auditace.tech

AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

ANDY

DATED : 21 July 23'

MANUAL TESTING

Centralization – swaps are disabled by default

Severity: High
function: EnableTrading
Status: Not Resolved
Overview:
The smart contract owner must enable trades for holders. If trading remain disabled, no one would be able to buy/sell/transfer tokens.

```
function enableTrading() external onlyOwner{
    require(!tradingEnabled, "Trading already enabled.");
    tradingEnabled = true;
    swapEnabled = true;
}
```

Suggestion

To mitigate this centralization issue, we propose the following options:

1. Renounce Ownership: Consider relinquishing control of the smart contract by renouncing ownership. This would remove the ability for a single entity to manipulate the router, reducing centralization risks.
2. Multi-signature Wallet: Transfer ownership to a multi-signature wallet. This would require multiple approvals for any changes to the mainRouter, adding an additional layer of security and reducing the centralization risk.
3. Transfer ownership to a trusted and valid 3rd party in order to guarantee enabling of the trades

AUDIT SUMMARY

Project name – ANDY

Date: 21 July, 2023

Scope of Audit- Audit Ace was consulted to conduct the smart contract audit of the solidity source codes.

Audit Status: **Passed with High Risk**

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	1	1	0	1
Acknowledged	0	0	0	0	0
Resolved	0	0	0	0	0

USED TOOLS

Tools:

1- Manual Review:
A line by line code review has been performed by audit ace team.

2- BSC Test Network: All tests were conducted on the BSC Test network, and each test has a corresponding transaction attached to it. These tests can be found in the "Functional Tests" section of the report.

3- Slither :
The code has undergone static analysis using Slither.

Testnet version:
The tests were performed using the contract deployed on the BSC Testnet, which can be found at the following address:
https://testnet.bscscan.com/token/0x9Fd02e21B2b8F329d8E03Df8A7AF3E374ccfc0B1

Token Information

Token Name : Andy Coin

Token Symbol: ANDY

Decimals: 18

Token Supply: 420,690,000

Token Address:
0x05Bc7ceb04ce02b880024889c4F42621639eA88C

Checksum:
4288d966137e156f7e6ec06092b873817bd9da58

Owner:
0x0F2F3a60dee0d5D8417e64cc4Ac0CBa58C6D8CFd
(at time of writing the audit)

Deployer:
0x0F2F3a60dee0d5D8417e64cc4Ac0CBa58C6D8CFd

TOKEN OVERVIEW

Fees:

Buy Fees: 0-5%

Sell Fees: 0-5%

Transfer Fees: 0-5%

Fees Privilege: owner

Ownership: owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: no

Blacklist: No

Other Privileges: Initial distribution of the tokens

modifying fees

enabling trades

AUDIT METHODOLOGY

The auditing process will follow a routine as special considerations by Auditace:

- Review of the specifications, sources, and instructions provided to Auditace to make sure the contract logic meets the intentions of the client without exposing the user's funds to risk.
- Manual review of the entire codebase by our experts, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
- Specification comparison is the process of checking whether the code does what the specifications, sources, and instructions provided to Auditace describe.
- Test coverage analysis determines whether the test cases are covering the code and how much code isexercised when we run the test cases.
- Symbolic execution is analysing a program to determine what inputs cause each part of a program to execute.
- Reviewing the codebase to improve maintainability, security, and control based on the established industry and academic practices.

VULNERABILITY CHECKLIST

- [x] Return values of low-level calls
- [x] Private modifier
- [x] Multiple Sends
- [x] Using Suicide
- [x] Gas Limitand Loops
- [x] Address hardcoded
- [x] Exception Disorder
- [x] Using inline assembly
- [x] Divide before multiply
- [x] Missing Zero Address Validation
- [x] Compiler version not fixed
- [x] **Gasless Send**
- [x] Using block.timestamp
- [x] Re-entrancy
- [x] Tautology or contradiction
- [x] Timestamp Dependence
- [x] Revert/require functions
- [x] Use of tx.origin
- [x] Integer overflow/underflow
- [x] Dangerous strict equalities
- [x] Using SHA3
- [x] Using throw

