Ether Authority

www.EtherAuthority.io
audit@etherauthority.io

SMART CONTRACT

Security Audit Report

Project:	Catcoin Token
Website:	https://catcoin.io
Platform:	Binance Smart Chain
Language:	Solidity
Date:	March 16th, 2023

Table of contents

Email: audit@EtherAuthority.io

`

THIS IS SECURITY AUDIT REPORT DOCUMENT AND WHICH MAY CONTAIN INFORMATION WHICH IS CONFIDENTIAL. WHICH INCLUDES ANY POTENTIAL VULNERABILITIES AND MALICIOUS CODES WHICH CAN BE USED TO EXPLOIT THE SOFTWARE. THIS MUST BE REFERRED INTERNALLY AND ONLY SHOULD BE MADE AVAILABLE TO THE PUBLIC AFTER ISSUES ARE RESOLVED.

Email: audit@EtherAuthority.io

Introduction

EtherAuthority was contracted by the Catcoin team to perform the Security audit of the Catcoin smart contract code. The audit has been performed using manual analysis as well as using automated software tools. This report presents all the findings regarding the audit performed on March 16th, 2023.

The purpose of this audit was to address the following:

- Ensure that all claimed functions exist and function correctly.
- Identify any security vulnerabilities that may be present in the smart contract.

Project Background

- CatCoin's new contract upgrade provides the ultimate in functionality & security.
- Catcoin is one of the safest contracts ever created in the BSC space. It is thought to be simple and efficient, yet functional and secure.
- Catcoin is able to securely retain its crucial functions such as multiSendTokens, swapAndLiquify, setBuy, setSell, etc.

Audit scope

Name	**Code Review and Security Analysis Report for Catcoin Token Smart Contract**
Platform	**BSC / Solidity**
File	Catcoin.sol
Online Code	0x2f0c6e147974bfbf7da557b88643d74c324053a2
File MD5 Hash	ABED4AC446FCBE8839E9CA08709A49D2
Audit Date	March 16th, 2023

Claimed Smart Contract Features

Claimed Feature Detail	Our Observation
Tokenomics: • Name: Catcoin • Symbol: CATS • Decimals: 0 • Swap Treshold: 0.2% • Total Supply: 1 Quadrillion	**YES, This is valid.**
Ownership Control: • Multisig addresses can be set by the multisig owner. • Swap Threshold value can be set by the multisig owner. • Developer wallet address can be updated by the owner. • buy taxes, buy marketing taxes, liquidity taxes, dev taxes can be set by the multisig owner. • Swap and Liquify enabled status by the multisig owner. • Multi send tokens by the owner.	**YES, This is valid.**

Audit Summary

According to the standard audit assessment, Customer`s solidity based smart contracts are **“Secured”**. This token contract does contain owner control, which does not make it fully decentralized.

We used various tools like Slither, Solhint and Remix IDE. At the same time this finding is based on critical analysis of the manual audit.
All issues found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the Audit overview section. General overview is presented in AS-IS section and all identified issues can be found in the Audit overview section.

We found 0 critical, 0 high, 0 medium and 1 low and some very low level issues.

Investors Advice: Technical audit of the smart contract does not guarantee the ethical nature of the project. Any owner controlled functions should be executed by the owner with responsibility. All investors/users are advised to do their due diligence before investing in the project.

Technical Quick Stats

Main Category	Subcategory	Result
Contract Programming	Solidity version not specified	Passed
	Solidity version too old	Passed
	Integer overflow/underflow	Passed
	Function input parameters lack of check	Passed
	Function input parameters check bypass	Passed
	Function access control lacks management	Passed
	Critical operation lacks event log	Passed
	Human/contract checks bypass	Passed
	Random number generation/use vulnerability	N/A
	Fallback function misuse	Passed
	Race condition	Passed
	Logical vulnerability	Passed
	Features claimed	Passed
	Other programming issues	Passed
Code Specification	Function visibility not explicitly declared	Passed
	Var. storage location not explicitly declared	Passed
	Use keywords/functions to be deprecated	Passed
	Unused code	Passed
Gas Optimization	“Out of Gas” Issue	Passed
	High consumption ‘for/while’ loop	Passed
	High consumption ‘storage’ storage	Passed
	Assert() misuse	Passed
Business Risk	The maximum limit for mintage not set	Passed
	“Short Address” Attack	Passed
	“Double Spend” Attack	Passed

Overall Audit Result: **PASSED**

Code Quality

This audit scope has 1 smart contract. Smart contract contains Libraries, Smart contracts, inherits and Interfaces. This is a compact and well written smart contract.

The libraries in Catcoin Token are part of its logical algorithm. A library is a different type of smart contract that contains reusable code. Once deployed on the blockchain (only once), it is assigned a specific address and its properties / methods can be reused many times by other contracts in the Catcoin Token.

