Cyberscope

Audit Report

DualMiner

May 2022

Type	BEP20
Network	BSC
Address	0xc51e869f1fd44c9dab5f195477de5f2fd3e95d1a
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	DUAL
Compiler Version	v0.8.7+commit.e28d00a7
Optimization	5000 runs
Licence	MIT
Explorer	https://bscscan.com/token/0xC51E869F1FD44c9dAb5f195477dE5f2Fd3e95D1a
Symbol	DUAL
Decimals	18
Total Supply	340,000,000
Domain	dualminer.money

Source Files

Filename	**SHA256**
contract.sol	5b85680207fac3adab354b8c657caf1b6c202d3e56c0d69aed5f72afcc15ba34

Audit Updates

Initial Audit	27th May 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#L1413

Description

The contract owner has the authority to stop the sales for all users excluding the owner. The owner may take advantage of it by setting the `txfee` to a high value. Additionally, the contract owner has the authority to change the `TwentyFourhours` value. As a result the `tradeAmount` will be stuck and the users will not be able to sell and their funds will be trapped into a **HONEYPOT**.

```
function ActivateAntiWhale(address from,uint amount) private {

    uint blkTime = block.timestamp;

    uint256 onePercent = balanceOf(from).mul(txfee).div(100); //Should use
variable
    require(amount <= onePercent, "ERR: Can't sell more than 1%");

    if( blkTime > tradeData[from].lastTradeTime + TwentyFourhours) {
        tradeData[from].lastTradeTime = blkTime;
        tradeData[from].tradeAmount = amount;
    }
    else if( (blkTime < tradeData[from].lastTradeTime + TwentyFourhours) && ((
blkTime > tradeData[from].lastTradeTime)) ){
        require(tradeData[from].tradeAmount + amount <= onePercent, "ERR: Can't
sell more than 1% in One day");
        tradeData[from].tradeAmount = tradeData[from].tradeAmount + amount;
    }
}
```

Recommendation

The contract could embody a check for not allowing setting the txfee and TwentyFourhours 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.

ULTW - Unlimited Liquidity to Team Wallet

Criticality	minor
Location	contract.sol#L1211

Description

The contract owner has the authority to transfer funds without limit to the team wallet. These funds have been accumulated from fees collected from the contract. The owner may take advantage of it by calling the clearStuckBalance or rescueToken.

```
function clearStuckBalance(address _receiver) external onlyOwner {
    uint256 balance = address(this).balance;
    payable(_receiver).transfer(balance);
}

function rescueToken(address tokenAddress, uint256 tokens,address _receiver)
external onlyOwner returns (bool success){
    return IERC20(tokenAddress).transfer(_receiver, tokens);
}
```

Recommendation

The contract could embody a check for the maximum amount of funds that can be swapped. Since a huge amount may volatile the token's price.

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
●	CR	Code Repetition
●	L01	Public Function could be Declared External
●	L02	State Variables could be Declared Constant
●	L04	Conformance to Solidity Naming Conventions
●	L05	Unused State Variable
●	L07	Missing Events Arithmetic
●	L09	Dead Code Elimination
●	L12	Using Variables before Declaration
●	L14	Uninitialized Variables in Local Scope
●	L15	Local Scope Variable Shadowing

CR - Code Repetition

Criticality	minor
Location	contract.sol#L1306

Description

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

The `swapManual()` method could be reused by the transfer's swap feature.

```
swapping = true;
if(AmountMarketingFee > 0) swapAndSendToFee(AmountMarketingFee);
if(AmountLiquidityFee > 0) swapAndLiquify(AmountLiquidityFee);
if(AmountTokenRewardsFee > 0) swapAndSendDividends(AmountTokenRewardsFee);
if(AmountGasFee > 0) swapAndGasFee(AmountGasFee);
swapping = false;
```

Recommendation

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

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L44,48,229,233,237,249,254,258,263,273,278,589,602,624,632,748,797,841,900,1109,1113,1129,1145,1171,1191,1195,1252,1256,1260,1264,1306,1430

Description

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

```
getCirculatingSupply
swapManual
isExcludedFromDividends
dividendTokenBalanceOf
withdrawableDividendOf
isExcludedFromFees
setIsSellLimitExempt
setSwapTokensAtAmount
updateGasForProcessing
...
```

Recommendation

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

L02 - State Variables could be Declared Constant

Criticality	minor
Location	contract.sol#L991,990,966,694

Description

Constant state variables should be declared constant to save gas.

```
_rewardTokenAddress
rewardToken
deadWallet
ZeroWallet
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L341,505,506,523,624,628,632,636,573,575,724,752,900,903,909,913,917,928,694,1001,1150,1182,1195,1199,1203,1207,1211,1216,1220,1413,982,983,984,985,987,988,991,1006

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.