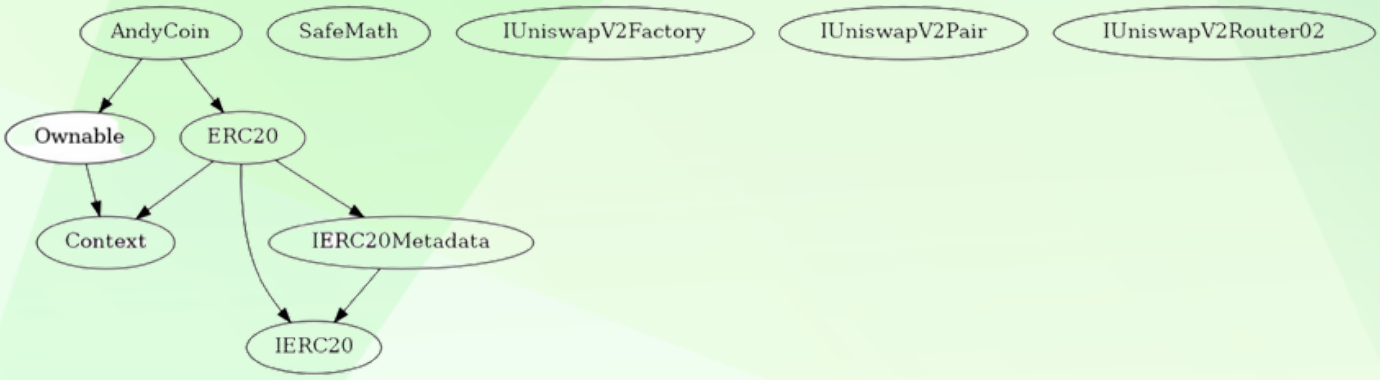
CLASSIFICATION OF RISK

Severity	Description
Critical	These vulnerabilities could be exploited easily and can lead to asset loss, data loss, asset, or data manipulation. They should be fixed right away.
High-Risk	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.
Medium-Risk	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.
Low-Risk	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.
Gas Optimization /Suggestion	A vulnerability that has an informational character but is not affecting any of the code.

Findings

Severity	Found
Critical	0
High-Risk	1
Medium-Risk	1
Low-Risk	0
Gas Optimization / Suggestions	1

INHERITANCE TREE



POINTS TO NOTE

- **Owner is able to update buy/sell fees in range 0-5%**
- **Owner is not able to set fee on transfers (0% transfer fee)**
- Owner is not able to blacklist an arbitrary address.
- Owner is not able to disable trades
- Owner is not able to mint new tokens
- Owner is able to set maximum wallet and maximum buy/sell limits
- **Owner must enable trades manually**

CONTRACT ASSESMENT

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
Context	Implementation			
└	_msgSender	Internal 🔒		
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO ❗
└	owner	Public ❗		NO ❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_transferOwnership	Internal 🔒	🛑	
IERC20	Interface			
└	totalSupply	External ❗		NO ❗
└	balanceOf	External ❗		NO ❗
└	transfer	External ❗	🛑	NO ❗
└	allowance	External ❗		NO ❗
└	approve	External ❗	🛑	NO ❗
└	transferFrom	External ❗	🛑	NO ❗
IERC20Metadata	Interface	IERC20		
└	name	External ❗		NO ❗
└	symbol	External ❗		NO ❗
└	decimals	External ❗		NO ❗
ERC20	Implementation	Context, IERC20, IERC20Metadata		
└	<Constructor>	Public ❗	🛑	NO ❗
└	name	Public ❗		NO ❗
└	symbol	Public ❗		NO ❗
└	decimals	Public ❗		NO ❗
└	totalSupply	Public ❗		NO ❗
└	balanceOf	Public ❗		NO ❗

