StakingRewards Audit Report

Summary

Project Metrics

- Number of lines: 1144 (+ 0 in dependencies, + 0 in tests)
- Number of assembly lines: 0
- Number of contracts: 10 (+ 0 in dependencies, + 0 tests)
- Number of optimization issues: 0
- Number of informational issues: 34
- Number of low issues: 10
- Number of medium issues: 2
- Number of high issues: 1
- ERCs: ERC20

Contract Details

Name	Functions	ERCS	ERC20 Info	Complex Code	Features
IERC20Permit	3			No	
IERC20	6	ERC20	No Minting, Approve Race Cond.	No	
Address	13			No	Send ETH, Delegatecall, Assembly
SafeERC20	7			No	Send ETH, Tokens interaction
StakingRewards	44			No	Send ETH, Tokens interaction

Note: The table above shows the details of the contracts used in the project. The `Functions` column shows the number of functions in each contract. The `ERC20 Info` column shows additional information about the ERC20 implementation used in the contract. The `Complex Code` column indicates whether the contract contains complex code. The `Features` column lists the features supported by the contract.

Contract Summary

Interface IStakingRewards

- From IStakingRewards
  - balanceOf(address) (external)
  - claim() (external)
  - earned(address) (external)
  - exit() (external)
  - getRewardForDuration() (external)
  - lastTimeRewardApplicable() (external)
  - rewardPerToken() (external)
  - stake(uint256) (external)
  - unstake(uint256) (external)

Contract StakingRewards

- From ReentrancyGuard
  - _nonReentrantAfter() (private)
  - _nonReentrantBefore() (private)
  - constructor() (internal)
- From Pausable
  - _pause() (internal)
  - _requireNotPaused() (internal)
  - _requirePaused() (internal)
  - _unpause() (internal)
  - paused() (public)
- From Context
  - _msgData() (internal)
  - _msgSender() (internal)
- From Ownable
  - _checkOwner() (internal)
  - _transferOwnership(address) (internal)
  - owner() (public)
  - renounceOwnership() (public)
  - transferOwnership(address) (public)
- From StakingRewards
  - _earned(address) (internal)
  - _lastTimeRewardApplicable() (internal)
  - _rewardPerToken() (internal)
  - balanceOf(address) (external)
  - claim() (public)
  - constructor(address,address,uint256) (public)
  - earned(address) (external)
  - exit() (public)
  - fund(uint256) (public)
  - getRewardForDuration() (external)
  - lastTimeRewardApplicable() (external)
  - recoverERC20(address,uint256) (external)
  - rewardPerToken() (external)
  - setRewardsDuration(uint256) (external)
  - stake(uint256) (public)
  - unstake(uint256) (public)

Function Summary

Interface IStakingRewards

- Contract vars: []
- Inheritance:: []

Function	Visibility	Modifiers	Read	Write	Internal Calls	External Calls
balanceOf(address)	external	[]	[]	[]	[]	[]
earned(address)	external	[]	[]	[]	[]	[]
getRewardForDuration()	external	[]	[]	[]	[]	[]
lastTimeRewardApplicable()	external	[]	[]	[]	[]	[]
rewardPerToken()	external	[]	[]	[]	[]	[]

Function	Visibility	Modifiers	Read	Write	Internal Calls	External Calls
stake(uint256)	external	[]	[]	[]	[]	[]
unstake(uint256)	external	[]	[]	[]	[]	[]
claim()	external	[]	[]	[]	[]	[]
exit()	external	[]	[]	[]	[]	[]

Modifiers	Visibility	Read	Write	Internal Calls	External Calls
[]	external	[]	[]	[]	[]

Contract StakingRewards

- Contract vars: ['_owner', '_paused', '_NOT_ENTERED', '_ENTERED', '_status', 'stakingToken', 'rewardsToken', 'periodFinish', 'rewardRate', 'rewardsDuration', 'lastUpdateTime', 'rewardPerTokenStored', 'userRewardPerTokenPaid', 'rewards', 'totalSupply', 'balances']
- Inheritance:: ['ReentrancyGuard', 'Pausable', 'Ownable', 'Context', 'IStakingRewards']

