Cyberscope

Audit Report

Alaska Token

August 2022

Type	ERC20
Network	DOGE
Address	0x0AE5cBdd4992344020db4F4Bc78Db6d1d73702B9
Audited by	© cyberscope

Table of Contents

Contract Review

Explorer	https://explorer.dogechain.dog/address/0x0AE5cBdd4992344020db4F4Bc78Db6d1d73702B9
Optimization	200 runs
Compiler Version	v0.8.13+commit.abaa5c0e
Contract name	Alaska Token
Decimals	9
Total supply	1,000,000,000,000,000
Symbol	ALT

Source Files

Filename	**SHA256**
contract.sol	f7a345e6f65608baf344791a4f9ccafd4ce122e5ce088aee14bdf66b8cfe7765

Audit Updates

Initial Audit	21st August 2022
Corrected	

Contract Analysis

● Critical ● Medium ● Minor / Informative ● Pass

Severity	Code	Description	Status
●	ST	Stops Transactions	Passed
●	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	Unresolved
●	BC	Blacklists Addresses	Unresolved

OCTD - Transfers Contract's Tokens

Criticality	minor
Location	contract.sol#L461
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 `rescueToken` function.

```
function rescueToken(address tokenAddress, uint256 tokens)
    public
    onlyOwner
    returns (bool success)
{
    return IERC20(tokenAddress).transfer(msg.sender, tokens);
}
```

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
Location	contract.sol#L395
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 `setFee` function with the value 30 in buys.

```
function setFee (uint256 _sellFeeRate, uint256 _buyFeeRate) external onlyOwner {
    require (_sellFeeRate <= 14, "Fee can't exceed 14%");
    require (_buyFeeRate <= 30, "Fee can't exceed 30%");
    sellFeeRate = _sellFeeRate;
    buyFeeRate = _buyFeeRate;
}
```

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.

ULTW - Transfers Liquidity to Team Wallet

Criticality	minor
Location	contract.sol#L368
Status	Unresolved

Description

The contract owner has the authority to transfer funds without limit to the team wallet. These funds have been accumulated from fees collected from the contract. The owner may take advantage of it by calling the `swapToken` method.

```
function swapToken() public onlyOwner {

    uint256 contractTokenBalance = balanceOf(address(this));

    bool overMinTokenBalance = contractTokenBalance >=
numTokensSellToAddToLiquidity;

    bool shouldSwapBack = (overMinTokenBalance && balanceOf(address(this)) > 0);
    if(shouldSwapBack){
        swapTokensForEth(numTokensSellToAddToLiquidity);
        }
}
```

Recommendation

The contract could embody a check for the maximum amount of funds that can be swapped. Since a huge amount may volatile the token's price.

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.

BT - Burns Tokens

Criticality	minor
Location	contract.sol#L402
Status	Unresolved

Description

The contract owner has the authority to burn the contract's address tokens. The owner may take advantage of it by calling the `manualBurn` function. As a result the contract address will lose the corresponding tokens.

```
function manualBurn(uint256 amount) external onlyOwner returns (bool) {
    return _basicTransfer(address(this), DEAD, 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 - Blacklists Addresses

Criticality	medium
Location	contract.sol#L297
Status	Unresolved

Description

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

```
if (blacklistEnabled) {
    require (!isBot[sender] && !isBot[recipient], "Bot!");
}
```

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
●	CO	Code Optimization	Unresolved
●	L01	Public Function could be Declared External	Unresolved
●	L02	State Variables could be Declared Constant	Unresolved
●	L04	Conformance to Solidity Naming Conventions	Unresolved
●	L05	Unused State Variable	Unresolved
●	L07	Missing Events Arithmetic	Unresolved
●	L11	Unnecessary Boolean equality	Unresolved

CO - Code Optimization

Criticality	minor
Location	contract.sol
Status	Unresolved

Description

There are code segments that are not used by the contract. Those sections could be eliminated from the contract's code base.