```
TwentyFourhours
ZeroWallet
_gasFeeCollector
_marketingWalletAddress
AmountGasFee
AmountMarketingFee
AmountTokenRewardsFee
AmountLiquidityFee
ActivateAntiWhale
...
```

Recommendation

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

L05 - Unused State Variable

Criticality	minor
Location	contract.sol#L144

Description

There are segments that contain unused state variables.

```
MAX_INT256
```

Recommendation

Remove unused state variables.

L07 - Missing Events Arithmetic

Criticality	minor
Location	contract.sol#L1191,1199,1203,1224,1232

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.

```
sellTokenRewardsFee = rewardsFee
buyTokenRewardsFee = rewardsFee
TwentyFourhours = _time
txfee = _value
swapTokensAtAmount = amount
```

Recommendation

Emit an event for critical parameter changes.

L09 - Dead Code Elimination

Criticality	minor
Location	contract.sol#L641,190

Description

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

```
abs
_transfer
```

Recommendation

Remove unused functions.

L12 - Using Variables before Declaration

Criticality	minor
Location	contract.sol#L1406

Description

The contract is using a variable before the declaration. This is usually happening either if it has not been declared yet or the variable has been declared in a different scope.

```
claims
lastProcessedIndex
iterations
```

Recommendation

The variables should be declared before any usage of them.

L14 - Uninitialized Variables in Local Scope

Criticality	minor
Location	contract.sol#L1406,1361

Description

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

```
claims
iterations
fees
lastProcessedIndex
```

Recommendation

All the local scoped variables should be initialized.

L15 - Local Scope Variable Shadowing

Criticality	minor
Location	contract.sol#L584,624,628,632,636,1054

Description

The are variables that are defined in the local scope containing the same name from an upper scope.