The Catcoin Token team has **not** provided scenario and unit test scripts, which would have helped to determine the integrity of the code in an automated way.

Code parts are **not well** commented on in the smart contracts. Ethereum's NatSpec commenting style is recommended.

Documentation

We were given a Catcoin Token smart contract code in the form of a bscscan web link. The hash of that code is mentioned above in the table.

As mentioned above, code parts are **not well** commented. But the logic is straightforward. So it is easy to quickly understand the programming flow as well as complex code logic. Comments are very helpful in understanding the overall architecture of the protocol.

Another source of information was its official website: https://catcoin.io which provided rich information about the project architecture and tokenomics.

Use of Dependencies

As per our observation, the libraries are used in this smart contract infrastructure that are based on well known industry standard open source projects.

Apart from libraries, its functions are used in external smart contract calls.

AS-IS overview

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	onlyMultisig	modifier	Passed	No Issue
7	lockTheSwap	modifier	Passed	No Issue
8	name	external	Passed	No Issue
9	symbol	external	Passed	No Issue
10	decimals	external	Passed	No Issue
11	totalSupply	external	Passed	No Issue
12	balanceOf	read	Passed	No Issue
13	setMultisig	external	Missing-zero-address-validation	Refer to audit findings
14	transfer	external	Passed	No Issue
15	allowance	read	Passed	No Issue
16	approve	external	Passed	No Issue
17	transferFrom	external	Passed	No Issue
18	increaseAllowance	external	Passed	No Issue
19	decreaseAllowance	external	Passed	No Issue
20	isExcludedFromReward	read	Passed	No Issue
21	totalFees	read	Passed	No Issue
22	setSwapTreshold	write	access only Multisig	No Issue
22	reflectionFromToken	read	Passed	No Issue
23	tokenFromReflection	read	Passed	No Issue
24	updateWithdrawContract	external	access only Multisig	No Issue
25	updateDevWallet	external	Passed	No Issue
26	addBotToBlacklist	external	access only Multisig	No Issue
27	removeBotFromBlacklist	external	access only Multisig	No Issue
28	excludeFromReward	external	access only Multisig	No Issue
29	includeInReward	external	access only Multisig	No Issue
30	excludeFromFee	external	access only Multisig	No Issue
31	includeInFee	external	access only Multisig	No Issue
32	setFees	external	access only Multisig	No Issue
33	excessFundWithdrawal	external	Centralize risk	Refer to audit findings
34	setSwapAndLiquifyEnabled	external	access only Multisig	No Issue
35	receive	external	Passed	No Issue
36	_reflectFee	write	Passed	No Issue
37	_getTValues	read	Passed	No Issue
38	_getRValues	write	Passed	No Issue
39	_getRate	read	Passed	No Issue

40	_getCurrentSupply	read	Passed	No Issue
41	_takeLiquidity	write	Passed	No Issue
42	_takeWalletFee	write	Passed	No Issue
43	calculateTaxFee	read	Passed	No Issue
44	calculateLiquidityFee	read	Passed	No Issue
45	calculateMarketingFee	read	Passed	No Issue
46	calculateDevFee	read	Passed	No Issue
47	removeAllFee	write	Passed	No Issue
48	setBuy	write	Passed	No Issue
49	setSell	write	Passed	No Issue
50	isExcludedFromFee	read	Passed	No Issue
51	_approve	write	Passed	No Issue
52	openTrading	external	Missing-zero-address-validation	Refer to audit findings
53	_transfer	write	Passed	No Issue
54	swapAndLiquify	write	Passed	No Issue
55	swapTokensForEth	write	Passed	No Issue
56	addLiquidity	write	Centralize risk, Not handle properly return value	Refer to audit findings
57	_tokenTransfer	write	Passed	No Issue
58	_transferStandard	write	Passed	No Issue
59	_transferToExcluded	write	Passed	No Issue
60	_transferFromExcluded	write	Passed	No Issue
61	_transferBothExcluded	write	Passed	No Issue
62	multiSendTokens	write	access only Owner	No Issue

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to token loss etc.
High	High-level vulnerabilities are difficult to exploit; however, they also have significant impact on smart contract execution, e.g. public access to crucial
Medium	Medium-level vulnerabilities are important to fix; however, they can't lead to tokens lose
Low	Low-level vulnerabilities are mostly related to outdated, unused etc. code snippets, that can't have significant impact on execution
Lowest / Code Style / Best Practice	Lowest-level vulnerabilities, code style violations and info statements can't affect smart contract execution and can be ignored.

Email: audit@EtherAuthority.io

Audit Findings

Critical Severity

No Critical severity vulnerabilities were found.

High Severity

No High severity vulnerabilities were found.

Medium

No Medium severity vulnerabilities were found.

