auditace.tech

AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

Itchy Scratchy Inu

DATED : 4 June 23'

ISSUES FOUND

Centralisation – Trades must be enabled

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 startTrading() external onlyOwner {
    require(!tradingEnabled, "Trading already enabled");
    tradingEnabled = 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 – Itchy Scratchy Inu

Date: 4 June, 2023

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

Audit Status: Passed

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	1	0	0	0
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 done on BSC Test network, each test has its transaction has attached to it.

3- Slither : Static Analysis

Testnet Link: all tests were done using this contract, tests are done on BSC Testnet

https://testnet.bscscan.com/token/0xd719eefa258132b183ff5e3213696bf9b5368c04#code

Token Information

Token Name : Itchy Scratchy Inu

Token Symbol: ISI

Decimals: 9

Token Supply: 10,000,000,000

Token Address:
0xF1bD68258254197aDFe2B40AACc7F0C2ba870Ebb

Checksum:
6552075746215d4a0f2704bc5592c550f79d6c0d

Owner:
0x53d22ac9A891De7A0EC955EF5e2cdE5f36753B92
(at time of writing the audit)

Deployer:
0x6a627454C40Dc779725c5c1CB34F18eC2D41d18f

TOKEN OVERVIEW

Fees:

Buy Fees: 3%

Sell Fees: 3 %

Transfer Fees: 3%

Fees Privilige: None (Immutable fees)

Ownership : Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: none

Blacklist: No

Other Priviliges: - changing swap threshold

- modifying swap settings
- enabling trades
- initial distribution of tokens

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

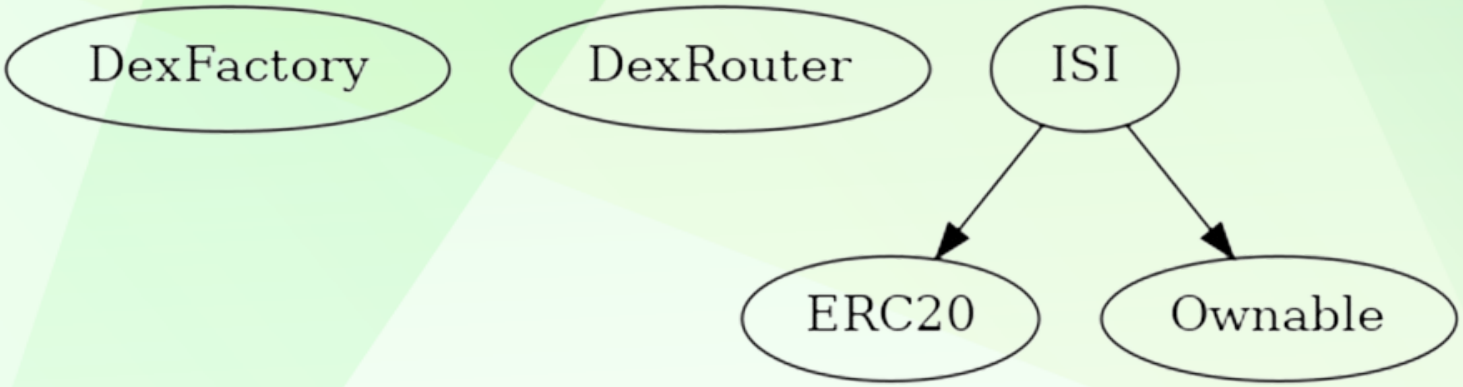
CLASSIFICATION OF RISK

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	0
Low-Risk	0
Gas Optimization / Suggestions	0

INHERITANCE TREE



POINTS TO NOTE

- Owner is not able to change buy/sell/transfer fees (5% static)
- Owner is not able to set max buy/sell/transfer/hold amount
- Owner is not able to blacklist an arbitrary wallet
- Owner is not able to mint new tokens
- Owner is not able to disable trades

CONTRACT ASSESMENT

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
DexFactory	Interface				
└	createPair	External ❗	🛑	NO❗
DexRouter	Interface				
└	factory	External ❗		NO❗
└	WETH	External ❗		NO❗
└	addLiquidityETH	External ❗	💵	NO❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
ISI	Implementation	ERC20, Ownable			
└	<Constructor>	Public ❗	🛑	ERC20
└	setmarketingWallet	External ❗	🛑	onlyOwner
└	setSwapTokensAtAmount	External ❗	🛑	onlyOwner
└	toggleSwapping	External ❗	🛑	onlyOwner
└	setWhitelistStatus	External ❗	🛑	onlyOwner
└	checkWhitelist	External ❗		NO❗
└	startTrading	External ❗	🛑	onlyOwner
└	_takeTax	Internal 🔒	🛑	
└	_transfer	Internal 🔒	🛑	
└	internalSwap	Internal 🔒	🛑	
└	swapToETH	Internal 🔒	🛑	
└	withdrawStuckETH	External ❗	🛑	onlyOwner
└	withdrawStuckTokens	External ❗	🛑	onlyOwner
└	<Receive Ether>	External ❗	💵	NO❗