Function	Visibility	Modifiers	Read	Write	Internal Calls	External Calls
constructor()	internal	[]	['_NOT_ENTERED']	['_status']	[]	[]
_nonReentrantBefore()	private	[]	['_ENTERED', '_status']	['_status']	['require(bool,string)']	[]
_nonReentrantAfter()	private	[]	['_NOT_ENTERED']	['_status']	[]	[]
paused()	public	[]	['_paused']	[]	[]	[]
_requireNotPaused()	internal	[]	[]	[]	['paused', 'require(bool,string)']	[]
_requirePaused()	internal	[]	[]	[]	['paused', 'require(bool,string)']	[]
_pause()	internal	['whenNotPaused']	[]	['_paused']	['_msgSender', 'whenNotPaused']	[]
_unpause()	internal	['whenPaused']	[]	['_paused']	['_msgSender', 'whenPaused']	[]
_msgSender()	internal	[]	['msg.sender']	[]	[]	[]
_msgData()	internal	[]	['msg.data']	[]	[]	[]
owner()	public	[]	['_owner']	[]	[]	[]
_checkOwner()	internal	[]	[]	[]	['_msgSender', 'owner', 'require(bool,string)']	[]
renounceOwnership()	public	['onlyOwner']	[]	[]	['onlyOwner', '_transferOwnership']	[]
transferOwnership(address)	public	['onlyOwner']	[]	[]	['_transferOwnership', 'onlyOwner', 'require(bool,string)']	[]
_transferOwnership(address)	internal	[]	['_owner']	['_owner']	[]	[]
balanceOf(address)	external	[]	[]	[]	[]	[]
earned(address)	external	[]	[]	[]	[]	[]
getRewardForDuration()	external	[]	[]	[]	[]	[]
lastTimeRewardApplicable()	external	[]	[]	[]	[]	[]
rewardPerToken()	external	[]	[]	[]	[]	[]
stake(uint256)	external	['nonReentrant', 'whenNotPaused', 'updateReward']	['balances', 'stakingToken', 'totalSupply', 'msg.sender', 'this']	['balances', 'totalSupply']	['revert ZeroAmount()', 'updateReward']	['stakingToken.safeTransferFrom(msg.sender,address(this),amount)']
unstake(uint256)	external	['nonReentrant', 'whenNotPaused', 'updateReward']	['balances', 'stakingToken', 'totalSupply', 'msg.sender']	['balances', 'totalSupply']	['revert NotEnoughBalance()', 'revert ZeroAmount()', 'updateReward']	['stakingToken.safeTransfer(msg.sender,amount)']
claim()	external	['nonReentrant', 'whenNotPaused', 'updateReward']	['rewards', 'rewardsToken', 'msg.sender']	['rewards']	['updateReward', 'whenNotPaused']	['rewardsToken.safeTransfer(msg.sender,reward)']
exit()	external	['whenNotPaused']	['balances', 'msg.sender']	[]	['whenNotPaused', 'unstake', 'updateReward']	[]
fund(uint256)	public	['onlyOwner', 'whenNotPaused', 'updateReward']	['periodFinish', 'rewardRate', 'rewardsDuration', 'rewardsToken', 'block.timestamp', 'msg.sender', 'this']	['lastUpdateTime', 'periodFinish']	['updateReward', 'whenNotPaused', 'onlyOwner', 'revert TooHighReward()']	['rewardsToken.balanceOf(address(this))', 'rewardsToken.safeTransferFrom(msg.sender,address(this),reward)']
recoverERC20(address,uint256)	external	['onlyOwner']	['stakingToken']	[]	[]	[]

Modifiers	Visibility	Read	Write	Internal Calls	External Calls
				['_nonReentrantBefore',	

Modifiers	Visibility	Read	Write	Internal Calls	External Calls
nonReentrant()	internal	[]	[]	'_nonReentrantAfter']	[]
whenNotPaused()	internal	[]	[]	['_requireNotPaused']	[]
whenPaused()	internal	[]	[]	['_requirePaused']	[]
onlyOwner()	internal	[]	[]	['_checkOwner']	[]
updateReward(address)	internal	['rewardPerTokenStored']	['lastUpdateTime', 'rewardPerTokenStored']	['_earned', '_lastTimeRewardApplicable']	[]
			['rewards', 'userRewardPerTokenPaid']	['_rewardPerToken']	

