expelee

Building the Futuristic Blockchain Ecosystem

SECURITY AUDIT REPORT

Baaz Coin

TOKEN OVERVIEW

Risk Findings

Severity	Found
High	3
Medium	1
Low	3
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 ?	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	**High Risk**
Audit Date	**06 December 2024**

CONTRACT DETAILS

Token Address: 0xeD6797a4f0F490AeD75264F0bA3Df95B42DB4704

Name: Baaz Coin

Symbol: BAAZ

Decimals: 18

Network: Bsc Scan

Token Type: BEP-20

Owner: 0x7bD4923caa55d6F28Decf82ea8D95610F16255cC

Deployer: 0x7bD4923caa55d6F28Decf82ea8D95610F16255cC

Token Supply: 50,000,000

Checksum: 86c3b879dc5db8d306e826729ea33a15

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

- Manual Review: The code has undergone a line-by-line review by the Ace team.
- 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.
- Slither: The code has undergone static analysis using Slither.

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

STATIC ANALYSIS

```
INFO:Detectors:
Address.isContract(address) (BaazCoin.sol#306-316) uses assembly
        - INLINE ASM (BaazCoin.sol#312-314)
Address.verifyCallResult(bool,bytes,string) (BaazCoin.sol#475-495) uses assembly
        - INLINE ASM (BaazCoin.sol#487-490)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
4 different versions of Solidity are used:
        - Version constraint >=0.6.2 is used by:
                ->=0.6.2 (BaazCoin.sol#17)
                ->=0.6.2 (BaazCoin.sol#115)
        - Version constraint >=0.5.0 is used by:
                ->=0.5.0 (BaazCoin.sol#161)
        - Version constraint ^0.8.0 is used by:
                -^0.8.0 (BaazCoin.sol#183)
                -^0.8.0 (BaazCoin.sol#209)
                -^0.8.0 (BaazCoin.sol#283)
                -^0.8.0 (BaazCoin.sol#502)
                -^0.8.0 (BaazCoin.sol#731)
        - Version constraint ^0.8.10 is used by:
                -^0.8.10 (BaazCoin.sol#812)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used
INFO:Detectors:
BaazCoin.includeInReward(address) (BaazCoin.sol#1110-1121) has costly operations inside a loop:
        - _excluded.pop() (BaazCoin.sol#1117)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-operations-inside-a-loop
INFO:Detectors:
Context._msgData() (BaazCoin.sol#200-202) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Version constraint >=0.6.2 contains known severe issues (https://solidity.readthedocs.io/en/latest/bugs.html)
        - MissingSideEffectsOnSelectorAccess
        - AbiReencodingHeadOverflowWithStaticArrayCleanup
        - DirtyBytesArrayToStorage
        - NestedCalldataArrayAbiReencodingSizeValidation
        - ABIDecodeTwoDimensionalArrayMemory
        - KeccakCaching
        - EmptyByteArrayCopy
        - DynamicArrayCleanup
        - MissingEscapingInFormatting
```

```
INFO:Detectors:
BaazCoin.addLiquidity(uint256,uint256) (BaazCoin.sol#1566-1579) ignores return value by uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),token
nAmount,0,0,owner(),block.timestamp) (BaazCoin.sol#1571-1578)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
INFO:Detectors:
BaazCoin.allowance(address,address).owner (BaazCoin.sol#984) shadows:
        - Ownable.owner() (BaazCoin.sol#239-241) (function)
BaazCoin._approve(address,address,uint256).owner (BaazCoin.sol#1410) shadows:
        - Ownable.owner() (BaazCoin.sol#239-241) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
BaazCoin.controlBuyBack(bool,uint256) (BaazCoin.sol#1142-1145) should emit an event for:
        - buybackLimit = amount (BaazCoin.sol#1144)
BaazCoin.setNumTokensSellToAddToLiquidity(uint256) (BaazCoin.sol#1161-1166) should emit an event for:
        - numTokensSellToAddToLiquidity = numTokens * 10 ** _decimals (BaazCoin.sol#1165)
BaazCoin.setMaxBuyAmount(uint256) (BaazCoin.sol#1179-1181) should emit an event for:
        - maxBuyAmount = value * 10 ** _decimals (BaazCoin.sol#1180)
BaazCoin.setMaxSellAmount(uint256) (BaazCoin.sol#1183-1185) should emit an event for:
        - maxSellAmount = value * 10 ** _decimals (BaazCoin.sol#1184)
BaazCoin.setMaxWallet(uint256) (BaazCoin.sol#1187-1189) should emit an event for:
        - maxWalletAmount = value * 10 ** _decimals (BaazCoin.sol#1188)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
BaazCoin.constructor(uint16,uint16,uint16,uint16,uint16,address,uint256,uint256,uint256,uint256,uint256)._marketing (BaazCoin.sol#899) lacks a zero-check on
 :
                - _marketingWallet = _marketing (BaazCoin.sol#945)
BaazCoin.updateRouter(address)._uniswapV2Pair (BaazCoin.sol#1170-1175) lacks a zero-check on :
                - uniswapV2Pair = _uniswapV2Pair (BaazCoin.sol#1176)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
```

```
INFO:Detectors:
Loop condition i < _excluded.length (BaazCoin.sol#1319) should use cached array length instead of referencing `length` member of the storage array.
 Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cache-array-length
INFO:Detectors:
BaazCoin._decimals (BaazCoin.sol#836) should be constant
BaazCoin._name (BaazCoin.sol#834) should be constant
BaazCoin._symbol (BaazCoin.sol#835) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
BaazCoin._marketingWallet (BaazCoin.sol#867) should be immutable
BaazCoin._tTotal (BaazCoin.sol#830) should be immutable
BaazCoin.deadBlocks (BaazCoin.sol#848) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
INFO:Slither:BaazCoin.sol analyzed (9 contracts with 94 detectors), 59 result(s) found
```

