOf()
- burn()

- burnFrom()
- decreaseAllowance()
- delegate()
- increaseAllowance()
- transfer()
- transferFrom()

The delegateBySig() function was also tested:

```
Calling -> contract_Stader.transfer(alice.address, 1000000, {'from': owner})
Transaction sent: 0x39fb5ede65011029140eb95f19a991f0c5a3174120b19c341d0f0bf22ca9bc1d
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 13
  Stader.transfer confirmed   Block: 13958342   Gas used: 85736 (1.28%)


Calling -> contract_Stader.delegate(alice.address, {'from': alice})
Transaction sent: 0x3a8881e92a1e23d7e72686d7818fa7fd86fdab94c5b02ff45bc595c90ffd856
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 1
  Stader.delegate confirmed   Block: 13958343   Gas used: 89944 (1.34%)

contract_Stader.balanceOf(alice.address) -> 1000000
contract_Stader.balanceOf(bob.address) -> 0
contract_Stader.getVotes(alice) -> 1000000
contract_Stader.getVotes(bob) -> 0

Calling -> contract_Stader.delegateBySig(bob.address, 0, 9999999999999999999999, 28, '0x3c6d9c819b1780f632431753e3f9fc96edfe34a92e515d1f61101ca5e26692cf', '0x0e5cbe5e5d649ce283c2973720cb9ba3bf631f49c1fbfaba55ad123cfd07ad21', {'from': bob
})
Transaction sent: 0x10373cb95cf317cb698ee8cf86236adbfc999140fad2b434df88d845b5006774ca
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 9
  Stader.delegateBySig confirmed   Block: 13958345   Gas used: 135178 (2.01%)

contract_Stader.getVotes(alice) -> 0
contract_Stader.getVotes(bob) -> 1000000
```

And also the permit() function:

```
contract_Stader.allowance(alice.address, bob.address) -> 0

Calling -> contract_Stader.permit(alice.address, bob.address, 1337, 9999999999999999999999, 27, '0x609ab5fdf9168acba19d18b4165846bbc35733b7df574d565e2c57e89abb63e', '0x0c9b9df4857f9a592e01013558183c687c11f641746907797537ab4dad216c82', {
'from': bob})
Transaction sent: 0xd91fe217e26d1b1d980c99d330e01ceb5983c8392a86aec330d747dcb70d2ba5
  Gas price: 0.0 gwei   Gas limit: 6721975   Nonce: 11
  Stader.permit confirmed   Block: 13958360   Gas used: 72107 (1.07%)

contract_Stader.allowance(alice.address, bob.address) -> 1337
```

No issues were found during the manual tests.

AUTOMATED TESTING

5.1 STATIC ANALYSIS REPORT

Description:

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

Slither results:

