Cyberscope

Audit Report

Nutgain

March 2022

Type	BEP20
Network	BSC
Address	0xb149b030cfa47880af0bde4cd36539e4c928b3eb
Audited by	© cyberscope

Table of Contents

Table of Contents 1
Contract Review 3
Audit Updates 3
Contract Analysis 4
ST - Stop Transactions 5
Description 5
Recommendation 5
ELFM - Exceed Limit Fees Manipulation 6
Description 6
Recommendation 6
BC - Blacklisted Contracts 7
Description 7
Recommendation 7
Contract Diagnostics 8
L01 - Public Function could be Declared External 9
Description 9
Recommendation 9
L02 - State Variables could be Declared Constant 10
Description 10
Recommendation 10
L04 - Conformance to Solidity Naming Conventions 11
Description 11
Recommendation 11
L11 - Unnecessary Boolean equality 12
Description 12
Recommendation 12

L07 - Missing Events Arithmetic 13
Description 13
Recommendation 13
L13 - Divide before Multiply Operation 14
Description 14
Recommendation 14
Contract Functions 15
Contract Flow 18
Domain Info 19
Summary 20
Disclaimer 21
About Cyberscope 22

Contract Review

Contract Name	NUTGAIN
Compiler Version	v0.8.7+commit.e28d00a7
Optimization	200 runs
Licence	Unlicense
Explorer	https://bscscan.com/token/0xb149b030cfa47880af0bde4cd36539e4c928b3eb
Symbol	NUTGV2
Decimals	9
Total Supply	1,500,000,000
Source	contract.sol
Domain	nutgain.io

Audit Updates

Initial Audit	18th March 2022
Corrected	

Contract Analysis

Critical Medium Minor Pass

Severity	Code	Description
●	ST	Contract Owner is not able to stop or pause transactions
●	OCTD	Contract Owner is not able to transfer tokens from specific address
●	OTUT	Owner Transfer User's Tokens
●	ELFM	Contract Owner is not able to increase fees more than a reasonable percent (25%)
●	ULTW	Contract Owner is not able to increase the amount of liquidity taken by dev wallet more than a reasonable percent
●	MT	Contract Owner is not able to mint new tokens
●	BT	Contract Owner is not able to burn tokens from specific wallet
●	BC	Contract Owner is not able to blacklist wallets from selling

ST - Stop Transactions

Criticality	critical
Location	contract.sol#L640,492

Description

The contract owner has the authority to stop transactions for all users excluding the owner. The owner may take advantage of it by setting the `tradingEnabled` to false.

```
if (!_isExcludedFromFee[from] && !_isExcludedFromFee[to]) {
    require(tradingEnabled, "Trading not active");
}
```

The contract owner has the authority to stop the sales for all users excluding the owner. This will cause the contract to operate like a honeypot. The owner may take advantage of it by setting `coolDownTime` to a high value

```
if (coolDownEnabled) {
    uint256 timePassed = block.timestamp - _lastSell[from];
    require(timePassed >= coolDownTime, "Cooldown enabled");
    _lastSell[from] = block.timestamp;
}
```

Recommendation

The contract could embody a check for not allowing setting the variables less or more than a reasonable amount.

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

ELFM - Exceed Limit Fees Manipulation

Criticality	minor
Location	contract.sol#L396

Description

The contract owner has the authority to increase over the allowed limit of 25%. The owner may take advantage of it by calling the `setSellTaxes` function with 30 percentage value.

```
   function setSellTaxes(
        uint256 _rfi,
        uint256 _marketing,
        uint256 _liquidity,
        uint256 _dev,
        uint256 _utility,
        uint256 _burn
    ) public onlyOwner {
        sellTaxes = Taxes(_rfi, _marketing, _liquidity, _dev, _utility, _burn);
        require((_rfi + _marketing + _utility + _liquidity + _dev + _burn) <=
30, "Must keep fees at 30% or less");
        emit FeesChanged();
    }
```

Recommendation

The contract could have a lower limit on fees.

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

BC - Blacklisted Contracts

Criticality	critical
Location	contract.sol#L616

Description

The contract owner has the authority to massively stop contacts from transactions. The owner may take advantage of it by calling the `bulkIsBlacklisted` function.

```
require(!_isBlacklisted[from] && !_isBlacklisted[to], "You are a bot");
```

Recommendation

The team should carefully manage the private keys of the owner’s account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

Contract Diagnostics

Critical Medium Minor

Severity	Code	Description
	L01	Public Function could be Declared External
	L02	State Variables could be Declared Constant
	L04	Conformance to Solidity Naming Conventions
	L11	Unnecessary Boolean equality
	L07	Missing Events Arithmetic
	L13	Divide before Multiply Operation

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L55,59,241,245,254,263,267,272,286,291 and 9 more

Description

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

```
rescueAnyBEP20Tokens
setSellTaxes
setTaxes
isExcludedFromFee
includeInFee
excludeFromFee
reflectionFromToken
isExcludedFromReward
transfer
...
```

Recommendation

Use the external attribute for functions never called from the contract

L02 - State Variables could be Declared Constant

Criticality	minor
Location	contract.sol#L142,153

Description

Constant state variables should be declared constant to save gas.

```
deadWallet
_tTotal
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L78,185,328,385,386,387,388,389,390,397 and 13 more

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.