Variables and Auth

Interface IStakingRewards

Function	State variables written	Conditions on msg.sender
balanceOf	[]	[]
earned	[]	[]
getRewardForDuration	[]	[]
lastTimeRewardApplicable	[]	[]
rewardPerToken	[]	[]
stake	[]	[]
unstake	[]	[]
claim	[]	[]
exit	[]	[]

Contract StakingRewards

Function	State variables written	Conditions on msg.sender
constructor	['_status']	[]
_nonReentrantBefore	['_status']	[]
_nonReentrantAfter	['_status']	[]
constructor	['_paused']	[]
paused	[]	[]
_requireNotPaused	[]	[]
_requirePaused	[]	[]
_pause	['_paused']	[]
_unpause	['_paused']	[]
_msgSender	[]	[]
_msgData	[]	[]
constructor	['_owner']	[]
owner	[]	[]
_checkOwner	[]	[]
renounceOwnership	['_owner']	[]
transferOwnership	['_owner']	[]
_transferOwnership	['_owner']	[]
balanceOf	[]	[]
earned	[]	[]
getRewardForDuration	[]	[]
lastTimeRewardApplicable	[]	[]
rewardPerToken	[]	[]
stake	[]	[]
unstake	[]	['balances[msg.sender] < amount']
claim	[]	[]
exit	[]	['balances[msg.sender] < amount']
constructor	['stakingToken', 'rewardsToken', 'rewardsDuration']	[]

Function	State variables written	Conditions on msg.sender
setRewardsDuration	['rewardsDuration']	[]
stake	['rewards', 'lastUpdateTime', 'totalSupply', '_status', 'rewardPerTokenStored', 'balances', 'userRewardPerTokenPaid']	[]
unstake	['rewards', 'lastUpdateTime', 'totalSupply', '_status', 'rewardPerTokenStored', 'balances', 'userRewardPerTokenPaid']	['balances[msg.sender] < amount']
claim	['rewards', 'lastUpdateTime', '_status', 'rewardPerTokenStored', 'userRewardPerTokenPaid']	[]
exit	['rewards', 'lastUpdateTime', 'totalSupply', '_status', 'rewardPerTokenStored', 'balances', 'userRewardPerTokenPaid']	['balances[msg.sender] < amount']
fund	['rewards', 'lastUpdateTime', 'periodFinish', 'rewardPerTokenStored', 'rewardRate', 'userRewardPerTokenPaid']	[]
recoverERC20	[]	[]
_lastTimeRewardApplicable	[]	[]
_rewardPerToken	[]	[]
_earned	[]	[]
balanceOf	[]	[]
lastTimeRewardApplicable	[]	[]
rewardPerToken	[]	[]
earned	[]	[]
getRewardForDuration	[]	[]
slitherConstructorVariables	['rewardsDuration']	[]
slitherConstructorConstantVariables	['_NOT_ENTERED', '_ENTERED']	[]

Findings

The following are the findings from the audit of the Solidity code for the StakingRewards contract:

1. Reentrancy vulnerability in StakingRewards.exit() function (lines 1020-1023): This function contains external calls to unstake() and claim() functions, which can be called recursively before the previous call completes. This can result in unexpected behavior and loss of funds. The function also writes to state variables after the external calls, which can lead to further vulnerabilities.
2. Divide-before-multiply vulnerability in StakingRewards.fund() function (lines 1032-1057): This function performs a multiplication on the result of a division, which can result in unexpected behavior and loss of funds.
3. Dangerous comparisons in StakingRewards.setRewardsDuration() function (lines 954-960) and StakingRewards.claim() function (lines 1007-1014): These functions use block.timestamp for comparisons, which can result in vulnerabilities due to the possibility of miners manipulating the timestamp.
4. Other reentrancy vulnerabilities in StakingRewards.claim() function (lines 1007-1014), StakingRewards.exit() function (lines 1020-1023), StakingRewards.fund() function (lines 1032-1057), StakingRewards.recoverERC20() function (lines 1066-1075), StakingRewards.stake() function (lines 969-979), and StakingRewards.unstake() function (lines 988-1001). These functions contain external calls that can be called recursively before the previous call completes, which can result in unexpected behavior and loss of funds.