Stader.sol

```
ERC20Votes._writeCheckpoint(ERC20Votes.Checkpoint[],function(uint256,uint256) returns(uint256),uint256) (node_modules/@openzeppelin/contracts/token/ERC20/extensions/ERC20Votes.sol#237-251) uses a dangerous strict equality:
        - pos > 0 && ckpts[pos - 1].fromBlock == block.number (node_modules/@openzeppelin/contracts/token/ERC20/extensions/ERC20Votes.sol#246)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities

ERC20Votes._moveVotingPower(address,address,uint256).newWeight_scope_1 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/ERC20Votes.sol#231) is a local variable never initialized
ERC20Votes._moveVotingPower(address,address,uint256).oldWeight_scope_0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/ERC20Votes.sol#231) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables

ERC20Permit.constructor(string).name (node_modules/@openzeppelin/contracts/token/ERC20/extensions/draft-ERC20Permit.sol#36) shadows:
        - ERC20.name() (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#62-64) (function)
        - IERC20Metadata.name() (node_modules/@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol#17) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing

Variable 'ERC20Votes._moveVotingPower(address,address,uint256).newWeight (node_modules/@openzeppelin/contracts/token/ERC20/extensions/ERC20Votes.sol#226)' in ERC20Votes._moveVotingPower(address,address,uint256) (node_modules/@openzeppeli
n/contracts/token/ERC20/extensions/ERC20Votes.sol#219-235) potentially used before declaration: (oldWeight,newWeight) = _writeCheckpoint(_checkpoints[dst],_add,amount) (node_modules/@openzeppelin/contracts/token/ERC20/extensions/ERC20Vot
es.sol#231)
Variable 'ERC20Votes._moveVotingPower(address,address,uint256).oldWeight (node_modules/@openzeppelin/contracts/token/ERC20/extensions/ERC20Votes.sol#226)' in ERC20Votes._moveVotingPower(address,address,uint256) (node_modules/@openzeppeli
n/contracts/token/ERC20/extensions/ERC20Votes.sol#219-235) potentially used before declaration: (oldWeight,newWeight) = _writeCheckpoint(_checkpoints[dst],_add,amount) (node_modules/@openzeppelin/contracts/token/ERC20/extensions/ERC20Vot
es.sol#231)
Variable 'ECDSA.tryRecover(bytes32,bytes).r (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#62)' in ECDSA.tryRecover(bytes32,bytes) (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#57-86) potentially
used before declaration: r = mload(uint256)(signature + 0x20) (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#79)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#pre-declaration-usage-of-local-variables

ERC20Votes.delegateBySig(address,uint256,uint256,uint8,bytes32,bytes32) (node_modules/@openzeppelin/contracts/token/ERC20/extensions/ERC20Votes.sol#144-161) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(block.timestamp <= expiry,ERC20Votes: signature expired) (node_modules/@openzeppelin/contracts/token/ERC20/extensions/ERC20Votes.sol#152)
ERC20Permit.permit(address,address,uint256,uint256,uint8,bytes32,bytes32) (node_modules/@openzeppelin/contracts/token/ERC20/extensions/draft-ERC20Permit.sol#41-60) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(block.timestamp <= deadline,ERC20Permit: expired deadline) (node_modules/@openzeppelin/contracts/token/ERC20/extensions/draft-ERC20Permit.sol#50)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp

ECDSA.tryRecover(bytes32,bytes) (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#57-86) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#67-71)
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#78-81)
ECDSA.tryRecover(bytes32,bytes32,bytes32) (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#115-127) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#122-125)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['^0.8.0', '^0.8.2']
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/ERC20Burnable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/ERC20Votes.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/draft-ERC20Permit.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/draft-IERC20Permit.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Counters.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Strings.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/cryptography/draft-EIP712.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/math/Math.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#4)
        - ^0.8.2 (contracts/Stader.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Context._msgData() (node_modules/@openzeppelin/contracts/utils/Context.sol#21-23) is never used and should be removed
Counters.decrement(Counters.Counter) (node_modules/@openzeppelin/contracts/utils/Counters.sol#32-38) is never used and should be removed
Counters.reset(Counters.Counter) (node_modules/@openzeppelin/contracts/utils/Counters.sol#40-42) is never used and should be removed
ECDSA.recover(bytes32,bytes) (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#102-106) is never used and should be removed
ECDSA.recover(bytes32,bytes32,bytes32) (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#134-142) is never used and should be removed
ECDSA.toEthSignedMessageHash(bytes) (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#218-220) is never used and should be removed
ECDSA.toEthSignedMessageHash(bytes32) (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#204-200) is never used and should be removed
ECDSA.tryRecover(bytes32,bytes) (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#57-86) is never used and should be removed
ECDSA.tryRecover(bytes32,bytes32,bytes32) (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#115-127) is never used and should be removed
ERC20Votes._add(uint256,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/extensions/ERC20Votes.sol#253-255) is never used and should be removed
ERC20Votes._subtract(uint256,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/extensions/ERC20Votes.sol#257-259) is never used and should be removed
Math.ceilDiv(uint256,uint256) (node_modules/@openzeppelin/contracts/utils/math/Math.sol#39-42) is never used and should be removed
Math.max(uint256,uint256) (node_modules/@openzeppelin/contracts/utils/math/Math.sol#13-15) is never used and should be removed
Math.min(uint256,uint256) (node_modules/@openzeppelin/contracts/utils/math/Math.sol#20-22) 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.toUint256(int256) (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#134-137) 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
Strings.toHexString(uint256) (node_modules/@openzeppelin/contracts/utils/Strings.sol#40-51) is never used and should be removed
Strings.toHexString(uint256,uint256) (node_modules/@openzeppelin/contracts/utils/Strings.sol#56-66) is never used and should be removed
Strings.toString(uint256) (node_modules/@openzeppelin/contracts/utils/Strings.sol#15-35) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
```

