# BNBEER

## Smart Contract Review

**Deliverable: Smart Contract Audit Report**

**Security Report**

**October 2021**

# Disclaimer

The information and views set out in this publication are those of the author(s) and do not necessarily reflect the official opinion of the Company. The content, conclusions and recommendations set out in this publication are elaborated in the specific for only project.

eNebula Solutions does not guarantee the authenticity of the project or organization or team of members that is connected/owner behind the project or nor accuracy of the data included in this study. All representations, warranties, undertakings and guarantees relating to the report are excluded, particularly concerning – but not limited to – the qualities of the assessed projects and products. Neither the Company nor any personating on the Company's behalf may be held responsible for the use that may be made of the information contained herein.

eNebula Solutions retains the right to display audit reports and other content elements as examples of their work in their portfolio and as content features in other projects with protecting all security purpose of customer. The report containing confidential information can be used internally by the Customer, or it can be disclosed publicly after all vulnerabilities fixed - upon a decision of the Customer.

# Report Summary

| | | | |
|---|---|---|---|
| **Title** | **BNBEER Smart Contract Audit** | | |
| **Project Owner** | BNBEER | | |
| | | | |
| **Type** | Public | | |
| **Reviewed by** | Vatsal Raychura | **Revision date** | 07/10/2021 |
| **Approved by** | eNebula Solutions Private Limited | **Approval date** | 07/10/2021 |
| | | **Nº Pages** | **38** |

# Overview

## Background

BNBeer's team requested that eNebula Solutions perform an Extensive Smart Contract audit of their Smart Contract.

## Project Dates

The following is the project schedule for this review and report:

- **October 07**: Smart Contract Review Completed *(Completed)*
- **October 07**: Delivery of Smart Contract Audit Report *(Completed)*

## Review Team

The following eNebula Solutions team member participated in this review:

- Sejal Barad, Security Researcher and Engineer
- Vatsal Raychura, Security Researcher and Engineer

# Coverage

## Target Specification and Revision

For this audit, we performed research, investigation, and review of the smart contract of BNBEER.

The following documentation repositories were considered in-scope for the review:

- BNBEER Project: https://bscscan.com/address/0x7fd659e61815a4b6e65bd373183b5146580d17cf#code

# Introduction

Given the opportunity to review BNBEER Project's smart contract source code, we in the report outline our systematic approach to evaluate potential security issues in the smart contract implementation, expose possible semantic inconsistencies between smart contract code and design document, and provide additional suggestions or recommendations for improvement. Our results show that the given version of smart contracts is ready to launch as, there are no critical or high or any security issues found related to business logic, security or performance.

About BNBEER: -

| Item | Description |
|---|---|
| **Issuer** | BNBeer |
| **Website** | https://www.bnbeer.io |
| **Type** | BEP20 |
| **Platform** | Solidity |
| **Audit Method** | Whitebox |
| **Latest Audit Report** | October 07, 2021 |

The Test Method Information: -

| Test method | Description |
|---|---|
| **Black box testing** | Conduct security tests from an attacker's perspective externally. |
| **Grey box testing** | Conduct security testing on code modules through the scripting tool, observing the internal running status, mining weaknesses. |
| **White box testing** | Based on the open-source code, non-open-source code, to detect whether there are vulnerabilities in programs such as nodes, SDK, etc. |

The vulnerability severity level information:

| Level | Description |
|---|---|
| **Critical** | Critical severity vulnerabilities will have a significant effect on the security of the DeFi project, and it is strongly recommended to fix the critical vulnerabilities. |
| **High** | High severity vulnerabilities will affect the normal operation of the DeFi project. It is strongly recommended to fix high-risk vulnerabilities. |
| **Medium** | Medium severity vulnerability will affect the operation of the DeFi project. It is recommended to fix medium-risk vulnerabilities. |
| **Low** | Low severity vulnerabilities may affect the operation of the DeFi project in certain scenarios. It is suggested that the project party should evaluate and consider whether these vulnerabilities need to be fixed. |
| **Weakness** | There are safety risks theoretically, but it is extremely difficult to reproduce in engineering. |

The Full List of Check Items:

| Category | Check Item |
|---|---|
| **Basic Coding Bugs** | Constructor Mismatch |
| | Ownership Takeover |
| | Redundant Fallback Function |
| | Overflows & Underflows |
| | Reentrancy |
| | MONEY-Giving Bug |
| | Blackhole |
| | Unauthorized Self-Destruct |
| | Revert DoS |
| | Unchecked External Call |
| | Gasless Send |
| | Send Instead of Transfer |
| | Costly Loop |
| | (Unsafe) Use of Untrusted Libraries |
| | (Unsafe) Use of Predictable Variables |
| | Transaction Ordering Dependence |
| | Deprecated Uses |
| **Semantic Consistency Checks** | Semantic Consistency Checks |
| | Business Logics Review |

| | |
|---|---|
| **Advanced DeFi Scrutiny** | Functionality Checks |
| | Authentication Management |
| | Access Control & Authorization |
| | Oracle Security |
| | Digital Asset Escrow |
| | Kill-Switch Mechanism |
| | Operation Trails & Event Generation |
| | ERC20 Idiosyncrasies Handling |
| | Frontend-Contract Integration |
| | Deployment Consistency |
| | Holistic Risk Management |
| **Additional Recommendations** | Avoiding Use of Variadic Byte Array |
| | Using Fixed Compiler Version |
| | Making Visibility Level Explicit |
| | Making Type Inference Explicit |
| | Adhering To Function Declaration Strictly |
| | Following Other Best Practices |

Common Weakness Enumeration (CWE) Classifications Used in This Audit:

| Category | Summary |
|---|---|
| **Configuration** | Weaknesses in this category are typically introduced during the configuration of the software. |
| **Data Processing Issues** | Weaknesses in this category are typically found in functionality that processes data. |
| **Numeric Errors** | Weaknesses in this category are related to improper calculation or conversion of numbers. |
| **Security Features** | Weaknesses in this category are concerned with topics like authentication, access control, confidentiality, cryptography, and privilege management. (Software security is not security software.) |
| **Time and State** | Weaknesses in this category are related to the improper management of time and state in an environment that supports simultaneous or near-simultaneous computation by multiple systems, processes, or threads. |
| **Error Conditions, Return Values, Status Codes** | Weaknesses in this category include weaknesses that occur if a function does not generate the correct return/status code, or if the application does not handle all possible return/status codes that could be generated by a function. |
| **Resource Management** | Weaknesses in this category are related to improper management of system resources. |

| | |
|---|---|
| **Behavioral Issues** | Weaknesses in this category are related to unexpected behaviors from code that an application uses. |
| **Business Logics** | Weaknesses in this category identify some of the underlying problems that commonly allow attackers to manipulate the business logic of an application. Errors in business logic can be devastating to an entire application. |
| **Initialization and Cleanup** | Weaknesses in this category occur in behaviors that are used for initialization and breakdown. |
| **Arguments and Parameters** | Weaknesses in this category are related to improper use arguments or parameters within function calls. |
| **Expression Issues** | Weaknesses in this category are related to incorrectly written expressions within code. |
| **Coding Practices** | Weaknesses in this category are related to coding practices that are deemed unsafe and increase the chances that an ex pilotable vulnerability will be present in the application. They may not directly introduce a vulnerability, but indicate the product has not been carefully developed or maintained. |

# Findings

## Summary

Here is a summary of our findings after analyzing the BNBEER's Smart Contract. During the first phase of our audit, we studied the smart contract sourcecode and ran our in-house static code analyzer through the Specific tools. The purpose here is to statically identify known coding bugs, and then manually verify (reject or confirm) issues reported by tools. We further manually review businesslogics, examine system operations, and place DeFi-related aspects under scrutinyto uncover possible pitfalls and/or bugs.

| Severity | No. of Issues |
|---|---|
| Critical | 0 |
| High | 0 |
| Medium | 0 |
| Low | 3 |
| Total | 3 |

We have so far identified that there are potential issues with severity of **0 Critical, 0 High, 0 Medium, and 3 Low**. Overall, these smart contracts are well- designed and engineered.

