Cyberscope

Audit Report

Quarashi Staking BSC

April 2022

Type	BEP20
Network	BSC
Address	0xee7b65e341de03621964c0f2cdaee78690e2cee9
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	QuaStaking
Compiler Version	v0.8.11+commit.d7f03943
Optimization	200000 runs
Licence	MIT
Explorer	https://bscscan.com/token/0xee7b65e341de03621964c0f2cdaee78690e2cee9

Source Files

Filename	**SHA256**
contract.sol	44757d01b05df3de640fa3381690fec00ea926a303367b2d8d2ee182304f313c

Audit Updates

Initial Audit	13th April 2022
Corrected	

Contract Analysis

The contract implements a basic staking feature. The users have the ability to deposit tokens to three different pools. Each pool provides a different combination of A.P.Y. (Annual Percentage Yield), locking period and commission. The commission is only applied if the user withdraws the tokens earlier than the locking period.

Pools

The pool options are 3 and cannot be changed.

Pool Id	A.P.Y. (percentage)	Locking Period (months)	Commission (percentage)
0	0.0055	1	0.01
1	0.0125	6	0.03
3	0.028	12	0.08

Reward calculation

The APY percentage is added every month to the previous month's APY. So for instance, if a user stake 10000 tokens in the pool id 1, then the withdrawn amount after 6 months will be 10773.9. As a result the APY does not work as an annual percentage but as an accumulated monthly percentage.

Early Withdraw

The depositors have the ability to withdraw the tokens earlier than the locking period. As a result the depositor will receive the APY percentage proportional to the time that has been elapsed. Additionally, the depositor will be taxed with a commission amount. The commission amount is calculated based on the initial deposit, not in the awarded amount.

Contract Owner privileges

- The Admin role is renounced
- The Admin role has the ability to set the commission address
- The Admin role has the ability to withdraw the contract’s excessed tokens.

Deposit Info Id Event Emit

Criticality	minor
Location	contract.sol#L591,598

Description

Since the TokensStaked and Withdraw() are based on the user's deposit info index, it would be more informative to emit the depositInfoId number in the event as well.

Recommendation

The depositInfoId could be emitted in the events.

Minimum Deposit Amount

Criticality	minor
Location	contract.sol#L666

Description

The calculation of award amount is a production of division. Hence, there is a minimum amount that the division will return zero.

The minimum amount are:

- if a user deposits 181 tokens in the pool id 1, then the awards amount will be zero.
- if a user deposits 79 tokens in the pool id 2, then the awards amount will be zero.
- if a user deposits 35 tokens in the pool id 3, then the awards amount will be zero.

```
_maxUnstakeAmount * pools[_poolId].APY / PERSENT_BASE;
```

Recommendation

The contract could have a minimum amount check, so it is guaranteed that all the depositors will receive rewards.

Contract Diagnostics

Critical Medium Minor

Severity	Code	Description
	L01	Public Function could be Declared External
	L04	Conformance to Solidity Naming Conventions
	L09	Dead Code Elimination
	L13	Divide before Multiply Operation
	L14	Uninitialized Variables in Local Scope

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L463,476,494

Description

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

```
renounceRole
revokeRole
grantRole
```

Recommendation

Use the external attribute for functions never called from the contract

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L642,657,697,753,784,785

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.

```
_depositInfoId
_user
_poolId
_commissionAddress
```

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	contract.sol#L527,248,223

Description

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

```
toString
toHexString
_setRoleAdmin
```

Recommendation

Remove unused functions.

L13 - Divide before Multiply Operation

Criticality	minor
Location	contract.sol#L783

Description

Performing divisions before multiplications may cause lose of prediction.

```
stakingDays = (block.timestamp - deposit.start) % MONTH / DAY
```

Recommendation

The multiplications should be prior to the divisions.