AUTOMATED TESTING

```
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#4) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#4) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/ERC20Burnable.sol#4) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/ERC20Votes.sol#4) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol#4) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/draft-ERC20Permit.sol#4) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/draft-IERC20Permit.sol#4) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol#4) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Counters.sol#4) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Strings.sol#4) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#4) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/cryptography/draft-EIP712.sol#4) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/math/Math.sol#4) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/math/SafeCast.sol#4) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.2 (contracts/Stader.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.4 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Function ERC20Permit.DOMAIN_SEPARATOR() (node_modules/@openzeppelin/contracts/token/ERC20/extensions/draft-ERC20Permit.sol#73-75) is not in mixedCase
Variable ERC20Permit._PERMIT_TYPEHASH (node_modules/@openzeppelin/contracts/token/ERC20/extensions/draft-ERC20Permit.sol#28-29) is not in mixedCase
Function IERC20Permit.DOMAIN_SEPARATOR() (node_modules/@openzeppelin/contracts/token/ERC20/extensions/draft-IERC20Permit.sol#59) is not in mixedCase
Variable EIP712._CACHED_DOMAIN_SEPARATOR (node_modules/@openzeppelin/contracts/utils/cryptography/draft-EIP712.sol#31) is not in mixedCase
Variable EIP712._CACHED_CHAIN_ID (node_modules/@openzeppelin/contracts/utils/cryptography/draft-EIP712.sol#32) is not in mixedCase
Variable EIP712._CACHED_THIS (node_modules/@openzeppelin/contracts/utils/cryptography/draft-EIP712.sol#33) is not in mixedCase
Variable EIP712._HASHED_NAME (node_modules/@openzeppelin/contracts/utils/cryptography/draft-EIP712.sol#35) is not in mixedCase
Variable EIP712._HASHED_VERSION (node_modules/@openzeppelin/contracts/utils/cryptography/draft-EIP712.sol#36) is not in mixedCase
Variable EIP712._TYPE_HASH (node_modules/@openzeppelin/contracts/utils/cryptography/draft-EIP712.sol#37) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Stader.constructor() (contracts/Stader.sol#10-12) uses literals with too many digits:
        - _mint(msg.sender,150000000 * 10 ** decimals()) (contracts/Stader.sol#11)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

ERC20Permit._PERMIT_TYPEHASH (node_modules/@openzeppelin/contracts/token/ERC20/extensions/draft-ERC20Permit.sol#28-29) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
```

- No major issues were found by Slither.

ERC20 checks:

