expelee

Building the Futuristic Blockchain Ecosystem

SECURITY AUDIT REPORT

CateCoin

TOKEN OVERVIEW

Risk Findings

Severity	Found
High	0
Medium	1
Low	1
Informational	2

Centralization Risks

Owner Privileges	Description
Can Owner Set Taxes >25% ?	Not Detected
Owner needs to enable trading ?	Not Detected
Can Owner Disable Trades ?	Not Detected
Can Owner Mint ?	Not Detected
Can Owner Blacklist ?	Not Detected
Can Owner set Max Wallet amount ?	Not Detected
Can Owner Set Max TX amount ?	Not Detected

TABLE OF CONTENTS

OVERVIEW

The Expelee team has performed a line-by-line manual analysis and automated review of the smart contract. The smart contract was analysed mainly for common smart contract vulnerabilities, exploits, and manipulation hacks. According to the smart contract audit:

Audit Result	**Passed**
KYC Verification	**-**
Audit Date	**08 April 2024**

CONTRACT DETAILS

Token Address: 0x1689D1fd1dCcd01D46E1FD06C1CE3D21F4982EB3

Name: CateCoin

Symbol: CATE

Decimals: 18

Network: BscScan

Token Type: BEP-20

Owner: Renounced

Deployer: 0xaA0E4e2276B9457203065O167d4744FACC1bb9BC

Token Supply: 21,000,000,000,000

Checksum: Ac6659e84744e0102ab19c1d1e78a321

Testnet:
https://testnet.bscscan.com/address/0xd5377c1e2d8302bf1c1d12e9d271a53099f741ad#code

AUDIT METHODOLOGY

Audit Details

Our comprehensive audit report provides a full overview of the audited system's architecture, smart contract codebase, and details on any vulnerabilities found within the system.

Audit Goals

The audit goal is to ensure that the project is built to protect investors and users, preventing potentially catastrophic vulnerabilities after launch , that lead to scams and rugpulls.

Code Quality

Our analysis includes both automatic tests and manual code analysis for the following aspects:

- Exploits
- Back-doors
- Vulnerability
- Accuracy
- Readability

Tools

- DE
- Open Zeppelin
- Code Analyzer
- Solidity Code
- Compiler
- Hardhat

VULNERABILITY CHECKS

Design Logic	Passed
Compiler warnings	Passed
Private user data leaks	Passed
Timestamps dependence	Passed
Integer overflow and underflow	Passed
Race conditions & reentrancy. Cross-function race conditions	Passed
Possible delays in data delivery	Passed
Oracle calls	Passed
Front Running	Passed
DoS with Revert	Passed
DoS with block gas limit	Passed
Methods execution permissions	Passed
Economy model	Passed
Impact of the exchange rate on the logic	Passed
Malicious event log	Passed
Scoping and declarations	Passed
Uninitialized storage pointers	Passed
Arithmetic accuracy	Passed
Cross-function race conditions	Passed
Safe Zepplin module	Passed

RISK CLASSIFICATION

When performing smart contract audits, our specialists look for known vulnerabilities as well as logical and acces control issues within the code. The exploitation of these issues by malicious actors may cause serious financial damage to projects that failed to get an audit in time. We categorize these vulnerabilities by the following levels:

High Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

Medium Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

Low Risk

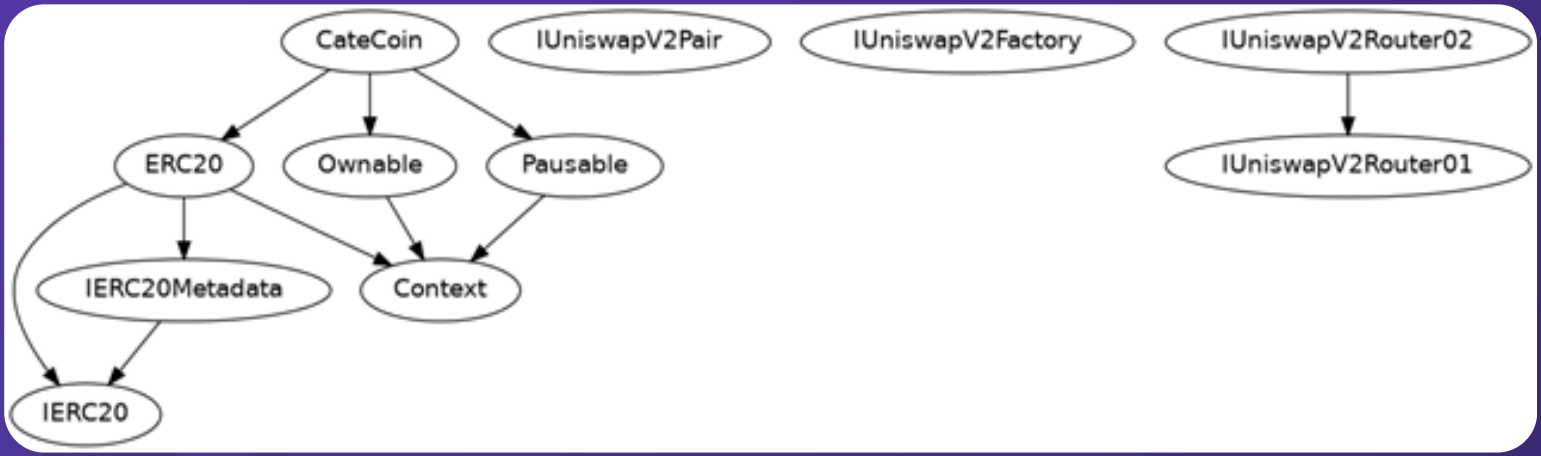
Issues on this level are minor details and warning that can remain unfixed.

