Cyberscope

Audit Report

Kakashi Sensei

June 2022

Type	BEP20
Network	BSC
Address	0x3ee99f1e4e88008ac56934d05a10f270d6fd691b
Audited by	© cyberscope

Table of Contents

Table of Contents 1
Contract Review 3
Source Files 3
Audit Updates 3
Contract Analysis 4
ST - Stop Transactions 5
Description 5
Recommendation 6
ELFM - Exceed Limit Fees Manipulation 7
Description 7
Recommendation 7
Contract Diagnostics 9
L01 - Public Function could be Declared External 10
Description 10
Recommendation 10
L02 - State Variables could be Declared Constant 11
Description 11
Recommendation 11
L04 - Conformance to Solidity Naming Conventions 12
Description 12
Recommendation 12
L07 - Missing Events Arithmetic 13
Description 13
Recommendation 13
L09 - Dead Code Elimination 14
Description 14

Recommendation 14
Contract Functions 15
Contract Flow 19
Domain Info 20
Summary 21
Disclaimer 22
About Cyberscope 23

Contract Review

Contract Name	KAKASHISENSEI
Compiler Version	v0.6.12+commit.27d51765
Optimization	200 runs
Licence	GNU GPLv2
Explorer	https://bscscan.com/token/0x3EE99F1e4e88008Ac56934d05A10F270d6Fd691b
Symbol	KAKASHI
Decimals	18
Total Supply	100,000,000,000
Domain	kakashisensei.net

Source Files

Filename	**SHA256**
contract.sol	71d9b09ceff3b3893d0de32f8e405447bfe76d2843fbe0491b386f37a9d736c0

Audit Updates

Initial Audit	2nd June 2022
Corrected	6th June 2022

Contract Analysis

Critical Medium Minor Pass

Severity	Code	Description	Status
●	ST	Contract Owner is not able to stop or pause transactions	Resolved
●	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%)	Resolved
●	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#L848, 834, 880, 721
Status	Resolved

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 `_maxTxAmount` to zero.

```
if(from != owner() && to != owner())
            require(amount <= _maxTxAmount, "Transfer amount exceeds the
maxTxAmount.");
```

The owner can also stop transactions for multiple users by calling the `setAntibotModeWhitelist` function.

```
if (!isAntibotModeEnabled) return;
        if (from == owner() || from == airdropContract) return;
        require(antibotModeWhitelist[from] && antibotModeWhitelist[to],
"Address not in antibot mode whitelist");
```

The contract owner can also convert the contract into a honeypot and prevent users from selling by increasing the selling taxes.

```
else if(isSell){
                _taxFee = _sellTaxFee;
                _advestisementFee = _sellAdvestisementFee;
                _burnFee = _sellBurnFee;
            }
```

```
uint256 tFee = calculateTaxFee(tAmount);
        uint256 tAdvertisement = calculateAdvestisementFee(tAmount);
        uint256 tBurn = calculateBurnFee(tAmount);

        uint256 tTransferAmount =
tAmount.sub(tFee).sub(tAdvertisement).sub(tBurn);
```

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 minimum amount should be more than a fixed percentage of the total supply.

The issue about antibotWhitelist will be resolved when the contract owner calls the `turnOffAntibotMode` function.

The contract could embody a check for not allowing setting the total tax fees more than 100%.

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.

Updated 06 June 2022

The team has renounced ownership and resolved the issues.

ELFM - Exceed Limit Fees Manipulation

Criticality	critical
Location	contract.sol#L685, 691, 696
Status	Resolved

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 `setTaxFeePercent` , `setBurnFee` and `setAdvestisementFeePercentage` function with a high percentage value.

```
function setTaxFeePercent(uint256 buyTaxFee, uint256 sellTaxFee) external
onlyOwner() {
        _buyTaxFee = buyTaxFee;
        _sellTaxFee = sellTaxFee;
    }
```

```
function setBurnFee(uint256 buyBurnFee, uint256 sellBurnFee) external
onlyOwner() {
        _buyBurnFee = buyBurnFee;
        _sellBurnFee = sellBurnFee;
    }
```

```
function setAdvestisementFeePercent(uint256 buyAdvestisementFee, uint256
sellAdvestisementFee) external onlyOwner() {
        _sellAdvestisementFee = sellAdvestisementFee;
        _buyAdvestisementFee = buyAdvestisementFee;
    }
```

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.