Stader.sol

```
# Check ERC20

## Check functions
[✓] totalSupply() is present
        [✓] totalSupply() -> () (correct return value)
        [✓] totalSupply() is view
[✓] balanceOf(address) is present
        [✓] balanceOf(address) -> () (correct return value)
        [✓] balanceOf(address) is view
[✓] transfer(address,uint256) is present
        [✓] transfer(address,uint256) -> () (correct return value)
        [✓] Transfer(address,address,uint256) is emitted
[✓] transferFrom(address,address,uint256) is present
        [✓] transferFrom(address,address,uint256) -> () (correct return value)
        [✓] Transfer(address,address,uint256) is emitted
[✓] approve(address,uint256) is present
        [✓] approve(address,uint256) -> () (correct return value)
        [✓] Approval(address,address,uint256) is emitted
[✓] allowance(address,address) is present
        [✓] allowance(address,address) -> () (correct return value)
        [✓] allowance(address,address) is view
[✓] name() is present
        [✓] name() -> () (correct return value)
        [✓] name() is view
[✓] symbol() is present
        [✓] symbol() -> () (correct return value)
        [✓] symbol() is view
[✓] decimals() is present
        [✓] decimals() -> () (correct return value)
        [✓] decimals() is view

## Check events
[✓] Transfer(address,address,uint256) is present
        [✓] parameter 0 is indexed
        [✓] parameter 1 is indexed
[✓] Approval(address,address,uint256) is present
        [✓] parameter 0 is indexed
        [✓] parameter 1 is indexed


# Check ERC20Burnable

## Check functions
[✓] totalSupply() is present
        [✓] totalSupply() -> () (correct return value)
        [✓] totalSupply() is view
[✓] balanceOf(address) is present
        [✓] balanceOf(address) -> () (correct return value)
        [✓] balanceOf(address) is view
[✓] transfer(address,uint256) is present
        [✓] transfer(address,uint256) -> () (correct return value)
        [✓] Transfer(address,address,uint256) is emitted
[✓] transferFrom(address,address,uint256) is present
        [✓] transferFrom(address,address,uint256) -> () (correct return value)
        [✓] Transfer(address,address,uint256) is emitted
[✓] approve(address,uint256) is present
        [✓] approve(address,uint256) -> () (correct return value)
        [✓] Approval(address,address,uint256) is emitted
[✓] allowance(address,address) is present
        [✓] allowance(address,address) -> () (correct return value)
        [✓] allowance(address,address) is view
[✓] name() is present
        [✓] name() -> () (correct return value)
        [✓] name() is view
[✓] symbol() is present
        [✓] symbol() -> () (correct return value)
        [✓] symbol() is view
[✓] decimals() is present
        [✓] decimals() -> () (correct return value)
        [✓] decimals() is view

## Check events
[✓] Transfer(address,address,uint256) is present
        [✓] parameter 0 is indexed
        [✓] parameter 1 is indexed
[✓] Approval(address,address,uint256) is present
        [✓] parameter 0 is indexed
        [✓] parameter 1 is indexed


# Check Stader

## Check functions
[✓] totalSupply() is present
        [✓] totalSupply() -> () (correct return value)
        [✓] totalSupply() is view
[✓] balanceOf(address) is present
        [✓] balanceOf(address) -> () (correct return value)
        [✓] balanceOf(address) is view
[✓] transfer(address,uint256) is present
        [✓] transfer(address,uint256) -> () (correct return value)
        [✓] Transfer(address,address,uint256) is emitted
[✓] transferFrom(address,address,uint256) is present
