Cyberscope

Audit Report

Opulence User Helper

March 2022

SHA256 c397f881b5c847b734b0ce56a5e78d333f3793aa71d35d0d1eb4a51077ed1aca

Audited by © cyberscope

Table of Contents

Source Files

Filename	SHA256
contract.sol	c397f881b5c847b734b0ce56a5e78d333f3793aa71d35d0d1eb4a51077ed1aca

Audit Updates

Initial Audit	28th March 2022
Corrected	

Contract Analysis

UserHelper is a wrapper of the management contract. The management contract is responsible for the essential functionality of the create node. Additionally, it is the medium to guarantee the safety of the process.

The auditing of the “**Management**” contract is **out of the scope of this audit**.

Create Nodes

- A user has the ability to create nodes.
- The users can give a name to the created node.
- The cost of a node can be configurable by the contract owner.
- When the user creates a node, the funds are transferred to the contract's address.
- If the contract's accumulated amount of tokens is more than a threshold, the amount is distributed proportionally to the liquidity pool, treasury address and burned address.
- The threshold can be configurable by the contract owner.
- The distribution portions can be configured by the contract owner.
- The entire UserHelper functionality can be paused by the contract owner.

management

- The user has the ability to claim the rewards.
- The management contract is responsible for checking if the depositor is applicable to claim the rewards.
- The awarded amount is taxed with a "claimTax".
- The claimTax configurable by the contract owner.
- The claimTax amount is converted to the native coin and transferred to the treasury and dev wallet proportionally.
- The proportions of treasury and dev amounts can be paused by the contract owner.

Contract Diagnostics

Critical Medium Minor

Severity	Code	Description
	FSA	Fixed Swap Address
	CR	Code Repetition
	MC	Missing Check
	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

FSA - Fixed Swap Address

Criticality	minor
Location	contract.sol#L899

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(_router);
```

Recommendation

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

CR - Code Repetition

Criticality	minor
Location	contract.sol#L937,1003

Description

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

```
if (!distributing && checkBalance) {
    distributing = true;

    uint256 amount = opec.balanceOf(address(this));

    uint256 liquidityAmount = amount * create_liquidity / 100;
    uint256 treasuryAmount = amount * create_treasury / 100;
    uint256 burnAmount = amount * create_burn / 100;

    swapAndLiquify(liquidityAmount);

    swapAndSendToFee(treasury, treasuryAmount);

    opec.safeTransfer(burn, burnAmount);

    distributing = false;
}
```

Recommendation

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

MC - Missing Check

Criticality	medium
Location	contract.sol#L942,967

Description

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

There are some properties like create_liquidity, create_treasury, create_burn that are used in order to get proportional fees from the amount. These fees are assumed to be a percentage, but in the code there is no guarantee that the fees can be configured to be more than 100%.

```
uint256 liquidityAmount = amount * create_liquidity / 100;
uint256 treasuryAmount = amount * create_treasury / 100;
uint256 burnAmount = amount * create_burn / 100;
//
swapAndSendToFee(treasury, cashoutAmount * claim_treasury / 100);
swapAndSendToFee(dev, cashoutAmount * claim_dev / 100);
//
```

Recommendation

The contract should properly check the variables according to the required specifications

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L816,835,844,907,911,916,930,958,973,988 and 10 more

Description

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

```
getNodeAvailableReward
getNoderewardPerDay
getNodeLastClaimTime
getNodeCreateTime
getNodeNames
getNodeCount
getTotalCount
getClaimInterval
getNodeLimit
...
```

Recommendation

Use the external attribute for functions never called from the contract

L02 - State Variables could be Declared Constant

Criticality	minor
Location	contract.sol#L860

Description

Constant state variables should be declared constant to save gas.

```
burn
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L134,930,958,988,871,872,873,876,877

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.

```
claim_dev
claim_treasury
create_burn
create_treasury
create_liquidity
_name
_index
WAVAX
```

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#L916

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.

```
create_liquidity = parameters[0]
```

Recommendation

Emit an event for critical parameter changes.

L09 - Dead Code Elimination

Criticality	minor
Location	contract.sol#L410,439,499,509,472,482,385,586,610,601

Description

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

```
safeIncreaseAllowance
safeDecreaseAllowance
safeApprove
sendValue
functionStaticCall
functionDelegateCall
functionCallWithValue
functionCall
...
```

Recommendation

Remove unused functions.