Informational

Issues on this level are minor details and warning that can remain unfixed.

expelee

INHERITANCE TREE



expelee

STATIC ANALYSIS

```
INFO:Detectors:
CateCoin.handleTax(address,address,uint256) (CateCoin.sol#543-633) ignores return value by taxWallets[charity].call{value: charityETH}() (CateCoin.sol#617)
CateCoin.handleTax(address,address,uint256) (CateCoin.sol#543-633) ignores return value by taxWallets[charity].call{value: ethGained - (liquidityETH + charityETH)}() (CateCoin.sol#620)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-low-level-calls
INFO:Detectors:
CateCoin.handleTax(address,address,uint256).tax (CateCoin.sol#549) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
CateCoin.handleTax(address,address,uint256) (CateCoin.sol#543-633) ignores return value by uniswapV2Router02.swapExactTokensForETH(toSell,0,sellPath,address(this),block.timestamp) (CateCoin.sol#585-591)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
INFO:Detectors:
CateCoin.setPayable(address).payableWallet (CateCoin.sol#536) lacks a zero-check on :
        - Router = payableWallet (CateCoin.sol#537)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in CateCoin.handleTax(address,address,uint256) (CateCoin.sol#543-633):
        External calls:
        - _transfer(from,address(this),tax) (CateCoin.sol#566)
                - uniswapV2Router02.swapExactTokensForETH(toSell,0,sellPath,address(this),block.timestamp) (CateCoin.sol#585-591)
                - (amountToken,amountETH,liquidity) = uniswapV2Router02.addLiquidityETH{value: liquidityETH}(address(this),liquidityToken,0,0,taxWallets[liquidity],block.timestamp) (CateCoin.sol#602-609)
                - taxWallets[charity].call{value: charityETH}() (CateCoin.sol#617)
                - taxWallets[charity].call{value: ethGained - (liquidityETH + charityETH)}() (CateCoin.sol#620)
        External calls sending eth:
        - _transfer(from,address(this),tax) (CateCoin.sol#566)
                - (amountToken,amountETH,liquidity) = uniswapV2Router02.addLiquidityETH{value: liquidityETH}(address(this),liquidityToken,0,0,taxWallets[liquidity],block.timestamp) (CateCoin.sol#602-609)
                - taxWallets[charity].call{value: charityETH}() (CateCoin.sol#617)
                - taxWallets[charity].call{value: ethGained - (liquidityETH + charityETH)}() (CateCoin.sol#620)
        State variables written after the call(s):
        - _approve(address(this),address(uniswapV2Router02),toSell) (CateCoin.sol#583)
                - _allowances[owner][spender] = amount (CateCoin.sol#280)
Reentrancy in CateCoin.handleTax(address,address,uint256) (CateCoin.sol#543-633):
        External calls:
        - _transfer(from,address(this),tax) (CateCoin.sol#566)
                - uniswapV2Router02.swapExactTokensForETH(toSell,0,sellPath,address(this),block.timestamp) (CateCoin.sol#585-591)
                - (amountToken,amountETH,liquidity) = uniswapV2Router02.addLiquidityETH{value: liquidityETH}(address(this),liquidityToken,0,0,taxWallets[liquidity],block.timestamp) (CateCoin.sol#602-609)
                - taxWallets[charity].call{value: charityETH}() (CateCoin.sol#617)
                - taxWallets[charity].call{value: ethGained - (liquidityETH + charityETH)}() (CateCoin.sol#620)
        - uniswapV2Router02.swapExactTokensForETH(toSell,0,sellPath,address(this),block.timestamp) (CateCoin.sol#585-591)
        External calls sending eth:
        - _transfer(from,address(this),tax) (CateCoin.sol#566)
                - (amountToken,amountETH,liquidity) = uniswapV2Router02.addLiquidityETH{value: liquidityETH}(address(this),liquidityToken,0,0,taxWallets[liquidity],block.timestamp) (CateCoin.sol#602-609)
                - taxWallets[charity].call{value: charityETH}() (CateCoin.sol#617)
                - taxWallets[charity].call{value: ethGained - (liquidityETH + charityETH)}() (CateCoin.sol#620)
        State variables written after the call(s):
        - _approve(address(this),address(uniswapV2Router02),liquidityToken) (CateCoin.sol#600)
                - _allowances[owner][spender] = amount (CateCoin.sol#280)
Reentrancy in CateCoin.handleTax(address,address,uint256) (CateCoin.sol#543-633):
        External calls:
        - _transfer(from,address(this),tax) (CateCoin.sol#566)
                - uniswapV2Router02.swapExactTokensForETH(toSell,0,sellPath,address(this),block.timestamp) (CateCoin.sol#585-591)
```

```
INFO:Detectors:
CateCoin.handleTax(address,address,uint256) (CateCoin.sol#543-633) uses timestamp for comparisons
        Dangerous comparisons:
        - tax > 0 (CateCoin.sol#555)
        - tax > 0 (CateCoin.sol#565)
        - taxSum == 0 (CateCoin.sol#574)
        - ethValue >= swapThreshold (CateCoin.sol#578)
        - remainingTokens > 0 (CateCoin.sol#613)
        - ethGained - (liquidityETH + charityETH) > 0 (CateCoin.sol#619)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Context._msgData() (CateCoin.sol#56-58) is never used and should be removed
ERC20._burn(address,uint256) (CateCoin.sol#175-190) is never used and should be removed
Pausable.pause() (CateCoin.sol#276-279) is never used and should be removed
Pausable.unpause() (CateCoin.sol#283-286) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
solc-0.8.24 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in CateCoin.handleTax(address,address,uint256) (CateCoin.sol#543-633):
        - taxWallets[charity].call{value: charityETH}() (CateCoin.sol#617)
        - taxWallets[charity].call{value: ethGained - (liquidityETH + charityETH)}() (CateCoin.sol#620)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (CateCoin.sol#302) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (CateCoin.sol#303) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (CateCoin.sol#320) is not in mixedCase
Function IUniswapV2Router01.WETH() (CateCoin.sol#356) is not in mixedCase
Variable CateCoin.Router (CateCoin.sol#501) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (CateCoin.sol#361) is too similar to IUniswapV2Router01.addLiquidity(address,address,uin
t256,uint256,uint256,uint256,address,uint256).amountBDesired (CateCoin.sol#362)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
CateCoin.slitherConstructorVariables() (CateCoin.sol#489-702) uses literals with too many digits:
        - swapThreshold = 500000000000 (CateCoin.sol#493)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
CateCoin.charityTaxBuy (CateCoin.sol#496) should be constant
CateCoin.charityTaxSell (CateCoin.sol#499) should be constant
CateCoin.denominator (CateCoin.sol#492) should be constant
CateCoin.liquidityTaxBuy (CateCoin.sol#495) should be constant
CateCoin.liquidityTaxSell (CateCoin.sol#498) should be constant
CateCoin.swapThreshold (CateCoin.sol#493) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
```

