Cyberscope

Audit Report

Squadwar

April 2022

Type	BEP20
Network	BSC
Address	0xE76755de72D6Be6744695De489A1d0701b4C9AD5
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	Squadwar
Compiler Version	v0.8.7+commit.e28d00a7
Optimization	200 runs
Licence	MIT
Explorer	https://bscscan.com/token/0xE76755de72D6Be6744695De489A1d0701b4C9AD5
Symbol	SWT
Decimals	18
Total Supply	1,000,000,000
Domain	squadwar.io

Source Files

Filename	**SHA256**
contract.sol	a39cc8451cd17e904cb8327b4684fd5544113ceef9942976fbe0485adb90acd4

Audit Updates

Initial Audit	7th April 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

ELFM - Exceed Limit Fees Manipulation

Criticality	medium
Location	contract.sol#L1527, 1831, 1855

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 updateMaxSellTransactionAmount function with a zero value.

```
if(!newCycle){

                    if(totalSellAmount >= maxSellTransactionAmount){
                        marketingFeeActual = 0;
                        teamFeeActual = 0;
                        liquidityFeeActual = 0;
                        buybackFeeActual = 30;
                        treasuryFeeActual = 0;
                        BNBRewardsFeeActual = 0;

                    }
```

Recommendation

The contract could set the buyBackFeeActual up to 25 in that case.

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**
●	CO	Code Optimization
●	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

CO - Code Optimization

Criticality	minor
Location	contract.sol#L1672, 1696

Description

There are code segments that could be optimized. A segment may be optimized so that it becomes a smaller size, consumes less memory, executes more rapidly, or performs fewer operations.

```
uint256 _totalBuyFees = newMarketingFee.add(newTeamFee).add(newLiquidityFee);
                _totalBuyFees =
_totalBuyFees.add(newBuybackFee).add(newTreasuryFee).add(newBNBRewardsFee);
        require(_totalBuyFees <= MAX_BUYFEE_RATE, "Max buy fee was 25");

        marketingBuyFees = newMarketingFee;
        teamBuyFees = newTeamFee;
        liquidityBuyFee = newLiquidityFee;
        buyBackBuyFee = newBuybackFee;
        gameTreasuryBuyFee = newTreasuryFee;
        BNBRewardsBuyFee = newBNBRewardsFee;

        emit UpdateBuyFees(newMarketingFee, newTeamFee, newLiquidityFee,
newBuybackFee, newTreasuryFee, newBNBRewardsFee);
        totalBuyFees =
marketingBuyFees.add(teamBuyFees).add(liquidityBuyFee).add(buyBackBuyFee).add(
gameTreasuryBuyFee).add(BNBRewardsBuyFee);
```

Recommendation

_totalBuyFees could be assigned to totalBuyFees so the runtime will be more performant.

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L356,364,381,413,426,443,466,851,1205,1243 and 19 more

Description

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

```
process
getAccountAtIndex
manualBuyBackAndBurn
dividendTokenBalanceOf
withdrawableDividendOf
isExcludedFromFees
updateSellFees
updateBuyFees
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#L1350,1354,1361

Description

Constant state variables should be declared constant to save gas.

```
swapTokensAtAmount
deadWallet
buyBackUpperLimit
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L49,51,82,931,1243,1250,1262,1276,1146,1532 and 20 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.

```
_account
_newMinimumBalance
BNBRewardsFeeActual
MaxSellLimitPeriod
_holderLast24hSellAmount
_holderLastSellTimestamp
BNBRewardsSellFee
BNBRewardsBuyFee
TreasuryEnabled
...
```

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#L863

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#L1527,1532

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.

```
MaxSellLimitPeriod = _hours * (3600)
maxSellTransactionAmount = newAmount * (10 ** 18)
```

Recommendation

Emit an event for critical parameter changes.

L09 - Dead Code Elimination

Criticality	minor
Location	contract.sol#L164,909

Description

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

```
abs
get
```

Recommendation

Remove unused functions.

L12 - Using Variables before Declaration

Criticality	minor
Location	contract.sol#L1961,1960,1962

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.

```
lastProcessedIndex
iterations
claims
```

Recommendation

The variables should be declared before any usage of them.