## Functional Overview

| | |
|---|---|
| ($) = payable function<br># = non-constant function | [Pub] public<br>[Ext] external<br>[Prv] private<br>[Int] internal |

+ Context
  - [Int] _msgSender
  - [Int] _msgData

+ [Int] IERC20
  - [Ext] totalSupply
  - [Ext] balanceOf
  - [Ext] transfer #
  - [Ext] allowance
  - [Ext] approve #
  - [Ext] transferFrom #

+ [Lib] SafeMath
  - [Int] add
  - [Int] sub
  - [Int] sub
  - [Int] mul
  - [Int] div
  - [Int] div
  - [Int] mod
  - [Int] mod

+ [Lib] Address
  - [Int] isContract
  - [Int] sendValue #
  - [Int] functionCall #
  - [Int] functionCall #
  - [Int] functionCallWithValue #
  - [Int] functionCallWithValue #
  - [Prv] _functionCallWithValue #

+ Ownable (Context)
  - [Pub] <Constructor> #
  - [Pub] owner
  - [Pub] renounceOwnership #
    - modifiers: onlyOwner
  - [Pub] transferOwnership #
    - modifiers: onlyOwner
  - [Pub] getUnlockTime
  - [Pub] getTime
  - [Pub] lock #
    - modifiers: onlyOwner
  - [Pub] unlock #

+ [Int] IUniswapV2Factory
  - [Ext] feeTo
  - [Ext] feeToSetter
  - [Ext] getPair
  - [Ext] allPairs
  - [Ext] allPairsLength
  - [Ext] createPair #
  - [Ext] setFeeTo #
  - [Ext] setFeeToSetter #

+ [Int] IUniswapV2Pair
  - [Ext] name
  - [Ext] symbol
  - [Ext] decimals
  - [Ext] totalSupply
  - [Ext] balanceOf
  - [Ext] allowance
  - [Ext] approve #
  - [Ext] transfer #
  - [Ext] transferFrom #
  - [Ext] DOMAIN_SEPARATOR
  - [Ext] PERMIT_TYPEHASH
  - [Ext] nonces
  - [Ext] permit #
  - [Ext] MINIMUM_LIQUIDITY
  - [Ext] factory
  - [Ext] token0
  - [Ext] token1
  - [Ext] getReserves
  - [Ext] price0CumulativeLast
  - [Ext] price1CumulativeLast
  - [Ext] kLast
  - [Ext] burn #
  - [Ext] swap #
  - [Ext] skim #
  - [Ext] sync #
  - [Ext] initialize #

+ [Int] IUniswapV2Router01
  - [Ext] factory
  - [Ext] WETH

- [Ext] addLiquidity #
- [Ext] addLiquidityETH ($)
- [Ext] removeLiquidity #
- [Ext] removeLiquidityETH #
- [Ext] removeLiquidityWithPermit #
- [Ext] removeLiquidityETHWithPermit #
- [Ext] swapExactTokensForTokens #
- [Ext] swapTokensForExactTokens #
- [Ext] swapExactETHForTokens ($)
- [Ext] swapTokensForExactETH #
- [Ext] swapExactTokensForETH #
- [Ext] swapETHForExactTokens ($)
- [Ext] quote
- [Ext] getAmountOut
- [Ext] getAmountIn
- [Ext] getAmountsOut
- [Ext] getAmountsIn

+ [Int] IUniswapV2Router02 (IUniswapV2Router01)
- [Ext] removeLiquidityETHSupportingFeeOnTransferTokens #
- [Ext] removeLiquidityETHWithPermitSupportingFeeOnTransferTokens #
- [Ext] swapExactTokensForTokensSupportingFeeOnTransferTokens #
- [Ext] swapExactETHForTokensSupportingFeeOnTransferTokens ($)
- [Ext] swapExactTokensForETHSupportingFeeOnTransferTokens #

+ BNBeer (Context, IERC20, Ownable)
- [Pub] <Constructor> #
- [Pub] name
- [Pub] symbol
- [Pub] decimals
- [Pub] totalSupply

- [Pub] balanceOf
- [Pub] transfer #
- [Pub] allowance
- [Pub] approve #
- [Pub] transferFrom #
- [Pub] increaseAllowance #
- [Pub] decreaseAllowance #
- [Pub] isExcludedFromReward
- [Pub] totalFees
- [Pub] minimumTokensBeforeSwapAmount
- [Pub] buyBackSellLimitAmount
- [Pub] deliver #
- [Pub] reflectionFromToken
- [Pub] tokenFromReflection
- [Pub] excludeFromReward #
  - modifiers: onlyOwner
- [Ext] includeInReward #
  - modifiers: onlyOwner
- [Prv] _approve #
- [Prv] _transfer #
- [Prv] swapTokens #
  - modifiers: lockTheSwap
- [Prv] buyBackTokens #
  - modifiers: lockTheSwap
- [Prv] swapTokensForEth #
- [Prv] swapETHForTokens #
- [Prv] addLiquidity #
- [Prv] _tokenTransfer #
- [Prv] _transferStandard #
- [Prv] _transferToExcluded #
- [Prv] _transferFromExcluded #

- [Prv] _transferBothExcluded #
- [Prv] _reflectFee #
- [Prv] _getValues
- [Prv] _getTValues
- [Prv] _getRValues
- [Prv] _getRate
- [Prv] _getCurrentSupply
- [Prv] _takeLiquidity #
- [Prv] calculateTaxFee
- [Prv] calculateLiquidityFee
- [Prv] removeAllFee #
- [Prv] restoreAllFee #
- [Pub] isExcludedFromFee
- [Pub] excludeFromFee #
  - modifiers: onlyOwner
- [Pub] includeInFee #
  - modifiers: onlyOwner
- [Prv] _getSellBnBAmount
- [Prv] _removeOldSellHistories #
- [Ext] SetBuyBackMaxTimeForHistories #
  - modifiers: onlyOwner
- [Ext] SetBuyBackDivisor #
  - modifiers: onlyOwner
- [Pub] GetBuyBackTimeInterval
- [Ext] SetBuyBackTimeInterval #
  - modifiers: onlyOwner
- [Ext] SetBuyBackRangeRate #
  - modifiers: onlyOwner
- [Pub] GetSwapMinutes
- [Ext] SetSwapMinutes #
  - modifiers: onlyOwner

- [Ext] setTaxFeePercent #
  - modifiers: onlyOwner
- [Ext] setBuyFee #
  - modifiers: onlyOwner
- [Ext] setSellFee #
  - modifiers: onlyOwner
- [Ext] setLiquidityFeePercent #
  - modifiers: onlyOwner
- [Ext] setBuyBackSellLimit #
  - modifiers: onlyOwner
- [Ext] setMaxTxAmount #
  - modifiers: onlyOwner
- [Ext] setMarketingDivisor #
  - modifiers: onlyOwner
- [Ext] setNumTokensSellToAddToBuyBack #
  - modifiers: onlyOwner
- [Ext] setMarketingAddress #
  - modifiers: onlyOwner
- [Pub] setSwapAndLiquifyEnabled #
  - modifiers: onlyOwner
- [Pub] setBuyBackEnabled #
  - modifiers: onlyOwner
- [Pub] setAutoBuyBackEnabled #
  - modifiers: onlyOwner
- [Ext] prepareForPreSale #
  - modifiers: onlyOwner
- [Ext] afterPreSale #
  - modifiers: onlyOwner
- [Prv] transferToAddressETH #
- [Pub] changeRouterVersion #
  - modifiers: onlyOwner

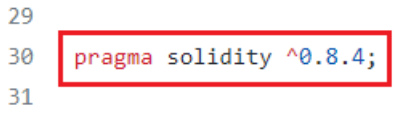
- [Ext] <Fallback> ($)
- [Pub] transferForeignToken #
  - modifiers: onlyOwner
- [Ext] Sweep #
  - modifiers: onlyOwner
- [Ext] setAddressFee #
  - modifiers: onlyOwner
- [Ext] setBuyAddressFee #
  - modifiers: onlyOwner
- [Ext] setSellAddressFee #
  - modifiers: onlyOwner

# Detailed Results

## Issues Checking Status

### 1. Floating Pragma

- SWC ID:103
- Severity: Low
- Location: BNBeer.sol
- Relationships: CWE-664: Improper Control of a Resource Through its Lifetime
- Description: A floating pragma is set. The current pragma Solidity directive is ""^0.8.4"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.




- Remediations: Lock the pragma version and also consider known bugs (https://github.com/ethereum/solidity/releases) for the compiler version that is chosen.

## 2. State Variable Default Visibility

- SWC ID:108
- Severity: Low
- Location: BNBeer.sol
- Relationships: CWE-710: Improper Adherence to Coding Standards
- Description: State variable visibility is not set. It is best practice to set the visibility of state variables explicitly. The default visibility for "inSwapAndLiquify" is internal. Other possible visibility settings are public and private.

```
511
512     bool inSwapAndLiquify;
513     bool public swapAndLiquifyEnabled = false;
514     bool public buyBackEnabled = true;
515
```

- Remediations: Variables can be specified as being public, internal or private. Explicitly define visibility for all state variables.

