Aderyn Analysis Report

This report was generated by Aderyn, a static analysis tool built by Cyfrin, a blockchain security company. This report is not a substitute for manual audit or security review. It should not be relied upon for any purpose other than to assist in the identification of potential security vulnerabilities. # Table of Contents

Summary

Files Summary

Key	Value
.sol Files	9
Total nSLOC	213

Files Details

Filepath	nSLOC
contracts/Conversion.sol	23
contracts/DCS.sol	31
contracts/Governance.sol	74
contracts/PeoCoin.sol	17

Filepath	nSLOC
contracts/Staking.sol	43
contracts/interfaces/IDCS.sol	4
contracts/interfaces/IPeoCoin.sol	7
contracts/interfaces/IStaking.sol	8
contracts/mocks/MockStaking.sol	6
Total	**213**

Issue Summary

Category	No. of Issues
High	0
Low	13

Low Issues

L-1: Centralization Risk for trusted owners

Contracts have owners with privileged rights to perform admin tasks and need to be trusted to not perform malicious updates or drain funds.

6 Found Instances

- Found in contracts/DCS.sol Line: 12

  ```
  contract DCS is Ownable {
  ```

- Found in contracts/DCS.sol Line: 65

  ```
      function updateWeights(uint256 _tokenWeight, uint256 _stakeWeight, uint256 _timeWei
  ```

- Found in contracts/Governance.sol Line: 12

  ```
  contract Governance is Ownable {
  ```

- Found in contracts/Governance.sol Line: 138

  ```
      function updateParameters(uint256 _votingPeriod, uint256 _quorum, uint256 _majority
  ```

- Found in contracts/PeoCoin.sol Line: 16

  ```
  contract PeoCoin is ERC20, Ownable {
  ```

- Found in contracts/PeoCoin.sol Line: 29

  ```
      function mint(address to, uint256 amount) external onlyOwner {
  ```

L-2: Unsafe ERC20 Operations should not be used

ERC20 functions may not behave as expected. For example: return values are not always meaningful. It is recommended to use OpenZeppelin's SafeERC20 library.

3 Found Instances

- Found in contracts/Conversion.sol Line: 51

```
bool success = peoToken.transferFrom(msg.sender, address(this), amount);
```

- Found in contracts/Staking.sol Line: 52

```
bool success = peoToken.transferFrom(msg.sender, address(this), amount);
```

- Found in contracts/Staking.sol Line: 95

```
bool success = peoToken.transfer(msg.sender, info.amount + reward);
```

L-3: Solidity pragma should be specific, not wide

Consider using a specific version of Solidity in your contracts instead of a wide version. For example, instead of `pragma solidity ^0.8.0;`, use `pragma solidity 0.8.0;`

9 Found Instances

- Found in contracts/Conversion.sol Line: 2

```
pragma solidity ^0.8.28;
```

- Found in contracts/DCS.sol Line: 2

```
pragma solidity ^0.8.28;
```

- Found in contracts/Governance.sol Line: 2

```
pragma solidity ^0.8.28;
```

- Found in contracts/PeoCoin.sol Line: 2

```
pragma solidity ^0.8.28;
```

- Found in contracts/Staking.sol Line: 2

```
pragma solidity ^0.8.28;
```

- Found in contracts/interfaces/IDCS.sol Line: 2

```
pragma solidity ^0.8.28;
```

- Found in contracts/interfaces/IPeoCoin.sol Line: 2

```
pragma solidity ^0.8.28;
```

- Found in contracts/interfaces/IStaking.sol Line: 2

```
pragma solidity ^0.8.28;
```

- Found in contracts/mocks/MockStaking.sol Line: 2

```
pragma solidity ^0.8.28;
```

L-4: Missing checks for `address(0)` when assigning values to address state variables

Check for `address(0)` when assigning values to address state variables.

7 Found Instances

- Found in contracts/Conversion.sol Line: 38

```
peoToken = IPeoCoin(_peoToken);
```

- Found in contracts/Conversion.sol Line: 39

```
backendService = _backendService;
```

- Found in contracts/DCS.sol Line: 33

```
peoToken = IPeoCoin(_peoToken);
```

- Found in contracts/DCS.sol Line: 34

```
staking = IStaking(_staking);
```

- Found in contracts/Governance.sol Line: 65

```
peoToken = IPeoCoin(_peoToken);
```

- Found in contracts/Governance.sol Line: 66

```
dcs = IDCS(_dcs);
```

- Found in contracts/Staking.sol Line: 41

```
peoToken = IPeoCoin(_peoToken);
```

L-5: Define and use `constant` variables instead of using literals

If the same constant literal value is used multiple times, create a constant state variable and reference it throughout the contract.

3 Found Instances

- Found in contracts/Staking.sol Line: 76

```
uint256 baseReward = (info.amount * baseAPY * stakedDays) / (365 * 100);
```

- Found in contracts/Staking.sol Line: 79

```
uint256 bonus = (baseReward * (tierMultiplier - 100)) / 100;
```

L-6: Event is missing `indexed` fields

Index event fields make the field more quickly accessible to off-chain tools that parse events. However, note that each index field costs extra gas during emission, so it's not necessarily best to index the maximum allowed per event (three fields). Each event should use three indexed fields if there are three or more fields, and gas usage is not particularly of concern for the events in question. If there are fewer than three fields, all of the fields should be indexed.

