auditace.tech

AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

BSCCAT

DATED : 27 May 24'

AUDIT SUMMARY

Project name – BSCCAT

Date: 27 May, 2024

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	0	0	3	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/address/0x9b334e7c065ff27981bf862e263f9cb1da9469a7#code

Token Information

Token Address:
0x0EA8286FFc0080061D60A156cce02Db24bB06858

Name: BSCCAT

Symbol: BCAT

Decimals: 9

Network: BscScan

Token Type: BEP-20

Owner: 0x8514B807D95F9318CECE629b8CB5C3f14319Bb29

Deployer:
0x8514B807D95F9318CECE629b8CB5C3f14319Bb29

Token Supply: 420000000000000000

Checksum: A2032c616934aeb47e6039f76b20d223

Testnet:
https://testnet.bscscan.com/address/0x9b334e7c065ff27981bf862e263f9cb1da9469a7#code

TOKEN OVERVIEW

Buy Fee: 0-0%

Sell Fee: 0-0%

Transfer Fee: 0-0%

Fee Privilege: Owner

Ownership: Owned

Minting: None

Max Tx: No

Blacklist: No

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	0
Medium-Risk	0
Low-Risk	3
Gas Optimization / Suggestions	1

INHERITANCE TREE

POINTS TO NOTE

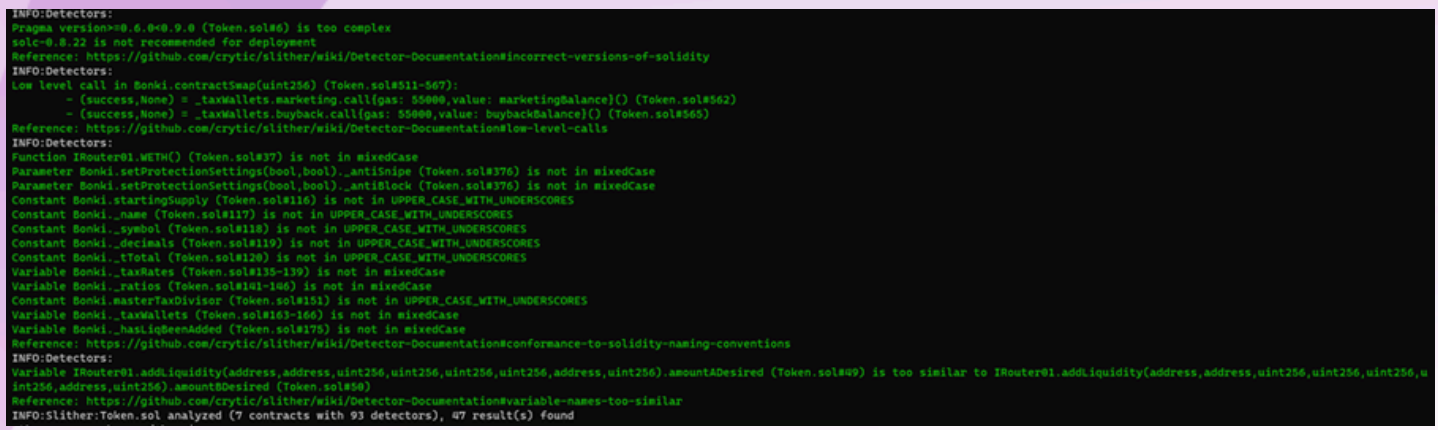
- The owner can transfer ownership.
- The owner can renounce ownership.
- The owner can set the buy and sell fees not more than 25%.
- The owner can exclude addresses from rewards.
- The owner can include addresses in the rewards.
- The owner can set a swapback amount of not less than 0.05% of the total supply.