Low

(1) Centralize risk:

```
function excessFundWithdrawal () external onlyMultisig {
    if (address(this).balance > 0){
        uint256 amountBNB = address(this).balance;
        payable(withdrawcontract).transfer(amountBNB);
    emit RecoverFunds();
    }
}
```

In this contract multisig can drain all the BNB into withdrawcontract wallet.

```
function addLiquidity(uint256 tokenAmount, uint256 ethAmount) private {
    _approve(address(this), address(uniswapV2Router), tokenAmount);
    uniswapV2Router.addLiquidityETH{value: ethAmount}(
        address(this),
        tokenAmount,
        0,
        0,
        withdrawcontract,
        block.timestamp
    );
}
```

In addLiquidity function call uniswapV2Router.addLiquidityETH function with the specific address "withdrawcontract" that acquire all the LP token that is generated by catcoin BNB pool. If withdrawcontract account will be EOA that can misbehave with LP token.

Resolution: We suggest centralized privileges or roles in the protocol be improved via a decentralized mechanism and using the contract itself (address.(this)) to make your LP token decentralized.

Very Low / Informational / Best practices:

(1) Missing-zero-address-validation:

```
function setMultisig(address _multisig) external onlyMultisig {
    multisig = _multisig;
    emit MultisigUpdate(_multisig);
}
```

In setMultisig there is zero address validation.

```
function openTrading(address payable _withdrawcontract) external onlyOwner {
    require(!tradingOpen,"trading is already open");
    buyFee.tax = 2;
    buyFee.liquidity = 1;
    buyFee.marketing = 2;
    buyFee.dev = 0;
    sellFee.tax = 2;
    sellFee.liquidity = 1;
    sellFee.marketing = 2;
    sellFee.dev = 1;
    tradingOpen = true;
    withdrawcontract = _withdrawcontract;
    emit OpenTrading();
}
```

Addresses should be checked before assignment or external call to make sure they are not zero addresses.

Resolution: We advise adding a zero-check for the passed-in address value to prevent unexpected errors and adding a required field and check zero address validation in the setMultisig function.

(2) Decimal is set to 0:

```
uint8 private constant _decimals = 0;
```

Decimal value is set to 0, there is no provision to change the decimal value. This will not have any fractional amount of the tokens. Only whole numbers will be there and no fractional value.

Resolution: We suggest setting appropriate values to the decimal variable.

(3) Use external function instead of public function:

A public function that is never called by the contract could be declared as external. external functions are more efficient than public functions.

Resolution: Consider using the external attribute for public functions that are never called within the contract.

(4) Not handle properly return value:

```
function addLiquidity(uint256 tokenAmount, uint256 ethAmount) private {
    _approve(address(this), address(uniswapV2Router), tokenAmount);
    uniswapV2Router.addLiquidityETH{value: ethAmount}(
        address(this),
        tokenAmount,
        0,
        0,
        withdrawcontract,
        block.timestamp
    );
}
```

Return value of addLiquidityETH not handled properly.

Resolution: We suggest using variables to receive the return value of the functions and handle both success and failure cases.

Centralization

This smart contract has some functions which can be executed by the Admin (Owner) only. If the admin wallet private key would be compromised, then it would create trouble. Following are Admin functions:

Catcoin.sol

- setMultisig: Multisig address can be set by the multisig owner.
- setSwapTreshold: Swap Threshold value can be set by the multisig owner.
- updateWithdrawContract: Withdraw contract address can be updated by the multisig owner.
- updateDevWallet: Developer wallet address can be updated by the owner.
- addBotToBlacklist: Bot addresses can be blacklisted by the multisig owner.
- removeBotFromBlacklist: Bot addresses can be removed from blacklisted by the multisig owner.
- excludeFromReward: The multisig owner can set excluded account status true.
- includeInReward: The multisig owner can set excluded account status false.
- excludeFromFee: The multisig owner can set excluded account fee status true.
- includeInFee: The multisig owner can set excluded account fee status false.
- setFees:buy taxes, buy marketing taxes, liquidity taxes, dev taxes can be set by the multisig owner.
- excessFundWithdrawal: Excess fund Withdrawal balance by the multisig owner.
- setSwapAndLiquifyEnabled: Swap and Liquify enabled status by the multisig owner.
- openTrading: Open Trading address executed by the owner.
- multiSendTokens: Multi send tokens by the owner.

Ownable.sol

- renounceOwnership: Deleting ownership will leave the contract without an owner, removing any owner-only functionality.
- transferOwnership: Current owner can transfer ownership of the contract to a new account.

To make the smart contract 100% decentralized, we suggest renouncing ownership in the smart contract once its function is completed.