        [✓] transferFrom(address,address,uint256) -> () (correct return value)
        [✓] Transfer(address,address,uint256) is emitted
[✓] approve(address,uint256) is present
        [✓] approve(address,uint256) -> () (correct return value)
        [✓] Approval(address,address,uint256) is emitted
[✓] allowance(address,address) is present
        [✓] allowance(address,address) -> () (correct return value)
        [✓] allowance(address,address) is view
[✓] name() is present
        [✓] name() -> () (correct return value)
        [✓] name() is view
[✓] symbol() is present
        [✓] symbol() -> () (correct return value)
        [✓] symbol() is view
[✓] decimals() is present
        [✓] decimals() -> () (correct return value)
        [✓] decimals() is view
```

```
## Check events
[✓] Transfer(address,address,uint256) is present
        [✓] parameter 0 is indexed
        [✓] parameter 1 is indexed
[✓] Approval(address,address,uint256) is present
        [✓] parameter 0 is indexed
        [✓] parameter 1 is indexed


# Check ERC20Votes

## Check functions
[✓] totalSupply() is present
        [✓] totalSupply() -> () (correct return value)
        [✓] totalSupply() is view
[✓] balanceOf(address) is present
        [✓] balanceOf(address) -> () (correct return value)
        [✓] balanceOf(address) is view
[✓] transfer(address,uint256) is present
        [✓] transfer(address,uint256) -> () (correct return value)
        [✓] Transfer(address,address,uint256) is emitted
[✓] transferFrom(address,address,uint256) is present
        [✓] transferFrom(address,address,uint256) -> () (correct return value)
        [✓] Transfer(address,address,uint256) is emitted
[✓] approve(address,uint256) is present
        [✓] approve(address,uint256) -> () (correct return value)
        [✓] Approval(address,address,uint256) is emitted
[✓] allowance(address,address) is present
        [✓] allowance(address,address) -> () (correct return value)
        [✓] allowance(address,address) is view
[✓] name() is present
        [✓] name() -> () (correct return value)
        [✓] name() is view
[✓] symbol() is present
        [✓] symbol() -> () (correct return value)
        [✓] symbol() is view
[✓] decimals() is present
        [✓] decimals() -> () (correct return value)
        [✓] decimals() is view

## Check events
[✓] Transfer(address,address,uint256) is present
        [✓] parameter 0 is indexed
        [✓] parameter 1 is indexed
[✓] Approval(address,address,uint256) is present
        [✓] parameter 0 is indexed
        [✓] parameter 1 is indexed


# Check Stader

## Check functions
[✓] totalSupply() is present
        [✓] totalSupply() -> () (correct return value)
        [✓] totalSupply() is view
[✓] balanceOf(address) is present
        [✓] balanceOf(address) -> () (correct return value)
        [✓] balanceOf(address) is view
[✓] transfer(address,uint256) is present
        [✓] transfer(address,uint256) -> () (correct return value)
        [✓] Transfer(address,address,uint256) is emitted
[✓] transferFrom(address,address,uint256) is present
        [✓] transferFrom(address,address,uint256) -> () (correct return value)
        [✓] Transfer(address,address,uint256) is emitted
[✓] approve(address,uint256) is present
        [✓] approve(address,uint256) -> () (correct return value)
        [✓] Approval(address,address,uint256) is emitted
[✓] allowance(address,address) is present
        [✓] allowance(address,address) -> () (correct return value)
        [✓] allowance(address,address) is view
[✓] name() is present
        [✓] name() -> () (correct return value)
