Cyberscope

Audit Report

ZILLION AAKAR XO

November 2022

Type	BEP20
Network	BSC
Address	0x9A2478C4036548864d96a97Fbf93f6a3341fedac
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	ZAX
Compiler Version	v0.8.7+commit.e28d00a7
Optimization	200 runs
Licence	Unlicense
Explorer	https://bscscan.com/token/0x9A2478C4036548864d96a97Fbf93f6a3341fedac
Symbol	ZAX
Decimals	9
Total Supply	21,000,000
Domain	https://www.zillionxo.io

Source Files

Filename	SHA256
contract.sol	2b4d316ff85d8e2acfa1a06564cbddb425a3a2f6515be98b2d281b4b8ad62b11

Audit Updates

Initial Audit	10th November 2022
Corrected	

Contract Analysis

● Critical ● Medium ● Minor / Informative ● Pass

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

ST - Stops Transactions

Criticality	critical
Location	contract.sol#L567
Status	Unresolved

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 `maxSellLimit` to zero. As a result, the contract may operate as a honeypot. Additionally, the owner has the authority to stop transactions by setting `maxBuyLimit` and `maxWalletLimit` to zero.

```
if (from == pair && !_isExcludedFromFee[to] && !swapping) {
    require(amount <= maxBuyLimit, "You are exceeding maxBuyLimit");
    require(
        balanceOf(to) + amount <= maxWalletLimit,
        "You are exceeding maxWalletLimit"
    );
}

if (
    from != pair && !_isExcludedFromFee[to] && !_isExcludedFromFee[from] && !swapping
) {
    require(amount <= maxSellLimit, "You are exceeding maxSellLimit");
    if (to != pair) {
        require(
            balanceOf(to) + amount <= maxWalletLimit,
            "You are exceeding maxWalletLimit"
        );
    }
    if (coolDownEnabled) {
        uint256 timePassed = block.timestamp - _lastSell[from];
        require(timePassed >= coolDownTime, "Cooldown enabled");
        _lastSell[from] = block.timestamp;
    }
}
```

Recommendation

The contract could embody a check for not allowing setting the maxSellLimit, maxBuyLimit, and maxWalletLimit 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.

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 - Transfers Contract's Tokens

Criticality	minor / informative
Location	contract.sol#L796
Status	Unresolved

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 `rescueAnyBEP20Tokens` function.

```
function rescueAnyBEP20Tokens(
        address _tokenAddr,
        address _to,
        uint256 _amount
    ) public onlyOwner {
        IERC20(_tokenAddr).transfer(_to, _amount);
}
```

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 - Exceeds Fees Limit

Criticality	minor / informative
Location	contract.sol#L390
Status	Unresolved

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 setSellTaxes function with a high percentage value.

```
function setSellTaxes(
        uint256 _rfi,
        uint256 _marketing,
        uint256 _liquidity,
        uint256 _buyback
    ) public onlyOwner {
        sellTaxes = Taxes(_rfi, _marketing, _liquidity, _buyback);
        require((_rfi + _marketing + _liquidity + _buyback) <= 30, "Must keep fees at 30% or
less");
        emit FeesChanged();
    }
```

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.

BC - Blacklists Addresses

Criticality	critical
Location	contract.sol#L760,764
Status	Unresolved

Description

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

```
function updateIsBlacklisted(address account, bool state) external onlyOwner {
    _isBlacklisted[account] = state;
  }

function bulkIsBlacklisted(address[] memory accounts, bool state) external onlyOwner {
    for (uint256 i = 0; i < accounts.length; i++) {
        _isBlacklisted[accounts[i]] = state;
    }
}
```

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 / Informative

Severity	Code	Description	Status
●	ZD	Zero Division	Unresolved
●	STC	Succeeded Transfer Check	Unresolved
●	SPC	Solidity Precision Concern	Unresolved
●	CO	Code Optimization	Unresolved
●	MC	Missing Check	Unresolved
●	L02	State Variables could be Declared Constant	Unresolved
●	L03	Redundant Statements	Unresolved
●	L04	Conformance to Solidity Naming Conventions	Unresolved
●	L07	Missing Events Arithmetic	Unresolved
●	L11	Unnecessary Boolean equality	Unresolved
●	L13	Divide before Multiply Operation	Unresolved

