Cyberscope

Audit Report

Santa Coin

November 2022

Type	BEP20
Network	BSC
Address	0xc483a137064Be3953e158f9ddA5f57F886dd11Db
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	SantaCoin
Compiler Version	v0.8.17+commit.8df45f5f
Optimization	200 runs
Licence	MIT
Explorer	https://bscscan.com/token/0xc483a137064Be3953e158f9ddA5f57F886dd11Db
Symbol	Santa
Decimals	18
Total Supply	1,000,000,000,000

Source Files

Filename	**SHA256**
contract.sol	94281a9822811ec3780cb85c45ed300a5bbdb880769fb641f4835db056b68ad5

Audit Updates

Initial Audit	21nd November 2022 https://github.com/cyberscope-io/audits/blob/main/7-smt/v1/audit.pdf
Corrected	22nd November 2022

Contract Analysis

● Critical ● Medium ● Minor / Informative ● Pass

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

Contract Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
	US	Untrusted Source	Unresolved
	RSL	Redundant SafeMath Library	Unresolved
	PVC	Price Volatility Concern	Unresolved
	L04	Conformance to Solidity Naming Conventions	Unresolved
	L05	Unused State Variable	Unresolved
	L07	Missing Events Arithmetic	Unresolved
	L09	Dead Code Elimination	Unresolved
	L11	Unnecessary Boolean equality	Unresolved
	L12	Using Variables before Declaration	Unresolved
	L13	Divide before Multiply Operation	Unresolved
	L14	Uninitialized Variables in Local Scope	Unresolved
	L15	Local Scope Variable Shadowing	Unresolved

US - Untrusted Source

Criticality	critical
Location	contract.sol#L1293
Status	Unresolved

Description

The contract uses an external contract in order to determine the transaction's flow. The external contract is untrusted. As a result it may produce security issues and harm the transactions.

```
dividendTracker.distributeDividends(balanceRewardToken);
```

Recommendation

The contract should use a trusted external source. A trusted source could be either a commonly recognized or an audited contract. The pointing addresses should not be able to change after the initialization.

PVC - Price Volatility Concern

Criticality	minor / informative
Location	contract.sol#L1298
Status	Unresolved

Description

The swapTokensAtAmount 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 setSwapTokensAtAmount(uint256 newAmount) external onlyOwner{
        require(newAmount > totalSupply() / 100000, "SwapTokensAtAmount must be
greater than 0.001% of total supply");
        swapTokensAtAmount = newAmount;
        emit SwapTokensAtAmountUpdated(newAmount);
}
```

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.

RSML - Redundant SafeMath Library

Criticality	minor / informative
Location	contract.sol#L49
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.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor / informative
Location	contract.sol#L1107,967,790,1318,756,235,252,626,234,672,680,1100,676,272,684
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.

```
_charityWallet
DEAD
_account
_maxWalletAmount
_newMinimumBalance
PERMIT_TYPEHASH
MINIMUM_LIQUIDITY
magnitude
DOMAIN_SEPARATOR
...
```

Recommendation

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

L05 - Unused State Variable

Criticality	minor / informative
Location	contract.sol#L103
Status	Unresolved

Description

There are segments that contain unused state variables.

```
MAX_INT256
```

Recommendation

Remove unused state variables.

L07 - Missing Events Arithmetic

Criticality	minor / informative
Location	contract.sol#L778
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.

```
lastProcessedIndex = index
```

Recommendation

Emit an event for critical parameter changes.

L09 - Dead Code Elimination

Criticality	minor / informative
Location	contract.sol#L446,130,450,455,689,419,442,438
Status	Unresolved

Description

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

```
functionCallWithValue
abs
_functionCallWithValue
_transfer
isContract
functionCall
```

Recommendation

Remove unused functions.

L11 - Unnecessary Boolean equality

Criticality	minor / informative
Location	contract.sol#L1114
Status	Unresolved

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.