expelee

STATIC ANALYSIS

```
INFO:Detectors:
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (CateCoin.sol#302) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (CateCoin.sol#303) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (CateCoin.sol#320) is not in mixedCase
Function IUniswapV2Router01.WETH() (CateCoin.sol#356) is not in mixedCase
Variable CateCoin.Router (CateCoin.sol#501) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (CateCoin.sol#361) is too similar to IUniswapV2Router01.addLiquidity(address,address,uin
t256,uint256,uint256,uint256,address,uint256).amountBDesired (CateCoin.sol#362)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
CateCoin.slitherConstructorVariables() (CateCoin.sol#489-702) uses literals with too many digits:
        - swapThreshold = 50000000000 (CateCoin.sol#493)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
CateCoin.charityTaxBuy (CateCoin.sol#496) should be constant
CateCoin.charityTaxSell (CateCoin.sol#499) should be constant
CateCoin.denominator (CateCoin.sol#492) should be constant
CateCoin.liquidityTaxBuy (CateCoin.sol#495) should be constant
CateCoin.liquidityTaxSell (CateCoin.sol#498) should be constant
CateCoin.swapThreshold (CateCoin.sol#493) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
CateCoin.initialSupply (CateCoin.sol#491) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
INFO:Slither:CateCoin.sol analyzed (11 contracts with 93 detectors), 54 result(s) found
```

TESTNET VERSION

1- Set Buy Tax (passed):
https://testnet.bscscan.com/tx/0x61a55272d95f689cf299e46d5b2c57367ae9438da9ce4836dd771297574bc973