### 3. Block values as a proxy for time

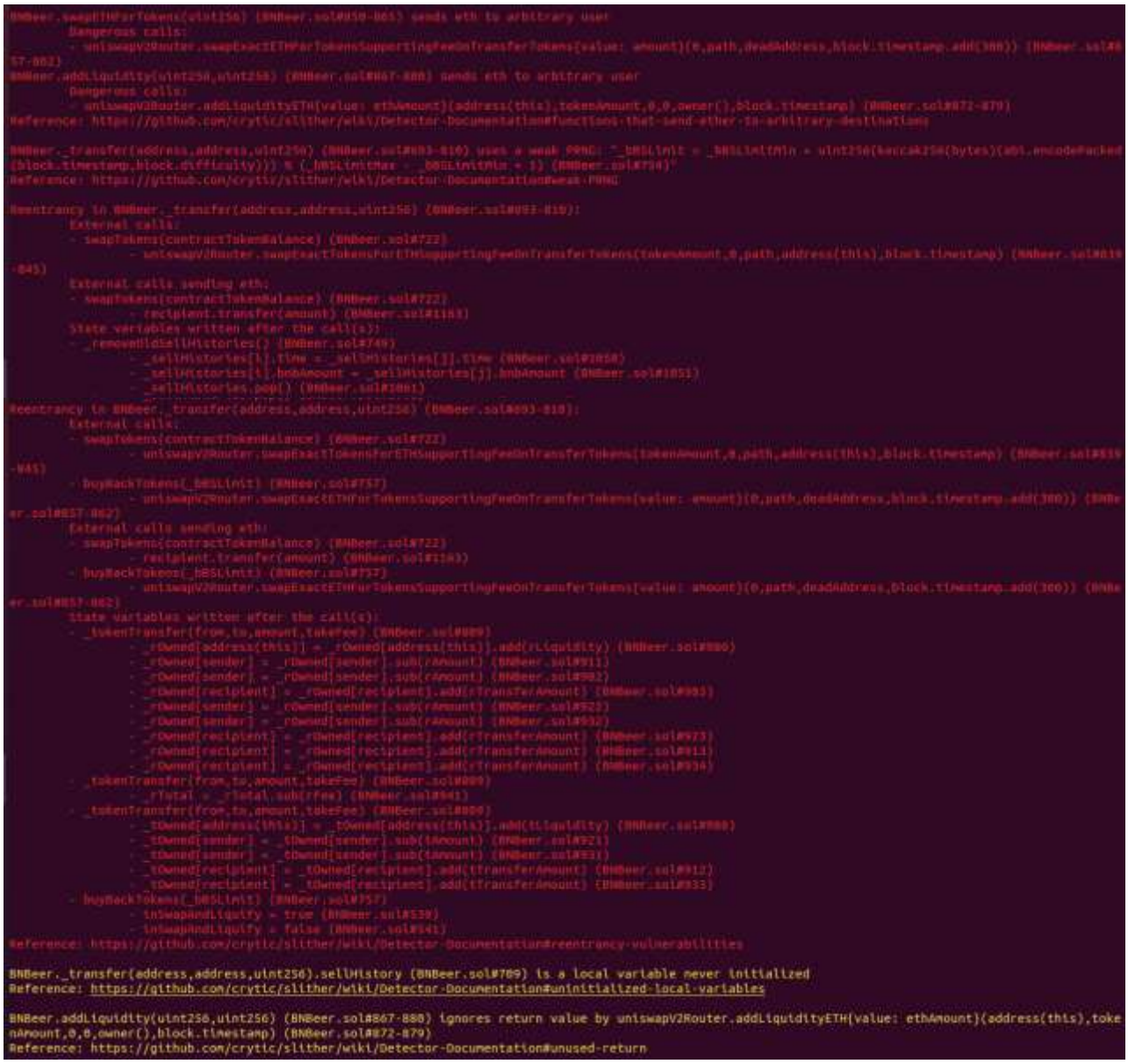
- SWC ID:116
- Severity: Low
- Location: BNBeer.sol
- Relationships: CWE-829: Inclusion of Functionality from Untrusted Control Sphere
- Description: A control flow decision is made based on The block.timestamp environment variable. The block.timestamp environment variable is used to determine a control flow decision. Note that the values of variables like coinbase, gaslimit, block number and timestamp are predictable and can be manipulated by a malicious miner. Also keep in mind that attackers know hashes of earlier blocks. Don't use any of those environment variables as sources of randomness and be aware that use of these variables introduces a certain level of trust into miners.

```
219       function unlock() public virtual {
220           require(_previousOwner == msg.sender, "You don't have permission to unlock");
221           require(block.timestamp > _lockTime , "Contract is locked until 7 days");
222           emit OwnershipTransferred(_owner, _previousOwner);
223           _owner = _previousOwner;
224       }
225   }
```

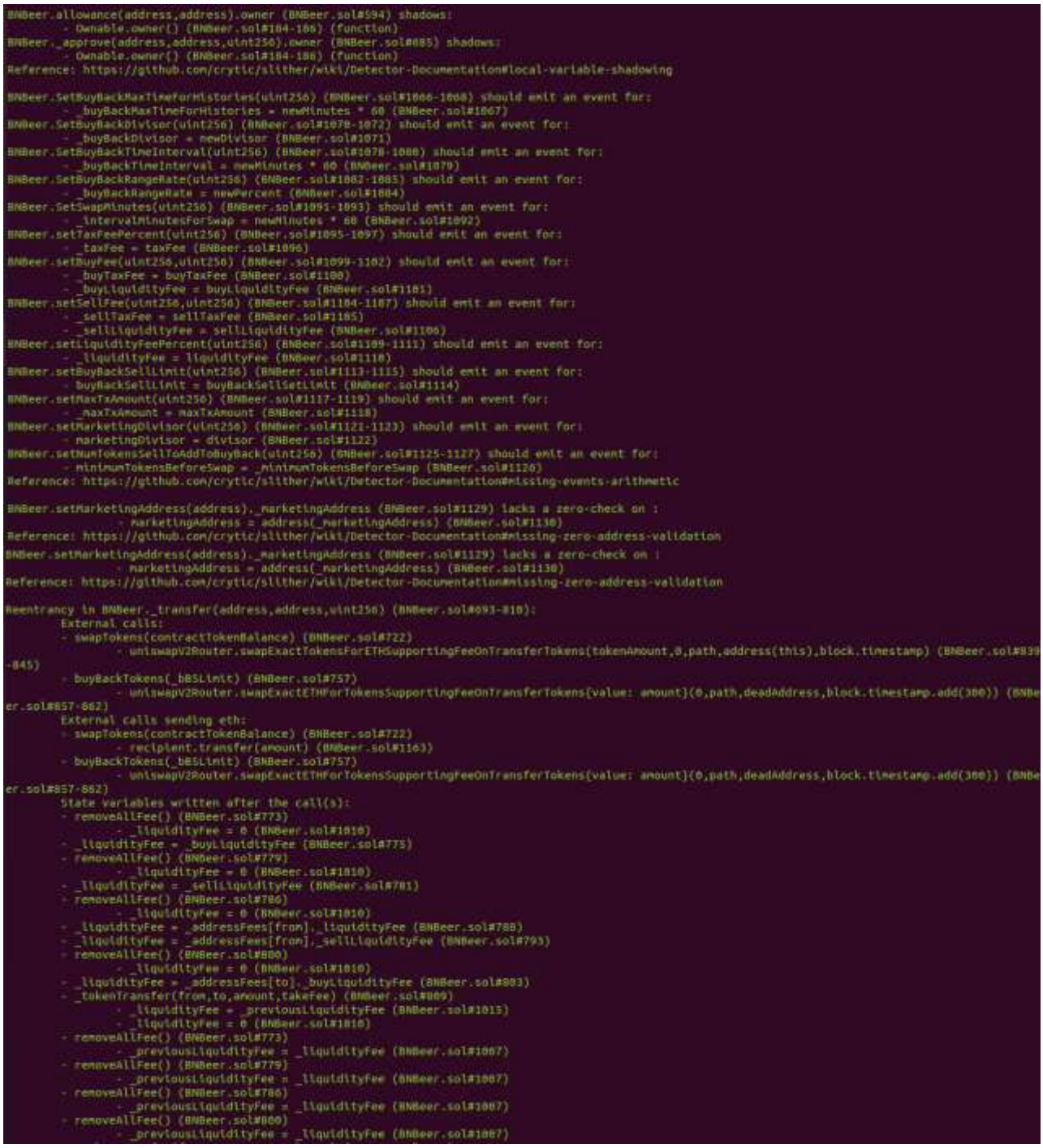
- Remediations: Developers should write smart contracts with the notion that block values are not precise, and the use of them can lead to unexpected effects. Alternatively, they may make use oracles.

