



# Smart Contract Audit

FOR

## BumbleBee

DATED : 25 June 23'



# High Risk Finding

## Centralization – Trades must be enabled

Severity: **High**

**function:** EnableTrading

**Status:** Not Resolved

### Overview:

The smart contract owner must enable trades for holders. If trading remain disabled, no one would be able to buy/sell/transfer tokens.

```
function EnableTrading() external onlyOwner {  
    require(!tradingEnabled, "Cannot re-enable trading");  
    tradingEnabled = true;  
}
```

### Suggestion

To mitigate this centralization issue, we propose the following options:

1. Renounce Ownership: Consider relinquishing control of the smart contract by renouncing ownership. This would remove the ability for a single entity to manipulate the router, reducing centralization risks.
2. Multi-signature Wallet: Transfer ownership to a multi-signature wallet. This would require multiple approvals for any changes to the mainRouter, adding an additional layer of security and reducing the centralization risk.
3. Transfer ownership to a trusted and valid 3<sup>rd</sup> party in order to guarantee enabling of the trades



# AUDIT SUMMARY

**Project name - BumbleBee**

**Date:** 25 June, 2023

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

**Audit Status: Passed With High Risk**

## Issues Found

| Status       | Critical | High | Medium | Low | Suggestion |
|--------------|----------|------|--------|-----|------------|
| Open         | 0        | 1    | 0      | 0   | 0          |
| Acknowledged | 0        | 0    | 0      | 0   | 0          |
| Resolved     | 0        | 0    | 0      | 0   | 0          |



# USED TOOLS

---

## Tools:

### 1- Manual Review:

A line by line code review has been performed by audit ace team.

**2- BSC Test Network:** All tests were 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/token/0x92204218F6BF0c42dC167572521E73b0bC2d9e9F>

---



# Token Information

---

**Token Name :** BumbleBee

**Token Symbol:** \$BBEE

**Decimals:** 9

**Token Supply:** 420,000,000,000

**Token Address:** 0xC8104F39d24D34b1262e54b167D5Debda7A47789

**Checksum:**

6bcc7c30b8c14cbb5151675488fa6710283eee8e

**Owner:** 0x1a8f1ec59FD2A14dD0f1E32b3DCAB8De81daE94E

**Deployer:** 0xf9e77f77b080AA4C819f3a018D821F279264090a

---



# TOKEN OVERVIEW

---

## Fees:

Buy Fees: 7%

Sell Fees: 7%

Transfer Fees: 7%

---

**Fees Privilege:** Owner

---

**Ownership:** owned

---

**Minting:** No mint function

---

**Max Tx Amount/ Max Wallet Amount:** No

---

**Blacklist:** No

---

**Other Privileges:** including in fees

- excluding from fees
  - initial distribution of the tokens
  - enabling trades
-



# 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 is exercised 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