```
function cooldownEnabled(bool _status, uint8 _interval) public onlyOwner {
    buyCooldownEnabled = _status;
    cooldownTimerInterval = _interval;
}

function blacklistBlockEnabled(bool _status) public onlyOwner {
    enableBlacklistBlock = _status;
}
```

Recommendation

Rewrite some code segments so the runtime will be more performant.

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L94,101,122,368,406,418,445,450,461
Status	Unresolved

Description

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

```
authorize
unauthorize
transferOwnership
swapToken
getCirculatingSupply
checkBot
cooldownEnabled
blacklistBlockEnabled
rescueToken
```

Recommendation

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

L02 - State Variables could be Declared Constant

Criticality	minor
Location	contract.sol#L188,189,195,210
Status	Unresolved

Description

Constant state variables should be declared constant to save gas.

```
DEAD
ZERO
_totalSupply
feeDenominator
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L137,386,395,410,445,450,187,188,189,191,192,193,196
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.

```
WETH
_address
_sellFeeRate
_buyFeeRate
_marketingWallet
_status
_interval
DEAD
ZERO
...
```

Recommendation

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

L05 - Unused State Variable

Criticality	minor
Location	contract.sol#L200,233
Status	Unresolved

Description

There are segments that contain unused state variables.

```
cooldown
cooldownTimer
```

Recommendation

Remove unused state variables.

L07 - Missing Events Arithmetic

Criticality	minor
Location	contract.sol#L395,427
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.

```
sellFeeRate = _sellFeeRate
numTokensSellToAddToLiquidity = amount * 10 ** 9
```

Recommendation

Emit an event for critical parameter changes.

L11 - Unnecessary Boolean equality

Criticality	minor
Location	contract.sol#L422
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.

```
require(bool,string)(blacklistEnabled == false,can only be called once)
```

Recommendation

Remove the equality to the boolean constant.