## Automated Tool Results

Slither: -



# Smart Contract Audit

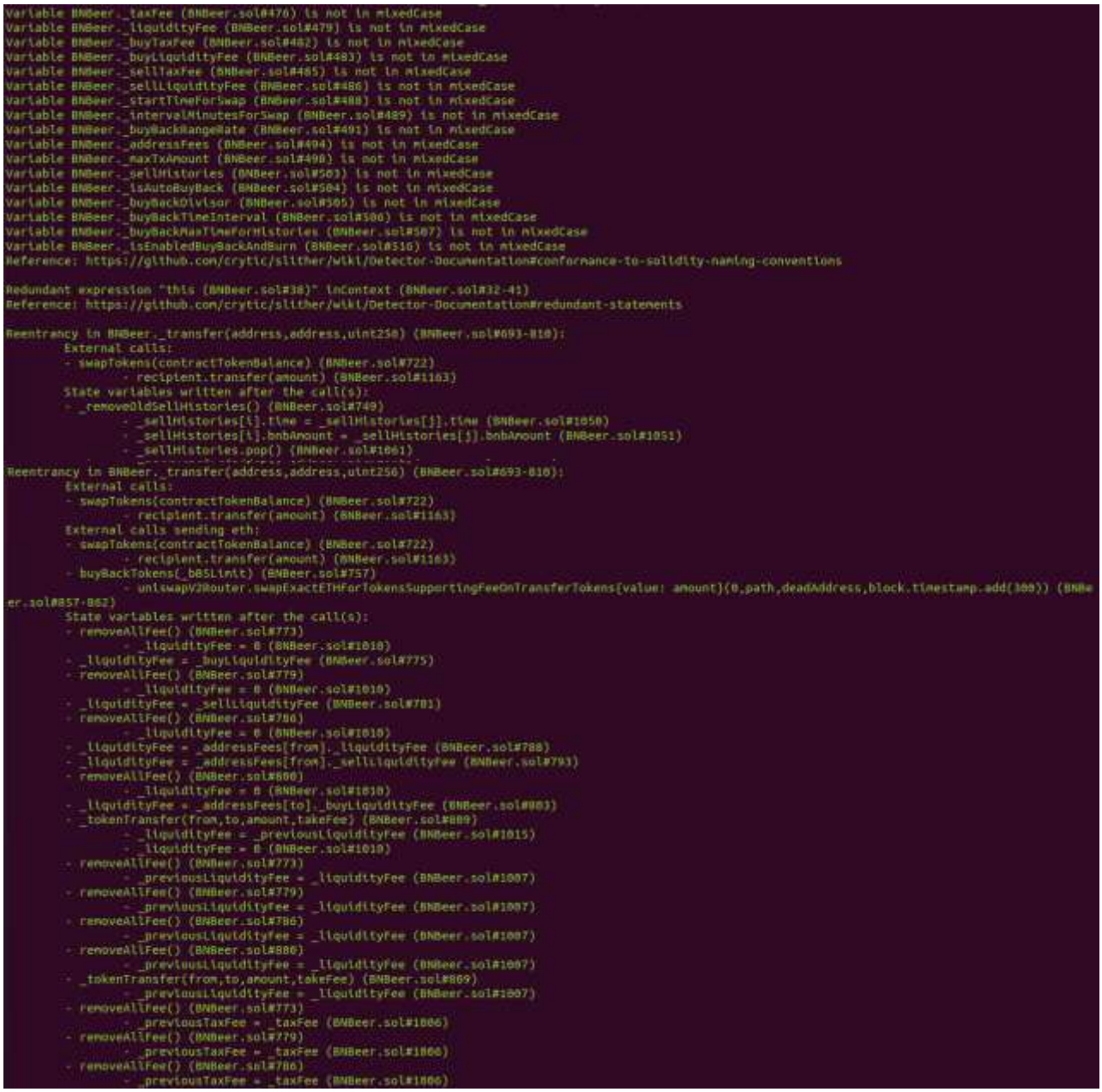


```
        - _tokenTransfer(from,to,amount,takeFee) (BNBeer.sol#809)
                - _previousLiquidityFee = _liquidityFee (BNBeer.sol#1007)
        - removeAllFee() (BNBeer.sol#773)
                - _previousTaxFee = _taxFee (BNBeer.sol#1006)
        - removeAllFee() (BNBeer.sol#779)
                - _previousTaxFee = _taxFee (BNBeer.sol#1006)
        - removeAllFee() (BNBeer.sol#786)
                - _previousTaxFee = _taxFee (BNBeer.sol#1006)
        - removeAllFee() (BNBeer.sol#800)
                - _previousTaxFee = _taxFee (BNBeer.sol#1006)
        - _tokenTransfer(from,to,amount,takeFee) (BNBeer.sol#809)
                - _previousTaxFee = _taxFee (BNBeer.sol#1006)
        - _tokenTransfer(from,to,amount,takeFee) (BNBeer.sol#809)
                - _tFeeTotal = _tFeeTotal.add(tFee) (BNBeer.sol#942)
        - removeAllFee() (BNBeer.sol#773)
                - _taxFee = 0 (BNBeer.sol#1009)
        - _taxFee = _buyTaxFee (BNBeer.sol#774)
        - removeAllFee() (BNBeer.sol#779)
                - _taxFee = 0 (BNBeer.sol#1009)
        - _taxFee = _sellTaxFee (BNBeer.sol#780)
        - removeAllFee() (BNBeer.sol#786)
                - _taxFee = 0 (BNBeer.sol#1009)
        - _taxFee = _addressFees[from]._taxFee (BNBeer.sol#787)
        - _taxFee = _addressFees[from]._sellTaxFee (BNBeer.sol#792)
        - removeAllFee() (BNBeer.sol#800)
                - _taxFee = 0 (BNBeer.sol#1009)
        - _taxFee = _addressFees[to]._buyTaxFee (BNBeer.sol#802)
        - _tokenTransfer(from,to,amount,takeFee) (BNBeer.sol#809)
                - _taxFee = _previousTaxFee (BNBeer.sol#1014)
                - _taxFee = 0 (BNBeer.sol#1009)
Reentrancy in BNBeer.changeRouterVersion(address) (BNBeer.sol#1166-1179):
        External calls:
        - _pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (BNBeer.sol#1172-1173)
        State variables written after the call(s):
        - uniswapV2Pair = _pair (BNBeer.sol#1175)
        - uniswapV2Router = _uniswapV2Router (BNBeer.sol#1178)
Reentrancy in BNBeer.changeRouterVersion(address) (BNBeer.sol#1166-1179):
        External calls:
        - _pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (BNBeer.sol#1172-1173)
        State variables written after the call(s):
        - uniswapV2Pair = _pair (BNBeer.sol#1175)
        - uniswapV2Router = _uniswapV2Router (BNBeer.sol#1178)
Reentrancy in BNBeer.constructor() (BNBeer.sol#544-566):
        External calls:
        - uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (BNBeer.sol#554-555)
        State variables written after the call(s):
        - _isExcludedFromFee[owner()] = true (BNBeer.sol#560)
        - _isExcludedFromFee[address(this)] = true (BNBeer.sol#561)
        - _startTimeForSwap = block.timestamp (BNBeer.sol#563)
        - uniswapV2Router = _uniswapV2Router (BNBeer.sol#557)
Reentrancy in BNBeer.transferFrom(address,address,uint256) (BNBeer.sol#683-687):
        External calls:
        - _transfer(sender,recipient,amount) (BNBeer.sol#684)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.add(300)) (BNBe
er.sol#857-862)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (BNBeer.sol#839
-845)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (BNBeer.sol#684)
                - recipient.transfer(amount) (BNBeer.sol#1163)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.add(300)) (BNBe
er.sol#857-862)
        State variables written after the call(s):
        - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (BNBeer.sol#685)
                - _allowances[owner][spender] = amount (BNBeer.sol#689)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
Reentrancy in BNBeer._transfer(address,address,uint256) (BNBeer.sol#693-810):
        External calls:
        - swapTokens(contractTokenBalance) (BNBeer.sol#722)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (BNBeer.sol#839
-845)
        - buyBackTokens(_bBSLimit) (BNBeer.sol#757)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.add(300)) (BNBe
er.sol#857-862)
        External calls sending eth:
        - swapTokens(contractTokenBalance) (BNBeer.sol#722)
                - recipient.transfer(amount) (BNBeer.sol#1163)
        - buyBackTokens(_bBSLimit) (BNBeer.sol#757)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.add(300)) (BNBe
er.sol#857-862)
        Event emitted after the call(s):
        - SwapETHForTokens(amount,path) (BNBeer.sol#864)
                - buyBackTokens(_bBSLimit) (BNBeer.sol#757)
        - Transfer(sender,recipient,tTransferAmount) (BNBeer.sol#908)
                - _tokenTransfer(from,to,amount,takeFee) (BNBeer.sol#809)
        - Transfer(sender,recipient,tTransferAmount) (BNBeer.sol#926)
                - _tokenTransfer(from,to,amount,takeFee) (BNBeer.sol#809)
        - Transfer(sender,recipient,tTransferAmount) (BNBeer.sol#916)
                - _tokenTransfer(from,to,amount,takeFee) (BNBeer.sol#809)
        - Transfer(sender,recipient,tTransferAmount) (BNBeer.sol#937)
                - _tokenTransfer(from,to,amount,takeFee) (BNBeer.sol#809)
Reentrancy in BNBeer.constructor() (BNBeer.sol#544-566):
        External calls:
        - uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (BNBeer.sol#554-555)
        Event emitted after the call(s):
        - Transfer(address(0),_msgSender(),_tTotal) (BNBeer.sol#565)
Reentrancy in BNBeer.swapETHForTokens(uint256) (BNBeer.sol#858-865):
        External calls:
        - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.add(300)) (BNBeer.sol#8
57-862)
        Event emitted after the call(s):
        - SwapETHForTokens(amount,path) (BNBeer.sol#864)
Reentrancy in BNBeer.swapTokensForEth(uint256) (BNBeer.sol#830-846):
        External calls:
        - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (BNBeer.sol#839-845)
        Event emitted after the call(s):
        - SwapTokensForETH(tokenAmount,path) (BNBeer.sol#847)
```

