Cyberscope

Audit Report

Diamond Dex Token

December 2022

Type	BEP20
Network	BSC
Address	0x4c287a968189dB50170f7E2b1F0284786F3448A7
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	DiamondDexToken
Compiler Version	v0.8.9+commit.e5eed63a
Optimization	200 runs
Licence	None
Explorer	https://bscscan.com/token/0x4c287a968189dB50170f7E2b1F0284786F3448A7
Symbol	DDT
Decimals	9
Total Supply	100,000,000,000,000,000

Source Files

Filename	**SHA256**
DiamondDexToken.sol	3625dcd404eaa4a487ca17c9340400d59af062ff89df06a4b390ddbbf5e384b8

Audit Updates

Initial Audit	12th December 2022 https://github.com/cyberscope-io/audits/tree/main/2-ddt/v1/audit.pdf
Corrected	16th December 2022

Contract Analysis

● Critical ● Medium ● Minor / Informative ● Pass

Severity	Code	Description	Status
●	ST	Stops Transactions	renounced
●	OCTD	Transfers Contract's Tokens	Passed
●	OTUT	Transfers User's Tokens	Passed
●	ELFM	Exceeds Fees Limit	renounced
●	ULTW	Transfers Liquidity to Team Wallet	Passed
●	MT	Mints Tokens	Passed
●	BT	Burns Tokens	Passed
●	BC	Blacklists Addresses	Passed

ST - Stops Transactions

Criticality	medium
Location	DiamondDexToken.sol#L1350
Status	renounced

Description

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

```
require(amount <= _maxTxAmount, "Transfer amount exceeds the maxTxAmount.");
```

Recommendation

The contract could embody a check for not allowing setting the _maxTxAmount less 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.

Team Update

The team renounced ownership in the following transaction: https://bscscan.com/tx/0x74ec1e9a99eaeda7ae6081eefdb63b5feba6e7bd71ffae17d653cca5c0389219

The variable _maxTxAmount is locked at 10% of the totalSupply.

ELFM - Exceeds Fees Limit

Criticality	critical
Location	DiamondDexToken.sol#L1168,1172,1180
Status	renounced

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 following functions with a high percentage value:

- setTaxFeePercent()
- setLiquidityFeePercent()
- setMarketingFeePercent()

```
function setTaxFeePercent(uint256 taxFee) external onlyOwner() {
    _taxFee = taxFee;
}
...
function setLiquidityFeePercent(uint256 liquidityFee) external onlyOwner() {
    _liquidityFee = liquidityFee;
}
...
function setMarketingFeePercent(uint256 marketingFee) external onlyOwner() {
    _MKTshare = marketingFee;
}
```

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.

Team Update

The team renounced ownership in the following transaction:
https://bscscan.com/tx/0x74ec1e9a99eaeda7ae6081eefdb63b5feba6e7bd71ffae17d653cca5c0389219

The fees are locked in the following values:

- taxFee: 3
- liquidityFee: 2
- marketingFee: 10

Contract Diagnostics

● Critical ● Medium ● Minor / Informative

Severity	Code	Description	Status
●	PVC	Price Volatility Concern	renounced
●	RSML	Redundant SafeMath Library	unresolved
●	L02	State Variables could be Declared Constant	unresolved
●	L04	Conformance to Solidity Naming Conventions	unresolved
●	L07	Missing Events Arithmetic	unresolved
●	L09	Dead Code Elimination	unresolved
●	L10	State Variables in Loop	unresolved

PVC - Price Volatility Concern

Criticality	minor
Location	DiamondDexToken.sol#L1195
Status	renounced

Description

The numTokensSellToAddToLiquidity could produce a dramatically price volatility. If the variable set to a high number, then the contract will sell a huge amount of tokens in a single transaction.

```
function changeNumTokensSellToAddToLiquidity(uint256
_numTokensSellToAddToLiquidity) external onlyOwner {
    numTokensSellToAddToLiquidity = _numTokensSellToAddToLiquidity;
}
```

Recommendation

The contract could ensure that it will not sell more than a reasonable amount of tokens once. A suggested implementation could check that the maximum amount should be less than a fixed percentage of the total supply.

Team Update

The team renounced ownership in the following transaction:
https://bscscan.com/tx/0x74ec1e9a99eaeda7ae6081eefdb63b5feba6e7bd71ffae17d653cca5c0389219

The amount of tokens to add to liquidity is locked at 0.000001% of the totalSupply.

RSML - Redundant SafeMath Library

Criticality	minor
Location	DiamondDexToken.sol#L103
Status	unresolved

Description

The Solidity versions that are greater than or equal to 0.8.0 do not need the use of SafeMath Library. The usage of the SafeMath library produces unnecessary additional gas.

```
library SafeMath {
...
}
```

Recommendation

The team is advised to remove the SafeMath library as it is safe to do math operations without it.