        [✓] name() is view
[✓] symbol() is present
        [✓] symbol() -> () (correct return value)
        [✓] symbol() is view
[✓] decimals() is present
        [✓] decimals() -> () (correct return value)
        [✓] decimals() is view

## Check events
[✓] Transfer(address,address,uint256) is present
        [✓] parameter 0 is indexed
        [✓] parameter 1 is indexed
[✓] Approval(address,address,uint256) is present
        [✓] parameter 0 is indexed
        [✓] parameter 1 is indexed


# Check ERC20Permit

## Check functions
[✓] totalSupply() is present
        [✓] totalSupply() -> () (correct return value)
        [✓] totalSupply() is view
[✓] balanceOf(address) is present
        [✓] balanceOf(address) -> () (correct return value)
        [✓] balanceOf(address) is view
[✓] transfer(address,uint256) is present
        [✓] transfer(address,uint256) -> () (correct return value)
        [✓] Transfer(address,address,uint256) is emitted
[✓] transferFrom(address,address,uint256) is present
        [✓] transferFrom(address,address,uint256) -> () (correct return value)
        [✓] Transfer(address,address,uint256) is emitted
[✓] approve(address,uint256) is present
        [✓] approve(address,uint256) -> () (correct return value)
        [✓] Approval(address,address,uint256) is emitted
[✓] allowance(address,address) is present
        [✓] allowance(address,address) -> () (correct return value)
        [✓] allowance(address,address) is view
[✓] name() is present
        [✓] name() -> () (correct return value)
        [✓] name() is view
[✓] symbol() is present
        [✓] symbol() -> () (correct return value)
        [✓] symbol() is view
[✓] decimals() is present
        [✓] decimals() -> () (correct return value)
        [✓] decimals() is view

## Check events
[✓] Transfer(address,address,uint256) is present
        [✓] parameter 0 is indexed
        [✓] parameter 1 is indexed
[✓] Approval(address,address,uint256) is present
        [✓] parameter 0 is indexed
        [✓] parameter 1 is indexed


# Check ERC20Votes

## Check functions
[✓] totalSupply() is present
        [✓] totalSupply() -> () (correct return value)
        [✓] totalSupply() is view
[✓] balanceOf(address) is present
        [✓] balanceOf(address) -> () (correct return value)
        [✓] balanceOf(address) is view
[✓] transfer(address,uint256) is present
        [✓] transfer(address,uint256) -> () (correct return value)
        [✓] Transfer(address,address,uint256) is emitted
[✓] transferFrom(address,address,uint256) is present
        [✓] transferFrom(address,address,uint256) -> () (correct return value)
        [✓] Transfer(address,address,uint256) is emitted
[✓] approve(address,uint256) is present
        [✓] approve(address,uint256) -> () (correct return value)
        [✓] Approval(address,address,uint256) is emitted
[✓] allowance(address,address) is present
        [✓] allowance(address,address) -> () (correct return value)
        [✓] allowance(address,address) is view
[✓] name() is present
        [✓] name() -> () (correct return value)
        [✓] name() is view
[✓] symbol() is present
        [✓] symbol() -> () (correct return value)
        [✓] symbol() is view
[✓] decimals() is present
        [✓] decimals() -> () (correct return value)
        [✓] decimals() is view

## Check events
[✓] Transfer(address,address,uint256) is present
        [✓] parameter 0 is indexed
        [✓] parameter 1 is indexed
[✓] Approval(address,address,uint256) is present
        [✓] parameter 0 is indexed
        [✓] parameter 1 is indexed
```