Updated 06 June 2022

The team has renounced ownership and resolved the issues.

Contract Diagnostics

Critical Medium Minor

Severity	Code	Description
	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

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L431,440,446,451,459,554,558,562,566,570,574,579,584,588,593,599,604,609,613,619,670,675,681,820,824,828

Description

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

```
setAntibotModeWhitelist
setAirdropContract
turnOffAntibotMode
includeInFee
manageAmmPairs
excludeFromFee
reflectionFromToken
totalFees
isExcludedFromReward
...
```

Recommendation

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

L02 - State Variables could be Declared Constant

Criticality	minor
Location	contract.sol#L533,503,501,502,497

Description

Constant state variables should be declared constant to save gas.

```
_tTotal
_symbol
_name
_decimals
BUSD
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L479,774,780,786,824,505,506,507,509,510,511,522,527,533

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.

```
BUSD
_maxTxAmount
_advestisementFee
_sellBurnFee
_sellAdvestisementFee
_sellTaxFee
_buyBurnFee
_buyAdvestisementFee
_buyTaxFee
...
```

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#L685,691,696,701

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.

```
_maxTxAmount = _tTotal.mul(maxTxPercent).div(10 ** 3)
_sellAdvestisementFee = sellAdvestisementFee
_buyBurnFee = buyBurnFee
_buyTaxFee = buyTaxFee
```

Recommendation

Emit an event for critical parameter changes.

L09 - Dead Code Elimination

Criticality	minor
Location	contract.sol#L357,317,327,342,352,264,291

Description

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