Conclusion

We were given a contract code in the form of bscscan web link. And we have used all possible tests based on given objects as files. We had observed 1 low issue and some informational issues in the smart contracts. But those are not critical ones. So, **it's good to go for the production**.

Since possible test cases can be unlimited for such smart contracts protocol, we provide no such guarantee of future outcomes. We have used all the latest static tools and manual observations to cover maximum possible test cases to scan everything.

Smart contracts within the scope were manually reviewed and analyzed with static analysis tools. Smart Contract's high-level description of functionality was presented in the As-is overview section of the report.

Audit report contains all found security vulnerabilities and other issues in the reviewed code.

Security state of the reviewed smart contract, based on standard audit procedure scope, is **"Secured".**

Our Methodology

We like to work with a transparent process and make our reviews a collaborative effort. The goals of our security audits are to improve the quality of systems we review and aim for sufficient remediation to help protect users. The following is the methodology we use in our security audit process.

Manual Code Review:

In manually reviewing all of the code, we look for any potential issues with code logic, error handling, protocol and header parsing, cryptographic errors, and random number generators. We also watch for areas where more defensive programming could reduce the risk of future mistakes and speed up future audits. Although our primary focus is on the in-scope code, we examine dependency code and behavior when it is relevant to a particular line of investigation.

Vulnerability Analysis:

Our audit techniques included manual code analysis, user interface interaction, and whitebox penetration testing. We look at the project's web site to get a high level understanding of what functionality the software under review provides. We then meet with the developers to gain an appreciation of their vision of the software. We install and use the relevant software, exploring the user interactions and roles. While we do this, we brainstorm threat models and attack surfaces. We read design documentation, review other audit results, search for similar projects, examine source code dependencies, skim open issue tickets, and generally investigate details other than the implementation.

Documenting Results:

We follow a conservative, transparent process for analyzing potential security vulnerabilities and seeing them through successful remediation. Whenever a potential issue is discovered, we immediately create an Issue entry for it in this document, even though we have not yet verified the feasibility and impact of the issue. This process is conservative because we document our suspicions early even if they are later shown to not represent exploitable vulnerabilities. We generally follow a process of first documenting the suspicion with unresolved questions, then confirming the issue through code analysis, live experimentation, or automated tests. Code analysis is the most tentative, and we strive to provide test code, log captures, or screenshots demonstrating our confirmation. After this we analyze the feasibility of an attack in a live system.

Suggested Solutions:

We search for immediate mitigations that live deployments can take, and finally we suggest the requirements for remediation engineering for future releases. The mitigation and remediation recommendations should be scrutinized by the developers and deployment engineers, and successful mitigation and remediation is an ongoing collaborative process after we deliver our report, and before the details are made public.

Disclaimers

EtherAuthority.io Disclaimer

EtherAuthority team has analyzed this smart contract in accordance with the best industry practices at the date of this report, in relation to: cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report, (Source Code); the Source Code compilation, deployment and functionality (performing the intended functions).