```
_symbol
_name
genesis_block
_decimals
_amount
_to
_tokenAddr
_enabled
_burn
...
```

Recommendation

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

L11 - Unnecessary Boolean equality

Criticality	minor
Location	contract.sol#L326

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.

```
state == true
```

Recommendation

Remove the equality to the boolean constant.

L07 - Missing Events Arithmetic

Criticality	minor
Location	contract.sol#L326,799,804,832,837

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.

```
maxWalletLimit = amount * 10 ** decimals()
maxBuyLimit = maxBuy * 10 ** decimals()
swapTokensAtAmount = amount * 10 ** _decimals
coolDownTime = time * 1
deadline = _deadline
```

Recommendation

Emit an event for critical parameter changes.

L13 - Divide before Multiply Operation

Criticality	minor
Location	contract.sol#L712

Description

Performing divisions before multiplications may cause lose of prediction.

```
unitBalance = deltaBalance / (denominator - temp.liquidity)
```

Recommendation

The multiplications should be prior to the divisions.

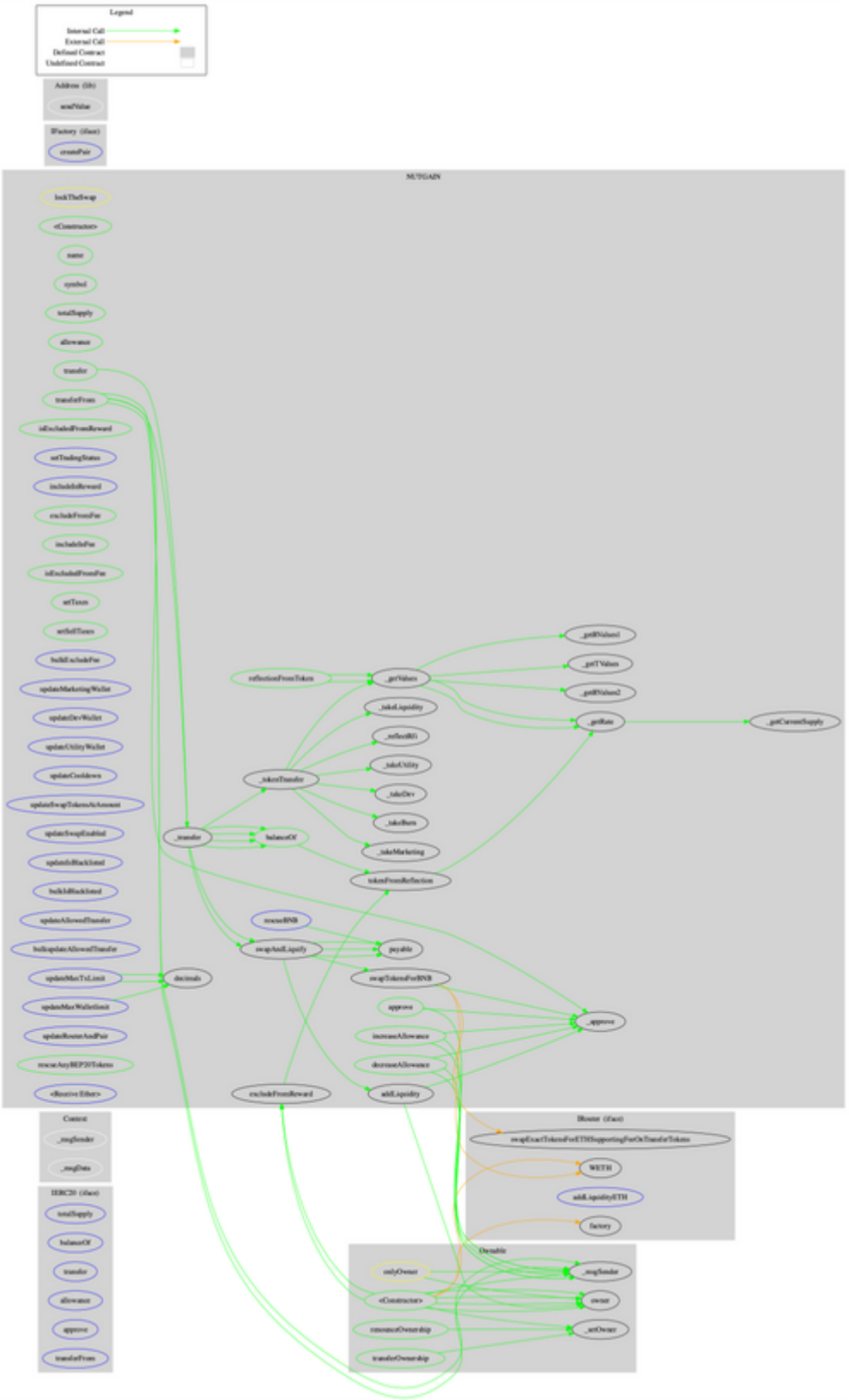
Contract Functions

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
Ownable	Implementation	Context		
	<Constructor>	Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_setOwner	Private	✓	
IFactory	Interface			
	createPair	External	✓	-
IRouter	Interface			
	factory	External		-
	WETH	External		-
	addLiquidityETH	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-

Address	Library			
	sendValue	Internal	✓	
NUTGAIN	Implementation	Context, IERC20, Ownable		
	<Constructor>	Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	transfer	Public	✓	-
	isExcludedFromReward	Public		-
	reflectionFromToken	Public		-
	setTradingStatus	External	✓	onlyOwner
	tokenFromReflection	Public		-
	excludeFromReward	Public	✓	onlyOwner
	includeInReward	External	✓	onlyOwner
	excludeFromFee	Public	✓	onlyOwner
	includeInFee	Public	✓	onlyOwner
	isExcludedFromFee	Public		-
	setTaxes	Public	✓	onlyOwner
	setSellTaxes	Public	✓	onlyOwner
	_reflectRfi	Private	✓	
	_takeLiquidity	Private	✓	
	_takeMarketing	Private	✓	
	_takeBurn	Private	✓	
	_takeDev	Private	✓	
	_takeUtility	Private	✓	
	_getValues	Private		

	_getTValues	Private		
	_getRValues1	Private		
	_getRValues2	Private		
	_getRate	Private		
	_getCurrentSupply	Private		
	_approve	Private	✓	
	_transfer	Private	✓	
	_tokenTransfer	Private	✓	
	swapAndLiquify	Private	✓	lockTheSwap
	addLiquidity	Private	✓	
	swapTokensForBNB	Private	✓	
	bulkExcludeFee	External	✓	onlyOwner
	updateMarketingWallet	External	✓	onlyOwner
	updateDevWallet	External	✓	onlyOwner
	updateUtilityWallet	External	✓	onlyOwner
	updateCooldown	External	✓	onlyOwner
	updateSwapTokensAtAmount	External	✓	onlyOwner
	updateSwapEnabled	External	✓	onlyOwner
	updateIsBlacklisted	External	✓	onlyOwner
	bulkIsBlacklisted	External	✓	onlyOwner
	updateAllowedTransfer	External	✓	onlyOwner
	bulkupdateAllowedTransfer	External	✓	onlyOwner
	updateMaxTxLimit	External	✓	onlyOwner
	updateMaxWalletlimit	External	✓	onlyOwner
	updateRouterAndPair	External	✓	onlyOwner
	rescueBNB	External	✓	onlyOwner
	rescueAnyBEP20Tokens	Public	✓	onlyOwner
	<Receive Ether>	External	Payable	-

Contract Flow



Domain Info

Domain Name	nutgain.io
Registry Domain ID	fab7836b511f4e12828701bb1b324ae5-DONUTS
Creation Date	2021-08-31T17:15:47Z
Updated Date	2022-03-18T10:29:03Z
Registry Expiry Date	2023-08-31T17:15:47Z
Registrar WHOIS Server	http://whois.cloudflare.com
Registrar URL	http://cloudflare.com
Registrar	Cloudflare, Inc
Registrar IANA ID	1910