Return values of low-level calls



**Gasless Send**



Private modifier



Using block.timestamp



Multiple Sends



Re-entrancy



Using Suicide



Tautology or contradiction



Gas Limit and Loops



Timestamp Dependence



Address hardcoded



Revert/require functions



Exception Disorder



Use of tx.origin



Using inline assembly



Integer overflow/underflow



Divide before multiply



Dangerous strict equalities



Missing Zero Address Validation



Using SHA3



Compiler version not fixed



Using throw



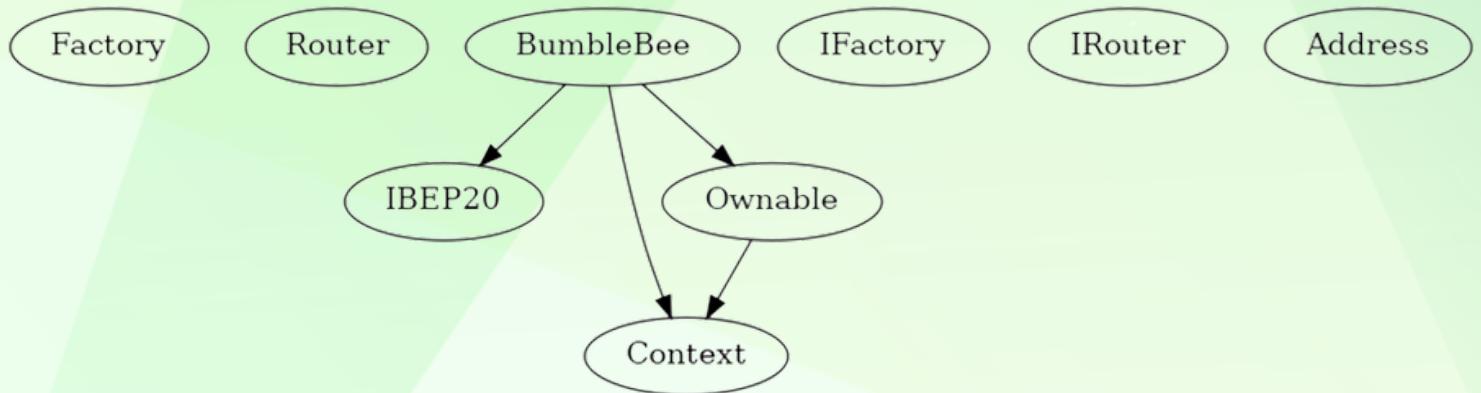
# CLASSIFICATION OF RISK

| Severity                        | Description  |
|---------------------------------|--|
| ◆ Critical                      | These vulnerabilities could be exploited easily and can lead to asset loss, data loss, asset, or data manipulation. They should be fixed right away. |
| ◆ High-Risk                     | A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.          |
| ◆ Medium-Risk                   | A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.  |
| ◆ Low-Risk                      | A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.                |
| ◆ Gas Optimization / Suggestion | A vulnerability that has an informational character but is not affecting any of the code.  |

## Findings

| Severity                         | Found |
|----------------------------------|-------|
| ◆ Critical                       | 0     |
| ◆ High-Risk                      | 1     |
| ◆ Medium-Risk                    | 0     |
| ◆ Low-Risk                       | 0     |
| ◆ Gas Optimization / Suggestions | 0     |

# INHERITANCE TREE





# CONTRACT ASSESSMENT

| Contract   | Type | Bases |  |  |  |
|--|------|-------|--|--|--|
| L   **Function Name**   **Visibility**   **Mutability**   **Modifiers**      |      |       |  |  |  |
| **Factory**   Interface  |      |       |  |  |  |
| L   createPair   External !   ● NO !   |      |       |  |  |  |
| **Router**   Interface   |      |       |  |  |  |
| L   WETH   External !   NO !   |      |       |  |  |  |
| L   factory   External !   NO !  |      |       |  |  |  |
| L   swapExactTokensForETHSupportingFeeOnTransferTokens   External !   ● NO ! |      |       |  |  |  |
| **IBEP20**   Interface   |      |       |  |  |  |
| L   totalSupply   External !   NO !  |      |       |  |  |  |
| L   balanceOf   External !   NO !  |      |       |  |  |  |
| L   transfer   External !   ● NO !   |      |       |  |  |  |
| L   allowance   External !   NO !  |      |       |  |  |  |
| L   approve   External !   ● NO !  |      |       |  |  |  |
| L   transferFrom   External !   ● NO !                                       |      |       |  |  |  |
| **Context**   Implementation   |      |       |  |  |  |
| L   _msgSender   Internal 🔒  |      |       |  |  |  |
| L   _msgData   Internal 🔒  |      |       |  |  |  |
| **Ownable**   Implementation   Context                                       |      |       |  |  |  |
| L   <Constructor>   Public !   ● NO !  |      |       |  |  |  |
| L   owner   Public !   NO !  |      |       |  |  |  |
| L   renounceOwnership   Public !   ● NO !                                    |      |       |  |  |  |
| L   transferOwnership   Public !   ● NO !                                    |      |       |  |  |  |
| L   _setOwner   Private 🔒   ● NO !   |      |       |  |  |  |
| **IFactory**   Interface   |      |       |  |  |  |
| L   createPair   External !   ● NO !   |      |       |  |  |  |
| **IRouter**   Interface  |      |       |  |  |  |
| L   factory   External !   NO !  |      |       |  |  |  |
| L   WETH   External !   NO !   |      |       |  |  |  |
| L   addLiquidityETH   External !   \$! NO !                                  |      |       |  |  |  |
| L   swapExactTokensForETHSupportingFeeOnTransferTokens   External !   ● NO ! |      |       |  |  |  |
| **Address**   Library  |      |       |  |  |  |
| L   sendValue   Internal 🔒   ● NO !  |      |       |  |  |  |
| **BumbleBee**   Implementation   Context, IBEP20, Ownable                    |      |       |  |  |  |
| L   <Constructor>   Public !   ● NO !  |      |       |  |  |  |
| L   name   Public !   NO !   |      |       |  |  |  |
| L   symbol   Public !   NO !   |      |       |  |  |  |
| L   decimals   Public !   NO !   |      |       |  |  |  |



