Cyberscope

Audit Report

AFRICA DAO

August 2022

Type	BEP20
Network	BSC
Address	0x663D675395215d4d51E04B8d5B87B67952b6B5F
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	AfricaDaoToken
Compiler Version	v0.8.11+commit.d7f03943
Optimization	1 runs
Explorer	https://bscscan.com/token/0x663D675395215d4d51E04B8d5B87B67952bb8B5F
Symbol	AFRICA
Decimals	18
Total Supply	8,000,000,000
Domain	http://africadao.co

Source Files

Filename	**SHA256**
contract.sol	8a0b070e7d09d37b17f8cef58f8171317a67f64ac1af22e7259af563b797f8e8

Audit Updates

Initial Audit	8th August 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	minor
Location	contract.sol#L1267,1412

Description

The contract owner has the authority to delay sell transactions for all users, after their first transactions, excluding the owner. The contract owner has the authority to set the delay period to a max of 1day. The owner may take advantage of it by setting the `sellFrequency` to 1 day.

```
if (automatedMarketMakerPairs[to] && !_isExcludedFromFees[from] && sellFrequency > 0) {
    require(sellTimestamp[from] == 0 || sellTimestamp[from] + sellFrequency < block.timestamp,
"Too frequent to sell");
    sellTimestamp[from] = block.timestamp;
}
```

The contract owner has the authority to delay buy transactions up to one per block for all users excluding the owner. The owner may take advantage of it by setting the `transferDelayEnabled` to true and restrict buy transactions to 1 per block.

```
if (transferDelayEnabled){
    if (to != owner() && to != address(uniswapV2Router) && to != address(uniswapV2Pair)){
        require(_holderLastTransferTimestamp[tx.origin] < block.number, "_transfer:: Transfer Delay
enabled.  Only one purchase per block allowed.");
        _holderLastTransferTimestamp[tx.origin] = block.number;
    }
}
```

Recommendation

The contract could embody a check for not allowing setting the `sellFrequency` less than a reasonable amount.

The contract owner could delay the buy transactions for a specific period when the trade opens.

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
	STC	Succeeded Transfer Check
	BLC	Business Logic Concern
	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
	L08	Tautology or Contradiction
	L09	Dead Code Elimination
	L13	Divide before Multiply Operation
	L15	Local Scope Variable Shadowing

STC - Succeeded Transfer Check

Criticality	minor
Location	contract.sol#L1443,1450

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.

```
(success,) = address(devWallet).call{value: ethForDev}("");

if(liquidityTokens > 0 && ethForLiquidity > 0){
    addLiquidity(liquidityTokens, ethForLiquidity);
    emit SwapAndLiquify(amountToSwapForETH, ethForLiquidity, tokensForLiquidity);
}

(success,) = address(marketingWallet).call{value: address(this).balance}("");
```

Recommendation

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

BLC - Business Logic Concern

Criticality	minor
Location	contract.sol#L1455

Description

The business logic seems peculiar. The implementation may not follow the expected behavior.

Unsigned integers are always greater than or equal to zero.

```
require(_percent <= 1000 && _percent >= 0, "Must set auto LP burn percent between 0% and 10%");
```

Recommendation

The team is advised to carefully check if the implementation follows the expected business logic.

CR - Code Repetition

Criticality	minor
Location	contract.sol#L1461,1483

Description

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

The following code segment is repetitive in the autoBurnLiquidityPairTokens and autoBurnLiquidityPairTokens methods.

```
lastLpBurnTime = block.timestamp;

// get balance of liquidity pair
uint256 liquidityPairBalance = this.balanceOf(uniswapV2Pair);

// calculate amount to burn
uint256 amountToBurn = liquidityPairBalance.mul(percentForLPBurn).div(10000);

// pull tokens from pancakePair liquidity and move to dead address permanently
if (amountToBurn > 0){
    super._transfer(uniswapV2Pair, address(0xdead), amountToBurn);
}

//sync price since this is not in a swap transaction!
IUniswapV2Pair pair = IUniswapV2Pair(uniswapV2Pair);
pair.sync();
```

Recommendation

Create an internal function that contains the code segment and remove it from all the sections.