The domain has been created 8 months before the creation of the audit. It will expire in over 1 year.

There is no public billing information, the creator is protected by the privacy settings.

Summary

There are some functions that can be abused by the owner, like blacklisting wallets and stopping transactions. The selling fees can be increased up to 30%. The contract can be converted into a honeypot and prevent users from selling if the owner abuses the admin functions. A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats.

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

Cyberscope

Audit Report

NUTGAIN Staking

April 2022

Type	BEP20
Network	BSC
Address	0xF1477207249C4429a0B872dFE91E276176DCedff
Audited by	© cyberscope

Table of Contents

Table of Contents 1
Contract Review 3
Source Files 3
Audit Updates 3
Contract Diagnostics 4
Notes 5
Data Structure Optimization 6
Description 6
Recommendation 6
CO - Code Optimization 7
Description 7
Recommendation 7
STC - Succeeded Transfer Check 8
Description 8
Recommendation 8
CR - Code Repetition 9
Description 9
Recommendation 9
MC - Missing Check 10
Description 10
Recommendation 10
L01 - Public Function could be Declared External 11
Description 11
Recommendation 11
L02 - State Variables could be Declared Constant 12
Description 12

Recommendation 12
L04 - Conformance to Solidity Naming Conventions 13
Description 13
Recommendation 13
L07 - Missing Events Arithmetic 14
Description 14
Recommendation 14
L09 - Dead Code Elimination 15
Description 15
Recommendation 15
L13 - Divide before Multiply Operation 16
Description 16
Recommendation 16
L14 - Uninitialized Variables in Local Scope 17
Description 17
Recommendation 17
Contract Functions 18
Contract Flow 21
Summary 22
Disclaimer 23
About Cyberscope 24

Contract Review

Contract Name	NutGainSingleStaking
Compiler Version	v0.8.7+commit.e28d00a7
Optimization	200 runs
Licence	MIT
Explorer	https://bscscan.com/token/0xf1477207249c4429a0b872dfe91e276176dcedff

Source Files

Filename	**SHA256**
contract.sol	1d7015d97935ed3fc35fbd55165d192b66105ecde365fbd4642989491e0dcca8

Audit Updates

Initial Audit	21st April 2022
Corrected	

Contract Diagnostics

Critical Medium Minor

Severity	Code	Description
	DSO	Data Structure Optimization
	CO	Code Optimization
	STC	Succeeded Transfer Check
	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
	L07	Missing Events Arithmetic
	L09	Dead Code Elimination
	L13	Divide before Multiply Operation
	L14	Uninitialized Variables in Local Scope

Notes

- if tokenSupply * apy is less than 1051200000, then no rewards will be distributed since the division will be less than 1.
- A user can withdraw the deposited amount with two ways. The emergencyWithdraw() that receives the entire amount, and the withdraw() that receives the amount redacted by an early withdrawal tax. The users have no reason to withdraw tokens using the withdraw() method since they have the option to call the emergencyWithdraw() method

Data Structure Optimization

Criticality	minor
Location	contract.sol#L778

Description

The poolInfo contains the array of the pools. The userInfo contains a map from the pool index to a map from the user's address to the user info. The first mapping of userInfo is mirroring the poolinfo array, as a result the code should keep in sync these to structures.

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

Recommendation

Since the userInfo is a subset of the poolInfo structure, the PoolInfo could embody the mapping(address => UserInfo) structure. Hence, the source will not have to keep in sync these to indexes.

