Audit Report

Valon Staking

January 2022

Type	BEP20
Network	BSC
Address	0x050026BC64dCfED18FcB3D57923827EcF11f3208
Audited by	© coinscope

Table of Contents

Table of Contents 1
Contract Review 3
Audit Updates 3
Contract Analysis 4
Contract Owner Can Disable the Rewards 4
Description 4
Recommendation 4
Contract Owner Can Disable the Unstaking 5
Description 5
Recommendation 5
Rewards Amount Consideration 6
Description 6
Recommendation 6
Contract Diagnostics 7
CR - Code Repetition 8
Description 8
Recommendation 8
L01 - Public Function could be Declared External 9
Description 9
Recommendation 9
L09 - Dead Code Elimination 10
Description 10
Recommendation 10
L11 - Unnecessary Boolean equality 11
Description 11
Recommendation 11

L07 - Missing Events Arithmetic 12
Description 12
Recommendation 12
Contract Functions 13
Contract Flow 17
Summary 18
Disclaimer 19
About Coinscope 20

Contract Review

Contract Name	ValonStaking
Compiler Version	v0.8.7+commit.e28d00a7
Optimization	200 runs
Licence	None
Explorer	https://bscscan.com/address/0x050026BC64dCfED18FcB3D57923827EcF11f3208
Source	contract.sol

Audit Updates

Initial Audit	30th January 2022
Corrected	

Contract Analysis

Contract Owner Can Disable the Rewards

Criticality	minor
Location	contract.sol#L1177

Description

The contract owner has the authority to disable the rewards for the users. The owner may take advantage of it by setting the *_rewardCap* to zero.

```
function claimRewards(address poolAddress) public nonReentrant {
    uint256 actualRewards = this.getActualRewards(poolAddress, msg.sender);
    require(actualRewards > 0, "BEP20: no rewards available");
    require(_totalRewards < _rewardCap, "BEP20: Reward cap reached");
    ...
}
```

Recommendation

The owner should carefully manage the credentials of the owner's account. We advised considering an extra-strong security mechanism that the actions may be quarantined by many users instead of one.

Contract Owner Can Disable the Unstaking

Criticality	minor
Location	contract.sol#L1205

Description

The contract owner has the authority to disable the unstacking for the users. The owner may take advantage of it by setting the *_timelockHeight* to a high value.

```
function removeStake(address poolAddress, uint256 lptAmount) public nonReentrant
{
    require(lptAmount > 0, "BEP20: amount needs to be more than 0");
    require(lptAmount <= this.getStake(poolAddress, msg.sender), "BEP20: amount
more than staking balance");
    if (_timelockActive) {
        uint256 blocks =
block.number.sub(blockHeights[poolAddress][msg.sender]);
        require(blocks >= _timelockHeight, "BEP20: Timelock still active");
    }
...
```

Recommendation

The owner should carefully manage the credentials of the owner's account. We advised considering an extra-strong security mechanism that the actions may be quarantined by many users instead of one.

Rewards Amount Consideration

Criticality	minor
Location	contract.sol#L1232

Description

Users can stack liquidity pool tokens in the pool. As a reward, they get Valon tokens. Currently there is no guarantee that the rewards are enough for the claimers. During the pool creation, the contract owner could also transfer Valon tokens in the stacking contract. That way, it will guarantee that the rewards will be sufficient for the users.

```
function claimRewards(address poolAddress) public nonReentrant {
    ...
    require(valon.transfer(msg.sender, reward), "Failed to transfer rewards");
}
```

Recommendation

The contract could transfer Valon tokens to the stacking contract during the pool creation. That way it will be more transparent, and there will always be funds to cover the user’s rewards.

Contract Diagnostics

Critical Medium Minor

Severity	Code	Description
	CR	Code Repetition
	L01	Public Function could be Declared External
	L09	Dead Code Elimination
	L11	Unnecessary Boolean equality
	L07	Missing Events Arithmetic

CR - Code Repetition

Criticality	minor
Location	contract.sol#L1177

Description

There are code segments that are repetitive in the contract. Those segments increase the code size of the contract unnecessarily.

```
function _updateStakeHolderCount(address poolAddress, address stakeHolder, bool
addingStake) private {
    if (addingStake) {
        if (getStake(poolAddress, stakeHolder) == 0) {
            pools[poolAddress].stakeHolderCount =
pools[poolAddress].stakeHolderCount.add(1);
        }
    } else {
        if (getStake(poolAddress, stakeHolder) == 0) {
            pools[poolAddress].stakeHolderCount =
pools[poolAddress].stakeHolderCount.sub(1);
        }
    }
}
```

Recommendation

Create an internal function that contains the code segment and remove it from all the sections.