Overall, the code contains multiple vulnerabilities that can result in unexpected behavior and loss of funds. It is recommended that the code be reviewed and updated to address these vulnerabilities.

Here are the details:

```
Reentrancy in StakingRewards.exit() (StakingRewards_Flatten.sol#1020-1023):
	External calls:
	- unstake(balances[msg.sender]) (StakingRewards_Flatten.sol#1021)
		- returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (StakingRewards_Flatten.sol#730)
		- (success,returndata) = target.call{value: value}(data) (StakingRewards_Flatten.sol#509)
		- stakingToken.safeTransfer(msg.sender,amount) (StakingRewards_Flatten.sol#999)
	- claim() (StakingRewards_Flatten.sol#1022)
		- returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (StakingRewards_Flatten.sol#730)
		- rewardsToken.safeTransfer(msg.sender,reward) (StakingRewards_Flatten.sol#1011)
		- (success,returndata) = target.call{value: value}(data) (StakingRewards_Flatten.sol#509)
	External calls sending eth:
	- unstake(balances[msg.sender]) (StakingRewards_Flatten.sol#1021)
		- (success,returndata) = target.call{value: value}(data) (StakingRewards_Flatten.sol#509)
	- claim() (StakingRewards_Flatten.sol#1022)
		- (success,returndata) = target.call{value: value}(data) (StakingRewards_Flatten.sol#509)
	State variables written after the call(s):
	- claim() (StakingRewards_Flatten.sol#1022)
		- _status = _NOT_ENTERED (StakingRewards_Flatten.sol#807)
		- _status = _ENTERED (StakingRewards_Flatten.sol#801)
	- claim() (StakingRewards_Flatten.sol#1022)
		- lastUpdateTime = _lastTimeRewardApplicable() (StakingRewards_Flatten.sol#924)
	- claim() (StakingRewards_Flatten.sol#1022)
		- rewardPerTokenStored = _rewardPerToken() (StakingRewards_Flatten.sol#923)
	- claim() (StakingRewards_Flatten.sol#1022)
		- rewards[msg.sender] = 0 (StakingRewards_Flatten.sol#1010)
		- rewards[account] = _earned(account) (StakingRewards_Flatten.sol#926)
	- claim() (StakingRewards_Flatten.sol#1022)
		- userRewardPerTokenPaid[account] = rewardPerTokenStored (StakingRewards_Flatten.sol#927)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities

StakingRewards.fund(uint256) (StakingRewards_Flatten.sol#1032-1057) performs a multiplication on the result of a division:
	- rewardRate = reward / rewardsDuration (StakingRewards_Flatten.sol#1036)
	- leftover = remaining * rewardRate (StakingRewards_Flatten.sol#1039)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-multiply

Reentrancy in StakingRewards.fund(uint256) (StakingRewards_Flatten.sol#1032-1057):
	External calls:
	- rewardsToken.safeTransferFrom(msg.sender,address(this),reward) (StakingRewards_Flatten.sol#1047)
	State variables written after the call(s):
	- lastUpdateTime = block.timestamp (StakingRewards_Flatten.sol#1053)
	- periodFinish = block.timestamp + rewardsDuration (StakingRewards_Flatten.sol#1054)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1

Reentrancy in StakingRewards.claim() (StakingRewards_Flatten.sol#1007-1014):
	External calls:
	- rewardsToken.safeTransfer(msg.sender,reward) (StakingRewards_Flatten.sol#1011)
	Event emitted after the call(s):
	- Claimed(msg.sender,reward) (StakingRewards_Flatten.sol#1012)
Reentrancy in StakingRewards.exit() (StakingRewards_Flatten.sol#1020-1023):
	External calls:
	- unstake(balances[msg.sender]) (StakingRewards_Flatten.sol#1021)
		- returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (StakingRewards_Flatten.sol#730)
		- (success,returndata) = target.call{value: value}(data) (StakingRewards_Flatten.sol#509)
		- stakingToken.safeTransfer(msg.sender,amount) (StakingRewards_Flatten.sol#999)
	- claim() (StakingRewards_Flatten.sol#1022)
		- returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (StakingRewards_Flatten.sol#730)
		- rewardsToken.safeTransfer(msg.sender,reward) (StakingRewards_Flatten.sol#1011)
		- (success,returndata) = target.call{value: value}(data) (StakingRewards_Flatten.sol#509)
	External calls sending eth:
	- unstake(balances[msg.sender]) (StakingRewards_Flatten.sol#1021)
		- (success,returndata) = target.call{value: value}(data) (StakingRewards_Flatten.sol#509)
```

```
    - claim() (StakingRewards_Flatten.sol#1022)
        - (success,returndata) = target.call{value: value}(data) (StakingRewards_Flatten.sol#509)
    Event emitted after the call(s):
    - Claimed(msg.sender,reward) (StakingRewards_Flatten.sol#1012)
        - claim() (StakingRewards_Flatten.sol#1022)
Reentrancy in StakingRewards.fund(uint256) (StakingRewards_Flatten.sol#1032-1057):
    External calls:
    - rewardsToken.safeTransferFrom(msg.sender,address(this),reward) (StakingRewards_Flatten.sol#1047)
    Event emitted after the call(s):
    - Funded(reward) (StakingRewards_Flatten.sol#1056)
Reentrancy in StakingRewards.recoverERC20(address,uint256) (StakingRewards_Flatten.sol#1066-1075):
    External calls:
    - IERC20(tokenAddress).safeTransfer(owner(),tokenAmount) (StakingRewards_Flatten.sol#1073)
    Event emitted after the call(s):
    - Recovered(tokenAddress,tokenAmount) (StakingRewards_Flatten.sol#1074)
Reentrancy in StakingRewards.stake(uint256) (StakingRewards_Flatten.sol#969-979):
    External calls:
    - stakingToken.safeTransferFrom(msg.sender,address(this),amount) (StakingRewards_Flatten.sol#977)
    Event emitted after the call(s):
    - Staked(msg.sender,amount) (StakingRewards_Flatten.sol#978)
Reentrancy in StakingRewards.unstake(uint256) (StakingRewards_Flatten.sol#988-1001):
    External calls:
    - stakingToken.safeTransfer(msg.sender,amount) (StakingRewards_Flatten.sol#999)
    Event emitted after the call(s):
    - Unstaked(msg.sender,amount) (StakingRewards_Flatten.sol#1000)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

StakingRewards.setRewardsDuration(uint256) (StakingRewards_Flatten.sol#954-960) uses timestamp for comparisons
    Dangerous comparisons:
    - block.timestamp < periodFinish (StakingRewards_Flatten.sol#955)
StakingRewards.claim() (StakingRewards_Flatten.sol#1007-1014) uses timestamp for comparisons
    Dangerous comparisons:
    - reward > 0 (StakingRewards_Flatten.sol#1009)
StakingRewards.fund(uint256) (StakingRewards_Flatten.sol#1032-1057) uses timestamp for comparisons
    Dangerous comparisons:
    - block.timestamp >= periodFinish (StakingRewards_Flatten.sol#1035)
    - rewardRate > balance / rewardsDuration (StakingRewards_Flatten.sol#1049)
StakingRewards._lastTimeRewardApplicable() (StakingRewards_Flatten.sol#1081-1083) uses timestamp for comparisons
    Dangerous comparisons:
    - block.timestamp < periodFinish (StakingRewards_Flatten.sol#1082)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp

Different versions of Solidity are used:
    - Version used: ['^0.8.0', '^0.8.1']
    - ^0.8.0 (StakingRewards_Flatten.sol#8)
    - ^0.8.0 (StakingRewards_Flatten.sol#36)
    - ^0.8.0 (StakingRewards_Flatten.sol#121)
    - ^0.8.0 (StakingRewards_Flatten.sol#228)
    - ^0.8.0 (StakingRewards_Flatten.sol#292)
    - ^0.8.1 (StakingRewards_Flatten.sol#378)
    - ^0.8.0 (StakingRewards_Flatten.sol#626)
    - ^0.8.0 (StakingRewards_Flatten.sol#744)
    - ^0.8.0 (StakingRewards_Flatten.sol#815)
    - ^0.8.0 (StakingRewards_Flatten.sol#841)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used
```

Note: Minimum solidity version should be 0.8.9 for StakingRewards.sol.