```
struct PoolInfo {
    uint256 apy;
    uint256 lastRewardBlock;
    uint256 accTokenPerShare;
    uint256 withdrawLockPeriod;
    uint256 balance;
    mapping(address => UserInfo) userInfo;
}
```

CO - Code Optimization

Criticality	minor
Location	contract.sol#L856,876

Description

The contract is declaring that there are properties that will be used for storage even if they are used for temporary calculations.

```
function pendingToken(uint256 _pid, address _user) public view returns (uint256)
{
    PoolInfo storage pool = poolInfo[_pid];
    UserInfo storage user = userInfo[_pid][_user];

...
function canHarvest(uint256 _pid, address _user) public view returns (bool) {
    UserInfo storage user = userInfo[_pid][_user];
    PoolInfo storage pool = poolInfo[_pid];
    return block.timestamp >= user.lastDepositTime.add(pool.withdrawLockPeriod);
}
```

Recommendation

The class properties that are not affected should be declared as *memory*.

STC - Succeeded Transfer Check

Criticality	minor
Location	contract.sol#L1045

Description

According to the ERC20 specification, the transfer methods should be checked if the result is successful. Otherwise, the contract may wrongly assume that the transfer has been established.

```
function safeTokenTransfer(address _to, uint256 _amount) internal {
    uint256 tokenBal = token.balanceOf(address(this));
    if (_amount > tokenBal) {
        token.transfer(_to, tokenBal);
    } else {
        token.transfer(_to, _amount);
    }
}
```

Recommendation

The contract should check if the result of the transfer methods is successful.

CR - Code Repetition

Criticality	minor
Location	contract.sol#L973,943,985,1001

Description

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

```
pool.balance = pool.balance.sub(amount);
user.amount = 0;
user.rewardDebt = 0;
user.rewardLockedUp = 0;

///

uint256 totalRewards = pending.add(user.rewardLockedUp);

// reset lockup
totalLockedUpRewards = totalLockedUpRewards.sub(user.rewardLockedUp);
user.rewardLockedUp = 0;
// send rewards
safeTokenTransfer(msg.sender, totalRewards);
```

Recommendation

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

MC - Missing Check

Criticality	minor
Location	contract.sol#L891,914,935,955,981,994

Description

The following functions require the pid or the sender’s address but are not checking if the pool id and sender’s address exist in the current structures.

- updatePool
- Deposit
- Harvest
- Withdraw
- emergencyWithdraw
- payOrLockupPendingToken

```
function deposit(uint256 _pid, uint256 _amount) external nonReentrant {
    PoolInfo storage pool = poolInfo[_pid];
    UserInfo storage user = userInfo[_pid][msg.sender];
    updatePool(_pid);
    payOrLockupPendingToken(_pid);
...
```

Recommendation

The contract should properly check if the indexes exist in the current structures.

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L671,680,856,1054

Description

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

```
getBlock
pendingToken
transferOwnership
renounceOwnership
```

Recommendation

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

L02 - State Variables could be Declared Constant

Criticality	minor
Location	contract.sol#L786

Description

Constant state variables should be declared constant to save gas.

```
DEAD
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L819,820,836,837,847,856,876,891,914,935,955,981,994,1016,1024,1045,786

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.

```
DEAD
_amount
_to
_earlyFeePercent
_pid
_user
_from
_withdrawLockPeriod
_withUpdate
...
```

Recommendation

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

L07 - Missing Events Arithmetic

Criticality	minor
Location	contract.sol#L1058

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.

```
treasure = treasure.sub(amount)
```

Recommendation

Emit an event for critical parameter changes.