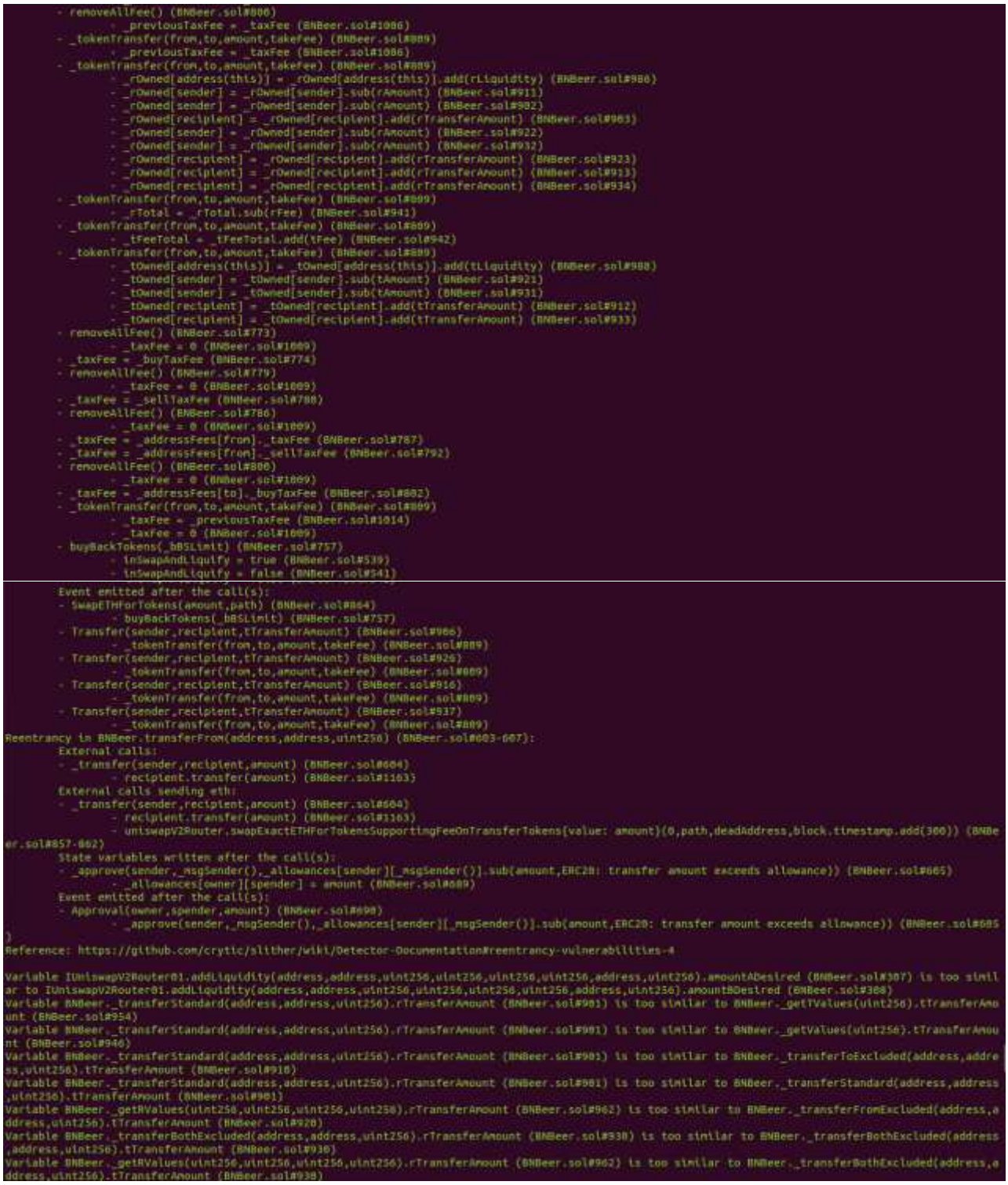
# Smart Contract Audit

```
Reentrancy in BNBeer.transferFrom(address,address,uint256) (BNBeer.sol#603-607):
        External calls:
        - _transfer(sender,recipient,amount) (BNBeer.sol#604)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.add(300)) (BNBe
er.sol#857-862)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (BNBeer.sol#839
-845)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (BNBeer.sol#604)
                - recipient.transfer(amount) (BNBeer.sol#1163)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.add(300)) (BNBe
er.sol#857-862)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (BNBeer.sol#690)
                - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (BNBeer.sol#605
)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Ownable.unlock() (BNBeer.sol#219-224) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(block.timestamp > _lockTime,Contract is locked until 7 days) (BNBeer.sol#221)
BNBeer._transfer(address,address,uint256) (BNBeer.sol#693-810) uses timestamp for comparisons
        Dangerous comparisons:
        - overMinimumTokenBalance && _startTimeForSwap + _intervalMinutesForSwap <= block.timestamp (BNBeer.sol#719)
        - _sellHistories[i].time >= startTime (BNBeer.sol#739)
        - balance > _bbSLimit (BNBeer.sol#756)
BNBeer.buyBackTokens(uint256) (BNBeer.sol#824-828) uses timestamp for comparisons
        Dangerous comparisons:
        - amount > 0 (BNBeer.sol#825)
BNBeer._removeOldSellHistories() (BNBeer.sol#1042-1064) uses timestamp for comparisons
        Dangerous comparisons:
        - _sellHistories[j].time >= maxStartTimeForHistories (BNBeer.sol#1048)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp

Address.isContract(address) (BNBeer.sol#114-123) uses assembly
        - INLINE ASM (BNBeer.sol#121)
Address._functionCallWithValue(address,bytes,uint256,string) (BNBeer.sol#151-168) uses assembly
        - INLINE ASM (BNBeer.sol#160-163)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Address._functionCallWithValue(address,bytes,uint256,string) (BNBeer.sol#151-168) is never used and should be removed
Address.functionCall(address,bytes) (BNBeer.sol#134-136) is never used and should be removed
Address.functionCall(address,bytes,string) (BNBeer.sol#138-140) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (BNBeer.sol#142-144) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (BNBeer.sol#146-149) is never used and should be removed
Address.isContract(address) (BNBeer.sol#114-123) is never used and should be removed
Address.sendValue(address,uint256) (BNBeer.sol#125-131) is never used and should be removed
BNBeer.addLiquidity(uint256,uint256) (BNBeer.sol#867-880) is never used and should be removed
Context._msgData() (BNBeer.sol#37-40) is never used and should be removed
SafeMath.mod(uint256,uint256) (BNBeer.sol#102-104) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (BNBeer.sol#106-109) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

BNBeer._rTotal (BNBeer.sol#454) is set pre-construction with a non-constant function or state variable:
        - (MAX - (MAX % _tTotal))
BNBeer._previousTaxFee (BNBeer.sol#477) is set pre-construction with a non-constant function or state variable:
        - _taxFee
BNBeer._previousLiquidityFee (BNBeer.sol#480) is set pre-construction with a non-constant function or state variable:
        - _liquidityFee
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#function-initializing-state

Pragma version^0.8.4 (BNBeer.sol#38) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
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) (BNBeer.sol#125-131):
        - (success) = recipient.call{value: amount}() (BNBeer.sol#129)
Low level call in Address._functionCallWithValue(address,bytes,uint256,string) (BNBeer.sol#151-168):
        - (success,returndata) = target.call{value: weiValue}(data) (BNBeer.sol#154)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function IUniswapV2Pair.DOMAIN_SEPARATOR() (BNBeer.sol#264) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (BNBeer.sol#265) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (BNBeer.sol#281) is not in mixedCase
Function IUniswapV2Router01.WETH() (BNBeer.sol#302) is not in mixedCase
Parameter BNBeer.calculateTaxFee(uint256)._amount (BNBeer.sol#991) is not in mixedCase
Parameter BNBeer.calculateLiquidityFee(uint256)._amount (BNBeer.sol#997) is not in mixedCase
Function BNBeer.setBuyBackMaxTimeForHistories(uint256) (BNBeer.sol#1066-1068) is not in mixedCase
Function BNBeer.setBuyBackDivisor(uint256) (BNBeer.sol#1070-1072) is not in mixedCase
Function BNBeer.getBuyBackTimeInterval() (BNBeer.sol#1074-1076) is not in mixedCase
Function BNBeer.setBuyBackTimeInterval(uint256) (BNBeer.sol#1078-1080) is not in mixedCase
Function BNBeer.setBuyBackRangeRate(uint256) (BNBeer.sol#1082-1085) is not in mixedCase
Function BNBeer.getSwapMinutes() (BNBeer.sol#1087-1089) is not in mixedCase
Function BNBeer.setSwapMinutes(uint256) (BNBeer.sol#1091-1093) is not in mixedCase
Parameter BNBeer.setNumTokensSellToAddToBuyBack(uint256)._minimumTokensBeforeSwap (BNBeer.sol#1125) is not in mixedCase
Parameter BNBeer.setMarketingAddress(address)._marketingAddress (BNBeer.sol#1129) is not in mixedCase
Parameter BNBeer.setSwapAndLiquifyEnabled(bool)._enabled (BNBeer.sol#1133) is not in mixedCase
Parameter BNBeer.setBuyBackEnabled(bool)._enabled (BNBeer.sol#1138) is not in mixedCase
Parameter BNBeer.setAutoBuyBackEnabled(bool)._enabled (BNBeer.sol#1143) is not in mixedCase
Parameter BNBeer.changeRouterVersion(address)._router (BNBeer.sol#1160) is not in mixedCase
Parameter BNBeer.transferForeignToken(address,address)._token (BNBeer.sol#1185) is not in mixedCase
Parameter BNBeer.transferForeignToken(address,address)._to (BNBeer.sol#1185) is not in mixedCase
Function BNBeer.Sweep() (BNBeer.sol#1191-1194) is not in mixedCase
Parameter BNBeer.setAddressFee(address,bool,uint256,uint256)._address (BNBeer.sol#1196) is not in mixedCase
Parameter BNBeer.setAddressFee(address,bool,uint256,uint256)._enable (BNBeer.sol#1196) is not in mixedCase
Parameter BNBeer.setAddressFee(address,bool,uint256,uint256)._addressTaxFee (BNBeer.sol#1196) is not in mixedCase
Parameter BNBeer.setAddressFee(address,bool,uint256,uint256)._addressLiquidityFee (BNBeer.sol#1196) is not in mixedCase
Parameter BNBeer.setBuyAddressFee(address,bool,uint256,uint256)._address (BNBeer.sol#1202) is not in mixedCase
Parameter BNBeer.setBuyAddressFee(address,bool,uint256,uint256)._enable (BNBeer.sol#1202) is not in mixedCase
Parameter BNBeer.setBuyAddressFee(address,bool,uint256,uint256)._addressTaxFee (BNBeer.sol#1202) is not in mixedCase
Parameter BNBeer.setBuyAddressFee(address,bool,uint256,uint256)._addressLiquidityFee (BNBeer.sol#1202) is not in mixedCase
Parameter BNBeer.setSellAddressFee(address,bool,uint256,uint256)._address (BNBeer.sol#1208) is not in mixedCase
Parameter BNBeer.setSellAddressFee(address,bool,uint256,uint256)._enable (BNBeer.sol#1208) is not in mixedCase
Parameter BNBeer.setSellAddressFee(address,bool,uint256,uint256)._addressTaxFee (BNBeer.sol#1208) is not in mixedCase
Parameter BNBeer.setSellAddressFee(address,bool,uint256,uint256)._addressLiquidityFee (BNBeer.sol#1208) is not in mixedCase
```