L14 - Uninitialized Variables in Local Scope

Criticality	minor
Location	contract.sol#L1960,1961,1962

Description

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

```
lastProcessedIndex
claims
iterations
```

Recommendation

All the local scoped variables should be initialized.

L15 - Local Scope Variable Shadowing

Criticality	minor
Location	contract.sol#L1166

Description

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

```
_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**
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
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	✓	-
IUniswapV2Factory	Interface			
	feeTo	External		-
	feeToSetter	External		-
	getPair	External		-
	allPairs	External		-
	allPairsLength	External		-
	createPair	External	✓	-
	setFeeTo	External	✓	-
	setFeeToSetter	External	✓	-
IterableMapping	Library			
	get	Internal		
	getIndexOfKey	Internal		
	getKeyAtIndex	Internal		
	size	Internal		
	set	Internal	✓	
	remove	Internal	✓	
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
IERC20Metadata	Interface	IERC20		
	name	External		-

	symbol	External		-
	decimals	External		-
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	✓	-
	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_beforeTokenTransfer	Internal	✓	
DividendPayingTokenOptionalInterface	Interface			
	withdrawableDividendOf	External		-
	withdrawnDividendOf	External		-
	accumulativeDividendOf	External		-
DividendPayingTokenInterface	Interface			
	dividendOf	External		-
	distributeDividends	External	Payable	-
	withdrawDividend	External	✓	-
SafeMath	Library			

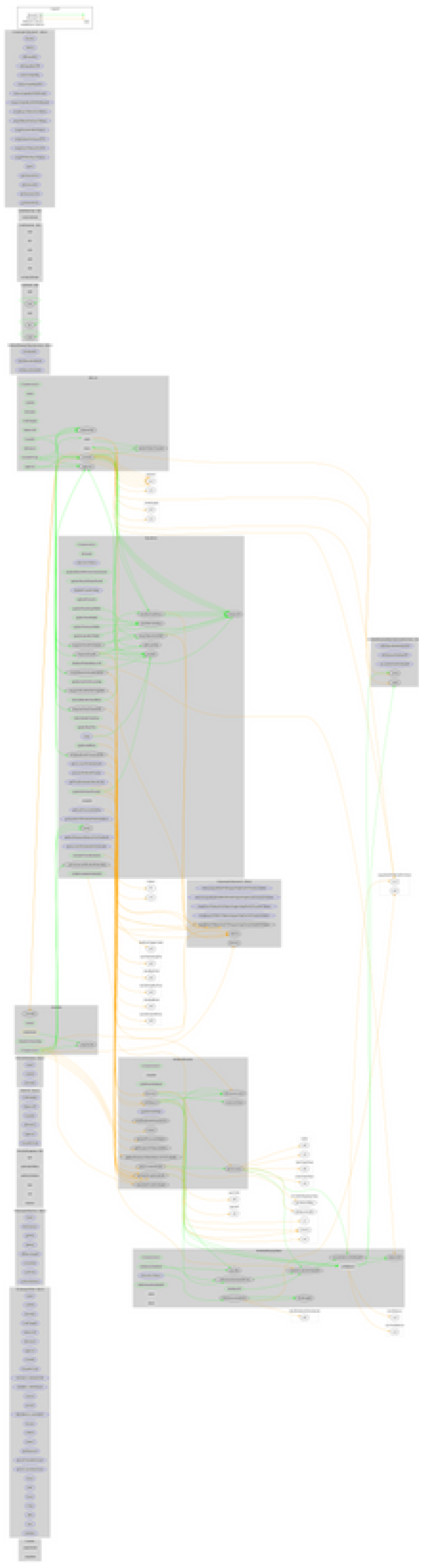
	add	Internal		
	sub	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		
	mod	Internal		
	mod	Internal		
Ownable	Implementation	Context		
	<Constructor>	Public	✓	-
	owner	Public		-
	transferOwnership	Public	✓	onlyOwner
SafeMathInt	Library			
	mul	Internal		
	div	Internal		
	sub	Internal		
	add	Internal		
	abs	Internal		
	toUint256Safe	Internal		
SafeMathUint	Library			
	toInt256Safe	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	✓	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
DividendPayingToken	Implementation	ERC20, DividendPayingTokenInterface, DividendPayingTokenOptionalInterface		
	<Constructor>	Public	✓	ERC20
	<Receive Ether>	External	Payable	-
	distributeDividends	Public	Payable	-
	withdrawDividend	Public	✓	-
	_withdrawDividendOfUser	Internal	✓	
	dividendOf	Public		-
	withdrawableDividendOf	Public		-
	withdrawnDividendOf	Public		-

	accumulativeDividendOf	Public		-
	_mint	Internal	✓	
	_burn	Internal	✓	
	_setBalance	Internal	✓	
Squadwar	Implementation	ERC20, Ownable		
	<Constructor>	Public	✓	ERC20
	decimals	Public		-
	<Receive Ether>	External	Payable	-
