Cyberscope

Audit Report

Sleepe Token

May 2022

Type	BEP20
Network	BSC
Address	0xd1daa2b21483240109eb30f88cc1ad90faa983c1
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	Token
Compiler Version	v0.7.4+commit.3f05b770
Optimization	200 runs
Licence	GNU GPLv3
Explorer	https://bscscan.com/token/0xd1DAA2b21483240109eb30F88CC1aD90faA983c1
Symbol	SLE
Decimals	18
Total Supply	1,000,000,000
Domain	sleepe.io

Source Files

Filename	**SHA256**
contract.sol	a4ffd3f386b2f2330c69c350b5a9431e2dd74d16d9e3dd79e2b996df1dbf452c

Audit Updates

Initial Audit	9th 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#L612, 1436

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 _feeTransfer with a higher value than PERCENT_DIVIDER by calling the changeFee function.

```
uint256 feeTransfer = amount.mul(_feeTransfer).div(PERCENTS_DIVIDER);
uint256 amountAfterFee = amount.sub(feeTransfer);
```

The contract owner can also convert the contract into a honeypot and prevent users from selling by setting the _numTokensSellToAddToLiquidity to zero.

```
if (amount > _numTokensSellToAddToLiquidity && limitSell== true &&
recipient==uniswapV2Pair) {
            revert("Limit Sell");
        }
```

Recommendation

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

The contract could embody a check for not allowing setting the _numTokensSellToAddToLiquidity less than a reasonable amount. A suggested implementation could check that the minimum 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.

OCTD - Owner Contract Tokens Drain

Criticality	minor
Location	contract.sol#L983

Description

The contract owner has the authority to claim all the balance of the contract. The owner may take advantage of it by calling the `transferToken` function.

```
function transferToken(
     address coinAddress,
     uint256 value,
     address payable to
 ) public onlyOwner {
     if (coinAddress == address(0)) {
         return to.transfer(value);
     }
     IERC20(coinAddress).transfer(to, value);
 }
```

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.

ELFM - Exceed Limit Fees Manipulation

Criticality	critical
Location	contract.sol#L1014, 1038

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 changeFee and/or setFeeTransfer functions with a high percentage value.

```
function changeFee(uint256 feeTransfer) public onlyOwner {
        _feeTransfer = feeTransfer;
    }
```

```
function setFeeTransfer(uint256 newFeeTransfer) public onlyOwner {
        require(
            newFeeTransfer < PERCENTS_DIVIDER,
            "Fee transfer cannot higher than 1000"
        );
        _feeTransfer = newFeeTransfer;
    }
```

Recommendation

The contract could embody a check for the maximum acceptable value without allowing the fees to be higher than 25%.

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.

MT - Mint Tokens

Criticality	critical
Location	contract.sol#L907

Description

The contract owner has the authority to mint tokens. The owner may take advantage of it by calling the mint function. As a result the contract tokens will be highly inflated

```
function mint(uint256 amount) public onlyOwner returns (bool) {
        require(_mintable, "this token is not mintable");
        _mint(_msgSender(), amount);
        return true;
    }
```

Recommendation

The owner should carefully manage the credentials of the owner's account. We advised considering an extra-strong security mechanism that the actions may be quarantined by many users instead of one. The owner could also renounce the contract ownership for a period of time or pass the access to the zero address.

BC - Blacklisted Contracts

Criticality	critical
Location	contract.sol#L780

Description

The contract owner has the authority to stop contracts from transactions. The owner may take advantage of it by calling the modifyBlackList function.

```
require(
     !blackList[sender],
     "ERC20: transfer to the black list address"
);
require(
     !blackList[recipient],
     "ERC20: transfer to the black list address"
);
```

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
	CR	Code Repetition
	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
	L11	Unnecessary Boolean equality
	L14	Uninitialized Variables in Local Scope

CR - Code Repetition

Criticality	minor
Location	contract.sol#L1010, 1034

Description

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

```
function changeFeeWallet(address feeWallet) public onlyOwner {
        _feeWallet = feeWallet;
}
```

```
function setFeeWallet(address newFeeWallet) public onlyOwner {
        _feeWallet = newFeeWallet;
}
```

Recommendation

Remove changeFeeWallet or setFeeWallet function and keep the other code segment.

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L426,435,444,449,457,549,557,574,581,588,606,635,652,675,716,743,907,913,918,923,935,939,951,955,967,983,994,1006,1010,1014,1018,1030,1034,1038,1417

Description

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

```
burn
setFeeTransfer
setFeeWallet
isExcludedFromPool
modifyWhiteListPool
changeFee
changeFeeWallet
isExcludedFromBot
modifyWhiteListBot
...
```

Recommendation

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

L02 - State Variables could be Declared Constant

Criticality	minor
Location	contract.sol#L1400

Description

Constant state variables should be declared constant to save gas.

```
maxSupply
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L913,971,975,979,520,521,523,1114,1116,1146,1190

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.