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS

```
Reentrancy in ISI._transfer(address,address,uint256) (contracts/Token.sol#1073-1096):
        External calls:
        - internalSwap() (contracts/Token.sol#1092)
                - uniswapRouter.swapExactTokensForETHSupportingFeeOnTransferTokens(_amount,0,path,address(this),block.timestamp) (contracts/Token.sol#1114-1120)
                - (success) = marketingWallet.call{value: address(this).balance}() (contracts/Token.sol#1104-1106)
        External calls sending eth:
        - internalSwap() (contracts/Token.sol#1092)
                - (success) = marketingWallet.call{value: address(this).balance}() (contracts/Token.sol#1104-1106)
        Event emitted after the call(s):
        - Transfer(from,to,amount) (contracts/Token.sol#424)
                - super._transfer(_from,_to,toTransfer) (contracts/Token.sol#1095)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Context._msgData() (contracts/Token.sol#117-119) is never used and should be removed
ERC20._burn(address,uint256) (contracts/Token.sol#464-480) is never used and should be removed
SafeMath.add(uint256,uint256) (contracts/Token.sol#683-685) is never used and should be removed
SafeMath.div(uint256,uint256) (contracts/Token.sol#725-727) is never used and should be removed
SafeMath.div(uint256,uint256,string) (contracts/Token.sol#781-790) is never used and should be removed
SafeMath.mod(uint256,uint256) (contracts/Token.sol#741-743) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (contracts/Token.sol#807-816) is never used and should be removed
SafeMath.mul(uint256,uint256) (contracts/Token.sol#711-713) is never used and should be removed
SafeMath.sub(uint256,uint256) (contracts/Token.sol#697-699) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (contracts/Token.sol#758-767) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (contracts/Token.sol#597-606) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (contracts/Token.sol#648-656) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (contracts/Token.sol#663-671) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (contracts/Token.sol#628-641) is never used and should be removed
SafeMath.trySub(uint256,uint256) (contracts/Token.sol#613-621) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.17 (contracts/Token.sol#8) 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 ISI.internalSwap() (contracts/Token.sol#1098-1107):
        - (success) = marketingWallet.call{value: address(this).balance}() (contracts/Token.sol#1104-1106)
Low level call in ISI.withdrawStuckETH() (contracts/Token.sol#1123-1128):
        - (success) = address(msg.sender).call{value: address(this).balance}() (contracts/Token.sol#1124-1126)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function DexRouter.WETH() (contracts/Token.sol#934) is not in mixedCase
Event ISImarketingWalletChanged(address) (contracts/Token.sol#985) is not in CapWords
Parameter ISI.setmarketingWallet(address)._newmarketing (contracts/Token.sol#1007) is not in mixedCase
Parameter ISI.setSwapTokensAtAmount(uint256)._newAmount (contracts/Token.sol#1016) is not in mixedCase
Parameter ISI.setWhitelistStatus(address,bool)._wallet (contracts/Token.sol#1030) is not in mixedCase
Parameter ISI.setWhitelistStatus(address,bool)._status (contracts/Token.sol#1031) is not in mixedCase
Parameter ISI.checkWhitelist(address)._wallet (contracts/Token.sol#1037) is not in mixedCase
Parameter ISI.swapToETH(uint256)._amount (contracts/Token.sol#1109) is not in mixedCase
Parameter ISI.withdrawStuckTokens(address).BEP20_token (contracts/Token.sol#1130) is not in mixedCase
Constant ISI._totalSupply (contracts/Token.sol#962) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

ISI.slitherConstructorVariables() (contracts/Token.sol#957-1139) uses literals with too many digits:
        - swapTokensAtAmount = _totalSupply / 100000 (contracts/Token.sol#977)
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

Router (PCS V2):
0xD99D1c33F9fC3444f8101754aBC46c52416550D1

All the functionalities have been tested, no issues were found

1- Adding liquidity (passed):
https://testnet.bscscan.com/tx/0x0647ea93818cd749d97aa8608cbbe5b95b3a4e030c30ac11098a8269e21dafd4

2- Buying when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0x576a5736042e9ea83a39ba5c78ff4168e4204725d209084fe35c22f336f0d83d

3- Selling when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0x027296fcb42efad84ea25ee87039760d204526bf7e1c6b5ed718f3ebf825b8a2

4- Transferring when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0xe325d89e21ff42da51860d05d41ad3cd3c7552deeede50ab7ab7fb468e33104b

5- Buying from a regular wallet (5% tax) (passed):
https://testnet.bscscan.com/tx/0xebc7d37936345342be806223f9b23b37fb513b8d3b680b6320ed62a1bd3ef8ef

6- Selling from a regular wallet (5% tax) (passed):
https://testnet.bscscan.com/tx/0x0fb63b314b82023480d284c08efbe2ed644db4c50196b2c61a45c5f0cf3d31ca

FUNCTIONAL TESTING

7- Transferring from a regular wallet (5% tax) (passed):
https://testnet.bscscan.com/tx/0x55ef457f3ee67fe24dfe46ac857fcd41026fe196cb0ba1bd394f1ae56fe9e2da

8-Internal swap (BNB Fees and auto-liquidity) ((passed):
https://testnet.bscscan.com/tx/0x0fb63b314b82023480d284c08efbe2ed644db4c50196b2c61a45c5f0cf3d31ca

ISSUES FOUND

Centralisation – Trades must be enabled

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 startTrading() external onlyOwner {
    require(!tradingEnabled, "Trading already enabled");
    tradingEnabled = 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

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