Contract Functions

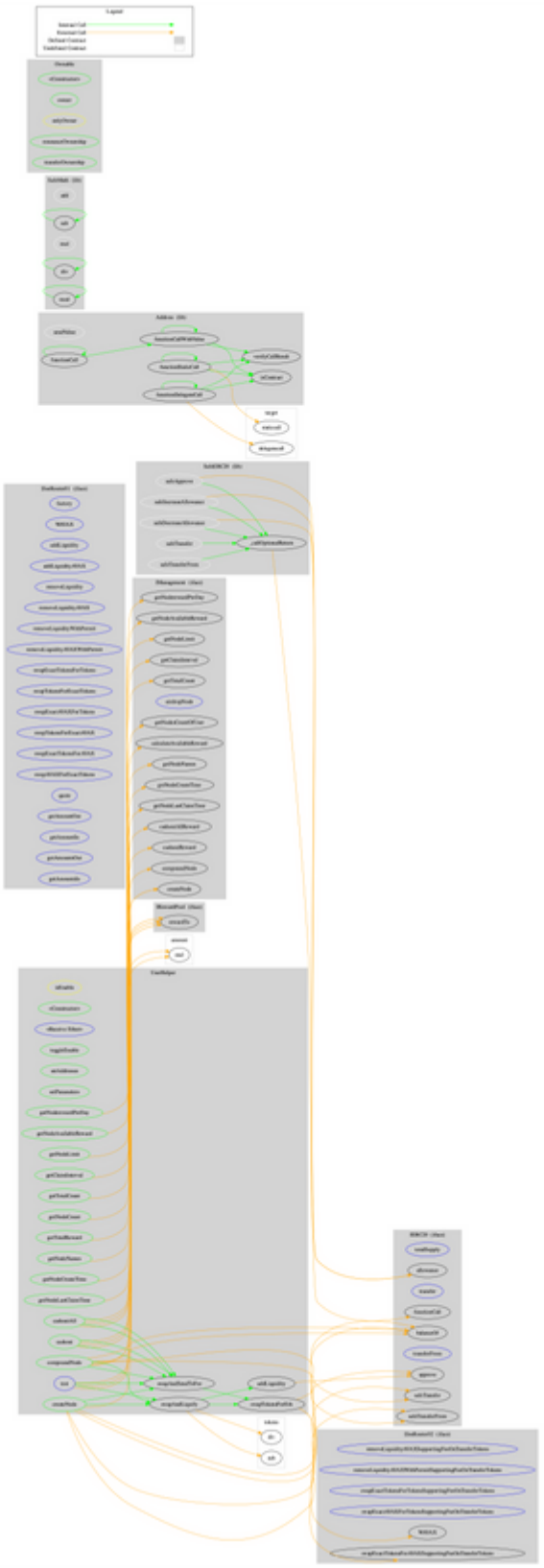
Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
IManagement	Interface			
	getNodeLimit	External		-
	getClaimInterval	External		-
	getTotalCount	External		-
	getNodesCountOfUser	External		-
	createNode	External	✓	-
	airdropNode	External	✓	-
	calculateAvailableReward	External		-
	calculateAvailableReward	External		-
	cashoutAllReward	External	✓	-
	cashoutReward	External	✓	-
	compoundNode	External	✓	-
	getNodeNames	External		-
	getNodeCreateTime	External		-
	getNodeLastClaimTime	External		-
	getNoderewardPerDay	External		-
	getNodeAvailableReward	External		-
IRewardPool	Interface			
	rewardTo	External	✓	-

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	✓	-
	swapExactAVAXForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForAVAXSupportingFeeOnTransferTokens	External	✓	-
Address	Library			
	isContract	Internal		
	sendValue	Internal	✓	

	functionCall	Internal	✓	
	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionStaticCall	Internal		
	functionStaticCall	Internal		
	functionDelegateCall	Internal	✓	
	functionDelegateCall	Internal	✓	
	verifyCallResult	Internal		
SafeERC20	Library			
	safeTransfer	Internal	✓	
	safeTransferFrom	Internal	✓	
	safeApprove	Internal	✓	
	safeIncreaseAllowance	Internal	✓	
	safeDecreaseAllowance	Internal	✓	
	_callOptionalReturn	Private	✓	
SafeMath	Library			
	add	Internal		
	sub	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		
	mod	Internal		
	mod	Internal		
Ownable	Implementation			
	<Constructor>	Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
UserHelper	Implementation	Ownable		

	<Constructor>	Public	✓	-
	<Receive Ether>	External	Payable	-
	toggleEnable	Public	✓	onlyOwner
	setAddresses	Public	✓	onlyOwner
	setParameters	Public	✓	onlyOwner
	createNode	Public	✓	isEnable
	cashout	Public	✓	isEnable
	cashoutAll	Public	✓	isEnable
	compoundNode	Public	✓	isEnable
	getTotalReward	Public		-
	getNodeLimit	Public		-
	getClaimInterval	Public		-
	getTotalCount	Public		-
	getNodeCount	Public		-
	getNodeNames	Public		-
	getNodeCreateTime	Public		-
	getNodeLastClaimTime	Public		-
	getNoderewardPerDay	Public		-
	getNodeAvailableReward	Public		-
	swapAndSendToFee	Private	✓	
	swapAndLiquify	Private	✓	
	swapTokensForEth	Private	✓	
	addLiquidity	Private	✓	
	test	External	✓	onlyOwner

Contract Flow



Summary

UserHelper wraps the functionality of the Management contract. In this audit we focus on the wrapping functionality. The management contract is out of the scope. This audit mentions some performance improvements, security concerns and optimizations.

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