Cyberscope

Audit Report

RBX Staking

June 2022

Type	BEP20
Network	BSC
Address	0x2A2Ab66a3a1269d1C0D0469B99E732bd8dB9d34F
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	RocketDropV1point8
Compiler Version	v0.8.13+commit.abaa5c0e
Optimization	99999 runs
Licence	
Explorer	https://bscscan.com/token/0x2A2Ab66a3a1269d1C0D0469B99E732bd8dB9d34F

Audit Updates

Initial Audit	20th May 2022
Corrected	4th June 2022

Source Files

Filename	SHA256
@openzeppelin/contracts/access/Ownable.sol	75e3c97011e75627ffb36f4a2799a4e887e1a3e27ed427490e82d7b6f51cc5c9
@openzeppelin/contracts/token/ERC20/IERC20.sol	94f23e4af51a18c2269b355b8c7cf4db8003d075c9c541019eb8dcf4122864d5
@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol	b5a1340c5232f387b15592574f27eef78f6017bdc66542a1cea512ad4f78a0d2
@openzeppelin/contracts/utils/Address.sol	aafa8f3e41700a8353aabcdf020e06735753e6bc4b615279b43de53cfbb4f2cd
@openzeppelin/contracts/utils/Context.sol	1458c260d010a08e4c20a4a517882259a23a4baa0b5bd9add9fb6d6a1549814a
@openzeppelin/contracts/utils/structs/EnumerableSet.sol	000b9ea0423e2384130d16e211a96a83ad0ad0f65622ed14b6650cf707a2d41d
contracts/AuditRefactor_RocketDrop.sol	bc8dde7f2affec08d5d346baddbfe83a6d16f3de81ba3a99350c6ceb23986a0e

Contract Diagnostics

🔴 Critical 🟠 Medium ● Minor

Severity	Code	Description
●	DSI	Data Structure Improvement
●	EUP	Execution on Uninitialized Pools
●	CO	Code Optimization
●	CR	Code Repetition
🟠	MC	Missing Check
●	L01	Public Function could be Declared External
●	L02	State Variables could be Declared Constant
●	L04	Conformance to Solidity Naming Conventions
●	L09	Dead Code Elimination

DSI - Data Structure Improvement

Criticality	minor
Location	contract.sol

Description

The poolInfo contains information regarding each staking pool indexed by the pid. The userInfo contains the user stake information indexed by the users address that is indexed by the pid. All the methods that access the userInfo, are accessing the poolInfo as well. Hence, the contract is keeping up to date two data structures with the same indexes.

```
// Info of each pool.
PoolInfo[] public poolInfo;
PoolExtras[] public poolExtras;

// Info of each user that stakes LP tokens.
mapping (uint256 => mapping (address => UserInfo)) public userInfo;
```

Recommendation

The contract could embed the userInfo mapping inside the poolInfo structure so there is no need for keeping up to date two indexes for data structures.

```
struct PoolInfo {
    IERC20 lpToken;
    uint256 lastRewardBlock;
    uint256 accERC20PerShare;
    IERC20 rewardToken;
    uint256 startBlock;
    uint256 endBlock;
    uint256 rewardPerBlock;
    uint256 paidOut;
    uint256 tokensStaked;
    uint256 gasAmount;
    uint256 minStake;
    uint256 maxStake;
    address payable partnerTreasury;
    uint256 partnerPercent;
    mapping (address => UserInfo) userInfo;
}
```

EUP - Execution on Uninitialized Pools

Criticality	minor
Location	contract.sol

Description

The users have the authority to call methods with pid that contain indexes that have not been initialized yet.

```
function emergencyWithdraw(uint256 _pid) external {
    PoolInfo storage pool = poolInfo[_pid];
    UserInfo storage user = userInfo[_pid][msg.sender];

    uint256 staked = user.amount;

    pool.tokensStaked -= staked;
    poolExtras[_pid].totalStakers--;

    user.amount = 0;
    user.rewardDebt = 0;

    pool.stakeToken.safeTransfer(address(msg.sender), staked);
    emit EmergencyWithdraw(msg.sender, _pid, staked);
}
```

Recommendation

All the methods that accept the pid as parameter should initially check if the pid is less than the active pool's length.