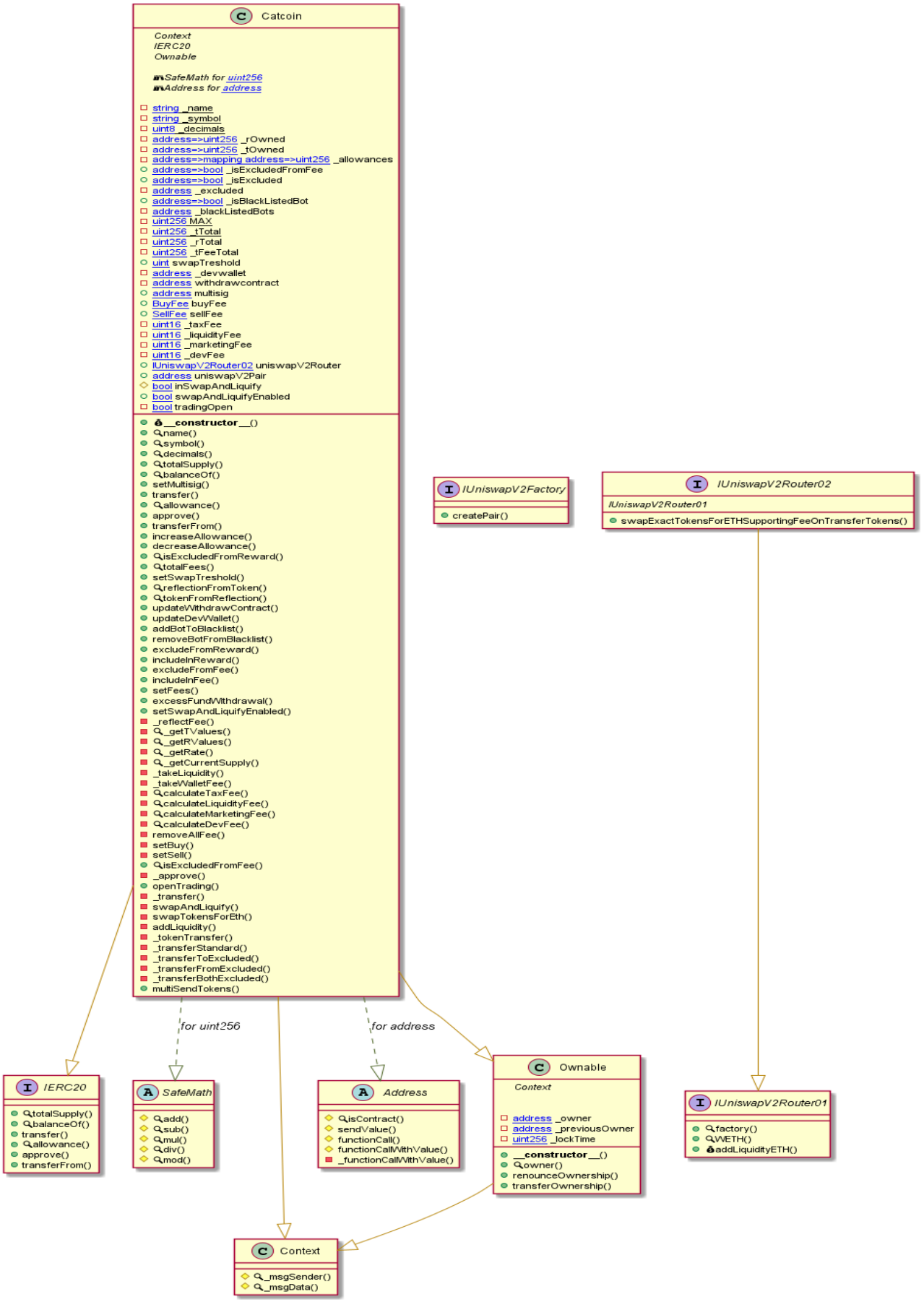
Due to the fact that the total number of test cases are unlimited, the audit makes no statements or warranties on security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bugfree status or any other statements of the contract. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only. We also suggest conducting a bug bounty program to confirm the high level of security of this smart contract.

Technical Disclaimer

Smart contracts are deployed and executed on the blockchain platform. The platform, its programming language, and other software related to the smart contract can have their own vulnerabilities that can lead to hacks. Thus, the audit can't guarantee explicit security of the audited smart contracts.