5 Found Instances

- Found in contracts/Conversion.sol Line: 25

  ```
  event ConversionRequested(address indexed user, uint256 amount, string mobileNumber
  ```

- Found in contracts/Conversion.sol Line: 31

  ```
  event ConversionConfirmed(address indexed user, uint256 amount, string mobileNumber
  ```

- Found in contracts/Governance.sol Line: 47

  ```
  event ProposalCreated(uint256 proposalId, address proposer, string description);
  ```

- Found in contracts/Governance.sol Line: 54

  ```
  event VoteCast(uint256 proposalId, address voter, bool support, uint256 weight);
  ```

- Found in contracts/Governance.sol Line: 58

  ```
  event ProposalExecuted(uint256 proposalId);
  ```

L-7: PUSH0 is not supported by all chains

Solc compiler version 0.8.20 switches the default target EVM version to Shanghai, which means that the generated bytecode will include PUSH0 opcodes. Be sure to select the appropriate EVM version in case you intend to deploy on a chain other than mainnet like L2 chains that may not support PUSH0, otherwise deployment of your contracts will fail.

9 Found Instances

- Found in contracts/Conversion.sol Line: 2

  ```
  pragma solidity ^0.8.28;
  ```

- Found in contracts/DCS.sol Line: 2

  ```
  pragma solidity ^0.8.28;
  ```

- Found in contracts/Governance.sol Line: 2

  ```
  pragma solidity ^0.8.28;
  ```

- Found in contracts/PeoCoin.sol Line: 2

  ```
  pragma solidity ^0.8.28;
  ```

- Found in contracts/Staking.sol Line: 2

  ```
  pragma solidity ^0.8.28;
  ```

- Found in contracts/interfaces/IDCS.sol Line: 2

  ```
  pragma solidity ^0.8.28;
  ```

- Found in contracts/interfaces/IPeoCoin.sol Line: 2

  ```
  pragma solidity ^0.8.28;
  ```

- Found in contracts/interfaces/IStaking.sol Line: 2

  ```
  pragma solidity ^0.8.28;
  ```

- Found in contracts/mocks/MockStaking.sol Line: 2

  ```
  pragma solidity ^0.8.28;
  ```

L-8: Large literal values multiples of 10000 can be replaced with scientific notation

Use `e` notation, for example: `1e18`, instead of its full numeric value.

1 Found Instances

- Found in contracts/PeoCoin.sol Line: 21

  ```
  _mint(msg.sender, 1_000_000 * 10**decimals());
  ```

L-9: Potentially missing inheritance for contract.

There is an interface / abstract contract that is potentially missing (not included in) the inheritance of this contract.

2 Found Instances

- Found in contracts/Staking.sol Line: 11

  Is this contract supposed to implement an interface? Consider extending one of the following: IStaking `solidity contract Staking {`

- Found in contracts/mocks/MockStaking.sol Line: 8

  Is this contract supposed to implement an interface? Consider extending one of the following: IStaking `solidity contract MockStaking {`

L-10: Unused Imports

Redundant import statement. Consider removing it.

1 Found Instances

- Found in contracts/Conversion.sol Line: 5

```
import "./interfaces/IDCS.sol";
```

L-11: State variable could be declared constant

State variables that are not updated following deployment should be declared constant to save gas. Add the `constant` attribute to state variables that never change.

3 Found Instances

- Found in contracts/Staking.sol Line: 27

```
uint256 public baseAPY = 10;
```

- Found in contracts/Staking.sol Line: 31

```
uint256 public lockPeriod = 30 days;
```

- Found in contracts/Staking.sol Line: 35

```
uint256 public bronzeMultiplier = 100;
```

L-12: State variable changes but no event is emitted.

State variable changes in this function but no event is emitted.

4 Found Instances

- Found in contracts/DCS.sol Line: 65

```
function updateWeights(uint256 _tokenWeight, uint256 _stakeWeight, uint256 _timeWei
```

- Found in contracts/Governance.sol Line: 138

```
function updateParameters(uint256 _votingPeriod, uint256 _quorum, uint256 _majority
```

- Found in contracts/Staking.sol Line: 48

```
function stake(uint256 amount) external {
```

- Found in contracts/Staking.sol Line: 87

```
function unstake() external {
```

L-13: State variable could be declared immutable

State variables that are should be declared immutable to save gas. Add the `immutable` attribute to state variables that are only changed in the constructor

7 Found Instances

- Found in contracts/Conversion.sol Line: 14

```
IPeoCoin public peoToken;
```

- Found in contracts/Conversion.sol Line: 18

```
address public backendService;
```

- Found in contracts/DCS.sol Line: 14

```
IPeoCoin public peoToken;
```

- Found in contracts/DCS.sol Line: 17

```
IStaking public staking;
```

- Found in contracts/Governance.sol Line: 14

```
IPeoCoin public peoToken;
```

- Found in contracts/Governance.sol Line: 17

```
IDCS public dcs;
```

- Found in contracts/Staking.sol Line: 13

```
IPeoCoin public peoToken;
```