CR - Code Repetition

Criticality	minor
Location	contract.sol

Description

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

```
IERC20 erc20 = pool.rewardToken;

uint256 startTokenBalance = erc20.balanceOf(address(this));
erc20.safeTransferFrom(address(msg.sender), address(this), _amount);
uint256 trueDepositedTo
```

Recommendation

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

MC - Missing Check

Criticality	medium
Location	contract.sol

Description

Normal Value Checks

The contract should check if the configured values may exploit the calculation results. For instance, if the result of `(poolEx.stakeTokenFee * endTokenBalance) / DIVISOR;` is greater than the `endTokenBalance` value, the expressions `endTokenBalance - startTokenBalance - depositFee;` will underflow.

Maximum Value Exceed

If variables like the `lockPeriod`, `gasAmount` set to a high value, the users will not be able to withdraw their rewards.

If the `poolEx.accessToken` is set to the dead address, then the user’s balance calculation will exploit.

```
if(poolEx.accessTokenRequired){
    require(poolEx.accessToken.balanceOf(msg.sender) >= poolEx.accessTokenMin,
'Must have minimum amount of access token!');
}
```

Recommendation

The contract should properly check the variables according to the required specifications

L01 - Public Function could be Declared External

Criticality	minor
Location	@openzeppelin/contracts/access/Ownable.sol#L54,62

Description

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

```
transferOwnership
renounceOwnership
```

Recommendation

Use the external attribute for functions never called from the contract.

L02 - State Variables could be Declared Constant

Criticality	minor
Location	contracts/AuditRefactor_RocketDrop.sol#L100

Description

Constant state variables should be declared constant to save gas.

```
DIVISOR
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contracts/AuditRefactor_RocketDrop.sol#L132,133,134,157,177,178,179,180,219,220,221,230,234,238,245,252,259,266,273,283,292,318,339,363,424,463,481,482,483,495,499,506,510,518,519,520,526,100

Description

Solidity defines a naming convention that should be followed. Rule exceptions:

- Allow constant variable name/symbol/decimals to be lowercase.
- Allow _ at the beginning of the mixed_case match for private variables and unused parameters.

```
DIVISOR
_newTreasury
_amount
_ERC20address
_recipient
_newBlock
_pid
_newReward
_newgas
...
```

Recommendation

Follow the Solidity naming convention.
https://docs.soliditylang.org/en/v0.4.25/style-guide.html#naming-conventions

L09 - Dead Code Elimination

Criticality	minor
Location	@openzeppelin/contracts/utils/Address.sol#L85,114,174,184,147,157,60 @openzeppelin/contracts/utils/structs/EnumerableSet.sol#L54,130,109,116,72,142,262,196,335,241,175,314,248,182,321,234,168,307,274,208,347 @openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#L45,69,60

Description

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