AUTOMATED TESTING

```
# Check Stader

## Check functions
[✓] totalSupply() is present
        [✓] totalSupply() -> () (correct return value)
        [✓] totalSupply() is view
[✓] balanceOf(address) is present
        [✓] balanceOf(address) -> () (correct return value)
        [✓] balanceOf(address) is view
[✓] transfer(address,uint256) is present
        [✓] transfer(address,uint256) -> () (correct return value)
        [✓] Transfer(address,address,uint256) is emitted
[✓] transferFrom(address,address,uint256) is present
        [✓] transferFrom(address,address,uint256) -> () (correct return value)
        [✓] Transfer(address,address,uint256) is emitted
[✓] approve(address,uint256) is present
        [✓] approve(address,uint256) -> () (correct return value)
        [✓] Approval(address,address,uint256) is emitted
[✓] allowance(address,address) is present
        [✓] allowance(address,address) -> () (correct return value)
        [✓] allowance(address,address) is view
[✓] name() is present
        [✓] name() -> () (correct return value)
        [✓] name() is view
[✓] symbol() is present
        [✓] symbol() -> () (correct return value)
        [✓] symbol() is view
[✓] decimals() is present
        [✓] decimals() -> () (correct return value)
        [✓] decimals() is view

## Check events
[✓] Transfer(address,address,uint256) is present
        [✓] parameter 0 is indexed
        [✓] parameter 1 is indexed
[✓] Approval(address,address,uint256) is present
        [✓] parameter 0 is indexed
        [✓] parameter 1 is indexed


# Check Stader

## Check functions
[✓] totalSupply() is present
        [✓] totalSupply() -> () (correct return value)
        [✓] totalSupply() is view
[✓] balanceOf(address) is present
        [✓] balanceOf(address) -> () (correct return value)
        [✓] balanceOf(address) is view
[✓] transfer(address,uint256) is present
        [✓] transfer(address,uint256) -> () (correct return value)
        [✓] Transfer(address,address,uint256) is emitted
[✓] transferFrom(address,address,uint256) is present
        [✓] transferFrom(address,address,uint256) -> () (correct return value)
        [✓] Transfer(address,address,uint256) is emitted
[✓] approve(address,uint256) is present
        [✓] approve(address,uint256) -> () (correct return value)
        [✓] Approval(address,address,uint256) is emitted
[✓] allowance(address,address) is present
        [✓] allowance(address,address) -> () (correct return value)
        [✓] allowance(address,address) is view

[✓] name() is present
        [✓] name() -> () (correct return value)
        [✓] name() is view
[✓] symbol() is present
        [✓] symbol() -> () (correct return value)
        [✓] symbol() is view
[✓] decimals() is present
        [✓] decimals() -> () (correct return value)
        [✓] decimals() is view

## Check events
[✓] Transfer(address,address,uint256) is present
        [✓] parameter 0 is indexed
        [✓] parameter 1 is indexed
[✓] Approval(address,address,uint256) is present
        [✓] parameter 0 is indexed
        [✓] parameter 1 is indexed


# Check Stader

## Check functions
[✓] totalSupply() is present
        [✓] totalSupply() -> () (correct return value)
        [✓] totalSupply() is view
[✓] balanceOf(address) is present
        [✓] balanceOf(address) -> () (correct return value)
        [✓] balanceOf(address) is view
[✓] transfer(address,uint256) is present
        [✓] transfer(address,uint256) -> () (correct return value)
        [✓] Transfer(address,address,uint256) is emitted
[✓] transferFrom(address,address,uint256) is present
        [✓] transferFrom(address,address,uint256) -> () (correct return value)
        [✓] Transfer(address,address,uint256) is emitted
[✓] approve(address,uint256) is present
        [✓] approve(address,uint256) -> () (correct return value)
        [✓] Approval(address,address,uint256) is emitted
[✓] allowance(address,address) is present
        [✓] allowance(address,address) -> () (correct return value)
        [✓] allowance(address,address) is view
[✓] name() is present
        [✓] name() -> () (correct return value)
        [✓] name() is view
[✓] symbol() is present
        [✓] symbol() -> () (correct return value)
        [✓] symbol() is view
[✓] decimals() is present
        [✓] decimals() -> () (correct return value)
        [✓] decimals() is view

## Check events
[✓] Transfer(address,address,uint256) is present
        [✓] parameter 0 is indexed
        [✓] parameter 1 is indexed
[✓] Approval(address,address,uint256) is present
        [✓] parameter 0 is indexed
        [✓] parameter 1 is indexed


        [✓] ERC20 has increaseAllowance(address,uint256)
        [✓] ERC20Burnable has increaseAllowance(address,uint256)
        [✓] Stader has increaseAllowance(address,uint256)
        [✓] ERC20Votes has increaseAllowance(address,uint256)
        [✓] Stader has increaseAllowance(address,uint256)
        [✓] ERC20Permit has increaseAllowance(address,uint256)
        [✓] ERC20Votes has increaseAllowance(address,uint256)
        [✓] Stader has increaseAllowance(address,uint256)
        [✓] Stader has increaseAllowance(address,uint256)
        [✓] Stader has increaseAllowance(address,uint256)
```

- All the Slither ERC20 checks were passed successfully.

5.2 AUTOMATED SECURITY SCAN

Description:

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

MythX results:

Stader.sol

Report for contracts/Stader.sol
https://dashboard.mythx.io/#/console/analyses/9ff91709-18db-495e-9170-66dd9ad7337d

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

- No major issues were found by MythX

THANK YOU FOR CHOOSING

// HALBORN