L09 - Dead Code Elimination

Criticality	minor
Location	contract.sol#L392,417,466,476,442,452,366,542,567,558

Description

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

```
safeIncreaseAllowance
safeDecreaseAllowance
safeApprove
sendValue
functionStaticCall
functionDelegateCall
functionCallWithValue
functionCall
...
```

Recommendation

Remove unused functions.

L13 - Divide before Multiply Operation

Criticality	minor
Location	contract.sol#L856,891

Description

Performing divisions before multiplications may cause lose of prediction.

```
tokenPerBlock = tokenSupply.mul(pool.apy).div(100).div(10512000)
```

Recommendation

The multiplications should be prior to the divisions.

L14 - Uninitialized Variables in Local Scope

Criticality	minor
Location	contract.sol#L962

Description

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

```
burnAmount
```

Recommendation

All the local scoped variables should be initialized.

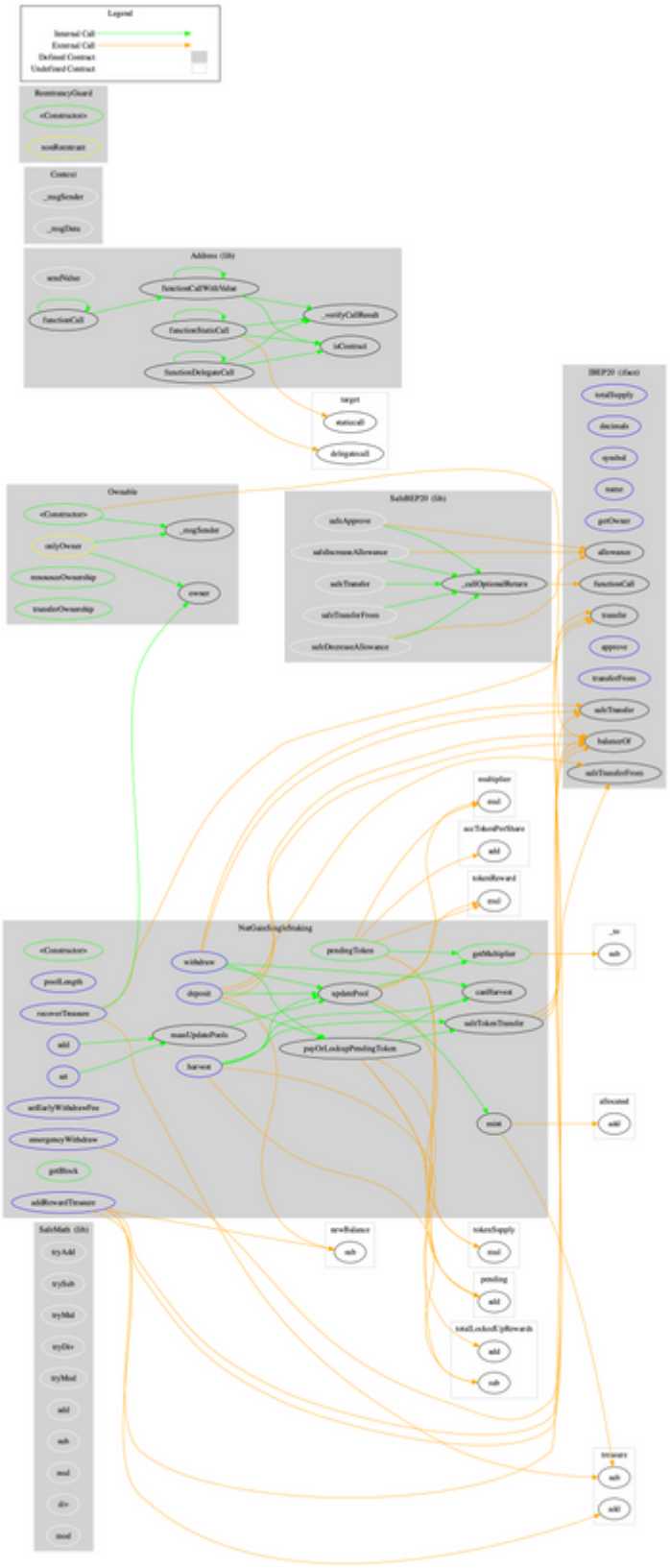
Contract Functions

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
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		
IBEP20	Interface			
	totalSupply	External		-
	decimals	External		-
	symbol	External		-
	name	External		-
	getOwner	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
Address	Library			
	isContract	Internal		

	sendValue	Internal	✓	
	functionCall	Internal	✓	
	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionStaticCall	Internal		
	functionStaticCall	Internal		
	functionDelegateCall	Internal	✓	
	functionDelegateCall	Internal	✓	
	_verifyCallResult	Private		
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
ReentrancyGuard	Implementation			
	<Constructor>	Public	✓	-
NutGainSingleStaking	Implementation	Ownable, ReentrancyGuard		

