Cyberscope

Audit Report

Opulence OPEC

March 2022

Type	ERC20
Network	AVAX
Address	0x283366bb42ef49a994913BAF22263c6562e588a4
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	OPEC
Compiler Version	v0.8.9+commit.e5eed63a
Optimization	2000 runs
Licence	MIT
Explorer	https://snowtrace.io/token/0x283366bb42ef49a994913BAF22263c6562e588a4
Symbol	OPEC
Decimals	18
Total Supply	6,000,000

Source Files

Filename	**SHA256**
contract.sol	1f62057224f30668eb99d33dddc5a6eaf36895a0cf82f7f87c0ccae9d0c544f9

Audit Updates

Initial Audit	28th March 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#L1142,1154,1161

Description

The contract owner has the authority to stop the sales for all users excluding the owner. The owner may take advantage of it by setting the `_liquidity_sell_tax` to a high value. This can cause the contract to operate as a honeypot.

```
if (!_isExcludedFromFee[from] && to == address(_uniswapV2Pair)) {
    fees = (amount / 100) * (_liquidity_sell_tax - passedDays * 10);
    amount = amount.sub(fees);
```

The contract owner has the authority to stop the buys and transfers for all users excluding the owner. The owner may take advantage of it by setting the `_balanceLimit` to zero.

```
if (!_isExcludedFromFee[to] && _isMaxLimit) {
    require(amount + balanceOf(to) <= _balanceLimit, "OPEC: TRANSFER RECIPIENT
BALANCE LIMIT");
}
```

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

```
if (block.timestamp - _launchTime < 10 minutes) {
    require(amount <= 10 * 1e18, "OPEC: TRANSFER BUY LIMIT ERROR");
} else {
    require(amount <= _maxBuyLimit, "OPEC: TRANSFER BUY LIMIT ERROR");
}
```

Recommendation

The contract could embody a check for not allowing setting the _balanceLimit and _balanceLimit 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.

Read more in the fees manipulation section.

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#L1115

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 `setUintParameter` function with SELL_TAX and a high percentage value.

```
function setUintParameter(PROTOCOL_PARAMETER _parameter, uint256 _value)
external onlyManager {
    if (_parameter == PROTOCOL_PARAMETER.SELL_TAX) {
        _liquidity_sell_tax = _value;
    } else if (_parameter == PROTOCOL_PARAMETER.LIMIT_AMOUNT) {
        _balanceLimit = _value;
    } else if (_parameter == PROTOCOL_PARAMETER.LIMIT_BUY) {
        _maxBuyLimit = _value;
    }
}
```

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.

MT - Mint Tokens

Criticality	critical
Location	contract.sol#L1087

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(address account_, uint256 amount_) external onlyPolicy {
    _mint(account_, amount_);
}
```

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	medium
Location	contract.sol#L1138

Description

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

```
require(!_isBlacklisted[from] && !_isBlacklisted[to], "OPEC: TRANSFER
BLACKLIST");
```

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
●	FSA	Fixed Swap Address
●	L01	Public Function could be Declared External
●	L02	State Variables could be Declared Constant
●	L04	Conformance to Solidity Naming Conventions
●	L06	Missing Events Access Control
●	L09	Dead Code Elimination
●	L11	Unnecessary Boolean equality
●	L13	Divide before Multiply Operation

FSA - Fixed Swap Address

Criticality	minor
Location	contract.sol#L1074

Description

The swap address is assigned once in the constructor and it can not be changed. The decentralized swaps sometimes create a new swap version or abandon the current. A contract that cannot change the swap address may not be able to catch-up the upgrade.

```
_uniswapV2Router = IJoeRouter02(uniV2Router);

_uniswapV2Pair =
IUniswapV2Pair(IJoeFactory(_uniswapV2Router.factory()).createPair(address(this),
_uniswapV2Router.WAVAX()));
```

Recommendation

It could be better to allow the swap address mutation in case of future swap updates.

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L658,666,683,690,715,728,745,768,797,824 and 3 more

Description

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

```
transferOwnership
renounceOwnership
owner
decreaseAllowance
increaseAllowance
transferFrom
approve
allowance
transfer
...
```

Recommendation

Use the external attribute for functions never called from the contract

L02 - State Variables could be Declared Constant

Criticality	minor
Location	contract.sol#L1033

Description

Constant state variables should be declared constant to save gas.