CONTRACT ASSESMENT

| └ | transfer | Public ❗ | 🛑 |NO ❗ |
| └ | allowance | Public ❗ | |NO ❗
└	approve	Public ❗	🛑	NO ❗
└	transferFrom	Public ❗	🛑	NO ❗
└	increaseAllowance	Public ❗	🛑	NO ❗
└	decreaseAllowance	Public ❗	🛑	NO ❗
└	_transfer	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_approve	Internal 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
└	_afterTokenTransfer	Internal 🔒	🛑	
SafeMath	Library			
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
IUniswapV2Factory	Interface			
└	feeTo	External ❗	NO ❗
└	feeToSetter	External ❗		NO ❗
└	getPair	External ❗		NO ❗
└	allPairs	External ❗	NO ❗
└	allPairsLength	External ❗		NO ❗
└	createPair	External ❗	🛑	NO ❗
└	setFeeTo	External ❗	🛑	NO ❗
└	setFeeToSetter	External ❗	🛑	NO ❗
IUniswapV2Pair	Interface			
└	name	External ❗	NO ❗
└	symbol	External ❗		NO ❗
└	decimals	External ❗	NO ❗
└	totalSupply	External ❗		NO ❗

CONTRACT ASSESMENT

└	balanceOf	External ❗		NO ❗
└	allowance	External ❗		NO ❗
└	approve	External ❗	🛑	NO ❗
└	transfer	External ❗	🛑	NO ❗
└	transferFrom	External ❗	🛑	NO ❗
└	DOMAIN_SEPARATOR	External ❗		NO ❗
└	PERMIT_TYPEHASH	External ❗		NO ❗
└	nonces	External ❗		NO ❗
└	permit	External ❗	🛑	NO ❗
└	MINIMUM_LIQUIDITY	External ❗		NO ❗
└	factory	External ❗		NO ❗
└	token0	External ❗		NO ❗
└	token1	External ❗		NO ❗
└	getReserves	External ❗		NO ❗
└	price0CumulativeLast	External ❗		NO ❗
└	price1CumulativeLast	External ❗		NO ❗
└	kLast	External ❗		NO ❗
└	mint	External ❗	🛑	NO ❗
└	burn	External ❗	🛑	NO ❗
└	swap	External ❗	🛑	NO ❗
└	skim	External ❗	🛑	NO ❗
└	sync	External ❗	🛑	NO ❗
└	initialize	External ❗	🛑	NO ❗
IUniswapV2Router02	Interface			
└	factory	External ❗		NO ❗
└	WETH	External ❗	NO ❗
└	addLiquidity	External ❗	🛑	NO ❗
└	addLiquidityETH	External ❗	💲	NO ❗
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗
🛑 |NO ❗ |

CONTRACT ASSESMENT

└	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💵	NO ❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
AndyCoin	Implementation	ERC20, Ownable		
└	<Constructor>	Public ❗	🛑	ERC20
└	<Receive Ether>	External ❗	💵	NO ❗
└	enableTrading	External ❗	🛑	onlyOwner
└	removeLimits	External ❗	🛑	onlyOwner
└	updateSwapTokensAtAmount	External ❗	🛑	onlyOwner
└	updateMaxTxnAmount	External ❗	🛑	onlyOwner
└	updateMaxWalletAmount	External ❗	🛑	onlyOwner
└	excludeFromMaxTransaction	Public ❗	🛑	onlyOwner
└	updateSwapEnabled	External ❗	🛑	onlyOwner
└	updateBuyFees	External ❗	🛑	onlyOwner
└	updateSellFees	External ❗	🛑	onlyOwner
└	excludeFromFees	Public ❗	🛑	onlyOwner
└	setAutomatedMarketMakerPair	Public ❗	🛑	onlyOwner
└	_setAutomatedMarketMakerPair	Private 🔐	🛑	
└	updateMarketingWallet	External ❗	🛑	onlyOwner
└	updateLPWallet	External ❗	🛑	onlyOwner
└	updateDevWallet	External ❗	🛑	onlyOwner
└	isExcludedFromFees	Public ❗		NO ❗
└	_transfer	Internal 🔒	🛑	
└	swapTokensForEth	Private 🔐	🛑	
└	addLiquidity	Private 🔐	🛑	
└	swapBack	Private 🔐	🛑	