	updateMaxSellTransactionAmount	Public	✓	onlyOwner
	updateMaxSellLimitPeriod	Public	✓	onlyOwner
	disableTransferDelay	External	✓	onlyOwner
	updateFeatures	Public	✓	onlyOwner
	updateDividendTracker	Public	✓	onlyOwner
	updateMinimumBalanceForDividends	External	✓	onlyOwner
	updateLiquidityWallet	Public	✓	onlyOwner
	updateMarketingWallet	Public	✓	onlyOwner
	updateTeamWallet	Public	✓	onlyOwner
	updateTreasuryWallet	Public	✓	onlyOwner
	excludeFromFees	Public	✓	onlyOwner
	excludeFromDividends	Public	✓	onlyOwner
	_setAutomatedMarketMakerPair	Private	✓	
	updateGasForProcessing	Public	✓	onlyOwner
	updateBuyFees	Public	✓	onlyOwner
	updateSellFees	Public	✓	onlyOwner
	getTotalDividendsDistributed	External		-
	isExcludedFromFees	Public		-
	withdrawableDividendOf	Public		-
	dividendTokenBalanceOf	Public		-
	getAccountDividendsInfo	External		-
	getAccountDividendsInfoAtIndex	External		-
	processDividendTracker	External	✓	-
	claim	External	✓	-
	getLastProcessedIndex	External		-
	getNumberOfDividendTokenHolders	External		-

	_transfer	Internal	✓	
	swapAndLiquify	Private	✓	
	swapTokensForBNB	Private	✓	
	addLiquidity	Private	✓	
	swapAndSendDividends	Private	✓	
	swapTokenForBuyBackBNB	Private	✓	
	swapAndSendMarketingBNB	Private	✓	
	swapAndSendTeamBNB	Private	✓	
	swapAndSendTreasuryBNB	Private	✓	
	buyBackAndBurn	Private	✓	
	manualBuyBackAndBurn	Public	✓	onlyOwner
DividendTracker	Implementation	DividendPayingToken, Ownable		
	<Constructor>	Public	✓	DividendPayingToken
	_transfer	Internal		
	withdrawDividend	Public		-
	updateMinimumTokenBalanceForDividends	External	✓	onlyOwner
	excludeFromDividends	External	✓	onlyOwner
	updateClaimWait	External	✓	onlyOwner
	getLastProcessedIndex	External		-
	getNumberOfTokenHolders	External		-
	getAccount	Public		-
	getAccountAtIndex	Public		-
	canAutoClaim	Private		
	setBalance	External	✓	onlyOwner
	process	Public	✓	-
	processAccount	Public	✓	onlyOwner

Contract Flow



Domain Info

Domain Name	squadwar.io
Registry Domain ID	70a67d9dec9340beb84f008213feb433-DONUTS
Creation Date	2022-03-12T10:32:36Z
Updated Date	2022-03-17T10:32:40Z
Registry Expiry Date	2023-03-12T10:32:36Z
Registrar WHOIS Server	whois.namecheap.com
Registrar URL	https://www.namecheap.com/
Registrar	NameCheap, Inc.
Registrar IANA ID	1068

The domain has been created 26 days 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

Squadwar is an interesting project that has a friendly and growing community. The Smart Contract analysis reported no compiler error or critical issues. The maximum fee percentage that can be set is 30%. The contract Owner can access some admin functions that can not be used in a malicious way to disturb the users’ transactions.

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