Ether Authority

www.EtherAuthority.io
audit@etherauthority.io

SMART CONTRACT

Security Audit Report

Project: BLB Token
Website: https://blb.homes
Platform: Binance Smart Chain
Language: Solidity
Date: April 9th, 2022

Table of contents

`

THIS IS SECURITY AUDIT REPORT DOCUMENT AND WHICH MAY CONTAIN INFORMATION WHICH IS CONFIDENTIAL. WHICH INCLUDES ANY POTENTIAL VULNERABILITIES AND MALICIOUS CODES WHICH CAN BE USED TO EXPLOIT THE SOFTWARE. THIS MUST BE REFERRED INTERNALLY AND ONLY SHOULD BE MADE AVAILABLE TO THE PUBLIC AFTER ISSUES ARE RESOLVED.

Introduction

EtherAuthority was contracted by the Blue Bird team to perform the Security audit of the BLB Token smart contract code. The audit has been performed using manual analysis as well as using automated software tools. This report presents all the findings regarding the audit performed on April 9th, 2022.

The purpose of this audit was to address the following:

- Ensure that all claimed functions exist and function correctly.

- Identify any security vulnerabilities that may be present in the smart contract.

Project Background

The Blue Bird contract is a BE20 standard smart contract, having functions like addLiquidity, swapTokens, calculateTaxFee, setTaxFee, presale, etc.

Audit scope

Name	Code Review and Security Analysis Report for BLB Token Smart Contract
Platform	**BSC / Solidity**
File	BuyBackToken.sol
File MD5 Hash	7DD1CFF6428B2BBE6FCFD7485177193C
Online Code Link	0x1cd4c653c8a8775e26cc14ab30a52bc12b3d7fd2
Audit Date	April 9th, 2022

Claimed Smart Contract Features

Claimed Feature Detail	Our Observation
Tokenomics: ● Name: BLUE BIRD ● Symbol: BLB ● Decimals: 9 ● Marketing Fee: 5% ● Maximum Transaction Amount: 47 Billion ● Buy Back Upper Limit Amount: 1 Billion ● Minimum Tokens Before Swap Amount: 9.4 Million ● Total Supply: 47 Billion	**YES, This is valid.** **Owner authorized wallet can set some percentage value and we suggest handling the private key of that wallet securely.**

Audit Summary

According to the standard audit assessment, Customer`s solidity smart contracts are **"Secured"**. This token contract does contain owner control, which does not make it fully decentralized.

Insecure	Poor secured	Secure	Well-secured

You are here →

We used various tools like Slither, Solhint and Remix IDE. At the same time this finding is based on critical analysis of the manual audit.

All issues found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the Audit overview section. General overview is presented in AS-IS section and all identified issues can be found in the Audit overview section.

We found 0 critical, 0 high, 0 medium and 5 low and some very low level issues.

All these issues have been acknowledged.

Investors Advice: Technical audit of the smart contract does not guarantee the ethical nature of the project. Any owner controlled functions should be executed by the owner with responsibility. All investors/users are advised to do their due diligence before investing in the project.

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.

Email: audit@EtherAuthority.io

Technical Quick Stats

Main Category	Subcategory	Result
Contract Programming	Solidity version not specified	Passed
	Solidity version too old	Passed
	Integer overflow/underflow	Passed
	Function input parameters lack of check	Passed
	Function input parameters check bypass	Passed
	Function access control lacks management	Passed
	Critical operation lacks event log	Moderated
	Human/contract checks bypass	Passed
	Random number generation/use vulnerability	N/A
	Fallback function misuse	Passed
	Race condition	Passed
	Logical vulnerability	Passed
	Features claimed	Passed
	Other programming issues	Moderated
Code Specification	Function visibility not explicitly declared	Passed
	Var. storage location not explicitly declared	Passed
	Use keywords/functions to be deprecated	Passed
	Unused code	Passed
Gas Optimization	"Out of Gas" Issue	Passed
	High consumption 'for/while' loop	Moderated
	High consumption 'storage' storage	Passed
	Assert() misuse	Passed
Business Risk	The maximum limit for mintage not set	Passed
	"Short Address" Attack	Passed
	"Double Spend" Attack	Passed

Overall Audit Result: PASSED

Code Quality

This audit scope has 1 smart contract file. Smart contract contains Libraries, Smart contracts, inherits and Interfaces. This is a compact and well written smart contract.