CONTRACT ASSESMENT

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Pragma version^0.8.17 (contracts/Token.sol#16) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16
solc-0.8.20 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in AndyCoin.swapBack() (contracts/Token.sol#684-720):
        - (success,None) = address(devWallet).call{value: ethForDev}() (contracts/Token.sol#712)
        - (success,None) = address(marketingWallet).call{value: address(this).balance}() (contracts/Token.sol#719)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function IUniswapV2Pair.DOMAIN_SEPARATOR() (contracts/Token.sol#259) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (contracts/Token.sol#261) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (contracts/Token.sol#280) is not in mixedCase
Function IUniswapV2Router02.WETH() (contracts/Token.sol#312) is not in mixedCase
Event AndyCoinmarketingWalletUpdated(address,address) (contracts/Token.sol#413) is not in CapWords
Event AndyCoindevWalletUpdated(address,address) (contracts/Token.sol#415) is not in CapWords
Event AndyCoinlpWalletUpdated(address,address) (contracts/Token.sol#417) is not in CapWords
Parameter AndyCoin.updateBuyFees(uint256,uint256,uint256)._marketingFee (contracts/Token.sol#519) is not in mixedCase
Parameter AndyCoin.updateBuyFees(uint256,uint256,uint256)._liquidityFee (contracts/Token.sol#519) is not in mixedCase
Parameter AndyCoin.updateBuyFees(uint256,uint256,uint256)._devFee (contracts/Token.sol#519) is not in mixedCase
Parameter AndyCoin.updateSellFees(uint256,uint256,uint256)._marketingFee (contracts/Token.sol#527) is not in mixedCase
Parameter AndyCoin.updateSellFees(uint256,uint256,uint256)._liquidityFee (contracts/Token.sol#527) is not in mixedCase
Parameter AndyCoin.updateSellFees(uint256,uint256,uint256)._devFee (contracts/Token.sol#527) is not in mixedCase
Constant AndyCoin.deadAddress (contracts/Token.sol#363) is not in UPPER_CASE_WITH_UNDERSCORES
Variable AndyCoin._isExcludedMaxTransactionAmount (contracts/Token.sol#399) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Variable IUniswapV2Router02.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (contracts/Token.sol#317) is too si
milar to IUniswapV2Router02.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (contracts/Token.sol#318)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

AndyCoin.updateSwapTokensAtAmount(uint256) (contracts/Token.sol#493-498) uses literals with too many digits:
        - require(bool,string)(newAmount >= (totalSupply() * 1) / 100000,Swap amount cannot be lower than 0.001% total supply.) (contracts/Token.sol#494)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
```

Result => A static analysis of contract's source code has been performed using slither,

No major issues were found in the output

FUNCTIONAL TESTING

1- Adding liquidity (passed):

https://testnet.bscscan.com/tx/0xb1c8fab47c9a6d40711f9ea42ed4c962e944d9cd700883b09d5c6eabee01a3f0

2- Buying when excluded from fees (0% tax) (passed):

https://testnet.bscscan.com/tx/0xa306c9e0d9fc028dceb04f7e11a15f6023504eac95a551430e2d7bede6f223d1

3- Selling when excluded from fees (0% tax) (passed):

https://testnet.bscscan.com/tx/0x3f53edd993e9f0664a1e79e448e93abf21593a7cd37073cfb084fa94dd9ef33a

4- Transferring when excluded from fees (0% tax) (passed):

https://testnet.bscscan.com/tx/0x976596846503cb40a802d2c14533dfac76ac7556f70c55c007773a649a7e26f5

5- Buying when not excluded from fees (0-5% tax) (passed):

https://testnet.bscscan.com/tx/0xe0927befd360b17adb3a87be9309bde974c483f2bad9bff20e0b05c7c395f3db

6- Selling when not excluded from fees (0-5% tax) (passed):

https://testnet.bscscan.com/tx/0x60170a8f131e12b6bd43eb49fb84ede607593f3b8ee379f807e551d9cdb43dfe

FUNCTIONAL TESTING

7- Transferring when not excluded from fees (0% tax) (passed):

https://testnet.bscscan.com/tx/0x84b813f514afa366d05651b404b1e7aab36872c84fa0128213839bc59e4290e9

8- Internal swap (passed):

https://testnet.bscscan.com/tx/0x60170a8f131e12b6bd43eb49fb84ede607593f3b8ee379f807e551d9cdb43dfe

MANUAL TESTING

Centralization – swaps are disabled by default

Severity: High
function: EnableTrading
Status: Not Resolved
Overview:
The smart contract owner must enable trades for holders. If trading remain disabled, no one would be able to buy/sell/transfer tokens.