```
safeIncreaseAllowance
safeDecreaseAllowance
safeApprove
values
remove
length
contains
at
_values
...
```

Recommendation

Remove unused functions.

Contract Functions

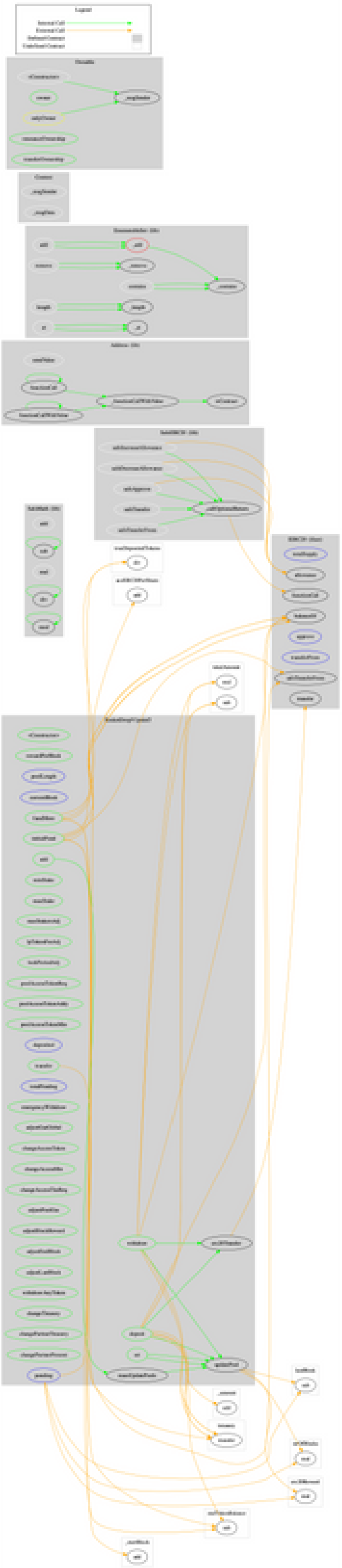
Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
Ownable	Implementation	Context		
	<Constructor>	Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_transferOwnership	Internal	✓	
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
SafeERC20	Library			
	safeTransfer	Internal	✓	
	safeTransferFrom	Internal	✓	
	safeApprove	Internal	✓	
	safeIncreaseAllowance	Internal	✓	
	safeDecreaseAllowance	Internal	✓	
	_callOptionalReturn	Private	✓	
Address	Library			
	isContract	Internal		
	sendValue	Internal	✓	
	functionCall	Internal	✓	

	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionStaticCall	Internal		
	functionStaticCall	Internal		
	functionDelegateCall	Internal	✓	
	functionDelegateCall	Internal	✓	
	verifyCallResult	Internal		
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
EnumerableSet	Library			
	_add	Private	✓	
	_remove	Private	✓	
	_contains	Private		
	_length	Private		
	_at	Private		
	_values	Private		
	add	Internal	✓	
	remove	Internal	✓	
	contains	Internal		
	length	Internal		
	at	Internal		
	values	Internal		
	add	Internal	✓	
	remove	Internal	✓	
	contains	Internal		
	length	Internal		
	at	Internal		
	values	Internal		
	add	Internal	✓	
	remove	Internal	✓	
	contains	Internal		

	length	Internal		
	at	Internal		
	values	Internal		
RocketDropV1point8	Implementation	Ownable		
	<Constructor>	Public	✓	-
	rewardPerBlock	External		-
	poolLength	External		-
	currentBlock	External		-
	initialFund	External	✓	-
	fundMore	External	✓	-
	add	External	✓	onlyOwner
	set	External	✓	onlyOwner
	minStake	External	✓	onlyOwner
	maxStake	External	✓	onlyOwner
	maxStakersAdj	External	✓	onlyOwner
	stakeTokenFeeAdj	External	✓	onlyOwner
	lockPeriodAdj	External	✓	onlyOwner
	poolAccessTokenReq	External	✓	onlyOwner
	poolAccessTokenAddy	External	✓	onlyOwner
	poolAccessTokenMin	External	✓	onlyOwner
	deposited	External		-
	pending	External		-
	totalPending	External		-
	massUpdatePools	Public	✓	-
	updatePool	Public	✓	-
	deposit	External	Payable	-
	withdraw	External	Payable	-
	emergencyWithdraw	External	✓	-
	erc20Transfer	Internal	✓	
	adjustGasGlobal	External	✓	onlyOwner
	adjustPoolGas	External	✓	onlyOwner
	adjustBlockReward	External	✓	onlyOwner
	adjustEndBlock	External	✓	onlyOwner
	adjustLastBlock	External	✓	onlyOwner

	withdrawAnyToken	External	✓	onlyOwner
	changeTreasury	External	✓	onlyOwner
	transfer	External	✓	onlyOwner

Contract Flow



Summary

The contract implements a staking functionality. Users have the ability to deposit an amount and receive rewards proportional to the time that has elapsed. This audit focuses on the business logic implementation, the security concerns and some potential performance improvements.

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.

Cyberscope 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 Cyberscope 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 Cyberscope 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 Cyberscope team disclaims any liability for the resulting losses.

About Cyberscope

Coinscope audit and K.Y.C. service has been rebranded to Cyberscope.

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.

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

The Cyberscope team

https://www.cyberscope.io