ZD - Zero Division

Criticality	critical
Location	contract.sol#L670
Status	Unresolved

Description

The contract is using variables that may be set to zero as denominators. The variables liquidity, marketing, and buyback can be set to zero. As a result, the transactions will revert.

```
function swapAndLiquify(uint256 contractBalance, Taxes memory temp) private lockTheSwap {
        uint256 denominator = (temp.liquidity +
            temp.marketing +
            temp.buyback) * 2;
        uint256 tokensToAddLiquidityWith = (contractBalance * temp.liquidity) / denominator;
```

Recommendation

The contract should prevent those variables to be set to zero or should not allow executing the corresponding statements.

STC - Succeeded Transfer Check

Criticality	minor / informative
Location	contract.sol#L805
Status	Unresolved

Description

According to the ERC20 specification, the transfer methods should be checked if the result is successful. Otherwise, the contract may wrongly assume that the transfer has been established.

```
IERC20(_tokenAddr).transfer(_to, _amount);
```

Recommendation

The contract should check if the result of the transfer methods is successful.

SPC - Solidity Precision Concern

Criticality	minor / informative
Location	contract.sol#L690,694
Status	Unresolved

Description

In Solidity, all integer division rounds down to the nearest integer. The contract distributes the funds proportional to the recipients. These calculations may produce unexpected leftover funds for the contract.

```
uint256 marketingAmt = unitBalance * 2 * temp.marketing;
if (marketingAmt > 0) {
    payable(marketingWallet).sendValue(marketingAmt);
}
uint256 buybackAmt = unitBalance * 2 * temp.buyback;
if (buybackAmt > 0) {
    payable(buybackWallet).sendValue(buybackAmt);
}
```

Recommendation

In Solidity, all integer division rounds down to the nearest integer. The contract distributes the funds proportional to the recipients. These calculations may produce unexpected leftover funds for the contract.

CO - Code Optimization

Criticality	minor / informative
Location	contract.sol#L126
Status	Unresolved

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.

The data structure allowedTransfer is redundant. Because it is not used in the contracts' implementation.

```
mapping(address => bool) public allowedTransfer;
```

Recommendation

Rewrite some code segments so the runtime will be more performant. The contract could remove the allowedTransfer data structure.

MC - Missing Check

Criticality	minor / informative
Location	contract.sol#L739,743
Status	Unresolved

Description

The contract is processing variables that have not been properly sanitized and checked that they form the proper shape. These variables may produce vulnerability issues. The address aguments are not sanitized properly.

```
function updateMarketingWallet(address newWallet) external onlyOwner {
    marketingWallet = newWallet;
}

function updatebuybackWallet(address newWallet) external onlyOwner {
    buybackWallet = newWallet;
}
```

Recommendation

The contract should properly check the variables according to the required specifications. The address should not be set to zero address.

L02 - State Variables could be Declared Constant

Criticality	minor / informative
Location	contract.sol#L142,153
Status	Unresolved

Description

Constant state variables should be declared constant to save gas.

```
_tTotal
deadWallet
```

Recommendation

Add the constant attribute to state variables that never change.

L03 - Redundant Statements

Criticality	minor / informative
Location	contract.sol#L26
Status	Unresolved

Description

The contract contains statements that are not used and have no effect. As a result, those segments increase the code size of the contract unnecessarily.

Context

Recommendation

Remove the redundant statements in order to decrease the code size.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor / informative
Location	contract.sol#L798,797,180,158,388,377,157,389,378,150,387,752,78,321,376,390,139,379,799
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.

```
_to
_tokenAddr
valuesFromGetValues
_symbol
_marketing
_name
_liquidity
genesis_block
_rfi
...
```

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 / informative
Location	contract.sol#L776,319,748,781,743
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.

```
maxBuyLimit = maxBuy * 10 ** decimals()
deadline = _deadline
swapTokensAtAmount = amount * 10 ** _decimals
maxWalletLimit = amount * 10 ** decimals()
coolDownTime = time * 1
```

Recommendation

Emit an event for critical parameter changes.