```
totalSupply
_owner
_symbol
_name
```

Recommendation

The local variables should have different names from the upper scoped variables.

Contract Functions

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
IPinkAntiBot	Interface			
	setTokenOwner	External	✓	-
	onPreTransferCheck	External	✓	-
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
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	✓	-
IERC20Metadata	Interface	IERC20		
	name	External		-
	symbol	External		-
	decimals	External		-

SafeMath	Library			
	add	Internal		
	sub	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		
	mod	Internal		
	mod	Internal		
SafeMathInt	Library			
	mul	Internal		
	div	Internal		
	sub	Internal		
	add	Internal		
	abs	Internal		
	toUint256Safe	Internal		
SafeMathUint	Library			
	toInt256Safe	Internal		
ERC20	Implementation	Context, IERC20, IERC20Metadata		
	<Constructor>	Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-

	decreaseAllowance	Public	✓	-
	_transfer	Internal	✓	
	_cast	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_beforeTokenTransfer	Internal	✓	
IUniswapV2Router01	Interface			
	factory	External		-
	WETH	External		-
	addLiquidity	External	✓	-
	addLiquidityETH	External	Payable	-
	removeLiquidity	External	✓	-
	removeLiquidityETH	External	✓	-
	removeLiquidityWithPermit	External	✓	-
	removeLiquidityETHWithPermit	External	✓	-
	swapExactTokensForTokens	External	✓	-
	swapTokensForExactTokens	External	✓	-
	swapExactETHForTokens	External	Payable	-
	swapTokensForExactETH	External	✓	-
	swapExactTokensForETH	External	✓	-
	swapETHForExactTokens	External	Payable	-
	quote	External		-
	getAmountOut	External		-
	getAmountIn	External		-
	getAmountsOut	External		-
	getAmountsIn	External		-
IUniswapV2Router02	Interface	IUniswapV2Router01		
	removeLiquidityETHSupportingFeeOnTransferTokens	External	✓	-
	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External	✓	-
	swapExactTokensForTokensSupporti	External	✓	-

	ngFeeOnTransferTokens			
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
IUniswapV2Factory	Interface			
	feeTo	External		-
	feeToSetter	External		-
	getPair	External		-
	allPairs	External		-
	allPairsLength	External		-
	createPair	External	✓	-
	setFeeTo	External	✓	-
	setFeeToSetter	External	✓	-
IUniswapV2Pair	Interface			
	name	External		-
	symbol	External		-
	decimals	External		-
	totalSupply	External		-
	balanceOf	External		-
	allowance	External		-
	approve	External	✓	-
	transfer	External	✓	-
	transferFrom	External	✓	-
	DOMAIN_SEPARATOR	External		-
	PERMIT_TYPEHASH	External		-
	nonces	External		-
	permit	External	✓	-
	MINIMUM_LIQUIDITY	External		-
	factory	External		-
	token0	External		-
	token1	External		-

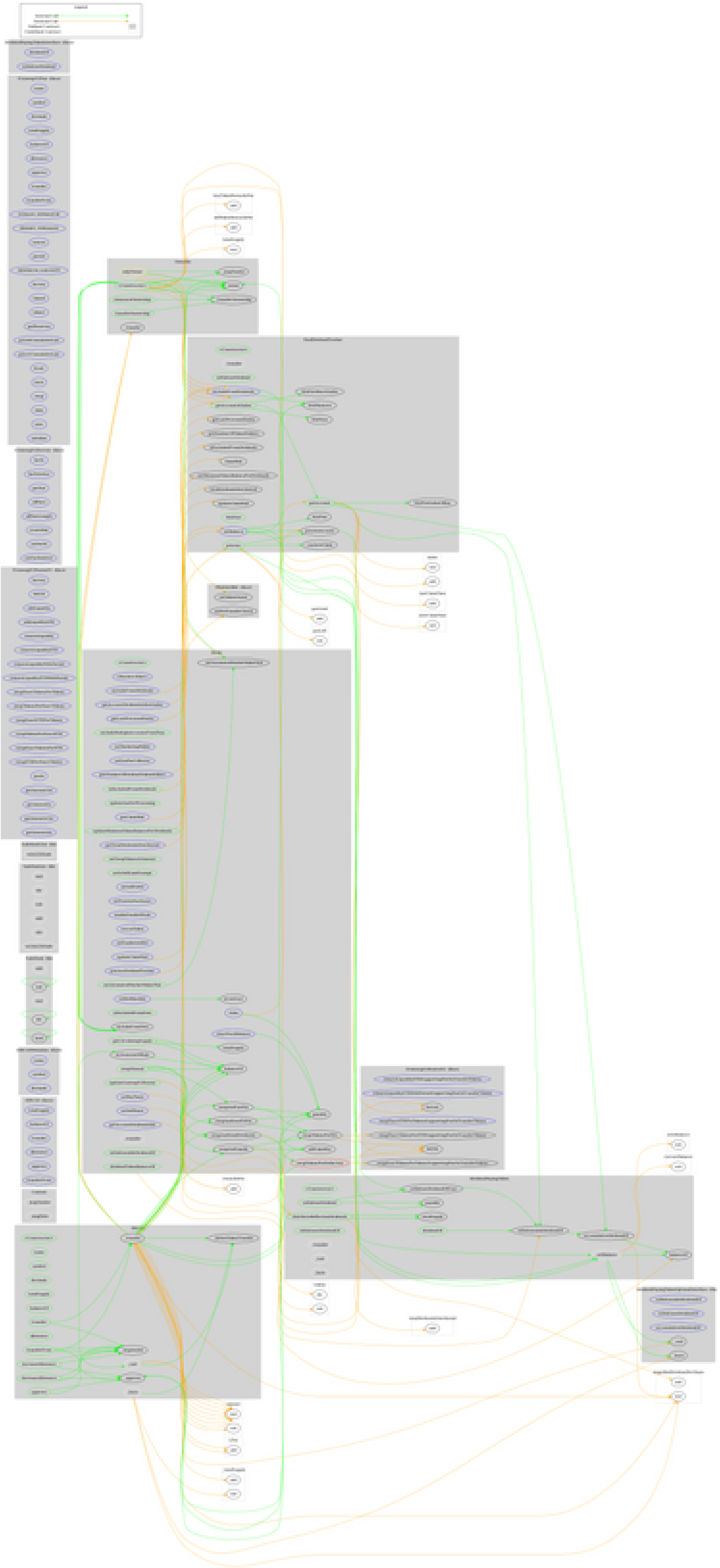
	getReserves	External		-
	price0CumulativeLast	External		-
	price1CumulativeLast	External		-
	kLast	External		-
	burn	External	✓	-
	swap	External	✓	-
	skim	External	✓	-
	sync	External	✓	-
	initialize	External	✓	-
DividendPayingTokenInterface	Interface			
	dividendOf	External		-
	withdrawDividend	External	✓	-
DividendPayingTokenOptionalInterface	Interface			
	withdrawableDividendOf	External		-
	withdrawnDividendOf	External		-
	accumulativeDividendOf	External		-
DividendPayingToken	Implementation	ERC20, Ownable, DividendPayingTokenInterface, DividendPayingTokenOptionalInterface		
	<Constructor>	Public	✓	ERC20
	distributeReflectionDividends	Public	✓	onlyOwner
	withdrawDividend	Public	✓	-
	_withdrawDividendOfUser	Internal	✓	