STATIC ANALYSIS

```
INFO:Detectors:
LiquidityGeneratorToken.addLiquidity(uint256,uint256) (LiquidityGeneratorToken.sol#1541-1554) ignores return value by uniswapV2Router.addLiquidityETH{value:
 ethAmount}(address(this),tokenAmount,0,0,address(0xdead),block.timestamp) (LiquidityGeneratorToken.sol#1546-1553)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
INFO:Detectors:
LiquidityGeneratorToken.allowance(address,address).owner (LiquidityGeneratorToken.sol#1078) shadows:
        - Ownable.owner() (LiquidityGeneratorToken.sol#150-152) (function)
LiquidityGeneratorToken._approve(address,address,uint256).owner (LiquidityGeneratorToken.sol#1449) shadows:
        - Ownable.owner() (LiquidityGeneratorToken.sol#150-152) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
LiquidityGeneratorToken.setTaxFeePercent(uint256) (LiquidityGeneratorToken.sol#1241-1247) should emit an event for:
        - _taxFee = taxFeeBps (LiquidityGeneratorToken.sol#1242)
LiquidityGeneratorToken.setLiquidityFeePercent(uint256) (LiquidityGeneratorToken.sol#1249-1258) should emit an event for:
        - _liquidityFee = liquidityFeeBps (LiquidityGeneratorToken.sol#1253)
LiquidityGeneratorToken.setCharityFeePercent(uint256) (LiquidityGeneratorToken.sol#1260-1266) should emit an event for:
        - _charityFee = charityFeeBps (LiquidityGeneratorToken.sol#1261)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
LiquidityGeneratorToken.constructor(string,string,uint256,address,address,uint16,uint16,uint16,address,uint256).serviceFeeReceiver_ (LiquidityGeneratorToken
.sol#987) lacks a zero-check on :
                - address(serviceFeeReceiver_).transfer(serviceFee_) (LiquidityGeneratorToken.sol#1045)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
```

```
INFO:Detectors:
Address.isContract(address) (LiquidityGeneratorToken.sol#445-455) uses assembly
        - INLINE ASM (LiquidityGeneratorToken.sol#451-453)
Address.verifyCallResult(bool,bytes,string) (LiquidityGeneratorToken.sol#614-634) uses assembly
        - INLINE ASM (LiquidityGeneratorToken.sol#626-629)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
LiquidityGeneratorToken.includeInReward(address) (LiquidityGeneratorToken.sol#1200-1211) has costly operations inside a loop:
        - _excluded.pop() (LiquidityGeneratorToken.sol#1207)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-operations-inside-a-loop
INFO:Detectors:
Address.functionCall(address,bytes) (LiquidityGeneratorToken.sol#498-500) is never used and should be removed
Address.functionCall(address,bytes,string) (LiquidityGeneratorToken.sol#508-514) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (LiquidityGeneratorToken.sol#527-533) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (LiquidityGeneratorToken.sol#541-552) is never used and should be removed
Address.functionDelegateCall(address,bytes) (LiquidityGeneratorToken.sol#587-589) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (LiquidityGeneratorToken.sol#597-606) is never used and should be removed
Address.functionStaticCall(address,bytes) (LiquidityGeneratorToken.sol#560-562) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (LiquidityGeneratorToken.sol#570-579) is never used and should be removed
Address.isContract(address) (LiquidityGeneratorToken.sol#445-455) is never used and should be removed
Address.sendValue(address,uint256) (LiquidityGeneratorToken.sol#473-478) is never used and should be removed
Address.verifyCallResult(bool,bytes,string) (LiquidityGeneratorToken.sol#614-634) is never used and should be removed
Context._msgData() (LiquidityGeneratorToken.sol#110-112) is never used and should be removed
SafeMath.div(uint256,uint256,string) (LiquidityGeneratorToken.sol#380-389) is never used and should be removed
SafeMath.mod(uint256,uint256) (LiquidityGeneratorToken.sol#340-342) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (LiquidityGeneratorToken.sol#406-415) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (LiquidityGeneratorToken.sol#211-217) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (LiquidityGeneratorToken.sol#253-258) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (LiquidityGeneratorToken.sol#265-270) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (LiquidityGeneratorToken.sol#236-246) is never used and should be removed
SafeMath.trySub(uint256,uint256) (LiquidityGeneratorToken.sol#224-229) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version=0.8.17 (LiquidityGeneratorToken.sol#911) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
```

