Cyberscope

Audit Report

GrowRich

August 2022

Type	ERC20
Network	ROPSTEN ETH
Address	0xb2C3cf9f2aB29E04e498Af5216dDfAe0544eC658
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	GrowRich
Compiler Version	v0.8.16+commit.07a7930e
Optimization	200 runs
Licence	MIT
Explorer	https://ropsten.etherscan.io/address/0xb2C3cf9f2aB29E04e498Af5216dDfAe0544eC658
Symbol	GrowRich
Decimals	18
Total Supply	1,000,000,000

Source Files

Filename	**SHA256**
contract.sol	18ea8492a10f4869d6ff3993645bdc9d10b2c1c95d4bceef7a248ff0a583ca64

Audit Updates

Initial Audit	22nd August 2022
Corrected	

Contract Analysis

● Critical ● Medium ● Minor / Informative ● Pass

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

ST - Stops Transactions

Criticality	critical
Location	contract.sol#L924
Status	Unresolved

Description

The contract does not properly manage the amount between the contract's tokens, the strategicBurnReserves and the totalSCD. As a result, in many occustances, the contract's balance will be less than the strategicBurnReserves and totalSCD and the contract will revert.

```
if (
        (from == uniswapV2Pair || to == uniswapV2Pair) &&
            _liquidityAdded &&
            !inSwapAndLiquify
        ) {
            if (from != uniswapV2Pair) {
                uint256 totalSCD = _totalStakingAmount + _totalClubGrowRichAmount +
                    _totalDevelopmentAmount;
                uint256 contractLiquidityBalance = (balanceOf(address(this)) -
                    strategicBurnReserves - totalSCD);
```

Recommendation

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

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
	STC	Succeeded Transfer Check	Unresolved
	CO	Code Optimization	Unresolved
	FSA	Fixed Swap Address	Unresolved
	L01	Public Function could be Declared External	Unresolved
	L02	State Variables could be Declared Constant	Unresolved
	L04	Conformance to Solidity Naming Conventions	Unresolved

STC - Succeeded Transfer Check

Criticality	minor / informative
Location	contract.sol#L1049,1050,1051
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(addressUSDC).transfer(stakingContractWallet, stakingShare);
IERC20(addressUSDC).transfer(clubGrowRichWallet, clubGrowRichShare);
IERC20(addressUSDC).transfer(developmentWallet, developmentShare);
```

Recommendation

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

CO - Code Optimization

Criticality	minor / informative
Location	contract.sol#L1046,1047,954
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 following wallets clubGrowRichWallet and developmentWallet point to the same address. That means that their share can be calculated together.

```
uint256 clubGrowRichShare = ((contractBalanceUSDC * 3) / 11);
uint256 developmentShare = ((contractBalanceUSDC * 2) / 11);
```

The variable sumAmount reserves extra memory.

```
    uint256 sumAmount = stakingAmount +
        liquidityAmount +
        clubGrowRichAmount +
        developmentAmount;

    return sumAmount;
```

Recommendation

The return value can be calculated on the return statement.

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

FSA - Fixed Swap Address

Criticality	minor / informative
Location	contract.sol#L887
Status	Unresolved

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.

```
uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory())
IUniswapV2Router02 _uniswapV2Router = IUniswapV2Router02(
    0x7a250d5630B4cF539739dF2C5dAcb4c659F2488D
);
uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory())
    .createPair(address(this), _uniswapV2Router.WETH());
```

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 / informative
Location	contract.sol#L633,680,523,585,656,515,530,606
Status	Unresolved

Description

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

```
transferFrom
decreaseAllowance
symbol
transfer
increaseAllowance
name
totalSupply
approve
```

Recommendation

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

L02 - State Variables could be Declared Constant

Criticality	minor / informative
Location	contract.sol#L834,835,838,832,833
Status	Unresolved

Description

Constant state variables should be declared constant to save gas.

```
clubGrowRichPercentage
developmentPercentage
burnRateStrategicBurnReserves
stakingPercentage
liquidityPercentage
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor / informative
Location	contract.sol#L70,101,68,147
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.

```
PERMIT_TYPEHASH
MINIMUM_LIQUIDITY
DOMAIN_SEPARATOR
WETH
```