```
_isExcludedFromMaxWalletLimit[from] == false &&
_isExcludedFromMaxWalletLimit[to] == false && to != uniswapV2Pair
```

Recommendation

Remove the equality to the boolean constant.

L12 - Using Variables before Declaration

Criticality	minor / informative
Location	contract.sol#L1237
Status	Unresolved

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
claims
iterations
```

Recommendation

The variables should be declared before any usage of them.

L13 - Divide before Multiply Operation

Criticality	minor / informative
Location	contract.sol#L1114
Status	Unresolved

Description

Performing divisions before multiplications may cause lose of prediction.

```
fees = (amount * _totalFees) / 100
```

Recommendation

The multiplications should be prior to the divisions.

L14 - Uninitialized Variables in Local Scope

Criticality	minor / informative
Location	contract.sol#L1146,1237,1192,1008
Status	Unresolved

Description

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

```
liquidityTokens
claims
burnTaxShare
lastProcessedIndex
router
iterations
```

Recommendation

All the local scoped variables should be initialized.

L15 - Local Scope Variable Shadowing

Criticality	minor / informative
Location	contract.sol#L635,684,676,672,680
Status	Unresolved

Description

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

```
_name
_owner
_symbol
```

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		
Ownable	Implementation	Context		
	<Constructor>	Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_transferOwnership	Internal	✓	
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		
IterableMapping	Library			
	get	Public		-
	getIndexOfKey	Public		-
	getKeyAtIndex	Public		-
	size	Public		-
	set	Public	✓	-
	remove	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		-

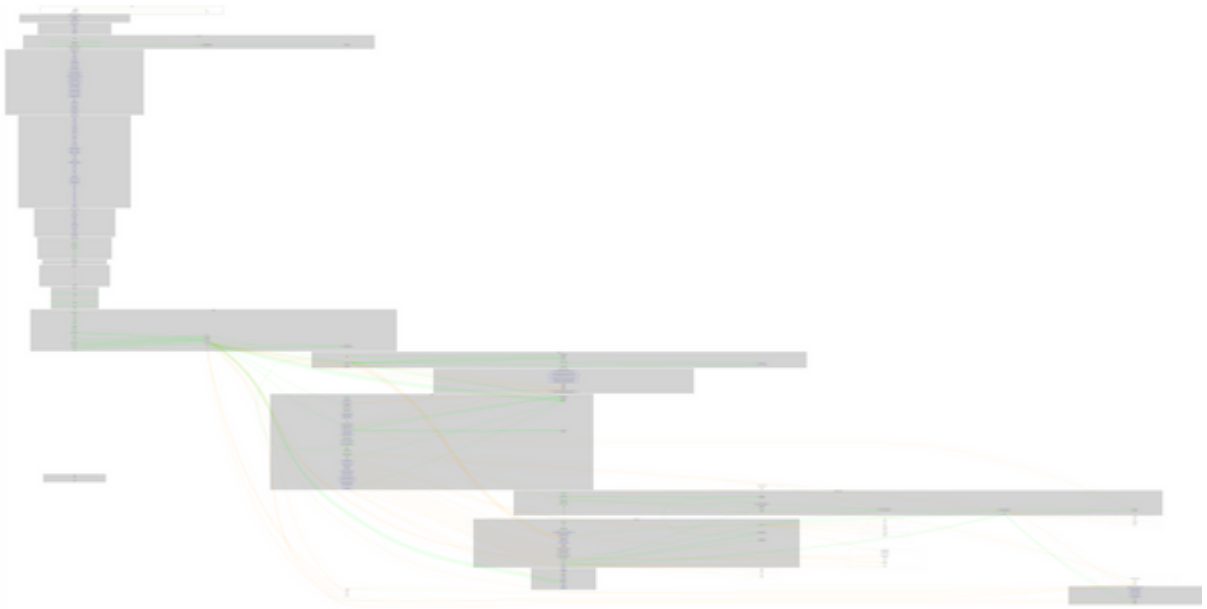
	getAmountIn	External		-
	getAmountsOut	External		-
	getAmountsIn	External		-
IUniswapV2Router02	Interface	IUniswapV2Router01		
	removeLiquidityETHSupportingFeeOnTransferTokens	External	✓	-
	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External	✓	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	allowance	External		-
	transfer	External	✓	-
	approve	External	✓	-
	transferFrom	External	✓	-
Address	Library			
	isContract	Internal		
	sendValue	Internal	✓	
	functionCall	Internal	✓	
	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	
	_functionCallWithValue	Private	✓	
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	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_beforeTokenTransfer	Internal	✓	
DividendPayingTokenInterface	Interface			
	dividendOf	External		-
	withdrawDividend	External	✓	-
DividendPayingTokenOptionalInterface	Interface			
	withdrawableDividendOf	External		-
	withdrawnDividendOf	External		-
	accumulativeDividendOf	External		-