STATIC ANALYSIS



```
INFO:Detectors:
Pragma version>=0.6.0<0.9.0 (Token.sol#6) is too complex
solc-0.8.22 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in Bonki.contractSwap(uint256) (Token.sol#511-567):
        - (success,None) = _taxWallets.marketing.call{gas: 55000,value: marketingBalance}() (Token.sol#562)
        - (success,None) = _taxWallets.buyback.call{gas: 55000,value: buybackBalance}() (Token.sol#565)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IRouter01.WETH() (Token.sol#37) is not in mixedCase
Parameter Bonki.setProtectionSettings(bool,bool)._antiSnipe (Token.sol#376) is not in mixedCase
Parameter Bonki.setProtectionSettings(bool,bool)._antiBlock (Token.sol#376) is not in mixedCase
Constant Bonki.startingSupply (Token.sol#116) is not in UPPER_CASE_WITH_UNDERSCORES
Constant Bonki._name (Token.sol#117) is not in UPPER_CASE_WITH_UNDERSCORES
Constant Bonki._symbol (Token.sol#118) is not in UPPER_CASE_WITH_UNDERSCORES
Constant Bonki._decimals (Token.sol#119) is not in UPPER_CASE_WITH_UNDERSCORES
Constant Bonki._tTotal (Token.sol#120) is not in UPPER_CASE_WITH_UNDERSCORES
Variable Bonki._taxRates (Token.sol#135-139) is not in mixedCase
Variable Bonki._ratios (Token.sol#141-146) is not in mixedCase
Constant Bonki.masterTaxDivisor (Token.sol#151) is not in UPPER_CASE_WITH_UNDERSCORES
Variable Bonki._taxWallets (Token.sol#163-166) is not in mixedCase
Variable Bonki._hasLiqBeenAdded (Token.sol#175) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable IRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (Token.sol#49) is too similar to IRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (Token.sol#50)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Slither:Token.sol analyzed (7 contracts with 93 detectors), 47 result(s) found
```

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- Approve (passed):

https://testnet.bscscan.com/tx/0xb1b4b48710ce2c75ada7e71d4773a7d929502221229f31183f08660de48bef44

2- Set Charity Fee Percent (passed):

https://testnet.bscscan.com/tx/0x3b0390935b522230dd0fdbb1e2bd29d29a35d036e18654ef91df586dbc66164b

3- Set Liquidity Fee Percent (passed):

https://testnet.bscscan.com/tx/0xbf9b4fbb4a49b9f3310af34b3c82806669f3a52af085b0bda7ae87e1e86b4c63

4- Set Tax Fee Percent (passed):

https://testnet.bscscan.com/tx/0xa5120edd7f3c7ce3c593e08eeacf5be32495774e6d8d4c02ece40d08ca6d30e6

MANUAL TESTING

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 setTaxFeePercent(uint256 taxFeeBps) external onlyOwner {
   _taxFee = taxFeeBps;
   require(
     _taxFee + _liquidityFee + _charityFee <= MAX_FEE,
     "Total fee is over 25%"
   );
 }
function setLiquidityFeePercent(uint256 liquidityFeeBps)
   external
   onlyOwner
 {
   _liquidityFee = liquidityFeeBps;
   require(
     _taxFee + _liquidityFee + _charityFee <= MAX_FEE,
     "Total fee is over 25%"
   );
 }
 function setCharityFeePercent(uint256 charityFeeBps) external onlyOwner {
   _charityFee = charityFeeBps;
   require(
     _taxFee + _liquidityFee + _charityFee <= MAX_FEE,
     "Total fee is over 25%"
   );
 }
```

MANUAL TESTING

Centralization – Missing Visibility

Severity: Low

Subject: Visibility

Status: Open

Overview:

It's simply saying that no visibility was specified, so it's going with the default. This has been related to security issues in contracts.

```
bool inSwapAndLiquify;
```

Suggestion:

You can easily silence the warning by adding the public/private.

MANUAL TESTING

Centralization – Local variable Shadowing

Severity: Low

Subject: Variable Shadowing

Status: Open

Overview:

```
function allowance(address owner, address spender)
    public
    view
    override
    returns (uint256)
  {
    return _allowances[owner][spender];
  }
function _approve(
    address owner,
    address spender,
    uint256 amount
  ) private {
    require(owner != address(0), "ERC20: approve from the zero address");
    require(spender != address(0), "ERC20: approve to the zero address");

    _allowances[owner][spender] = amount;
    emit Approval(owner, spender, amount);
  }
```

Suggestion:

Rename the local variables that shadow another component.

MANUAL TESTING

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.

```
event MinTokensBeforeSwapUpdated(uint256
minTokensBeforeSwap);
```

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