	<Constructor>	Public	✓	-
	poolLength	External		-
	add	External	✓	onlyOwner
	set	External	✓	onlyOwner
	getMultiplier	Public		-
	pendingToken	Public		-
	canHarvest	Public		-
	massUpdatePools	Public	✓	-
	updatePool	Public	✓	-
	deposit	External	✓	nonReentrant
	harvest	External	✓	nonReentrant
	withdraw	External	✓	nonReentrant
	emergencyWithdraw	External	✓	nonReentrant
	payOrLockupPendingToken	Internal	✓	
	setEarlyWithdrawFee	External	✓	onlyOwner
	addRewardTreasure	External	✓	-
	mint	Internal	✓	
	safeTokenTransfer	Internal	✓	
	getBlock	Public		-
	recoverTreasure	External	✓	onlyOwner

Contract Flow



Summary

NUTGAIN staking contract implements a staking related functionality. There are pools that combine different APYs and Locking periods. The users have the ability to deposit funds in one of the pools. The users can either redeem the rewards or withdraw the entire amount. This audit focuses on some security concerns, performance optimizations and business logic 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

Cyberscope

Audit Report

Nutgain

March 2022

Type	BEP20
Network	BSC
Address	0x40a3e5F7dF89cDE44ef7E91dC47a4c5345c32A9b
Audited by	© cyberscope

Table of Contents

Table of Contents 1
Contract Review 3
Audit Updates 3
Contract Analysis 4
ST - Stop Transactions 5
Description 5
Recommendation 6
ELFM - Exceed Limit Fees Manipulation 7
Description 7
Recommendation 7
BC - Blacklisted Contracts 8
Description 8
Recommendation 8
Contract Diagnostics 9
L01 - Public Function could be Declared External 10
Description 10
Recommendation 10
L02 - State Variables could be Declared Constant 11
Description 11
Recommendation 11
L04 - Conformance to Solidity Naming Conventions 12
Description 12
Recommendation 12
L11 - Unnecessary Boolean equality 13
Description 13
Recommendation 13

L07 - Missing Events Arithmetic 14
Description 14
Recommendation 14
L13 - Divide before Multiply Operation 15
Description 15
Recommendation 15
Contract Functions 16
Contract Flow 19
Domain Info 20
Summary 21
Disclaimer 22
About Cyberscope 23

Contract Review

Contract Name	NUTGAIN
Compiler Version	v0.8.7+commit.e28d00a7
Optimization	200 runs
Licence	Unlicense
Explorer	https://bscscan.com/token/0x40a3e5F7dF89cDE44ef7E91dC47a4c5345c32A9b
Symbol	NUTGV2
Decimals	9
Total Supply	1,500,000,000
Source	contract.sol
Domain	nutgain.io

Audit Updates

Initial Audit	12th March 2022
Corrected	

Contract Analysis

Critical Medium Minor Pass

Severity	Code	Description
●	ST	Contract Owner is not able to stop or pause transactions
●	OCTD	Contract Owner is not able to transfer tokens from specific address
●	OTUT	Owner Transfer User's Tokens
●	ELFM	Contract Owner is not able to increase fees more than a reasonable percent (25%)
●	ULTW	Contract Owner is not able to increase the amount of liquidity taken by dev wallet more than a reasonable percent
●	MT	Contract Owner is not able to mint new tokens
●	BT	Contract Owner is not able to burn tokens from specific wallet
●	BC	Contract Owner is not able to blacklist wallets from selling

ST - Stop Transactions

Criticality	critical
Location	contract.sol#L618,623,640,492

Description

The contract owner has the authority to stop transactions for all users excluding the owner. The owner may take advantage of it by setting the tradingEnabled to false or the `maxBuyLimit`, `maxWalletLimit` or `_maxTxAmount` to zero.

```
if (!_isExcludedFromFee[from] && !_isExcludedFromFee[to]) {
    require(tradingEnabled, "Trading not active");
}
```

```
require(amount <= maxBuyLimit, "You are exceeding maxBuyLimit");
require(
    balanceOf(to) + amount <= maxWalletLimit,
    "You are exceeding maxWalletLimit"
);
```

The contract owner has the authority to stop the sales for all users excluding the owner. This will cause the contract to operate like a honeypot. The owner may take advantage of it by setting `coolDownTime` or `sellTaxes` to a high value

```
if (coolDownEnabled) {
    uint256 timePassed = block.timestamp - _lastSell[from];
    require(timePassed >= coolDownTime, "Cooldown enabled");
    _lastSell[from] = block.timestamp;
}
```

```
Taxes memory temp;
if (isSell && !useLaunchTax) temp = sellTaxes;
else if (!useLaunchTax) temp = taxes;
else temp = launchtax;
```

Recommendation

The contract could embody a check for not allowing setting the variables less or more than a reasonable amount. A suggested implementation could check that the maximum amount should be more than a fixed percentage of the total supply.