L02 - State Variables could be Declared Constant

Criticality	minor
Location	DiamondDexToken.sol#L918,919,917,913
Status	unresolved

Description

Constant state variables should be declared constant to save gas.

```
_symbol
_decimals
_name
_tTotal
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	DiamondDexToken.sol#L921,1298,611,1195,642,936,924,609,900,935,690,927,1185,1302,1190
Status	unresolved

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.

```
_taxFee
_amount
PERMIT_TYPEHASH
_numTokensSellToAddToLiquidity
MINIMUM_LIQUIDITY
_maxWalletAmount
_liquidityFee
DOMAIN_SEPARATOR
MARKETING_ADDRESS
_maxTxAmount
WETH
_MKTshare
_add
_amount
_enabled
```

Recommendation

Follow the Solidity naming convention.
https://docs.soliditylang.org/en/v0.8.17/style-guide.html#naming-conventions.

L07 - Missing Events Arithmetic

Criticality	minor
Location	DiamondDexToken.sol#L1168,1172,1176,1180,1195
Status	unresolved

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.

```
_taxFee = taxFee
_liquidityFee = liquidityFee
_maxTxAmount = _tTotal.mul(maxTxPercent).div(10 ** 2)
_MKTshare = marketingFee
numTokensSellToAddToLiquidity = _numTokensSellToAddToLiquidity
```

Recommendation

Emit an event for critical parameter changes.

L09 - Dead Code Elimination

Criticality	minor
Location	DiamondDexToken.sol#L320,384,404,417,290,365,352
Status	unresolved

Description

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

```
sendValue
functionCallWithValue
functionCallWithValue
_functionCallWithValue
isContract
functionCall
functionCall
```

Recommendation

Remove unused functions.

L10 - State Variables in Loop

Criticality	minor
Location	DiamondDexToken.sol#L1125
Status	unresolved

Description

Costly operations inside a loop might waste gas, so optimizations are justified. Incrementing state variables in a loop incurs a lot of gas because of expensive SSTOREs, which might lead to an out-of-gas.