	dividendOf	Public		-
	withdrawableDividendOf	Public		-
	withdrawnDividendOf	Public		-

	accumulativeDividendOf	Public		-
	_transfer	Internal	✓	
	_cast	Internal	✓	
	_burn	Internal	✓	
	_setBalance	Internal	✓	
DualDividendTracker	Implementation	Ownable, DividendPayingToken		
	<Constructor>	Public	✓	DividendPayingToken
	_transfer	Internal		
	withdrawDividend	Public		-
	setMinimumTokenBalanceForDividends	External	✓	onlyOwner
	excludeFromDividends	External	✓	onlyOwner
	updateClaimWait	External	✓	onlyOwner
	getLastProcessedIndex	External		-
	getNumberOfTokenHolders	External		-
	isExcludedFromDividends	Public		-
	getAccount	Public		-
	getAccountAtIndex	Public		-
	canAutoClaim	Private		
	setBalance	External	✓	onlyOwner
	process	Public	✓	-
	processAccount	Public	✓	onlyOwner
	MAPGet	Public		-
	MAPGetIndexOfKey	Public		-
	MAPGetKeyAtIndex	Public		-
	MAPSize	Public		-
	MAPSet	Public	✓	-
	MAPRemove	Public	✓	-
DUAL	Implementation	ERC20, Ownable		
	<Constructor>	Public	✓	ERC20
	<Receive Ether>	External	Payable	-

	updateMinimumTokenBalanceForDividends	Public	✓	onlyOwner
	updateUniswapV2Router	Public	✓	onlyOwner
	excludeFromFees	Public	✓	onlyOwner
	excludeMultipleAccountsFromFees	Public	✓	onlyOwner
	setMarketingWallet	External	✓	onlyOwner
	setGasFeeCollector	External	✓	onlyOwner
	setAutomatedMarketMakerPair	Public	✓	onlyOwner
	setBotBlacklist	External	✓	onlyOwner
	isContract	Internal		
	_setAutomatedMarketMakerPair	Private	✓	
	updateGasForProcessing	Public	✓	onlyOwner
	updateClaimWait	External	✓	onlyOwner
	excludeFromDividends	External	✓	onlyOwner
	processDividendTracker	External	✓	-
	setSwapTokensAtAmount	Public	✓	onlyOwner
	setIsSellLimitExempt	Public	✓	onlyOwner
	setSellTxFee	External	✓	onlyOwner
	setTwentyFourhours	External	✓	onlyOwner
	enableDisableWhale	External	✓	onlyOwner
	clearStuckBalance	External	✓	onlyOwner
	rescueToken	External	✓	onlyOwner
	setEnableAntiBot	External	✓	onlyOwner
	setBuyTaxes	External	✓	onlyOwner
	setSellTaxes	External	✓	onlyOwner
	getClaimWait	External		-
	getTotalDividendsDistributed	External		-
	isExcludedFromFees	Public		-
	withdrawableDividendOf	Public		-
	dividendTokenBalanceOf	Public		-
	isExcludedFromDividends	Public		-
	getAccountDividendsInfo	External		-
	getAccountDividendsInfoAtIndex	External		-
	claim	External	✓	-
	getLastProcessedIndex	External		-

	getNumberOfDividendTokenHolders	External		-
	swapManual	Public	✓	onlyOwner
	_transfer	Internal	✓	
	ActivateAntiWhale	Private	✓	
	getCirculatingSupply	Public		-
	swapAndGasFee	Private	✓	
	swapAndSendToFee	Private	✓	
	swapAndLiquify	Private	✓	
	swapTokensForEth	Private	✓	
	swapTokensForReflection	Private	✓	
	addLiquidity	Private	✓	
	swapAndSendDividends	Private	✓	

Contract Flow



Domain Info

Domain Name	dualminer.money
Registry Domain ID	a9bf2de23ceb441d854e8d6f2a4c6a1c-DONUTS
Creation Date	2022-04-19T06:52:14Z
Updated Date	2022-04-24T06:52:24Z
Registry Expiry Date	2023-04-19T06:52:14Z
Registrar WHOIS Server	whois.godaddy.com/
Registrar URL	http://www.godaddy.com/domains/search.aspx?ci=8990
Registrar	GoDaddy.com, LLC
Registrar IANA ID	146

The domain has been created about 1 month before the creation of the audit. It will expire in 11 months.

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 stopping transactions and transferring funds to the team’s wallet. 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 provide all the essential tools to assist users draw their own conclusions.

The Cyberscope team

https://www.cyberscope.io