# Smart Contract Audit

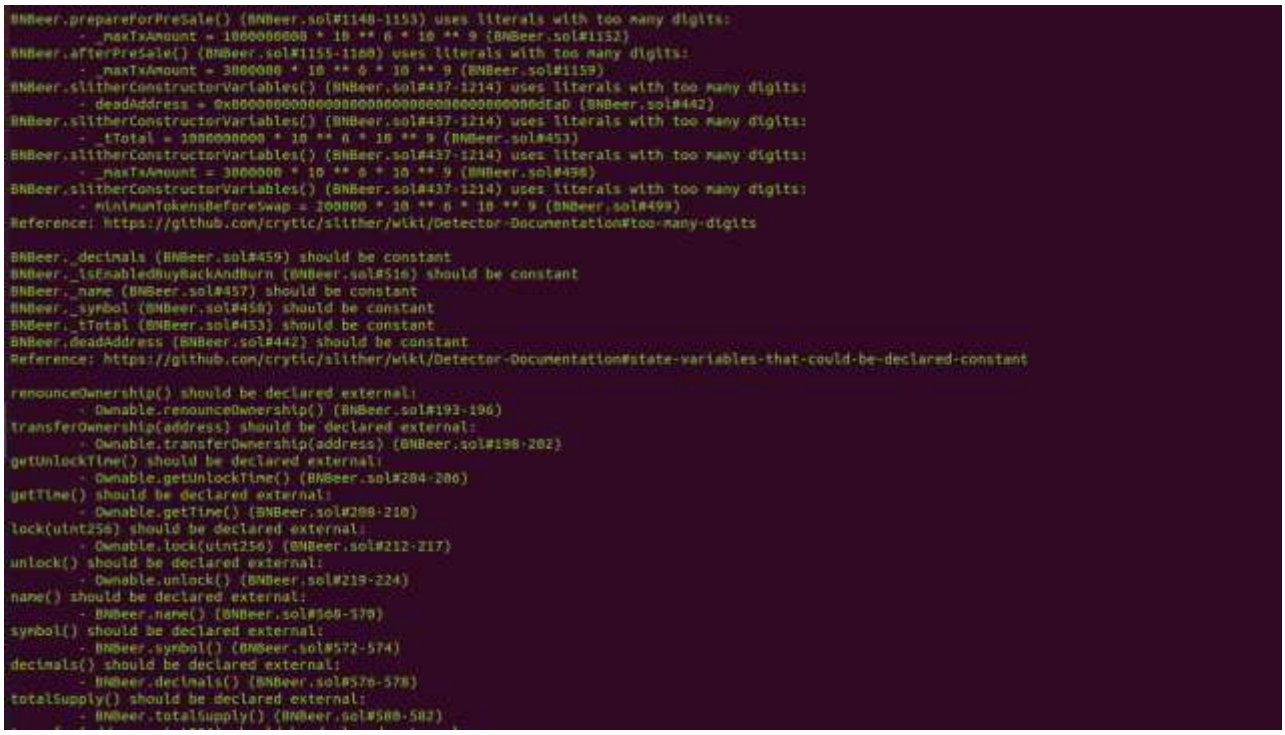


# Smart Contract Audit



```
Variable BNBeer._transferFromExcluded(address,address,uint256).rTransferAmount (BNBeer.sol#920) is too similar to BNBeer._transferBothExcluded(address
,address,uint256).tTransferAmount (BNBeer.sol#930)
Variable BNBeer._transferFromExcluded(address,address,uint256).rTransferAmount (BNBeer.sol#920) is too similar to BNBeer._transferFromExcluded(address
,address,uint256).tTransferAmount (BNBeer.sol#920)
Variable BNBeer._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (BNBeer.sol#962) is too similar to BNBeer._getTValues(uint256).tTransferA
mount (BNBeer.sol#954)
Variable BNBeer._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (BNBeer.sol#962) is too similar to BNBeer._getValues(uint256).tTransferAm
ount (BNBeer.sol#946)
Variable BNBeer.reflectionFromToken(uint256,bool).rTransferAmount (BNBeer.sol#651) is too similar to BNBeer._transferBothExcluded(address,address,uint
256).tTransferAmount (BNBeer.sol#930)
Variable BNBeer._transferFromExcluded(address,address,uint256).rTransferAmount (BNBeer.sol#920) is too similar to BNBeer._getValues(uint256).tTransfer
Amount (BNBeer.sol#946)
Variable BNBeer._transferStandard(address,address,uint256).rTransferAmount (BNBeer.sol#901) is too similar to BNBeer._transferFromExcluded(address,add
ress,uint256).tTransferAmount (BNBeer.sol#920)
Variable BNBeer._transferStandard(address,address,uint256).rTransferAmount (BNBeer.sol#901) is too similar to BNBeer._transferBothExcluded(address,add
ress,uint256).tTransferAmount (BNBeer.sol#930)
Variable BNBeer._getValues(uint256).rTransferAmount (BNBeer.sol#947) is too similar to BNBeer._transferBothExcluded(address,address,uint256).tTransfer
Amount (BNBeer.sol#930)
Variable BNBeer._transferBothExcluded(address,address,uint256).rTransferAmount (BNBeer.sol#930) is too similar to BNBeer._transferToExcluded(address,a
ddress,uint256).tTransferAmount (BNBeer.sol#910)
Variable BNBeer._getValues(uint256).rTransferAmount (BNBeer.sol#947) is too similar to BNBeer._getValues(uint256).tTransferAmount (BNBeer.sol#946)
Variable BNBeer._transferBothExcluded(address,address,uint256).rTransferAmount (BNBeer.sol#930) is too similar to BNBeer._transferStandard(address,add
ress,uint256).tTransferAmount (BNBeer.sol#901)
Variable BNBeer._transferFromExcluded(address,address,uint256).rTransferAmount (BNBeer.sol#920) is too similar to BNBeer._transferStandard(address,add
ress,uint256).tTransferAmount (BNBeer.sol#901)
Variable BNBeer.reflectionFromToken(uint256,bool).rTransferAmount (BNBeer.sol#651) is too similar to BNBeer._transferStandard(address,address,uint256)
.tTransferAmount (BNBeer.sol#901)
Variable BNBeer._transferToExcluded(address,address,uint256).rTransferAmount (BNBeer.sol#910) is too similar to BNBeer._getTValues(uint256).tTransferA
mount (BNBeer.sol#954)
Variable BNBeer._transferBothExcluded(address,address,uint256).rTransferAmount (BNBeer.sol#930) is too similar to BNBeer._getValues(uint256).tTransfer
Amount (BNBeer.sol#946)
Variable BNBeer.reflectionFromToken(uint256,bool).rTransferAmount (BNBeer.sol#651) is too similar to BNBeer._getValues(uint256).tTransferAmount (BNBeer.sol#946)
Variable BNBeer._transferToExcluded(address,address,uint256).rTransferAmount (BNBeer.sol#910) is too similar to BNBeer._transferToExcluded(address,add
ress,uint256).tTransferAmount (BNBeer.sol#910)
Variable BNBeer._transferToExcluded(address,address,uint256).rTransferAmount (BNBeer.sol#910) is too similar to BNBeer._transferStandard(address,addre
ss,uint256).tTransferAmount (BNBeer.sol#901)
Variable BNBeer._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (BNBeer.sol#962) is too similar to BNBeer._transferToExcluded(address,add
ress,uint256).tTransferAmount (BNBeer.sol#910)
Variable BNBeer._getValues(uint256).rTransferAmount (BNBeer.sol#947) is too similar to BNBeer._transferFromExcluded(address,address,uint256).tTransfer
Amount (BNBeer.sol#920)
Variable BNBeer._transferToExcluded(address,address,uint256).rTransferAmount (BNBeer.sol#910) is too similar to BNBeer._getValues(uint256).tTransferAm
ount (BNBeer.sol#946)
Variable BNBeer.reflectionFromToken(uint256,bool).rTransferAmount (BNBeer.sol#651) is too similar to BNBeer._getValues(uint256).tTransferAmount (BNBeer.sol#946)
Variable BNBeer._transferToExcluded(address,address,uint256).rTransferAmount (BNBeer.sol#910) is too similar to BNBeer._transferToExcluded(address,add
ress,uint256).tTransferAmount (BNBeer.sol#910)
Variable BNBeer._transferToExcluded(address,address,uint256).rTransferAmount (BNBeer.sol#910) is too similar to BNBeer._transferStandard(address,addre
ss,uint256).tTransferAmount (BNBeer.sol#901)
Variable BNBeer._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (BNBeer.sol#962) is too similar to BNBeer._transferToExcluded(address,add
ress,uint256).tTransferAmount (BNBeer.sol#910)
Variable BNBeer._getValues(uint256).rTransferAmount (BNBeer.sol#947) is too similar to BNBeer._transferFromExcluded(address,address,uint256).tTransfer
Amount (BNBeer.sol#920)
Variable BNBeer._transferToExcluded(address,address,uint256).rTransferAmount (BNBeer.sol#910) is too similar to BNBeer._getValues(uint256).tTransferAm
ount (BNBeer.sol#946)
Variable BNBeer._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (BNBeer.sol#962) is too similar to BNBeer._transferStandard(address,addre
ss,uint256).tTransferAmount (BNBeer.sol#901)
Variable BNBeer.reflectionFromToken(uint256,bool).rTransferAmount (BNBeer.sol#651) is too similar to BNBeer._transferFromExcluded(address,address,uint
256).tTransferAmount (BNBeer.sol#920)
Variable BNBeer._transferBothExcluded(address,address,uint256).rTransferAmount (BNBeer.sol#930) is too similar to BNBeer._transferFromExcluded(address
,address,uint256).tTransferAmount (BNBeer.sol#920)
Variable BNBeer._getValues(uint256).rTransferAmount (BNBeer.sol#947) is too similar to BNBeer._getTValues(uint256).tTransferAmount (BNBeer.sol#954)
Variable BNBeer._transferToExcluded(address,address,uint256).rTransferAmount (BNBeer.sol#910) is too similar to BNBeer._transferBothExcluded(address,a
ddress,uint256).tTransferAmount (BNBeer.sol#930)
Variable BNBeer._getValues(uint256).rTransferAmount (BNBeer.sol#947) is too similar to BNBeer._transferToExcluded(address,address,uint256).tTransferAm
ount (BNBeer.sol#910)
Variable BNBeer._transferFromExcluded(address,address,uint256).rTransferAmount (BNBeer.sol#920) is too similar to BNBeer._getTValues(uint256).tTransfe