```
function enableTrading() external onlyOwner{
    require(!tradingEnabled, "Trading already enabled.");
    tradingEnabled = true;
    swapEnabled = true;
}
```

Suggestion

To mitigate this centralization issue, we propose the following options:

1. Renounce Ownership: Consider relinquishing control of the smart contract by renouncing ownership. This would remove the ability for a single entity to manipulate the router, reducing centralization risks.
2. Multi-signature Wallet: Transfer ownership to a multi-signature wallet. This would require multiple approvals for any changes to the mainRouter, adding an additional layer of security and reducing the centralization risk.
3. Transfer ownership to a trusted and valid 3rd party in order to guarantee enabling of the trades

MANUAL TESTING

Centralization – LP tokens sent to an EOA

Severity: Medium
function: enableTrading
Status: Not Resolved
Overview:
the LP tokens generated from auto-liquidity are sent to an EOA (lpWallet).
This LP tokens could be used to remove a portion of the liquidity pool (depending on size of accumulated LP tokens)

```
function addLiquidity(uint256 tokenAmount, uint256 ethAmount) private {
    // approve token transfer to cover all possible scenarios
    _approve(address(this), address(uniswapV2Router), tokenAmount);

    // add the liquidity
    uniswapV2Router.addLiquidityETH{value: ethAmount}(
        address(this),
        tokenAmount,
        0, // slippage is unavoidable
        0, // slippage is unavoidable
        lpWallet,
        block.timestamp
    );
}
```

Suggestion
To mitigate this centralization issue, you can either burn or Lock new LP tokens

MANUAL TESTING

Centralization – wallet and swap limitations

Severity: Informational
function: enableTrading
Status: Not Resolved
Overview:

Owner is able to set a limitation for maximum wallet size (I.e how many tokens a user can hold in his/her wallet) and maximum amount of tokens that can be bought (ETH => token) or sold (token => ETH).
This limitations can be adjusted in between 1% - 100% of total supply.

```
    function updateMaxTxnAmount(uint256 newNum) external onlyOwner {
        require(newNum >= ((totalSupply() * 1) / 100) / 1e18, "Cannot set maxTransactionAmount
lower than 1%");
        maxTransactionAmount = newNum * (10 ** 18);
    }

    function updateMaxWalletAmount(uint256 newNum) external onlyOwner {
        require(newNum >= ((totalSupply() * 1) / 100) / 1e18, "Cannot set maxWallet lower than
1%");
        maxWallet = newNum * (10 ** 18);
    }
```

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. 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 Auditace 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 Auditace 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 Auditace team disclaims any liability for the resulting losses.

ABOUT AUDITACE

We specializes in providing thorough and reliable audits for Web3 projects. With a team of experienced professionals, we use cutting-edge technology and rigorous methodologies to evaluate the security and integrity of blockchain systems. We are committed to helping our clients ensure the safety and transparency of their digital assets and transactions.

https://auditace.tech/

https://t.me/Audit_Ace

https://twitter.com/auditace_

https://github.com/Audit-Ace