L11 - Unnecessary Boolean equality

Criticality	minor / informative
Location	contract.sol#L319
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.

```
state == true
```

Recommendation

Remove the equality to the boolean constant.

L13 - Divide before Multiply Operation

Criticality	minor / informative
Location	contract.sol#L666
Status	Unresolved

Description

Performing divisions before multiplications may cause lose of prediction.

```
unitBalance = deltaBalance / (denominator - temp.liquidity)
```

Recommendation

The multiplications should be prior to the divisions.

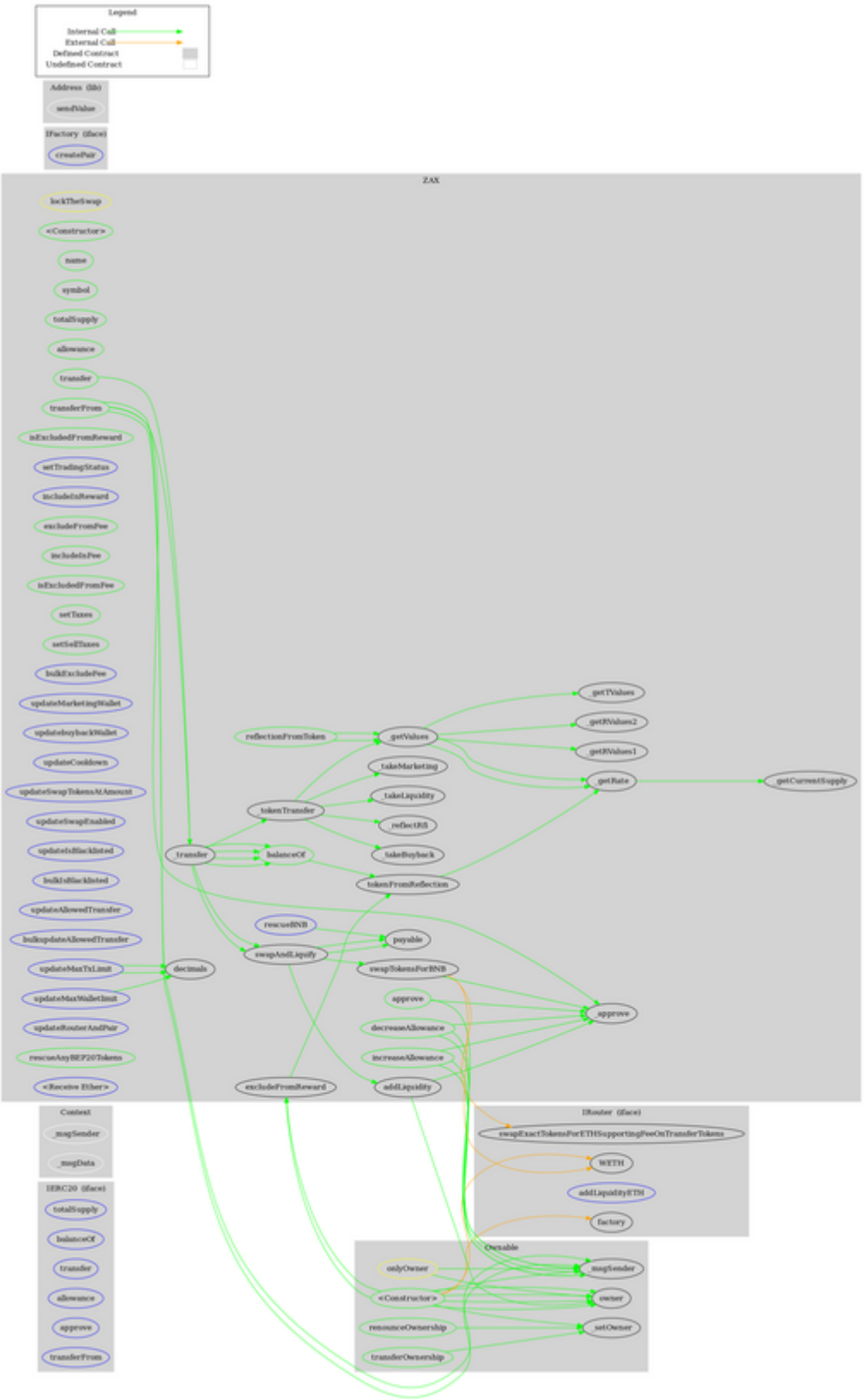
Contract Functions

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
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
	_setOwner	Private	✓	
IFactory	Interface			
	createPair	External	✓	-
IRouter	Interface			
	factory	External		-
	WETH	External		-
	addLiquidityETH	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-

Address	Library			
	sendValue	Internal	✓	
ZAX	Implementation	Context, IERC20, Ownable		
	<Constructor>	Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	transfer	Public	✓	-
	isExcludedFromReward	Public		-
	reflectionFromToken	Public		-
	setTradingStatus	External	✓	onlyOwner
	tokenFromReflection	Public		-
	excludeFromReward	Public	✓	onlyOwner
	includeInReward	External	✓	onlyOwner
	excludeFromFee	Public	✓	onlyOwner
	includeInFee	Public	✓	onlyOwner
	isExcludedFromFee	Public		-
	setTaxes	Public	✓	onlyOwner
	setSellTaxes	Public	✓	onlyOwner
	_reflectRfi	Private	✓	
	_takeLiquidity	Private	✓	
	_takeMarketing	Private	✓	
	_takeBuyback	Private	✓	
	_getValues	Private		
	_getTValues	Private		

	_getRValues1	Private		
	_getRValues2	Private		
	_getRate	Private		
	_getCurrentSupply	Private		
	_approve	Private	✓	