DividendPayingToken	Implementation	ERC20, Ownable, DividendPayingTokenInterface, DividendPayingTokenOptionalInterface		
	<Constructor>	Public	✓	ERC20
	distributeDividends	Public	✓	onlyOwner
	withdrawDividend	Public	✓	-
	_withdrawDividendOfUser	Internal	✓	
	dividendOf	Public		-
	withdrawableDividendOf	Public		-
	withdrawnDividendOf	Public		-
	accumulativeDividendOf	Public		-
	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_setBalance	Internal	✓	
DividendTracker	Implementation	Ownable, DividendPayingToken		
	<Constructor>	Public	✓	DividendPayingToken
	_transfer	Internal		
	withdrawDividend	Public		-
	updateMinimumTokenBalanceForDividends	External	✓	onlyOwner
	excludeFromDividends	External	✓	onlyOwner
	updateClaimWait	External	✓	onlyOwner
	setLastProcessedIndex	External	✓	onlyOwner
	getLastProcessedIndex	External		-
	getNumberOfTokenHolders	External		-
	getAccount	Public		-
	getAccountAtIndex	Public		-
	canAutoClaim	Private		

	setBalance	External	✓	onlyOwner
	process	Public	✓	-
	processAccount	Public	✓	onlyOwner
SantaCoin	Implementation	ERC20, Ownable		
	<Constructor>	Public	✓	ERC20
	<Receive Ether>	External	Payable	-
	claimStuckTokens	External	✓	onlyOwner
	excludeFromFees	External	✓	onlyOwner
	isExcludedFromFees	Public		-
	changeMarketingWallet	External	✓	onlyOwner
	changeCharityWallet	External	✓	onlyOwner
	_transfer	Internal	✓	
	swapAndLiquify	Private	✓	
	swapAndSendDividends	Private	✓	
	setSwapTokensAtAmount	External	✓	onlyOwner
	setEnableMaxWalletLimit	External	✓	onlyOwner
	setMaxWalletAmount	External	✓	onlyOwner
	setExcludeFromMaxWallet	External	✓	onlyOwner
	isExcludedFromMaxWalletLimit	Public		-
	getClaimWait	External		-
	getTotalDividendsDistributed	External		-
	withdrawableDividendOf	Public		-
	dividendTokenBalanceOf	Public		-
	totalRewardsEarned	Public		-
	excludeFromDividends	External	✓	onlyOwner
	getAccountDividendsInfo	External		-
	getAccountDividendsInfoAtIndex	External		-
	processDividendTracker	External	✓	-
	claim	External	✓	-
	getLastProcessedIndex	External		-
	getNumberOfDividendTokenHolders	External		-

Contract Flow



Summary

Santa Coin is an interesting project that has a friendly and growing community. The Smart Contract analysis reported no compiler error or critical issues. The contract Owner can access some admin functions that can not be used in a malicious way to disturb the users' transactions. There is also a limit of max 10% fees.

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