The team should carefully manage the private keys of the owner’s account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

ELFM - Exceed Limit Fees Manipulation

Criticality	critical
Location	contract.sol#L396

Description

The contract owner has the authority to increase over the allowed limit of 25%. The owner may take advantage of it by calling the `setSellTaxes` function with a high percentage value.

```
function setSellTaxes(
    uint256 _rfi,
    uint256 _marketing,
    uint256 _liquidity,
    uint256 _dev,
    uint256 _utility,
    uint256 _burn
) public onlyOwner {
    sellTaxes = Taxes(_rfi, _marketing, _liquidity, _dev, _utility, _burn);
    emit FeesChanged();
}
```

Recommendation

The contract could embody a check for the maximum acceptable value.

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

BC - Blacklisted Contracts

Criticality	critical
Location	contract.sol#L616

Description

The contract owner has the authority to massively stop contacts from transactions. The owner may take advantage of it by calling the `bulkIsBlacklisted` function.

```
require(!_isBlacklisted[from] && !_isBlacklisted[to], "You are a bot");
```

Recommendation

The team should carefully manage the private keys of the owner’s account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

Contract Diagnostics

Critical Medium Minor

Severity	Code	Description
	L01	Public Function could be Declared External
	L02	State Variables could be Declared Constant
	L04	Conformance to Solidity Naming Conventions
	L11	Unnecessary Boolean equality
	L07	Missing Events Arithmetic
	L13	Divide before Multiply Operation

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L55,59,241,245,254,263,267,272,286,291 and 9 more

Description

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

```
rescueAnyBEP20Tokens
setSellTaxes
setTaxes
isExcludedFromFee
includeInFee
excludeFromFee
reflectionFromToken
isExcludedFromReward
transfer
...
```

Recommendation

Use the external attribute for functions never called from the contract

L02 - State Variables could be Declared Constant

Criticality	minor
Location	contract.sol#L142,153

Description

Constant state variables should be declared constant to save gas.

```
deadWallet
_tTotal
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L78,185,328,385,386,387,388,389,390,397 and 13 more

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.

```
_symbol
_name
genesis_block
_decimals
_amount
_to
_tokenAddr
_enabled
_burn
...
```

Recommendation

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

L11 - Unnecessary Boolean equality

Criticality	minor
Location	contract.sol#L326

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.

```
state == true
```

Recommendation

Remove the equality to the boolean constant.

L07 - Missing Events Arithmetic

Criticality	minor
Location	contract.sol#L326,799,804,832,837

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.

```
maxWalletLimit = amount * 10 ** decimals()
maxBuyLimit = maxBuy * 10 ** decimals()
swapTokensAtAmount = amount * 10 ** _decimals
coolDownTime = time * 1
deadline = _deadline
```

Recommendation

Emit an event for critical parameter changes.

L13 - Divide before Multiply Operation

Criticality	minor
Location	contract.sol#L712

Description

Performing divisions before multiplications may cause lose of prediction.