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L285,1261,567,214,328,197,240,308,575,189,1232,263

Description

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

```
approve
setAutomatedMarketMakerPair
name
transferOwnership
increaseAllowance
transfer
symbol
decreaseAllowance
decimals
...
```

Recommendation

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

L02 - State Variables could be Declared Constant

Criticality	minor
Location	contract.sol#L1043

Description

Constant state variables should be declared constant to save gas.

```
manualBurnFrequency
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L1027,1072,1206,1088,1214,1453,986,1003,985,1090,1222,818

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.

```
_liquidityFee
_marketingFee
WETH
_freq
_percent
devWalletUpdated
_Enabled
_devFee
DOMAIN_SEPARATOR
...
```

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#L1453,1187,1180,1222,1206,1214,1192

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.

```
maxWallet = newNum * (10 ** 18)
sellMarketingFee = _marketingFee
buyMarketingFee = _marketingFee
sellFrequency = _freq
swapTokensAtAmount = newAmount
maxTransactionAmount = newNum * (10 ** 18)
lpBurnFrequency = _frequencyInSeconds
```

Recommendation

Emit an event for critical parameter changes.

L08 - Tautology or Contradiction

Criticality	minor
Location	contract.sol#L1453

Description

Detects expressions that are tautologies or contradictions. For instance, an uint variable will always be greater than or equal to zero.

```
require(bool,string)(_percent <= 1000 && _percent >= 0,Must set auto LP burn percent between 0% and 10%)
```

Recommendation

Fix the incorrect comparison by changing the value type or the comparison.

L09 - Dead Code Elimination

Criticality	minor
Location	contract.sol#L407

Description

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

```
_burn
```

Recommendation

Remove unused functions.

L13 - Divide before Multiply Operation

Criticality	minor
Location	contract.sol#L1267

Description

Performing divisions before multiplications may cause lose of prediction.

```
fees = amount.mul(sellTotalFees).div(100)
fees = amount.mul(buyTotalFees).div(100)
tokensForMarketing += fees * sellMarketingFee / sellTotalFees
```

Recommendation

The multiplications should be prior to the divisions.

L15 - Local Scope Variable Shadowing

Criticality	minor
Location	contract.sol#L1121

Description

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