rAmount (BNBeer.sol#954)
Variable BNBeer.reflectionFromToken(uint256,bool).rTransferAmount (BNBeer.sol#651) is too similar to BNBeer._getTValues(uint256).tTransferAmount (BNBe
er.sol#954)
Variable BNBeer._transferToExcluded(address,address,uint256).rTransferAmount (BNBeer.sol#910) is too similar to BNBeer._transferFromExcluded(address,a
ddress,uint256).tTransferAmount (BNBeer.sol#920)
Variable BNBeer._getValues(uint256).rTransferAmount (BNBeer.sol#947) is too similar to BNBeer._transferStandard(address,address,uint256).tTransferAmou
nt (BNBeer.sol#901)
Variable BNBeer._transferBothExcluded(address,address,uint256).rTransferAmount (BNBeer.sol#930) is too similar to BNBeer._getTValues(uint256).tTransfe
rAmount (BNBeer.sol#954)
Variable BNBeer._transferFromExcluded(address,address,uint256).rTransferAmount (BNBeer.sol#920) is too similar to BNBeer._transferToExcluded(address,a
ddress,uint256).tTransferAmount (BNBeer.sol#910)
Variable BNBeer.reflectionFromToken(uint256,bool).rTransferAmount (BNBeer.sol#651) is too similar to BNBeer._transferToExcluded(address,address,uint25
6).tTransferAmount (BNBeer.sol#910)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
```

# Smart Contract Audit



# Smart Contract Audit

MythX: -

Report for BNBeer.sol
https://dashboard.mythx.io/#/console/analyses/9832658e-e2db-4e04-82cf-39d35da3323b

| Line | SWC Title | Severity | Short Description |
|---|---|---|---|
| 30 | (SWC-103) Floating Pragma | Low | A floating pragma is set. |
| 61 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "+" discovered |
| 73 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "-" discovered |
| 83 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "*" discovered |
| 84 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "/" discovered |
| 96 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "/" discovered |
| 108 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "%" discovered |
| 215 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "+" discovered |
| 453 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "**" discovered |
| 453 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "*" discovered |
| 454 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "%" discovered |
| 454 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "-" discovered |
| 489 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "*" discovered |
| 498 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "**" discovered |
| 498 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "*" discovered |
| 499 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "**" discovered |
| 499 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "*" discovered |
| 500 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "*" discovered |
| 503 | (SWC-110) Assert Violation | Unknown | Public state variable with array type causing reacheable exception by default. |
| 507 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "*" discovered |
| 512 | (SWC-108) State Variable Default Visibility | Low | State variable visibility is not set. |
| 674 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "++" discovered |
| 675 | (SWC-110) Assert Violation | Unknown | Out of bounds array access |
| 676 | (SWC-110) Assert Violation | Unknown | Out of bounds array access |
| 676 | (SWC-101) Integer Overflow and Underflow | Unknown | Compiler-rewritable "<uint> - 1" discovered |
| 676 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "-" discovered |
| 719 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "+" discovered |
| 734 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "-" discovered |
| 737 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "++" discovered |
| 739 | (SWC-110) Assert Violation | Unknown | Out of bounds array access |
| 740 | (SWC-110) Assert Violation | Unknown | Out of bounds array access |
| 741 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "+" discovered |
| 754 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "%" discovered |
| 754 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "-" discovered |
| 754 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "+" discovered |

| | | | |
|---|---|---|---|
| 833 | (SWC-110) Assert Violation | Unknown | Out of bounds array access |
| 834 | (SWC-110) Assert Violation | Unknown | Out of bounds array access |
| 853 | (SWC-110) Assert Violation | Unknown | Out of bounds array access |
| 854 | (SWC-110) Assert Violation | Unknown | Out of bounds array access |
| 974 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "++" discovered |
| 975 | (SWC-110) Assert Violation | Unknown | Out of bounds array access |
| 976 | (SWC-110) Assert Violation | Unknown | Out of bounds array access |
| 977 | (SWC-110) Assert Violation | Unknown | Out of bounds array access |
| 993 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "**" discovered |
| 999 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "**" discovered |
| 1034 | (SWC-110) Assert Violation | Unknown | Out of bounds array access |
| 1035 | (SWC-110) Assert Violation | Unknown | Out of bounds array access |
| 1039 | (SWC-110) Assert Violation | Unknown | Out of bounds array access |
| 1044 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "-" discovered |
| 1046 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "++" discovered |
| 1048 | (SWC-110) Assert Violation | Unknown | Out of bounds array access |
| 1050 | (SWC-110) Assert Violation | Unknown | Out of bounds array access |
| 1051 | (SWC-110) Assert Violation | Unknown | Out of bounds array access |
| 1053 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "+" discovered |
| 1057 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "-" discovered |
| 1059 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "++" discovered |
| 1067 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "*" discovered |
| 1079 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "*" discovered |
| 1092 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "*" discovered |
| 1152 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "*" discovered |
| 1152 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "**" discovered |
| 1159 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "*" discovered |
| 1159 | (SWC-101) Integer Overflow and Underflow | Unknown | Arithmetic operation "**" discovered |