Appendix

Code Flow Diagram - Catcoin Token



Slither Results Log

Slither Log >> Catcoin.sol

```
Catcoin.allowance(address,address).owner (Catcoin.sol#397) shadows:
        - Ownable.owner() (Catcoin.sol#188-190) (function)
Catcoin._approve(address,address,uint256).owner (Catcoin.sol#720) shadows:
        - Ownable.owner() (Catcoin.sol#188-190) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing

Catcoin.setMultisig(address)._multisig (Catcoin.sol#389) lacks a zero-check on :
                - multisig = _multisig (Catcoin.sol#390)
Catcoin.openTrading(address)._withdrawcontract (Catcoin.sol#727) lacks a zero-check on :
                - withdrawcontract = _withdrawcontract (Catcoin.sol#738)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

Catcoin.swapTokensForEth(uint256) (Catcoin.sol#819-833) has external calls inside a loop: path[1] = uniswapV2Router.WETH() (Ca
tcoin.sol#822)
Catcoin.swapTokensForEth(uint256) (Catcoin.sol#819-833) has external calls inside a loop: uniswapV2Router.swapExactTokensForET
HSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (Catcoin.sol#826-832)
Catcoin.addLiquidity(uint256,uint256) (Catcoin.sol#835-845) has external calls inside a loop: uniswapV2Router.addLiquidityETH{
value: ethAmount}(address(this),tokenAmount,0,0,withdrawcontract,block.timestamp) (Catcoin.sol#837-844)
Catcoin.swapAndLiquify(uint256) (Catcoin.sol#771-817) has external calls inside a loop: address(withdrawcontract).transfer(mar
ketingAmt) (Catcoin.sol#805)
Catcoin.swapAndLiquify(uint256) (Catcoin.sol#771-817) has external calls inside a loop: address(_devwallet).transfer(devAmt) (
Catcoin.sol#809)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-a-loop

Reentrancy in Catcoin._transfer(address,address,uint256) (Catcoin.sol#742-769):
        External calls:
        - swapAndLiquify(contractTokenBalance) (Catcoin.sol#761)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,withdrawcontract,block.times
tamp) (Catcoin.sol#837-844)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.ti
mestamp) (Catcoin.sol#826-832)
        External calls sending eth:
        - swapAndLiquify(contractTokenBalance) (Catcoin.sol#761)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,withdrawcontract,block.times
tamp) (Catcoin.sol#837-844)
```

```
Address._functionCallWithValue(address,bytes,uint256,string) (Catcoin.sol#151-173) uses assembly
        - INLINE ASM (Catcoin.sol#165-168)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Catcoin.removeBotFromBlacklist(address) (Catcoin.sol#498-511) has costly operations inside a loop:
        - _blackListedBots.pop() (Catcoin.sol#506)
Catcoin.includeInReward(address) (Catcoin.sol#523-535) has costly operations inside a loop:
        - _excluded.pop() (Catcoin.sol#530)
Catcoin.lockTheSwap() (Catcoin.sol#336-340) has costly operations inside a loop:
        - inSwapAndLiquify = true (Catcoin.sol#337)
Catcoin.lockTheSwap() (Catcoin.sol#336-340) has costly operations inside a loop:
        - inSwapAndLiquify = false (Catcoin.sol#339)
Catcoin.removeAllFee() (Catcoin.sol#695-700) has costly operations inside a loop:
        - _taxFee = 0 (Catcoin.sol#696)
Catcoin.removeAllFee() (Catcoin.sol#695-700) has costly operations inside a loop:
        - _liquidityFee = 0 (Catcoin.sol#697)
Catcoin.removeAllFee() (Catcoin.sol#695-700) has costly operations inside a loop:
        - _marketingFee = 0 (Catcoin.sol#698)
Catcoin.removeAllFee() (Catcoin.sol#695-700) has costly operations inside a loop:
        - _devFee = 0 (Catcoin.sol#699)
Catcoin.setBuy() (Catcoin.sol#702-707) has costly operations inside a loop:
        - _taxFee = buyFee.tax (Catcoin.sol#703)
Catcoin.setBuy() (Catcoin.sol#702-707) has costly operations inside a loop:
        - _liquidityFee = buyFee.liquidity (Catcoin.sol#704)
Catcoin.setBuy() (Catcoin.sol#702-707) has costly operations inside a loop:
        - _marketingFee = buyFee.marketing (Catcoin.sol#705)
Catcoin.setBuy() (Catcoin.sol#702-707) has costly operations inside a loop:
        - _devFee = buyFee.dev (Catcoin.sol#706)
Catcoin.setSell() (Catcoin.sol#709-714) has costly operations inside a loop:
        - _taxFee = sellFee.tax (Catcoin.sol#710)
Catcoin.setSell() (Catcoin.sol#709-714) has costly operations inside a loop:
        - _liquidityFee = sellFee.liquidity (Catcoin.sol#711)
Catcoin.setSell() (Catcoin.sol#709-714) has costly operations inside a loop:
        - _marketingFee = sellFee.marketing (Catcoin.sol#712)
Catcoin.setSell() (Catcoin.sol#709-714) has costly operations inside a loop:
        - _devFee = sellFee.dev (Catcoin.sol#713)
Catcoin._reflectFee(uint256,uint256) (Catcoin.sol#595-598) has costly operations inside a loop:
```

```
Catcoin.setSell() (Catcoin.sol#709-714) has costly operations inside a loop:
        - _marketingFee = sellFee.marketing (Catcoin.sol#712)
Catcoin.setSell() (Catcoin.sol#709-714) has costly operations inside a loop:
        - _devFee = sellFee.dev (Catcoin.sol#713)
Catcoin._reflectFee(uint256,uint256) (Catcoin.sol#595-598) has costly operations inside a loop:
        - _rTotal = _rTotal.sub(rFee) (Catcoin.sol#596)
Catcoin._reflectFee(uint256,uint256) (Catcoin.sol#595-598) has costly operations inside a loop:
        - _tFeeTotal = _tFeeTotal.add(tFee) (Catcoin.sol#597)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-operations-inside-a-loop
```

```
Pragma version=0.8.4 (Catcoin.sol#5) allows old versions
solc-0.8.4 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (Catcoin.sol#101-111):
        - (success) = recipient.call{value: amount}() (Catcoin.sol#106)