2- Set Sell Tax (passed):
https://testnet.bscscan.com/tx/0x29202e8a5e64b1acf79f44c0f1bbb3a484082ce20c7455f495987a843742a90b

3- Set Tax Wallets (passed):
https://testnet.bscscan.com/tx/0xe1e331d2e8c8d4da595419c48ba8c5691991d50cda2bc9d89dd298abc722f373

4- Disable Tax (passed):
https://testnet.bscscan.com/tx/0x8c8279378feca8aa0d7252dd5509cc02189752dda760f1dfa37b22c1c3542cac

5- Enable Tax (passed):
https://testnet.bscscan.com/tx/0x1a58b392eff33b4eda8589353b86819472c6122738c8b4c74802a5c5f0f37425

MANUAL REVIEW

Severity Criteria

Expelee assesses the severity of disclosed vulnerabilities according to methodology based on OWASP standarts.

Vulnerabilities are dividend into three primary risk categroies:
High
Medium
Low

High-level considerations for vulnerabilities span the following key areas when conducting assessments:

- Malicious input handling
- Escalation of privileges
- Arithmetic
- Gas use

Overall Risk Severity				
Impact	HIGH	Medium	High	Critical
	MEDIUM	Low	Medium	High
	LOW	Note	Low	Medium
		LOW	MEDIUM	HIGH
	Likelihood			

MEDIUM RISK FINDING

Centralization – Liquidity is added to EOA

Severity: Medium

subject: handleTax

Status: Open

Overview:

Liquidity is adding to EOA. It may be drained by the taxWallets.

```
(uint amountToken, uint amountETH, uint liquidity) =
uniswapV2Router02.addLiquidityETH{value: liquidityETH}(
address(this),
        liquidityToken,
0,
0,
        taxWallets["liquidity"],
        block.timestamp
      );
```

Suggestion:

It is suggested that the address should be a contract address or a dead address.

LOW RISK FINDING

Centralization – Missing Events

Severity: Low

subject: Missing Events

Status: Open

Overview:
They serve as a mechanism for emitting and recording data onto the blockchain, making it transparent and easily accessible.

```
function setBuyTax(uint256 dev, uint256 marketing, uint256 liquidity,
uint256 charity) public onlyOwner {
    buyTaxes["dev"] = dev;
    buyTaxes["marketing"] = marketing;
    buyTaxes["liquidity"] = liquidity;
    buyTaxes["charity"] = charity;
  }
function setSellTax(uint256 dev, uint256 marketing, uint256 liquidity,
uint256 charity) public onlyOwner {
    sellTaxes["dev"] = dev;
    sellTaxes["marketing"] = marketing;
    sellTaxes["liquidity"] = liquidity;
    sellTaxes["charity"] = charity;
  }
function setTaxWallets(address charity) public onlyOwner {
   taxWallets["charity"] = charity;
  }
```

Suggestion:
Emit an event for critical changes.

INFORMATIONAL & OPTIMIZATIONS

Optimization

Severity: Informational

subject: Floating Pragma

Status: Open

Overview:

It is considered best practice to pick one compiler version and stick with it. With a floating pragma, contracts may accidentally be deployed using an outdated.

pragma solidity ^0.8.0;

Suggestion:

Adding the latest constant version of solidity is recommended, as this prevents the unintentional deployment of a contract with an outdated compiler that contains unresolved bugs.

INFORMATIONAL & OPTIMIZATIONS

Optimization

Severity: Optimization

subject: Remove unused code.

Status: Open

Overview:

Unused variables are allowed in Solidity, and they do. not pose a direct security issue. It is the best practice. though to avoid them

```
 function _msgData() internal view virtual returns (bytes calldata) {
return msg.data;
  }
function _pause() internal virtual whenNotPaused {
    _paused = true;
emit Paused(_msgSender());
  }
function _unpause() internal virtual whenPaused {
    _paused = false;
emit Unpaused(_msgSender());
  }
```

ABOUT EXPELEE

Expelee is a product-based aspirational Web3 start-up. Coping up with numerous solutions for blockchain security and constructing a Web3 ecosystem from deal making platform to developer hosting open platform, while also developing our own commercial and sustainable blockchain.

www.expelee.com

expeleeofficial

expelee

Expelee

expelee

expelee_official

expelee-co

expelee

Building the Futuristic Blockchain Ecosystem

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 guarantess against the sale of team tokens or the removal of liquidity by the project audited in this document.

Always do your own research and project yourselves from being scammed. The Expelee 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 Expelee receive a payment to manipulate those results or change the awarding badge that we will be adding in our website. Alway 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 Expelee team disclaims any liability for the resulting losses.