```
WETH
MINIMUM_LIQUIDITY
PERMIT_TYPEHASH
DOMAIN_SEPARATOR
_feeWallet
_feeTransfer
_numTokensSellToAddToLiquidity
_enable
_pmintable
...
```

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#L918

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.

```
_numTokensSellToAddToLiquidity = numTokensSellToAddToLiquidity
```

Recommendation

Emit an event for critical parameter changes.

L09 - Dead Code Elimination

Criticality	minor
Location	contract.sol#L874

Description

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

```
_setupDecimals
```

Recommendation

Remove unused functions.

L11 - Unnecessary Boolean equality

Criticality	minor
Location	contract.sol#L606,675,1421

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.

```
amount > _numTokensSellToAddToLiquidity && limitSell == true && recipient ==
uniswapV2Pair
sender != owner() && whiteListSender[_msgSender()] == false &&
whiteListSender[sender] == false && whiteListReceiver[recipient] == false
_msgSender() != owner() && whiteListSender[_msgSender()] == false &&
whiteListReceiver[recipient] == false
_msgSender() == owner() || whiteListSender[_msgSender()] == true ||
whiteListReceiver[recipient] == true
```

Recommendation

Remove the equality to the boolean constant.

L14 - Uninitialized Variables in Local Scope

Criticality	minor
Location	contract.sol#L998,962,1022,1001,959,930,927,946,943,1025

Description

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

```
index_scope_0
index
...
```

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		
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	✓	-

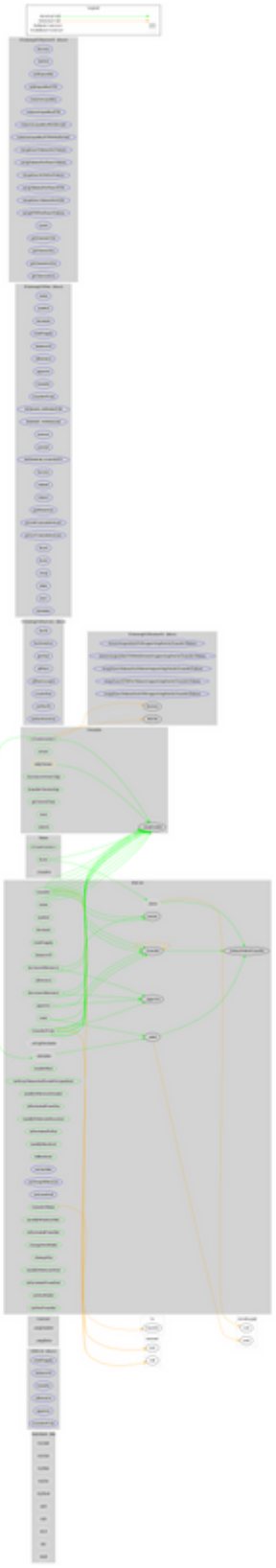
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	geUnlockTime	Public		-
	lock	Public	✓	onlyOwner
	unlock	Public	✓	-
ERC20	Implementation	Context, IERC20, Ownable		
	_initialize	Internal	✓	
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_setupDecimals	Internal	✓	
	_beforeTokenTransfer	Internal	✓	
	mint	Public	✓	onlyOwner
	enableMint	Public	✓	onlyOwner
	setNumTokensSellToAddToLiquidityt	Public	✓	onlyOwner
	modifyWhiteListSender	Public	✓	onlyOwner
	isExcludedFromFee	Public		-
	modifyWhiteListReceiver	Public	✓	onlyOwner
	isExcludedToFee	Public		-

	modifyBlackList	Public	✓	onlyOwner
	isBlackList	Public		-
	setAntiBot	External	✓	onlyOwner
	setSwapWhiteList	External	✓	onlyOwner
	setLimitSell	External	✓	onlyOwner
	transferToken	Public	✓	onlyOwner
	modifyWhiteListBot	Public	✓	onlyOwner
	isExcludedFromBot	Public		-
	changeFeeWallet	Public	✓	onlyOwner
	changeFee	Public	✓	onlyOwner
	modifyWhiteListPool	Public	✓	onlyOwner
	isExcludedFromPool	Public		-
	setFeeWallet	Public	✓	onlyOwner
	setFeeTransfer	Public	✓	onlyOwner
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	✓	-
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	✓	-
Token	Implementation	ERC20		
	<Constructor>	Public	✓	-
	burn	Public	✓	-
	_transfer	Internal	✓	

Contract Flow



Domain Info

Domain Name	sleepe.io
Registry Domain ID	18529b9b6ec341fba16e599c7b315b95-DONUTS
Creation Date	2022-04-08T11:43:06Z
Updated Date	2022-04-13T11:43:08Z
Registry Expiry Date	2023-04-08T11:43:06Z
Registrar WHOIS Server	whois.discount-domain.com
Registrar URL	http://www.onamae.com
Registrar	GMO Internet, Inc. d/b/a Onamae.com
Registrar IANA ID	49

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, transferring tokens to the team’s wallet, manipulating fees, minting tokens and massively blacklisting addresses. The contract can be converted into a honeypot and prevent users from selling if the owner abuses the admin functions. The contract tokens will be highly inflated. 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