Low level call in Address._functionCallWithValue(address,bytes,uint256,string) (Catcoin.sol#151-173):
        - (success,returndata) = target.call{value: weiValue}(data) (Catcoin.sol#158-160)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
```

```
Function IUniswapV2Router01.WETH() (Catcoin.sol#223) is not in mixedCase
Event CatcoinbotAddedToBlacklist(address) (Catcoin.sol#309) is not in CapWords
Event CatcoinbotRemovedFromBlacklist(address) (Catcoin.sol#310) is not in CapWords
Event CatcoinexcludeUserFromReward(address) (Catcoin.sol#314) is not in CapWords
Event CatcoinincludeUserInReward(address) (Catcoin.sol#315) is not in CapWords
Event CatcoinexcludeUserFromFees(address) (Catcoin.sol#316) is not in CapWords
Event CatcoinincludeUserInFees(address) (Catcoin.sol#317) is not in CapWords
Parameter Catcoin.setMultisig(address)._multisig (Catcoin.sol#389) is not in mixedCase
Parameter Catcoin.setFees(uint16,uint16,uint16,uint16,uint16,uint16,uint16,uint16).buy_tax (Catcoin.sol#548) is not in mixedCa
se
Parameter Catcoin.setFees(uint16,uint16,uint16,uint16,uint16,uint16,uint16,uint16).buy_liquidity (Catcoin.sol#549) is not in m
ixedCase
Parameter Catcoin.setFees(uint16,uint16,uint16,uint16,uint16,uint16,uint16,uint16).buy_marketing (Catcoin.sol#550) is not in m
ixedCase
Parameter Catcoin.setFees(uint16,uint16,uint16,uint16,uint16,uint16,uint16,uint16).buy_dev (Catcoin.sol#551) is not in mixedCa
se
Parameter Catcoin.setFees(uint16,uint16,uint16,uint16,uint16,uint16,uint16,uint16).sell_tax (Catcoin.sol#552) is not in mixedC
ase
Parameter Catcoin.setFees(uint16,uint16,uint16,uint16,uint16,uint16,uint16,uint16).sell_liquidity (Catcoin.sol#553) is not in
mixedCase
Parameter Catcoin.setFees(uint16,uint16,uint16,uint16,uint16,uint16,uint16,uint16).sell_marketing (Catcoin.sol#554) is not in
mixedCase
Parameter Catcoin.setFees(uint16,uint16,uint16,uint16,uint16,uint16,uint16,uint16).sell_dev (Catcoin.sol#555) is not in mixedC
ase
Parameter Catcoin.setSwapAndLiquifyEnabled(bool)._enabled (Catcoin.sol#588) is not in mixedCase
Parameter Catcoin.calculateTaxFee(uint256)._amount (Catcoin.sol#679) is not in mixedCase
Parameter Catcoin.calculateLiquidityFee(uint256)._amount (Catcoin.sol#683) is not in mixedCase
Parameter Catcoin.calculateMarketingFee(uint256)._amount (Catcoin.sol#687) is not in mixedCase
Parameter Catcoin.calculateDevFee(uint256)._amount (Catcoin.sol#691) is not in mixedCase
Parameter Catcoin.openTrading(address)._withdrawcontract (Catcoin.sol#727) is not in mixedCase
Constant Catcoin._name (Catcoin.sol#255) is not in UPPER_CASE_WITH_UNDERSCORES
Constant Catcoin._symbol (Catcoin.sol#256) is not in UPPER_CASE_WITH_UNDERSCORES
Constant Catcoin._decimals (Catcoin.sol#257) is not in UPPER_CASE_WITH_UNDERSCORES
```

```
Constant Catcoin._symbol (Catcoin.sol#256) is not in UPPER_CASE_WITH_UNDERSCORES
Constant Catcoin._decimals (Catcoin.sol#257) is not in UPPER_CASE_WITH_UNDERSCORES
Variable Catcoin._isExcludedFromFee (Catcoin.sol#262) is not in mixedCase
Variable Catcoin._isExcluded (Catcoin.sol#263) is not in mixedCase
Variable Catcoin._isBlackListedBot (Catcoin.sol#265) is not in mixedCase
Constant Catcoin._tTotal (Catcoin.sol#269) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Redundant expression "this (Catcoin.sol#88)" inContext (Catcoin.sol#83-91)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
```

```
Variable Catcoin._transferFromExcluded(address,address,uint256).rTransferAmount (Catcoin.sol#926) is too similar to Catcoin._g
etTValues(uint256).tTransferAmount (Catcoin.sol#611)
Variable Catcoin.reflectionFromToken(uint256,bool).rTransferAmount (Catcoin.sol#455) is too similar to Catcoin._transferToExcl
uded(address,address,uint256).tTransferAmount (Catcoin.sol#897)
Variable Catcoin.reflectionFromToken(uint256,bool).rTransferAmount (Catcoin.sol#455) is too similar to Catcoin._transferBothEx
cluded(address,address,uint256).tTransferAmount (Catcoin.sol#945)
Variable Catcoin._transferToExcluded(address,address,uint256).rTransferAmount (Catcoin.sol#902) is too similar to Catcoin._tra
nsferToExcluded(address,address,uint256).tTransferAmount (Catcoin.sol#897)
Variable Catcoin._transferFromExcluded(address,address,uint256).rTransferAmount (Catcoin.sol#926) is too similar to Catcoin._t
ransferFromExcluded(address,address,uint256).tTransferAmount (Catcoin.sol#921)
Variable Catcoin._getRValues(uint256,uint256,uint256,uint256,uint256).rTransferAmount (Catcoin.sol#636-639) is too similar to
Catcoin._transferToExcluded(address,address,uint256).tTransferAmount (Catcoin.sol#897)
Variable Catcoin._transferFromExcluded(address,address,uint256).rTransferAmount (Catcoin.sol#926) is too similar to Catcoin._t
ransferToExcluded(address,address,uint256).tTransferAmount (Catcoin.sol#897)
Variable Catcoin._transferBothExcluded(address,address,uint256).rTransferAmount (Catcoin.sol#950) is too similar to Catcoin._t
ransferToExcluded(address,address,uint256).tTransferAmount (Catcoin.sol#897)
Variable Catcoin._transferFromExcluded(address,address,uint256).rTransferAmount (Catcoin.sol#926) is too similar to Catcoin._t
ransferStandard(address,address,uint256).tTransferAmount (Catcoin.sol#873)
Variable Catcoin._transferToExcluded(address,address,uint256).rTransferAmount (Catcoin.sol#902) is too similar to Catcoin._tra
nsferBothExcluded(address,address,uint256).tTransferAmount (Catcoin.sol#945)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

Catcoin.uniswapV2Pair (Catcoin.sol#301) should be immutable
Catcoin.uniswapV2Router (Catcoin.sol#300) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
Catcoin.sol analyzed (9 contracts with 84 detectors), 124 result(s) found
```