	_transfer	Private	✓	
	_tokenTransfer	Private	✓	
	swapAndLiquify	Private	✓	lockTheSwap
	addLiquidity	Private	✓	
	swapTokensForBNB	Private	✓	
	bulkExcludeFee	External	✓	onlyOwner
	updateMarketingWallet	External	✓	onlyOwner
	updatebuybackWallet	External	✓	onlyOwner
	updateCooldown	External	✓	onlyOwner
	updateSwapTokensAtAmount	External	✓	onlyOwner
	updateSwapEnabled	External	✓	onlyOwner
	updateIsBlacklisted	External	✓	onlyOwner
	bulkIsBlacklisted	External	✓	onlyOwner
	updateAllowedTransfer	External	✓	onlyOwner
	bulkupdateAllowedTransfer	External	✓	onlyOwner
	updateMaxTxLimit	External	✓	onlyOwner
	updateMaxWalletlimit	External	✓	onlyOwner
	updateRouterAndPair	External	✓	onlyOwner
	rescueBNB	External	✓	onlyOwner
	rescueAnyBEP20Tokens	Public	✓	onlyOwner
	<Receive Ether>	External	Payable	-

Contract Flow



Domain Info

Domain Name	zillionxo.io
Registry Domain ID	277e51e4eeb54205a43693710f47019b-DONUTS
Creation Date	2022-05-12T09:36:25Z
Updated Date	2022-06-08T17:16:26Z
Registry Expiry Date	2023-05-12T09:36:25Z
Registrar WHOIS Server	whois.advancedregistrar.com
Registrar URL	http://www.netearthone.com
Registrar	NetEarth One Inc. dba NetEarth
Registrar IANA ID	1005

The domain was created 6 months before the creation of the audit. It will expire in 6 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, transferring funds to the team's wallet, 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 implements a sell cooldown mechanism on consecutive sell transactions. Additionally, the contract applies a launch tax. The launch tax is initialized when the trading opens.

A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats. There is also a limit of max 20% buy fees and a max of 30% of sell fees.

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