L14 - Uninitialized Variables in Local Scope

Criticality	minor
Location	contract.sol#L665,764,800,633,806

Description

The are variables that are defined in the local scope and are not initialized.

```
i
commissionAmount
```

Recommendation

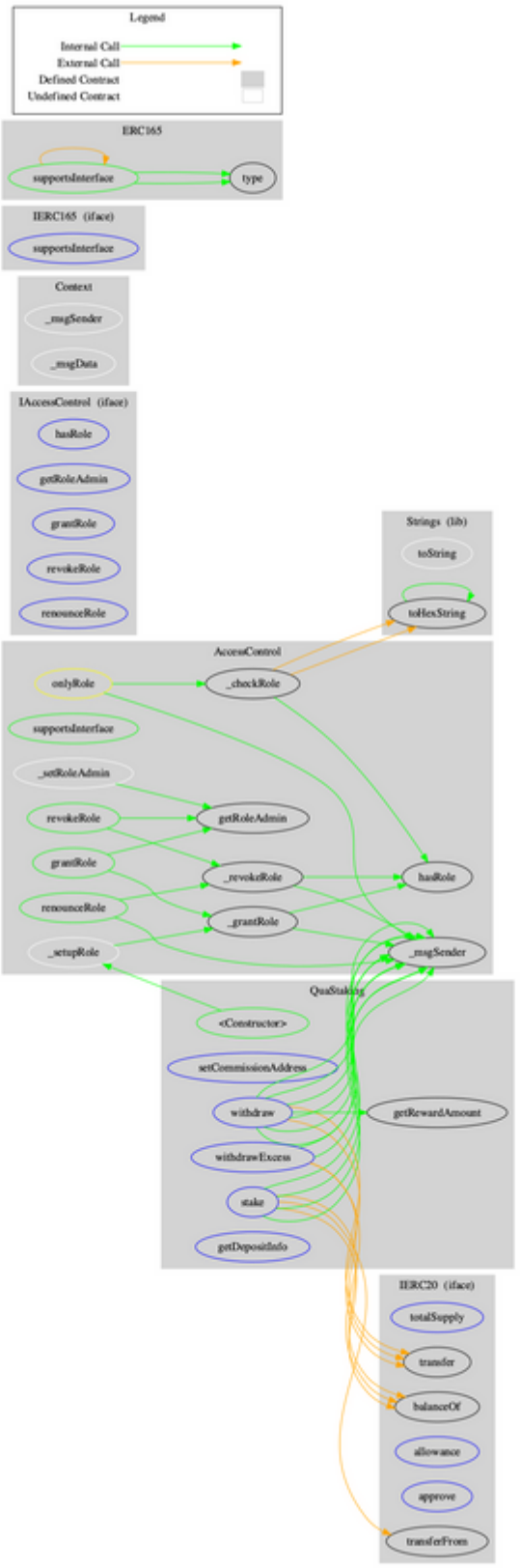
All the local scoped variables should be initialized.

Contract Functions

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
IAccessControl	Interface			
	hasRole	External		-
	getRoleAdmin	External		-
	grantRole	External	✓	-
	revokeRole	External	✓	-
	renounceRole	External	✓	-
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
Strings	Library			
	toString	Internal		
	toHexString	Internal		
	toHexString	Internal		
IERC165	Interface			
	supportsInterface	External		-
ERC165	Implementation	IERC165		

	supportsInterface	Public		-
AccessControl	Implementation	Context, IAccessControl, ERC165		
	supportsInterface	Public		-
	hasRole	Public		-
	_checkRole	Internal		
	getRoleAdmin	Public		-
	grantRole	Public	✓	onlyRole
	revokeRole	Public	✓	onlyRole
	renounceRole	Public	✓	-
	_setupRole	Internal	✓	
	_setRoleAdmin	Internal	✓	
	_grantRole	Internal	✓	
	_revokeRole	Internal	✓	
QuaStaking	Implementation	AccessControl		
	<Constructor>	Public	✓	-
	setCommissionAddress	External	✓	onlyRole
	stake	External	✓	-
	withdraw	External	✓	-
	withdrawExcess	External	✓	onlyRole
	getDepositInfo	External		-
	getRewardAmount	Public		-

Contract Flow



Summary

Quarashi Staking is a typical implementation of staking functionality. The users have the ability to stake tokens and get the rewards once the locked period has elapsed. This audit focuses on the business logic and potential optimizations.

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