```
_isSwapping
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L8,238,240,271,340,1091,1095,1099,1107,1115 and 22 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.

```
_liquidity_sell_tax
_maxBuyLimit
_balanceLimit
_launchTime
_isLaunched
_isMaxLimit
_isSwapping
_isTaxable
_policy
...
```

Recommendation

Follow the Solidity naming convention.
https://docs.soliditylang.org/en/v0.4.25/style-guide.html#naming-conventions

L06 - Missing Events Access Control

Criticality	minor
Location	contract.sol#L1091

Description

Detected missing events for critical access control parameters. There are functions that have no event emitted, so it is difficult to track off-chain changes.

```
_policy = _address
```

Recommendation

Emit an event for critical parameter changes.

L09 - Dead Code Elimination

Criticality	minor
Location	contract.sol#L902

Description

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

```
_burn
```

Recommendation

Remove unused functions.

L11 - Unnecessary Boolean equality

Criticality	minor
Location	contract.sol#L1043

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.

```
require(bool,string)(_managers[msg.sender] == true,NOT MANAGER)
```

Recommendation

Remove the equality to the boolean constant.

L13 - Divide before Multiply Operation

Criticality	minor
Location	contract.sol#L1136

Description

Performing divisions before multiplications may cause lose of prediction.

```
fees = (amount / 100) * (_liquidity_sell_tax - passedDays * 10)
```

Recommendation

The multiplications should be prior to the divisions.

Contract Functions

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
IJoeRouter01	Interface			
	factory	External		-
	WAVAX	External		-
	addLiquidity	External	✓	-
	addLiquidityAVAX	External	Payable	-
	removeLiquidity	External	✓	-
	removeLiquidityAVAX	External	✓	-
	removeLiquidityWithPermit	External	✓	-
	removeLiquidityAVAXWithPermit	External	✓	-
	swapExactTokensForTokens	External	✓	-
	swapTokensForExactTokens	External	✓	-
	swapExactAVAXForTokens	External	Payable	-
	swapTokensForExactAVAX	External	✓	-
	swapExactTokensForAVAX	External	✓	-
	swapAVAXForExactTokens	External	Payable	-
	quote	External		-
	getAmountOut	External		-
	getAmountIn	External		-
	getAmountsOut	External		-
	getAmountsIn	External		-
IJoeRouter02	Interface	IJoeRouter01		
	removeLiquidityAVAXSupportingFeeOnTransferTokens	External	✓	-
	removeLiquidityAVAXWithPermitSupportingFeeOnTransferTokens	External	✓	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactAVAXForTokensSupporting	External	Payable	-

	FeeOnTransferTokens			
	swapExactTokensForAVAXSupportingFeeOnTransferTokens	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	✓	-
IJoeFactory	Interface			
	feeTo	External		-

	feeToSetter	External		-
	migrator	External		-
	getPair	External		-
	allPairs	External		-
	allPairsLength	External		-
	createPair	External	✓	-
	setFeeTo	External	✓	-
	setFeeToSetter	External	✓	-
	setMigrator	External	✓	-
IOPEC	Interface			
	setCheckParameter	External	✓	-
	setBooleanParameter	External	✓	-
	setUintParameter	External	✓	-
	setAddressParameter	External	✓	-
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
IERC20Metadata	Interface	IERC20		
	name	External		-
	symbol	External		-
	decimals	External		-
SafeMath	Library			
	add	Internal		
	sub	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		

	div	Internal		
	mod	Internal		
	mod	Internal		
ERC20	Implementation	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	✓	
Ownable	Implementation			
	<Constructor>	Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
OPEC	Implementation	IOPEC, ERC20, Ownable		
	<Constructor>	Public	✓	ERC20
	<Receive Ether>	External	Payable	-
	mint	External	✓	onlyPolicy

	setPolicy	External	✓	onlyOwner
	setManager	External	✓	onlyOwner
	setCheckParameter	External	✓	onlyManager
	setBooleanParameter	External	✓	onlyManager
	setUintParameter	External	✓	onlyManager
	setAddressParameter	External	✓	onlyManager
	setLaunch	External	✓	onlyManager
	_transfer	Internal	✓	

Contract Flow

Summary

There are some functions that can be abused by the owner, like manipulating fees, stopping transactions, minting tokens and blacklisting addresses. 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.

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