# CONTRACT ASSESSMENT

|   |
|---|
| L   totalSupply   Public !   [NO !                  |
| L   balanceOf   Public !   [NO !                    |
| L   allowance   Public !   [NO !                    |
| L   approve   Public !   ●   [NO !                  |
| L   transferFrom   Public !   ●   [NO !             |
| L   increaseAllowance   Public !   ●   [NO !        |
| L   decreaseAllowance   Public !   ●   [NO !        |
| L   transfer   Public !   ●   [NO !                 |
| L   isExcludedFromReward   Public !   [NO !         |
| L   reflectionFromToken   Public !   [NO !          |
| L   EnableTrading   External !   ●   onlyOwner      |
| L   tokenFromReflection   Public !   [NO !          |
| L   excludeFromReward   Public !   ●   onlyOwner    |
| L   includeInReward   External !   ●   onlyOwner    |
| L   excludeFromFee   Public !   ●   onlyOwner       |
| L   includeInFee   Public !   ●   onlyOwner         |
| L   isExcludedFromFee   Public !   [NO !            |
| L   _reflectRfi   Private 🔒   ●                     |
| L   _takeMarketing   Private 🔒   ●                  |
| L   _getValues   Private 🔒                          |
| L   _getTValues   Private 🔒                         |
| L   _getRValues1   Private 🔒                        |
| L   _getRate   Private 🔒                            |
| L   _getCurrentSupply   Private 🔒                   |
| L   _approve   Private 🔒   ●                        |
| L   _transfer   Private 🔒   ●                       |
| L   _tokenTransfer   Private 🔒   ●                  |
| L   InternalSwap   Internal 🔒   ●   LockSwap        |
| L   bulkExcludeFee   External !   ●   onlyOwner     |
| L   rescueBNB   External !   ●   onlyOwner          |
| L   rescueAnyBEP20Tokens   Public !   ●   onlyOwner |
| L   <Receive Ether>   External !   💸   [NO !        |

### Legend

| Symbol | Meaning                   |
|--------|---------------------------|
| ●      | Function can modify state |
| 💸      | Function is payable       |



## POINTS TO NOTE

---

- Owner is not able to change buy/sell/transfer fees (7% each)
- Owner is not able to blacklist an arbitrary address.
- Owner is not able to disable trades
- Owner is not able to set max buy/sell/transfer/hold amount to 0
- Owner is not able to mint new tokens
- **Owner must enable trades manually**



# STATIC ANALYSIS

```
BumbleBee.includeInReward(address) (contracts/Token.sol#295-306) has costly operations inside a loop:  
- excluded.pop() (contracts/Token.sol#302)  
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-operations-inside-a-loop  
  
Address.sendValue(address,uint256) (contracts/Token.sol#114-118) is never used and should be removed  
Context._msgData() (contracts/Token.sol#46-49) is never used and should be removed  
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code  
  
BumbleBee._rTotal (contracts/Token.sol#138) is set pre-construction with a non-constant function or state variable:  
- (MAX - (MAX % _tTotal))  
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#function-initializing-state  
  
Pragma version^0.8.17 (contracts/Token.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16  
solc-0.8.20 is not recommended for deployment  
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity  
  
Low level call in Address.sendValue(address,uint256) (contracts/Token.sol#114-118):  
- (success) = recipient.call{value: amount}() (contracts/Token.sol#116)  
Low level call in BumbleBee.InternalSwap() (contracts/Token.sol#449-463):  
- (success) = address(marketingWallet).call{value: address(this).balance}() (contracts/Token.sol#462)  
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls  
  
Function Router.WETH() (contracts/Token.sol#10) is not in mixedCase  
Function IRouter.WETH() (contracts/Token.sol#93) is not in mixedCase  
Struct BumbleBee.valuesFromGetValues (contracts/Token.sol#161-169) is not in CapWords  
Function BumbleBee.EnableTrading() (contracts/Token.sol#275-278) is not in mixedCase  
Function BumbleBee.InternalSwap() (contracts/Token.sol#449-463) is not in mixedCase  
Parameter BumbleBee.rescueAnyBEP20Tokens(address,address,uint256)._tokenAddr (contracts/Token.sol#475) is not in mixedCase  
Parameter BumbleBee.rescueAnyBEP20Tokens(address,address,uint256)._to (contracts/Token.sol#475) is not in mixedCase  
Parameter BumbleBee.rescueAnyBEP20Tokens(address,address,uint256)._amount (contracts/Token.sol#475) is not in mixedCase  
Constant BumbleBee._decimals (contracts/Token.sol#134) is not in UPPER_CASE_WITH_UNDERSCORES  
Constant BumbleBee._name (contracts/Token.sol#142) is not in UPPER_CASE_WITH_UNDERSCORES  
Constant BumbleBee._symbol (contracts/Token.sol#143) is not in UPPER_CASE_WITH_UNDERSCORES  
Modifier BumbleBee.LockSwap() (contracts/Token.sol#177-181) is not in mixedCase  
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions  
  
Redundant expression "this (contracts/Token.sol#47)" inContext (contracts/Token.sol#41-50)  
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements  
  
BumbleBee._getTValues(uint256,bool,address,address) (contracts/Token.sol#347-368) uses literals with too many digits:  
- s.tRfi = (tAmount * temp.rfi) / 100000 (contracts/Token.sol#364)  
BumbleBee._getTValues(uint256,bool,address,address) (contracts/Token.sol#347-368) uses literals with too many digits:  
- s.tMarketing = (tAmount * temp.marketing) / 100000 (contracts/Token.sol#365)  
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits  
  
BumbleBee._tTotal (contracts/Token.sol#137) should be constant  
BumbleBee.marketingWallet (contracts/Token.sol#140) should be constant  
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant  
  
BumbleBee.pair (contracts/Token.sol#183) should be immutable  
BumbleBee.swapRouter (contracts/Token.sol#184) should be immutable  
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
```