Recommendation

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

Contract Functions

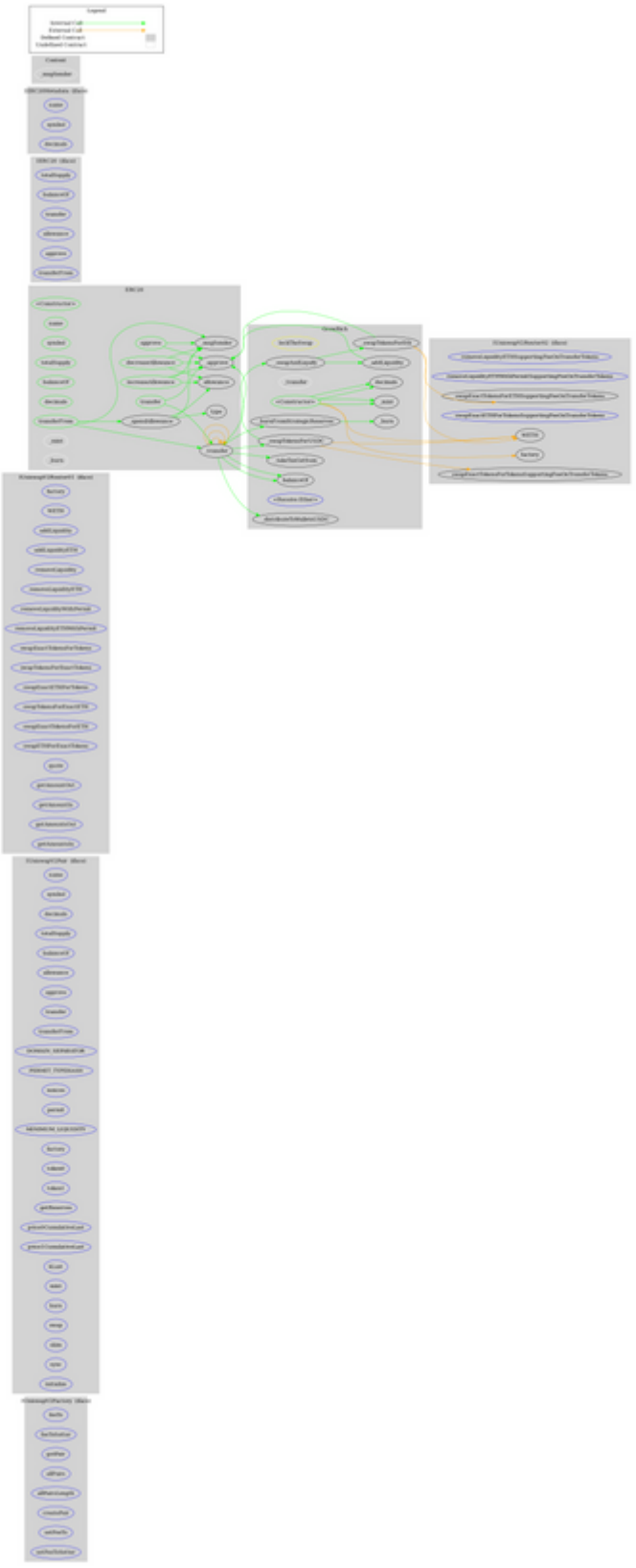
Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
IUniswapV2Factory	Interface			
	feeTo	External		-
	feeToSetter	External		-
	getPair	External		-
	allPairs	External		-
	allPairsLength	External		-
	createPair	External	✓	-
	setFeeTo	External	✓	-
	setFeeToSetter	External	✓	-
IUniswapV2Pair	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	✓	-
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		-
IUniswapV2Ro	Interface	IUniswapV2		

uter02		Router01		
	removeLiquidityETHSupportingFeeOnTransferTokens	External	✓	-
	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External	✓	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
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		
ERC20	Implementation	Context, IERC20, IERC20Metadata		
	<Constructor>	Public	✓	-
	name	Public		-
	symbol	Public		-
	totalSupply	Public		-
	balanceOf	Public		-

	decimals	Public		-
	allowance	Public		-
	transfer	Public	✓	-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	_mint	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_spendAllowance	Internal	✓	
	_transfer	Internal	✓	
GrowRich	Implementation	ERC20		
	<Constructor>	Public	✓	ERC20
	_transfer	Internal	✓	
	_takeTaxGetSum	Private	✓	
	_burnFromStrategicReserves	Private	✓	
	_swapAndLiquify	Private	✓	lockTheSwap
	_swapTokensForEth	Private	✓	
	_addLiquidity	Private	✓	
	_swapTokensForUSDC	Private	✓	lockTheSwap
	_distributeToWalletsUSDC	Private	✓	
	<Receive Ether>	External	Payable	-

Contract Flow



Summary

The Smart Contract analysis reported one critical severity issue. The contract owner has the authority to stop transactions. 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. There is also a fixed fee limit of 15%.

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