```
unitBalance = deltaBalance / (denominator - temp.liquidity)
```

Recommendation

The multiplications should be prior to the divisions.

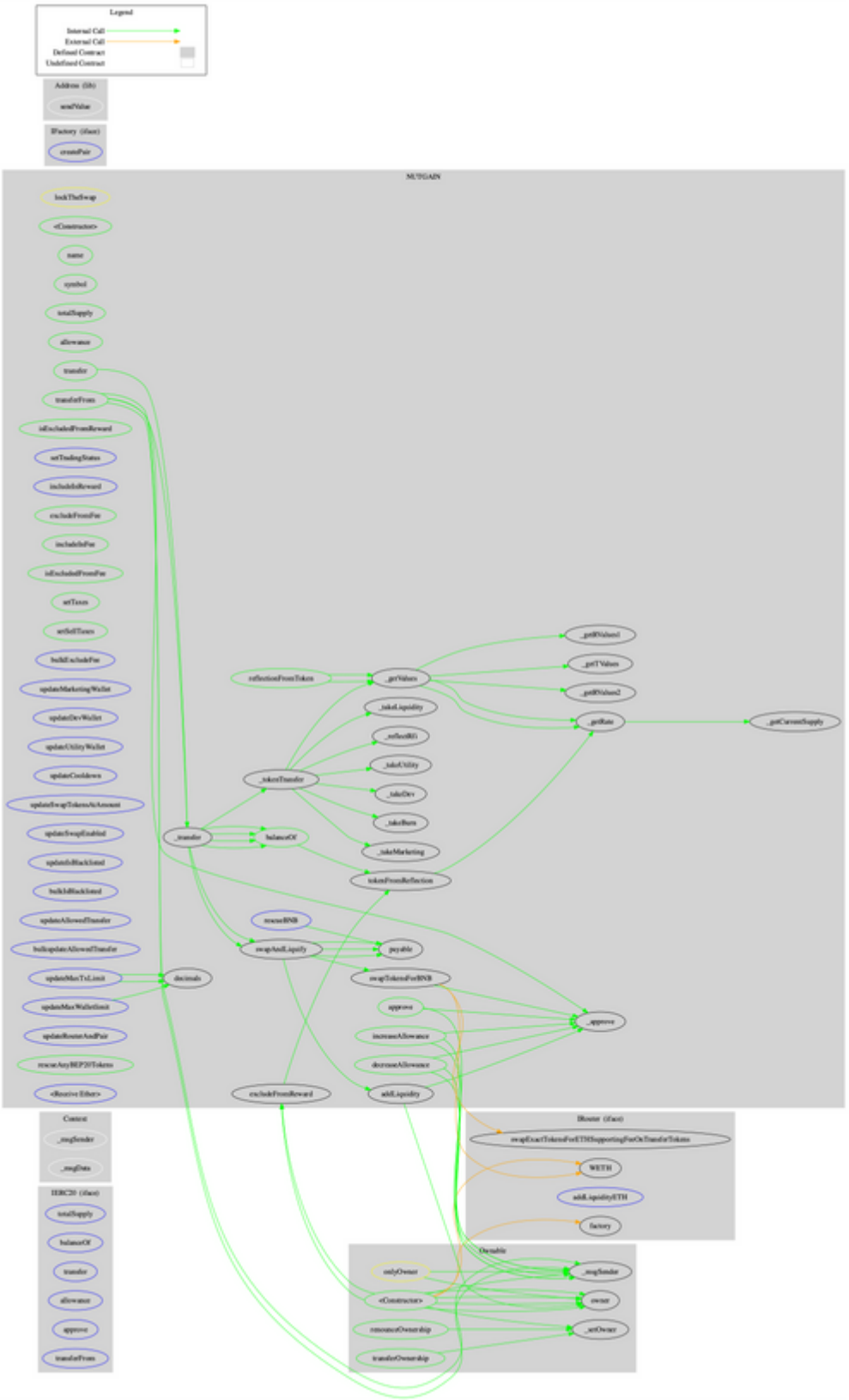
Contract Functions

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
Ownable	Implementation	Context		
	<Constructor>	Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_setOwner	Private	✓	
IFactory	Interface			
	createPair	External	✓	-
IRouter	Interface			
	factory	External		-
	WETH	External		-
	addLiquidityETH	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-

Address	Library			
	sendValue	Internal	✓	
NUTGAIN	Implementation	Context, IERC20, Ownable		
	<Constructor>	Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	transfer	Public	✓	-
	isExcludedFromReward	Public		-
	reflectionFromToken	Public		-
	setTradingStatus	External	✓	onlyOwner
	tokenFromReflection	Public		-
	excludeFromReward	Public	✓	onlyOwner
	includeInReward	External	✓	onlyOwner
	excludeFromFee	Public	✓	onlyOwner
	includeInFee	Public	✓	onlyOwner
	isExcludedFromFee	Public		-
	setTaxes	Public	✓	onlyOwner
	setSellTaxes	Public	✓	onlyOwner
	_reflectRfi	Private	✓	
	_takeLiquidity	Private	✓	
	_takeMarketing	Private	✓	
	_takeBurn	Private	✓	
	_takeDev	Private	✓	
	_takeUtility	Private	✓	
	_getValues	Private		

	_getTValues	Private		
	_getRValues1	Private		
	_getRValues2	Private		
	_getRate	Private		
	_getCurrentSupply	Private		
	_approve	Private	✓	
	_transfer	Private	✓	
	_tokenTransfer	Private	✓	
	swapAndLiquify	Private	✓	lockTheSwap
	addLiquidity	Private	✓	
	swapTokensForBNB	Private	✓	
	bulkExcludeFee	External	✓	onlyOwner
	updateMarketingWallet	External	✓	onlyOwner
	updateDevWallet	External	✓	onlyOwner
	updateUtilityWallet	External	✓	onlyOwner
	updateCooldown	External	✓	onlyOwner
	updateSwapTokensAtAmount	External	✓	onlyOwner
	updateSwapEnabled	External	✓	onlyOwner
	updateIsBlacklisted	External	✓	onlyOwner
	bulkIsBlacklisted	External	✓	onlyOwner
	updateAllowedTransfer	External	✓	onlyOwner
	bulkupdateAllowedTransfer	External	✓	onlyOwner
	updateMaxTxLimit	External	✓	onlyOwner
	updateMaxWalletlimit	External	✓	onlyOwner
	updateRouterAndPair	External	✓	onlyOwner
	rescueBNB	External	✓	onlyOwner
	rescueAnyBEP20Tokens	Public	✓	onlyOwner
	<Receive Ether>	External	Payable	-

Contract Flow



Domain Info

Domain Name	nutgain.io
Registry Domain ID	fab7836b511f4e12828701bb1b324ae5-DONUTS
Creation Date	2021-08-31T17:15:47Z
Updated Date	2022-01-28T10:29:03Z
Registry Expiry Date	2023-08-31T17:15:47Z
Registrar WHOIS Server	http://whois.cloudflare.com
Registrar URL	http://cloudflare.com
Registrar	Cloudflare, Inc
Registrar IANA ID	1910

The domain has been created 6 months before the creation of the audit. It will expire in over 1 year.

There is no public billing information, the creator is protected by the privacy settings.

Summary

There are some functions that can be abused by the owner, like manipulating fees, blacklisting wallets and stopping transactions. The contract can be converted into a honeypot and prevent users from selling if the owner abuses the admin functions. A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats.

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