```
sendValue
isContract
functionCallWithValue
functionCall
_functionCallWithValue
```

Recommendation

Remove unused functions.

Contract Functions

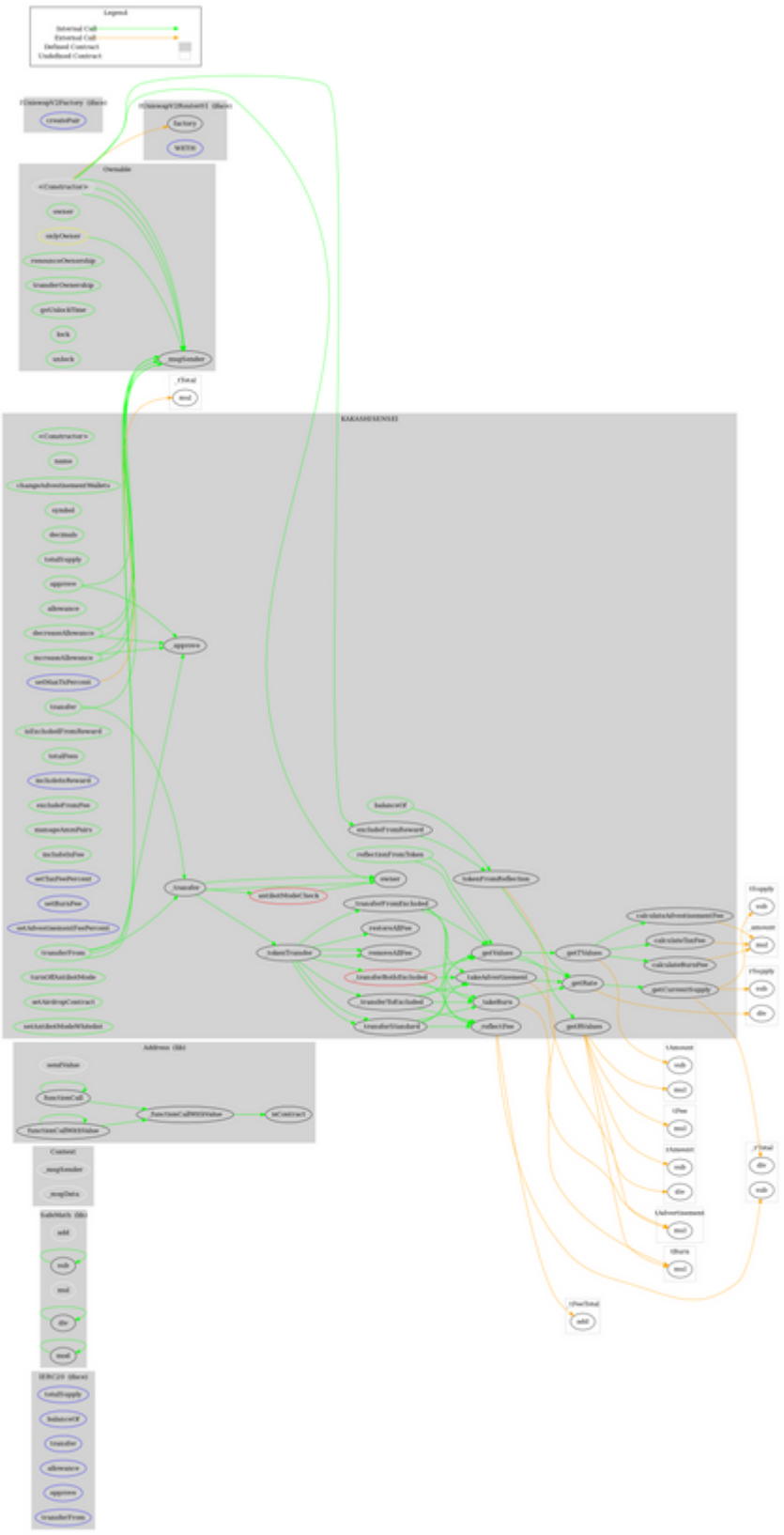
Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
SafeMath	Library			
	add	Internal		
	sub	Internal		
	sub	Internal		
	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>	Internal	✓	
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	geUnlockTime	Public		-
	lock	Public	✓	onlyOwner
	unlock	Public	✓	-
IUniswapV2Factory	Interface			
	createPair	External	✓	-
IUniswapV2Router01	Interface			
	factory	External		-
	WETH	External		-
KAKASHISENSEI	Implementation	Context, IERC20, Ownable		
	<Constructor>	Public	✓	-
	name	Public		-
	changeAdvestisementWallets	Public	✓	onlyOwner
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-

	decreaseAllowance	Public	✓	-
	isExcludedFromReward	Public		-
	totalFees	Public		-
	reflectionFromToken	Public		-
	tokenFromReflection	Public		-
	excludeFromReward	Public	✓	onlyOwner
	includeInReward	External	✓	onlyOwner
	_transferBothExcluded	Private	✓	
	excludeFromFee	Public	✓	onlyOwner
	manageAmmPairs	Public	✓	onlyOwner
	includeInFee	Public	✓	onlyOwner
	setTaxFeePercent	External	✓	onlyOwner
	setBurnFee	External	✓	onlyOwner
	setAdvestisementFeePercent	External	✓	onlyOwner
	setMaxTxPercent	External	✓	onlyOwner
	_reflectFee	Private	✓	
	_getValues	Private		
	_getTValues	Private		
	_getRValues	Private		
	_getRate	Private		
	_getCurrentSupply	Private		
	_takeAdvertisement	Private	✓	
	_takeBurn	Private	✓	
	calculateTaxFee	Private		
	calculateAdvestisementFee	Private		
	calculateBurnFee	Private		
	removeAllFee	Private	✓	
	restoreAllFee	Private	✓	
	_approve	Private	✓	
	turnOffAntibotMode	Public	✓	onlyOwner
	setAirdropContract	Public	✓	onlyOwner
	setAntibotModeWhitelist	Public	✓	onlyOwner
	antibotModeCheck	Private		
	_transfer	Private	✓	

	_tokenTransfer	Private	✓	
	_transferStandard	Private	✓	
	_transferToExcluded	Private	✓	
	_transferFromExcluded	Private	✓	

Contract Flow



Domain Info

Domain Name	kakashisensei.net
Registry Domain ID	2696462997_DOMAIN_NET-VRSN
Creation Date	2022-05-16T02:05:24.00Z
Updated Date	0001-01-01T00:00:00.00Z
Registry Expiry Date	2023-05-16T02:05:24.00Z
Registrar WHOIS Server	whois.namecheap.com
Registrar URL	http://www.namecheap.com
Registrar	NAMECHEAP INC
Registrar IANA ID	1068

The domain has been created 17 days before the creation of the audit. It will expire in 12 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 manipulating fees. 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.

Updated 06 June 2022

The team has renounced ownership and resolved the issues.

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