**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- Adding liquidity (**passed**):

<https://testnet.bscscan.com/tx/0xea5006f2295951eb333d17c500379d7934c278f60d0ac8a5db88513a5ec71db4>

## 2- Buying when excluded (0% tax) (**passed**):

<https://testnet.bscscan.com/tx/0x21c063d9ff2ef5c52bbc1d51521759ac69aca98e37bb630721010c05a99cd6ec>

## 3- Selling when excluded (0% tax) (**passed**):

<https://testnet.bscscan.com/tx/0xba81fbf2096306ad48d87b3469e7cdaf4e7fa768e52d29ef28bb256c1edc0471>

## 4- Transferring when excluded from fees (0% tax) (**passed**):

<https://testnet.bscscan.com/tx/0x0346597150e02c60ce8aa1f8c0d7261eefcf1a308edf0aba4a5b553972e4f77e>

## 5- Buying when not excluded from fees (7% tax) (**passed**):

<https://testnet.bscscan.com/tx/0x8307f5c99ebe667fee549e433130f66613d3f26f829d7b633d13c00f95dff4bd>

## 6- Selling when not excluded from fees (7% tax) (**passed**):

<https://testnet.bscscan.com/tx/0x2221b3a46e3c62bfbb710f399d6531b4b864d788749fc703d377047d98777c69>



# FUNCTIONAL TESTING

---

**7- Transferring when not excluded from fees (7% tax) (passed):**

<https://testnet.bscscan.com/tx/0x06efd915a2efd2f28ce887f98f0305ebe178c01b27552793ec78b7f3afbe2b1f>

**8- Internal swap (marketing bnb) (passed):**

<https://testnet.bscscan.com/address/0x485d6b3140d8f93289625cce1080f9f6f2e6eb36#internaltx>

# High Risk Finding

## Centralization – Trades must be enabled

Severity: **High**

**function:** EnableTrading

**Status:** Not Resolved

### Overview:

The smart contract owner must enable trades for holders. If trading remain disabled, no one would be able to buy/sell/transfer tokens.

```
function EnableTrading() external onlyOwner {  
    require(!tradingEnabled, "Cannot re-enable trading");  
    tradingEnabled = true;  
}
```

### Suggestion

To mitigate this centralization issue, we propose the following options:

1. Renounce Ownership: Consider relinquishing control of the smart contract by renouncing ownership. This would remove the ability for a single entity to manipulate the router, reducing centralization risks.
2. Multi-signature Wallet: Transfer ownership to a multi-signature wallet. This would require multiple approvals for any changes to the mainRouter, adding an additional layer of security and reducing the centralization risk.
3. Transfer ownership to a trusted and valid 3<sup>rd</sup> party in order to guarantee enabling of the trades



# DISCLAIMER

---

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment. Team provides no guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document. Always Do your own research and protect yourselves from being scammed. The Auditace team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools. Under no circumstances did Auditace receive a payment to manipulate those results or change the awarding badge that we will be adding in our website. Always Do your own research and protect yourselves from scams. This document should not be presented as a reason to buy or not buy any particular token. The Auditace team disclaims any liability for the resulting losses.



# ABOUT AUDITACE

---

We specialize 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://t.me/Audit_Ace)**



**[https://twitter.com/auditace\\_](https://twitter.com/auditace_)**



**<https://github.com/Audit-Ace>**

---