```
_excluded.pop()
```

Recommendation

Use a local variable to hold the loop computation result.

Contract Functions

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
SafeMath	Library			
	add	Internal		
	sub	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		
	mod	Internal		
	mod	Internal		
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
Address	Library			
	isContract	Internal		
	sendValue	Internal	✓	
	functionCall	Internal	✓	
	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	

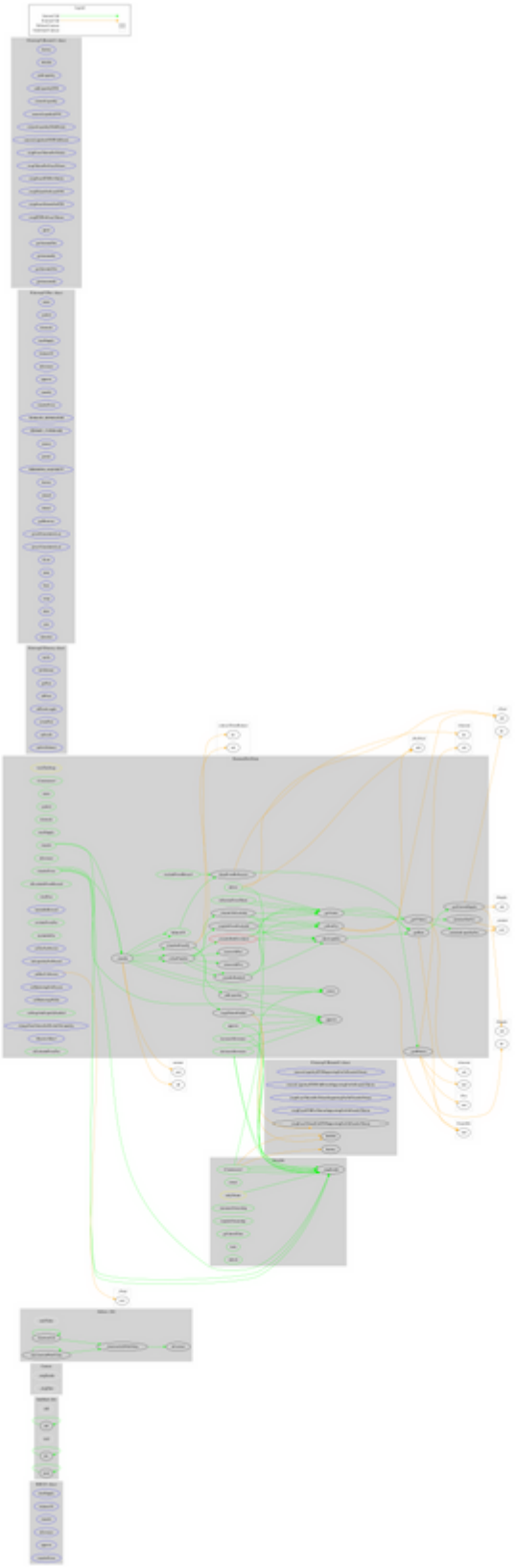
	_functionCallWithValue	Private	✓	
Ownable	Implementation	Context		
	<Constructor>	Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	geUnlockTime	Public		-
	lock	Public	✓	onlyOwner
	unlock	Public	✓	-
IUniswapV2Factory	Interface			
	feeTo	External		-
	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		-
	mint	External	✓	-
	burn	External	✓	-
	swap	External	✓	-
	skim	External	✓	-
	sync	External	✓	-
	initialize	External	✓	-
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	✓	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
DiamondDexToken	Implementation	Context, IERC20, Ownable		
	<Constructor>	Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	isExcludedFromReward	Public		-
	totalFees	Public		-
	deliver	Public	✓	-
	reflectionFromToken	Public		-
	tokenFromReflection	Public		-
	excludeFromReward	Public	✓	onlyOwner
	includeInReward	External	✓	onlyOwner
	_transferBothExcluded	Private	✓	

	excludeFromFee	Public	✓	onlyOwner
	includeInFee	Public	✓	onlyOwner
	setTaxFeePercent	External	✓	onlyOwner
	setLiquidityFeePercent	External	✓	onlyOwner
	setMaxTxPercent	External	✓	onlyOwner
	setMarketingFeePercent	External	✓	onlyOwner
	setMarketingWallet	External	✓	onlyOwner
	setSwapAndLiquifyEnabled	Public	✓	onlyOwner
	changeNumTokensSellToAddToLiquidity	External	✓	onlyOwner
	<Receive Ether>	External	Payable	-
	_reflectFee	Private	✓	
	_getValues	Private		
	_getTValues	Private		
	_getRValues	Private		
	_getRate	Private		
	_getCurrentSupply	Private		
	_takeLiquidity	Private	✓	
	calculateTaxFee	Private		
	calculateLiquidityFee	Private		
	removeAllFee	Private	✓	
	restoreAllFee	Private	✓	
	isExcludedFromFee	Public		-
	_approve	Private	✓	
	_transfer	Private	✓	
	swapAndLiquify	Private	✓	lockTheSwap
	swapTokensForEth	Private	✓	
	addLiquidity	Private	✓	
	_tokenTransfer	Private	✓	
	_transferStandard	Private	✓	
	_transferToExcluded	Private	✓	
	_transferFromExcluded	Private	✓	

Contract Flow



Summary

There are some functions that can be abused by the owner like stopping transactions and manipulating fees. A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats.

Team Update

The team renounced ownership in the following transaction:
https://bscscan.com/tx/0x74ec1e9a99eaeda7ae6081eefdb63b5feba6e7bd71ffae17d653cca5c0389219

Disclaimer

The information provided in this report does not constitute investment, financial or trading advice and you should not treat any of the document's content as such. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes nor may copies be delivered to any other person other than the Company without Cyberscope’s prior written consent. This report is not nor should be considered an “endorsement” or “disapproval” of any particular project or team. This report is not nor should be regarded as an indication of the economics or value of any “product” or “asset” created by any team or project that contracts Cyberscope to perform a security assessment. This document does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors' business, business model or legal compliance. This report should not be used in any way to make decisions around investment or involvement with any particular project. This report represents an extensive assessment process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk Cyberscope’s position is that each company and individual are responsible for their own due diligence and continuous security Cyberscope’s goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies and in no way claims any guarantee of security or functionality of the technology we agree to analyze. The assessment services provided by Cyberscope are subject to dependencies and are under continuing development. You agree that your access and/or use including but not limited to any services reports and materials will be at your sole risk on an as-is where-is and as-available basis Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives false negatives and other unpredictable results. The services may access and depend upon multiple layers of third parties.

About Cyberscope

Cyberscope is a blockchain cybersecurity company that was founded with the vision to make web3.0 a safer place for investors and developers. Since its launch, it has worked with thousands of projects and is estimated to have secured tens of millions of investors' funds.

Cyberscope is one of the leading smart contract audit firms in the crypto space and has built a high-profile network of clients and partners.

The Cyberscope team

https://www.cyberscope.io