Solhint: -

Linter results:

```
BNBeer.sol:30:1: Error: Compiler version ^0.8.4 does not satisfy the r semver requirement

BNBeer.sol:154:51: Error: Avoid to use low level calls.

BNBeer.sol:160:17: Error: Avoid to use inline assembly. It is acceptable only in rare cases

BNBeer.sol:178:5: Error: Explicitly mark visibility in function (Set ignoreConstructors to true if using
solidity >=0.7.0)

BNBeer.sol:209:16: Error: Avoid to make time-based decisions in your business logic

BNBeer.sol:215:21: Error: Avoid to make time-based decisions in your business logic

BNBeer.sol:221:17: Error: Avoid to make time-based decisions in your business logic

BNBeer.sol:264:5: Error: Function name must be in mixedCase

BNBeer.sol:265:5: Error: Function name must be in mixedCase

BNBeer.sol:281:5: Error: Function name must be in mixedCase

BNBeer.sol:302:5: Error: Function name must be in mixedCase

BNBeer.sol:437:1: Error: Contract has 40 states declarations but allowed no more than 15

BNBeer.sol:512:5: Error: Explicitly mark visibility of state

BNBeer.sol:544:5: Error: Explicitly mark visibility in function (Set ignoreConstructors to true if using
solidity >=0.7.0)

BNBeer.sol:563:29: Error: Avoid to make time-based decisions in your business logic
```

```
BNBeer.sol:710:32: Error: Avoid to make time-based decisions in your business logic

BNBeer.sol:719:95: Error: Avoid to make time-based decisions in your business logic

BNBeer.sol:720:41: Error: Avoid to make time-based decisions in your business logic

BNBeer.sol:734:45: Error: Avoid to make time-based decisions in your business logic

BNBeer.sol:754:91: Error: Avoid to make time-based decisions in your business logic

BNBeer.sol:844:13: Error: Avoid to make time-based decisions in your business logic

BNBeer.sol:861:13: Error: Avoid to make time-based decisions in your business logic

BNBeer.sol:878:13: Error: Avoid to make time-based decisions in your business logic

BNBeer.sol:1044:44: Error: Avoid to make time-based decisions in your business logic

BNBeer.sol:1066:5: Error: Function name must be in mixedCase

BNBeer.sol:1070:5: Error: Function name must be in mixedCase

BNBeer.sol:1074:5: Error: Function name must be in mixedCase

BNBeer.sol:1078:5: Error: Function name must be in mixedCase

BNBeer.sol:1082:5: Error: Function name must be in mixedCase

BNBeer.sol:1087:5: Error: Function name must be in mixedCase

BNBeer.sol:1091:5: Error: Function name must be in mixedCase
```

```
BNBeer.sol:1182:32: Error: Code contains empty blocks

BNBeer.sol:1191:5: Error: Function name must be in mixedCase
```

Mythril: -

```
==== Dependence on predictable environment variable ====
SWC ID: 116
Severity: Low
Contract: Ownable
Function name: unlock()
PC address: 400
Estimated Gas Usage: 1888 - 1903
A control flow decision is made based on The block.timestamp environment variable.
The block.timestamp environment variable is used to determine a control flow decision. Note that the values of variables like coinbase, gaslimit, bloc
k number and timestamp are predictable and can be manipulated by a malicious miner. Also keep in mind that attackers know hashes of earlier blocks. Do
n't use any of those environment variables as sources of randomness and be aware that use of these variables introduces a certain level of trust into
miners.
--------------------
In file: BNBeer.sol:221

require(block.timestamp > _lockTime , "Contract is locked until 7 days")

--------------------
Initial State:

Account: [CREATOR], balance: 0x1, nonce:0, storage:{}
Account: [ATTACKER], balance: 0x0, nonce:0, storage:{}
Account: [SOMEGUY], balance: 0x0, nonce:0, storage:{}

Transaction Sequence:

Caller: [CREATOR], calldata: , value: 0x0
Caller: [CREATOR], function: lock(uint256), txdata: 0xdd467064fffffffffffffffffffffffffffffffffffffffffffffffffffffffff, value: 0x0
Caller: [CREATOR], function: unlock(), txdata: 0xa69df4b5, value: 0x0
==== Dependence on predictable environment variable ====
SWC ID: 116
Severity: Low
Contract: Ownable
Function name: lock(uint256)
PC address: 1094
Estimated Gas Usage: 12812 - 52907
A control flow decision is made based on The block.timestamp environment variable.
The block.timestamp environment variable is used to determine a control flow decision. Note that the values of variables like coinbase, gaslimit, bloc
k number and timestamp are predictable and can be manipulated by a malicious miner. Also keep in mind that attackers know hashes of earlier blocks. Do
n't use any of those environment variables as sources of randomness and be aware that use of these variables introduces a certain level of trust into
miners.
--------------------
In file: #utility.yul:57

if

--------------------
Initial State:

Account: [CREATOR], balance: 0x0, nonce:0, storage:{}
Account: [ATTACKER], balance: 0x0, nonce:0, storage:{}
Account: [SOMEGUY], balance: 0x0, nonce:0, storage:{}

Transaction Sequence:

Caller: [CREATOR], calldata: , value: 0x0
Caller: [CREATOR], function: lock(uint256), txdata: 0xdd467064000000000000000000000000000000000000000000000000000000000000000000, value: 0x0
```

## Basic Coding Bugs

1. **Constructor Mismatch**
   - Description: Whether the contract name and its constructor are not identical to each other.
   - Result: PASSED
   - Severity: Critical
2. **Ownership Takeover**
   - Description: Whether the set owner function is not protected.
   - Result: PASSED
   - Severity: Critical
3. **Redundant Fallback Function**
   - Description: Whether the contract has a redundant fallback function.
   - Result: PASSED
   - Severity: Critical
4. **Overflows & Underflows**
   - Description: Whether the contract has general overflow or underflow vulnerabilities
   - Result: PASSED
   - Severity: High
5. **Reentrancy**
   - Description: Reentrancy is an issue when code can call back into your contract and change state, such as withdrawing ETHs.
   - Result: PASSED
   - Severity: Critical
6. **MONEY-Giving Bug**
   - Description: Whether the contract returns funds to an arbitrary address.
   - Result: PASSED
   - Severity: High

7. **Blackhole**

   - Description: Whether the contract locks ETH indefinitely: merely in without out.
   - Result: PASSED
   - Severity: High

8. **Unauthorized Self-Destruct**

   - Description: Whether the contract can be killed by any arbitrary address.
   - Result: PASSED
   - Severity: Medium

9. **Revert DoS**

   - Description: Whether the contract is vulnerable to DoS attack because of unexpected revert.
   - Result: PASSED
   - Severity: Medium

10. **Unchecked External Call**

    - Description: Whether the contract has any external call without checking the return value.
    - Result: PASSED
    - Severity: Medium

11. **Gasless Send**

    - Description: Whether the contract is vulnerable to gasless send.
    - Result: PASSED
    - Severity: Medium

12. **Send Instead of Transfer**

    - Description: Whether the contract uses send instead of transfer.
    - Result: PASSED
    - Severity: Medium

13. **Costly Loop**

    - Description: Whether the contract has any costly loop which may lead to Out-Of-Gas exception.
    - Result: PASSED
    - Severity: Medium

**14. (Unsafe)Use of Untrusted Libraries**

- Description: Whether the contract use any suspicious libraries.
- Result: PASSED
- Severity: Medium

**15. (Unsafe)Use of Predictable Variables**

- Description: Whether the contract contains any randomness variable, but its value can be predicated.
- Result: PASSED
- Severity: Medium

**16. Transaction Ordering Dependence**

- Description: Whether the final state of the contract depends on the order of the transactions.
- Result: PASSED
- Severity: Medium

**17. Deprecated Uses**

- Description: Whether the contract use the deprecated tx.origin to perform the authorization.
- Result: PASSED
- Severity: Medium

**Semantic Consistency Checks**

- Description: Whether the semantic of the white paper is different from the implementation of the contract.
- Result: PASSED
- Severity: Critical

## Conclusion

In this audit, we thoroughly analyzed BNBEER's Smart Contract. The current code base is well organized and there are promptly some low issues found in the first phase of Smart Contract Audit.

Meanwhile, we need to emphasize that smart contracts as a whole are still in an early, but exciting stage of development. To improve this report, we greatly appreciate any constructive feedbacks or suggestions, on our methodology, audit findings, or potential gaps in scope/coverage.

# About eNebula Solutions

We believe that people have a fundamental need to security and that the use of secure solutions enables every person to more freely use the Internet and every other connected technology. We aim to provide security consulting service to help others make their solutions more resistant to unauthorized access to data & inadvertent manipulation of the system. We support teams from the design phase through the production to launch and surely after.

The eNebula Solutions team has skills for reviewing code in C, C++, Python, Haskell, Rust, Node.js, Solidity, Go, and JavaScript for common security vulnerabilities & specific attack vectors. The team has reviewed implementations of cryptographic protocols and distributed system architecture, including in cryptocurrency, blockchains, payments, and smart contracts. Additionally, the team can utilize various tools to scan code & networks and build custom tools as necessary.

Although we are a small team, we surely believe that we can have a momentous impact on the world by being translucent and open about the work we do.

For more information about our security consulting,
please mail us at – contact@enebula.in