Solidity Static Analysis

Catcoin.sol

Security

Transaction origin:

Use of tx.origin: "tx.origin" is useful only in very exceptional cases. If you use it for authentication, you usually want to replace it by "msg.sender", because otherwise any contract you call can act on your behalf.
more
Pos: 749:35:

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in Catcoin.swapTokensForEth(uint256): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 819:4:

Gas & Economy

Gas costs:

Gas requirement of function Catcoin.removeBotFromBlacklist is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 498:4:

Gas costs:

Gas requirement of function Catcoin.includeInReward is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 523:4:

For loop over dynamic array:

Loops that do not have a fixed number of iterations, for example, loops that depend on storage values, have to be used carefully. Due to the block gas limit, transactions can only consume a certain amount of gas. The number of iterations in a loop can grow beyond the block gas limit which can cause the complete contract to be stalled at a certain point. Additionally, using unbounded loops incurs in a lot of avoidable gas costs. Carefully test how many items at maximum you can pass to such functions to make it successful.

more

Pos: 970:8:

Miscellaneous

Constant/View/Pure functions:

IUniswapV2Router02.swapExactTokensForETHSupportingFeeOnTransferTokens(uint256,u
: Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 241:4:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.

more

Pos: 971:12:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.

Pos: 788:30:

Solhint Linter

Catcoin.sol

```
Catcoin.sol:5:1: Error: Compiler version =0.8.10 does not satisfy the
r semver requirement
Catcoin.sol:183:5: Error: Explicitly mark visibility in function (Set
ignoreConstructors to true if using solidity >=0.7.0)
Catcoin.sol:223:5: Error: Function name must be in mixedCase
Catcoin.sol:250:1: Error: Contract has 25 states declarations but
allowed no more than 15
Catcoin.sol:257:28: Error: Constant name must be in capitalized
SNAKE_CASE
Catcoin.sol:269:30: Error: Constant name must be in capitalized
SNAKE_CASE
Catcoin.sol:303:5: Error: Explicitly mark visibility of
stateCatcoin.sol:315:5: Error: Event name must be in CamelCase
Catcoin.sol:316:5: Error: Event name must be in CamelCase
Catcoin.sol:317:5: Error: Event name must be in
CamelCaseCatcoin.sol:325:9: Error: Variable name must be in mixedCase
Catcoin.sol:326:9: Error: Variable name must be in mixedCase
Catcoin.sol:327:9: Error: Variable name must be in mixedCase
Catcoin.sol:328:9: Error: Variable name must be in mixedCase
Catcoin.sol:347:5: Error: Explicitly mark visibility in function (Set
ignoreConstructors to true if using solidity
>=0.7.0)Catcoin.sol:553:9: Error: Variable name must be in mixedCase
Catcoin.sol:554:9: Error: Variable name must be in mixedCase
Catcoin.sol:555:9: Error: Variable name must be in mixedCase
Catcoin.sol:593:32: Error: Code contains empty blocks
Catcoin.sol:749:36: Error: Avoid to use tx.origin
Catcoin.sol:831:13: Error: Avoid to make time-based decisions in your
business logic
Catcoin.sol:843:13: Error: Avoid to make time-based decisions in your
business logic
```

Software analysis result:

These software reported many false positive results and some are informational issues. So, those issues can be safely ignored.

Email: audit@EtherAuthority.io