```
totalSupply
```

Recommendation

The local variables should have different names from the upper scoped variables.

Contract Functions

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
IERC20Metadata	Interface	IERC20		
	name	External		-
	symbol	External		-
	decimals	External		-
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
ERC20	Implementation	Context, IERC20, IERC20Metadata		
	<Constructor>	Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-

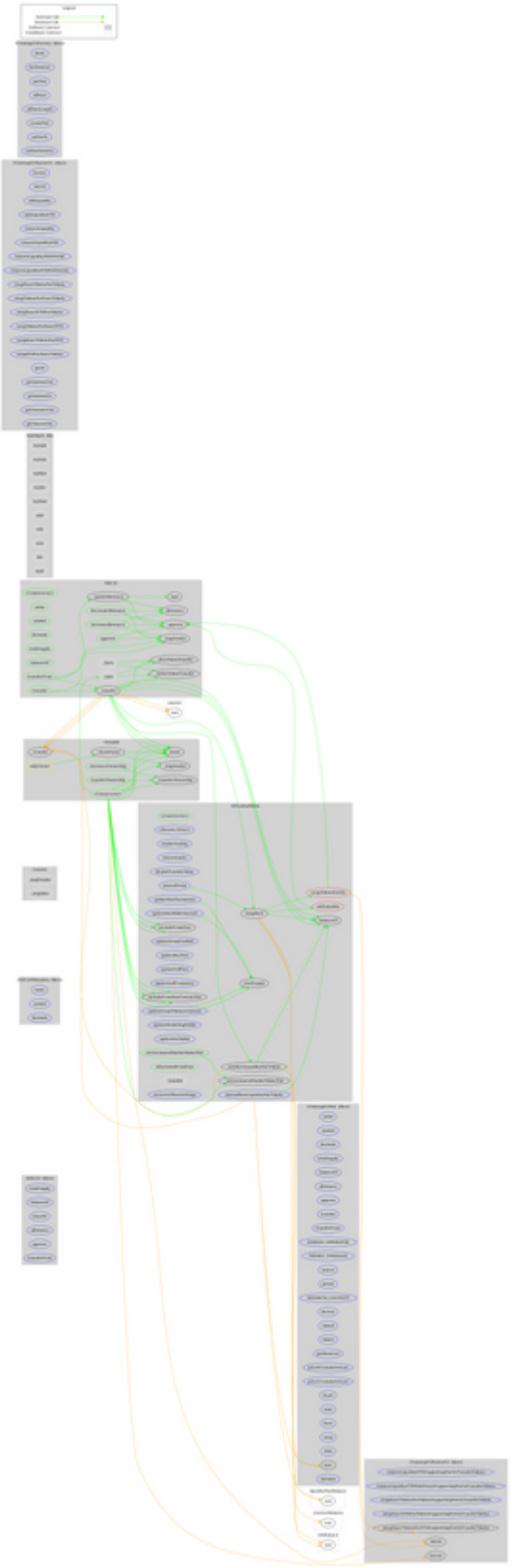
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_spendAllowance	Internal	✓	
	_beforeTokenTransfer	Internal	✓	
	_afterTokenTransfer	Internal	✓	
Ownable	Implementation	Context		
	<Constructor>	Public	✓	-
	owner	Public		-
	_checkOwner	Internal		
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_transferOwnership	Internal	✓	
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		

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	✓	-
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	✓	-
AfricaDaoToken	Implementation	ERC20, Ownable		
	<Constructor>	Public	✓	ERC20
	<Receive Ether>	External	Payable	-
	enableTrading	External	✓	onlyOwner
	removeLimits	External	✓	onlyOwner
	disableTransferDelay	External	✓	onlyOwner
	updateSwapTokensAtAmount	External	✓	onlyOwner
	updateMaxTxnAmount	External	✓	onlyOwner
	updateMaxWalletAmount	External	✓	onlyOwner
	excludeFromMaxTransaction	Public	✓	onlyOwner
	updateSwapEnabled	External	✓	onlyOwner
	updateBuyFees	External	✓	onlyOwner
	updateSellFees	External	✓	onlyOwner
	updateSellFrequency	External	✓	onlyOwner
	excludeFromFees	Public	✓	onlyOwner
	setAutomatedMarketMakerPair	Public	✓	onlyOwner
	_setAutomatedMarketMakerPair	Private	✓	
	updateMarketingWallet	External	✓	onlyOwner
	updateDevWallet	External	✓	onlyOwner
	manualSwap	External	✓	onlyOwner
	isExcludedFromFees	Public		-
	_transfer	Internal	✓	
	swapTokensForEth	Private	✓	
	addLiquidity	Private	✓	
	swapBack	Private	✓	
	setAutoLPBurnSettings	External	✓	onlyOwner
	autoBurnLiquidityPairTokens	Internal	✓	
	manualBurnLiquidityPairTokens	External	✓	onlyOwner

Contract Flow



Domain Info

Domain Name	africadao.co
Registry Domain ID	D3CC460ECF786482FB235AB8DF789A6B3-GDREG
Creation Date	2022-07-26T15:44:52Z
Updated Date	2022-07-31T15:44:53Z
Registry Expiry Date	2023-07-26T15:44:52Z
Registrar WHOIS Server	whois.namecheap.com
Registrar URL	http://www.namecheap.com
Registrar	NameCheap, Inc.
Registrar IANA ID	1068

The domain has been created in 12 months before the creation of the audit.

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 transferring funds to the team’s wallet. 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 max buy fee limit of 20% and a max sell fee of 25%.

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