Suggestion:

```
function _updateStakeHolderCount(address poolAddress, address stakeHolder, bool
addingStake) private {
    if (getStake(poolAddress, stakeHolder) > 0) return

    pools[poolAddress].stakeHolderCount = addingStake ?
pools[poolAddress].stakeHolderCount.add(1) :
pools[poolAddress].stakeHolderCount.sub(1)
}
```

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L1368,L1364,L1357 and 25 more

Description

Public functions that are never called by the contract should be declared external to save gas.

```
setDifficulty
setTotalRewards
setSharePoolRewards
...
```

Recommendation

Use the external attribute for functions never called from the contract

L09 - Dead Code Elimination

Criticality	minor
Location	contract.sol#L1065,L1060,L1025 and 19 more

Description

Functions that are not used in the contract, and make the code's size bigger.

```
_to18Digits
_burnFrom
_burn
...
```

Recommendation

Remove unused functions.

L11 - Unnecessary Boolean equality

Criticality	minor
Location	contract.sol#L1189

Description

The comparison to boolean constants is redundant. Boolean constants can be used directly and do not need to be compared to true or false.

```
require(bool,string)(pools[poolAddress].active == true,BEP20: pool is not
active)
```

Recommendation

Remove the equality to the boolean constant.

L07 - Missing Events Arithmetic

Criticality	minor
Location	contract.sol#L1368,L1364,L1349 and 2 more

Description

Detected missing events for critical arithmetic parameters. There are functions that have no event emitted, so it is difficult to track off-chain changes.

```
_difficulty = difficulty
_totalRewards = totalRewards
_rewardCap = amountEth
...
```

Recommendation

Emit an event for critical parameter changes.

Contract Functions

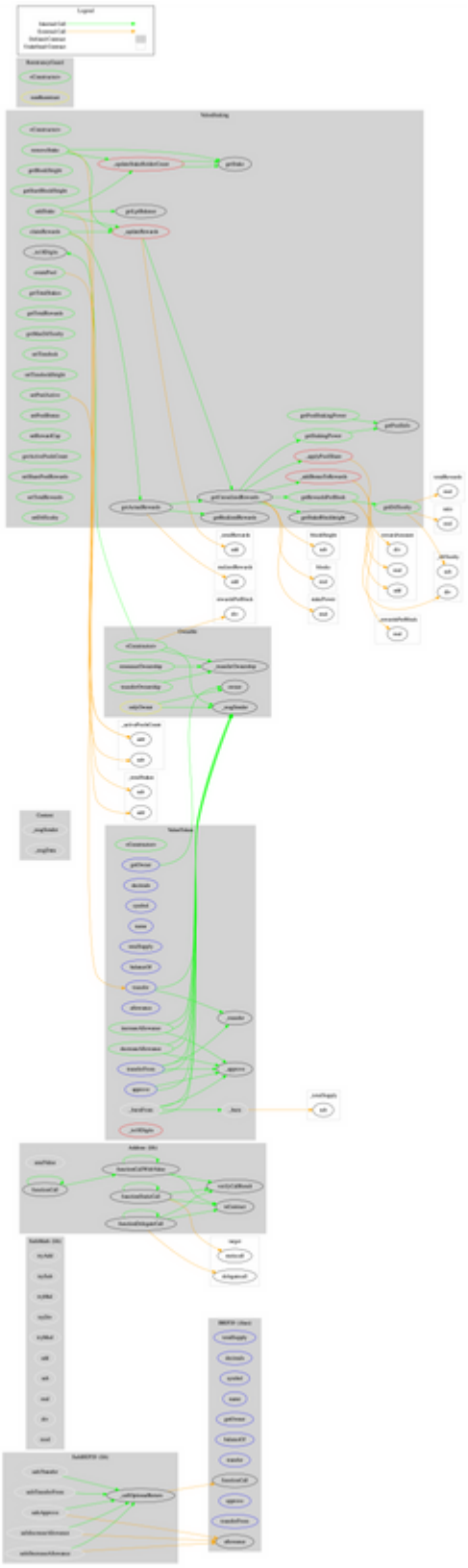
Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
IBEP20	Interface			
	totalSupply	External		-
	decimals	External		-
	symbol	External		-
	name	External		-
	getOwner	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
SafeMath	Library			
	tryAdd	Internal		
	trySub	Internal		
	tryMul	Internal		
	tryDiv	Internal		
	tryMod	Internal		
	add	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	mod	Internal		
	sub	Internal		
	div	Internal		
	mod	Internal		
Address	Library			
	isContract	Internal		