POINTS TO NOTE

- The owner can set buy/sell fee not more than 25%.
- The owner can exclude/include from/In rewards.
- The owner can launch.
- The owner can control the Buyback.
- The owner can blacklist.
- The owner can set max wallet.
- The owner can set the number of tokens sell to add liquidity.
- The owner can set swap and liquify enable.
- The owner can set Max wallet.
- The owner can set the Max Buy/Sell amount.
- The owner can claim stuck tokens.
- The owner can update the router.
- The owner can trigger buyback.

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			

HIGH RISK FINDING

Centralization – Enabling Trades

Severity: High

Function: Launch

Status: Fixed

Overview:

The Launch function permits only the contract owner to activate trading capabilities. Until this function is executed, no investors can buy, sell, or transfer their tokens. This places a high degree of control and centralization in the hands of the contract owner.

```
function launch() external onlyOwner {
require(antiBotBlocks == 0);
      antiBotBlocks = block.number + deadBlocks;
isTradingEnabled = true;
  }
```

Suggestion:

To reduce centralization and potential manipulation, consider one of the following approaches:

- Automatically enable trading after a specified condition, such as the completion of a presale, is met.
- If manual activation is still desired, consider transferring the ownership of the contract to a trustworthy, third-party entity like a certified "PinkSale Safu" developer. This can provide investors with more confidence in the eventual activation of trading capabilities, mitigating concerns of potential bad-faith actions by the original owner.

HIGH RISK FINDING

Centralization – Owner can blacklist wallets.

Severity: High

Function: setBlackList

Status: Open

Overview:

The owner can blacklist wallets from transferring tokens for an indefinite period which is not recommended. Which can lock the user's token.

```
function setBlackList(address addr, bool value) external onlyOwner {
    _isBlackListed[addr] = value;
```

Suggestion:

There should be a locking period so that the wallet cannot be locked for an indefinite period.

HIGH RISK FINDING

Centralization – Buy and Sell Fees.
Severity: High
Function: setBuy and setSell.
Status: FIXED

Overview:
The owner can set the buy and sell fees up to 100%, which is not recommended.

```
 function setBuyFee(uint16 liq, uint16 market, uint16 reward, uint16 tax) external
onlyOwner {
buyFee.liquidityFee = liq;
buyFee.marketingFee = market;
buyFee.buybackFee = reward;
      buyFee.taxFee = tax;
  }

   function setSellFee(uint16 liq, uint16 market, uint16 reward, uint16 tax)
external onlyOwner {
sellFee.liquidityFee = liq;
sellFee.marketingFee = market;
sellFee.buybackFee = reward;
sellFee.taxFee = tax;
  }
```

Suggestion:
It is recommended that no fees in the contract should be more than 25% of the contract

MEDIUM RISK FINDING

Centralization – Liquidity is added to EOA.
Severity: Medium
function: addLiquidity
Status: FIXED

Overview:
Liquidity is added to EOA. The owner may *drain it.*

```
 function addLiquidity(uint256 tokenAmount, uint256 ethAmount)
private lockTheSwap {
// 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
owner(),
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 excludeFromReward(address account) public onlyOwner {
    // require(account !=
0x7a250d5630B4cF539739dF2C5dAcb4c659F2488D, 'We can not exclude
Uniswap router.');
    require(!_isExcluded[account], "Account is already excluded");
    if (_rOwned[account] > 0) {
        _tOwned[account] = tokenFromReflection(_rOwned[account]);
    }
    _isExcluded[account] = true;
    _excluded.push(account);
}

function includeInReward(address account) external onlyOwner {
    require(_isExcluded[account], "Account is already excluded");
    for (uint256 i = 0; i < _excluded.length; i++) {
        if (_excluded[i] == account) {
            _excluded[i] = _excluded[_excluded.length - 1];
            _tOwned[account] = 0;
            _isExcluded[account] = false;
            _excluded.pop();
            break;
        }
    }
}
```