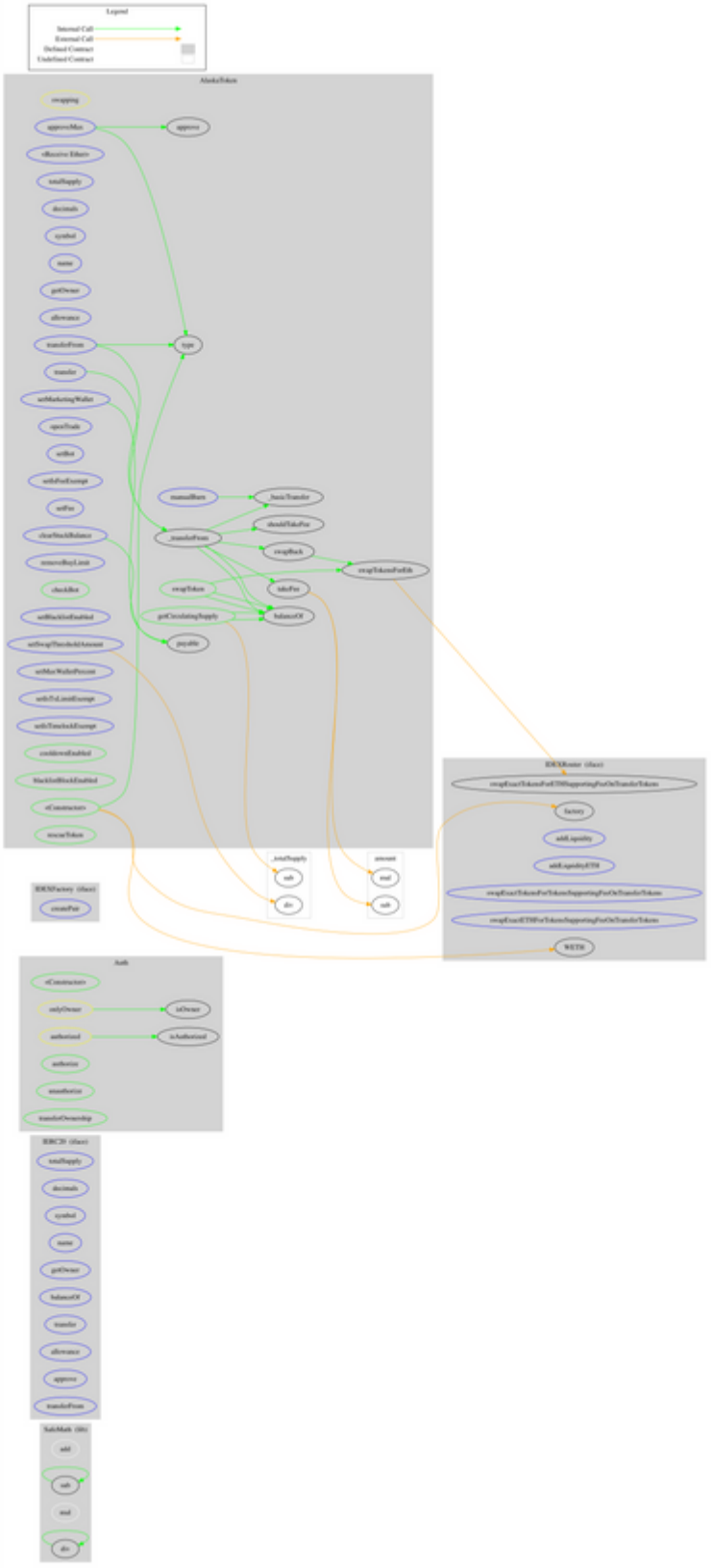
Contract Functions

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
SafeMath	Library			
	add	Internal		
	sub	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		
IERC20	Interface			
	totalSupply	External		-
	decimals	External		-
	symbol	External		-
	name	External		-
	getOwner	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
Auth	Implementation			
	<Constructor>	Public	✓	-
	authorize	Public	✓	onlyOwner
	unauthorize	Public	✓	onlyOwner
	isOwner	Public		-
	isAuthorized	Public		-
	transferOwnership	Public	✓	onlyOwner
IDEXFactory	Interface			

	createPair	External	✓	-
IDEXRouter	Interface			
	factory	External		-
	WETH	External		-
	addLiquidity	External	✓	-
	addLiquidityETH	External	Payable	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
AlaskaToken	Implementation	IERC20, Auth		
	<Constructor>	Public	✓	Auth
	<Receive Ether>	External	Payable	-
	totalSupply	External		-
	decimals	External		-
	symbol	External		-
	name	External		-
	getOwner	External		-
	balanceOf	Public		-
	allowance	External		-
	approve	Public	✓	-
	approveMax	External	✓	-
	transfer	External	✓	-
	transferFrom	External	✓	-
	_transferFrom	Internal	✓	
	_basicTransfer	Internal	✓	
	shouldTakeFee	Internal		
	takeFee	Internal	✓	
	swapBack	Internal	✓	swapping
	swapTokensForEth	Private	✓	
	swapToken	Public	✓	onlyOwner
	openTrade	External	✓	onlyOwner

	setBot	External	✓	onlyOwner
	setIsFeeExempt	External	✓	onlyOwner
	setFee	External	✓	onlyOwner
	manualBurn	External	✓	onlyOwner
	getCirculatingSupply	Public		-
	setMarketingWallet	External	✓	onlyOwner
	removeBuyLimit	External	✓	onlyOwner
	checkBot	Public		-
	setBlacklistEnabled	External	✓	onlyOwner
	setSwapThresholdAmount	External	✓	onlyOwner
	setMaxWalletPercent	External	✓	onlyOwner
	setIsTxLimitExempt	External	✓	onlyOwner
	setIsTimelockExempt	External	✓	onlyOwner
	cooldownEnabled	Public	✓	onlyOwner
	blacklistBlockEnabled	Public	✓	onlyOwner
	clearStuckBalance	External	✓	onlyOwner
	rescueToken	Public	✓	onlyOwner

Contract Flow



Summary

There are some functions that can be abused by the owner like transferring tokens to the team’s wallet, manipulating fees, transferring funds to the team’s wallet, burning tokens and blacklisting addresses. 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 30% fees in buys and 14% in sales.

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