	sendValue	Internal	✓	
	functionCall	Internal	✓	
	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionStaticCall	Internal		
	functionStaticCall	Internal		
	functionDelegateCall	Internal	✓	
	functionDelegateCall	Internal	✓	
	verifyCallResult	Internal		
SafeBEP20	Library			
	safeTransfer	Internal	✓	
	safeTransferFrom	Internal	✓	
	safeApprove	Internal	✓	
	safeIncreaseAllowance	Internal	✓	
	safeDecreaseAllowance	Internal	✓	
	_callOptionalReturn	Private	✓	
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
Ownable	Implementation	Context		
	<Constructor>	Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_transferOwnership	Internal	✓	
ReentrancyGuard	Implementation			
	<Constructor>	Public	✓	-
ValonToken	Implementation	Context, IBEP20,		

		Ownable		
	<Constructor>	Public	✓	-
	getOwner	External		-
	decimals	External		-
	symbol	External		-
	name	External		-
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	_transfer	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_burnFrom	Internal	✓	
	_to18Digits	Private		
ValonStaking	Implementation	Ownable, ReentrancyGuard		
	<Constructor>	Public	✓	-
	createPool	Public	✓	onlyOwner
	getPoolInfo	Public		-
	getBlockHeight	Public		-
	getStartBlockHeight	Public		-
	_updateRewards	Private	✓	
	_addBonusToRewards	Private		
	_applyPoolShare	Private		
	_updateStakeHolderCount	Private	✓	
	addStake	Public	✓	nonReentrant
	removeStake	Public	✓	nonReentrant
	claimRewards	Public	✓	nonReentrant
	_to18Digits	Private		

	getStake	Public		-
	getLptBalance	Public		-
	getStakeBlockheight	Public		-
	getTotalStakes	Public		-
	getTotalRewards	Public		-
	getRealizedRewards	Public		-
	getMaxDifficulty	Public		-
	getDifficulty	Public		-
	getPoolStakingPower	Public		-
	getRewardsPerBlock	Public		-
	getStakingPower	Public		-
	getUnrealizedRewards	Public		-
	getActualRewards	Public		-
	setTimelock	Public	✓	onlyOwner
	setTimelockHeight	Public	✓	onlyOwner
	setPoolActive	Public	✓	onlyOwner
	setPoolBonus	Public	✓	onlyOwner
	setRewardCap	Public	✓	onlyOwner
	getActivePoolsCount	Public		-
	setSharePoolRewards	Public	✓	onlyOwner
	setTotalRewards	Public	✓	onlyOwner
	setDifficulty	Public	✓	onlyOwner

Contract Flow



Summary

Valon Staking providers stacking for multiple LP tokens. Every LP token is a different pool in the contract. Users can deposit LP tokens for the corresponding pool. As a reward they gain Velon tokens. The rewards are calculated according to the stacking period and the supplied amount. The audit remarks some optimizations and security concerns.

Disclaimer

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment.

Coinscope team provides no guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document. Always Do your own research and protect yourselves from being scammed.

The Coinscope team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools. Under no circumstances did Coinscope receive a payment to manipulate those results or change the awarding badge that we will be adding in our website.

Always Do your own research and protect yourselves from scams. This document should not be presented as a reason to buy or not buy any particular token.

The Coinscope team disclaims any liability for the resulting losses.

About Coinscope

Coinscope is the leading early coin listing, voting and auditing authority firm. The audit process is analyzing and monitoring many aspects of the project. That way, it gives the community a good sense of security using an informative report and a generic score.

Coinscope is aiming to make crypto discoverable and efficient globally. It provides all the essential tools to assist users draw their own conclusions.

The Coinscope.co team

https://www.coinscope.co