LOW RISK FINDING

```
function launch() external onlyOwner {
        require(antiBotBlocks == 0);
        antiBotBlocks = block.number + deadBlocks;
        isTradingEnabled = true;
    }

    function setBuyFee(uint16 liq, uint16 market, uint16 reward, uint16 tax) external
onlyOwner {
        buyFee.liquidityFee = liq;
        buyFee.marketingFee = market;
        buyFee.buybackFee = reward;
        buyFee.taxFee = tax;
    }

    function setSellFee(uint16 liq, uint16 market, uint16 reward, uint16 tax) external
onlyOwner {
        sellFee.liquidityFee = liq;
        sellFee.marketingFee = market;
        sellFee.buybackFee = reward;
        sellFee.taxFee = tax;
    }

    function setMaxBuyAmount(uint256 value) external onlyOwner {
        maxBuyAmount = value * 10 ** _decimals;
    }

    function setMaxSellAmount(uint256 value) external onlyOwner {
        maxSellAmount = value * 10 ** _decimals;
    }

    function setMaxWallet(uint256 value) external onlyOwner {
        maxWalletAmount = value * 10 ** _decimals;
    }
```

LOW RISK FINDING

```
function setBlackList(address addr, bool value) external onlyOwner {
    _isBlackListed[addr] = value;


function setNumTokensSellToAddToLiquidity(uint256 numTokens)
    external
    onlyOwner
{
    numTokensSellToAddToLiquidity = numTokens * 10 ** _decimals;
}
```

Suggestion: Emit an event for critical changes.

LOW RISK FINDING

Centralization – Local variable Shadowing

Severity: Low

Subject: Variable Shadowing

Status: Open

Overview:

```
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);
    }

function allowance(address owner, address spender)
public
view
override
returns (uint256)
    {
return _allowances[owner][spender];
    }
```

Suggestion:

Rename the local variables that shadow another component.

LOW RISK FINDING

Centralization – Remove the safe math library.

Severity: Low

Status: Open

Line Number: 514-725

Overview:

The Safe Math library is no longer needed for Solidity version 0.8 and above. This is because Solidity 0.8 includes checked arithmetic operations by default. All of Safe Math's methods are now inherited into Solidity programming.

OPTIMIZATION

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 sendValue(address payable recipient, uint256 amount) internal {
require(address(this).balance >= amount, "Address: insufficient balance");

        (bool success, ) = recipient.call{value: amount}("");
require(success, "Address: unable to send value, recipient may have
reverted");
    }

 function functionCall(address target, bytes memory data) internal returns
(bytes memory) {
        return functionCall(target, data, "Address: low-level call failed");
    }

 function functionCallWithValue(
        address target,
        bytes memory data,
        uint256 value
    )
```

OPTIMIZATION

```
internal returns (bytes memory) {
        return functionCallWithValue(target, data, value, "Address: low-level call
with value failed");
    }

 function functionStaticCall(address target, bytes memory data) internal
view returns (bytes memory) {
        return functionStaticCall(target, data, "Address: low-level static call
failed");
    }

 function functionDelegateCall(address target, bytes memory data) internal
returns (bytes memory) {
        return functionDelegateCall(target, data, "Address: low-level delegate
call failed");
    }

 function tryAdd(uint256 a, uint256 b) internal pure returns (bool, uint256) {
        unchecked {
            uint256 c = a + b;
            if (c < a) return (false, 0);
            return (true, c);
        }
    }

 function trySub(uint256 a, uint256 b) internal pure returns (bool, uint256) {
        unchecked {
            if (b > a) return (false, 0);
            return (true, a - b);
        }
    }
 function tryMul(uint256 a, uint256 b) internal pure returns (bool, uint256) {
        unchecked {
```

OPTIMIZATION

```
        // Gas optimization: this is cheaper than requiring 'a' not being zero, but the
        // benefit is lost if 'b' is also tested.
        // See: https://github.com/OpenZeppelin/openzeppelin-contracts/pull/522
        if (a == 0) return (true, 0);
        uint256 c = a * b;
        if (c / a != b) return (false, 0);
        return (true, c);
    }
}

function tryDiv(uint256 a, uint256 b) internal pure returns (bool, uint256) {
    unchecked {
        if (b == 0) return (false, 0);
        return (true, a / b);
    }
}

function tryMod(uint256 a, uint256 b) internal pure returns (bool, uint256) {
    unchecked {
        if (b == 0) return (false, 0);
        return (true, a % b);
    }
}
function mod(uint256 a, uint256 b) internal pure returns (uint256) {
    return a % b;
}
function div(
    uint256 a,
    uint256 b,
    string memory errorMessage
) internal pure returns (uint256) {
    unchecked {
        require(b > 0, errorMessage);
        return a / b;
    }
}
```

OPTIMIZATION

```
function mod(
    uint256 a,
    uint256 b,
    string memory errorMessage
  ) internal pure returns (uint256) {
    unchecked {
      require(b > 0, errorMessage);
      return a % b;
    }
  }
}
```

OPTIMIZATION

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.

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

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.