The libraries in Blue bird Token are part of its logical algorithm. A library is a different type of smart contract that contains reusable code. Once deployed on the blockchain (only once), it is assigned a specific address and its properties / methods can be reused many times by other contracts in the Blue bird Token.

The Blue bird Token team has **not** provided scenario and unit test scripts, which would have helped to determine the integrity of the code in an automated way.

Code parts are **not well** commented on smart contracts.

Documentation

We were given a Blue bird Token smart contract code in the form of a BSCScan Web Link.The hash of that code is mentioned above in the table.

As mentioned above, code parts are **not well** commented. So it is not easy to quickly understand the programming flow as well as complex code logic. Comments are very helpful in understanding the overall architecture of the protocol.

Another source of information was its official website https://blb.homes which provided rich information about the project architecture and tokenomics.

Use of Dependencies

As per our observation, the libraries are used in this smart contract infrastructure that are based on well known industry standard open source projects.

Apart from libraries, its functions are used in external smart contract calls.

AS-IS overview

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	getUnlockTime	read	Passed	No Issue
7	getTime	read	Passed	No Issue
8	lock	write	access only Owner	No Issue
9	unlock	write	Regain ownership	Refer Audit Findings
10	name	read	Passed	No Issue
11	symbol	read	Passed	No Issue
12	decimals	read	Passed	No Issue
13	totalSupply	read	Passed	No Issue
14	balanceOf	read	Passed	No Issue
15	transfer	write	Passed	No Issue
16	allowance	read	Passed	No Issue
17	approve	write	Passed	No Issue
18	transferFrom	write	Passed	No Issue
19	increaseAllowance	write	Passed	No Issue
20	decreaseAllowance	write	Passed	No Issue
21	isExcludedFromReward	read	Passed	No Issue
22	totalFees	read	Passed	No Issue
23	minimumTokensBeforeSwapAmount	read	Passed	No Issue
24	buyBackUpperLimitAmount	read	Passed	No Issue
25	deliver	write	Critical operation lacks event log	Refer Audit Findings
26	reflectionFromToken	read	Passed	No Issue
27	tokenFromReflection	read	Passed	No Issue
28	excludeFromReward	write	access only Owner	No Issue
29	includeInReward	external	Infinite loops possibility	Refer Audit Findings
30	_approve	write	Passed	No Issue
31	_transfer	write	Passed	No Issue
32	swapTokens	write	Division before multiplication	Refer Audit Findings
33	buyBackTokens	write	Passed	No Issue
34	swapTokensForEth	write	Passed	No Issue
35	swapETHForTokens	write	Passed	No Issue
36	addLiquidity	write	Passed	No Issue
37	_tokenTransfer	write	Passed	No Issue
38	_transferStandard	write	Passed	No Issue

39	_transferToExcluded	write	Passed	No Issue
40	_transferFromExcluded	write	Passed	No Issue
41	_transferBothExcluded	write	Passed	No Issue
42	_reflectFee	write	Passed	No Issue
43	_getValues	read	Passed	No Issue
44	_getTValues	read	Passed	No Issue
45	_getRValues	write	Passed	No Issue
46	_getRate	read	Passed	No Issue
47	_getCurrentSupply	read	Infinite loops possibility	Refer Audit Findings
48	_takeLiquidity	write	Passed	No Issue
49	calculateTaxFee	read	Passed	No Issue
50	calculateLiquidityFee	read	Passed	No Issue
51	removeAllFee	write	Passed	No Issue
52	restoreAllFee	write	Passed	No Issue
53	isExcludedFromFee	read	Passed	No Issue
54	excludeFromFee	write	access only Owner	No Issue
55	includeInFee	write	access only Owner	No Issue
56	setTaxFee	external	Owner can set fee high	Refer Audit Findings
57	setBuybackFee	external	Owner can set fee high	Refer Audit Findings
58	setMaxTxAmount	external	access only Owner	No Issue
59	setMarketingFee	external	Owner can set fee high, Different error message for marketing fee setting	Refer Audit Findings
60	setNumTokensSellToAddToLiquidity	external	access only Owner	No Issue
61	setBuybackUpperLimit	external	access only Owner	No Issue
62	setMarketingAddress	external	access only Owner	No Issue
63	setSwapAndLiquifyEnabled	write	access only Owner	No Issue
64	setBuyBackEnabled	write	access only Owner	No Issue
65	presale	external	access only Owner	No Issue
66	transferToAddressETH	write	Passed	No Issue
67	receive	external	Passed	No Issue

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to token loss etc.
High	High-level vulnerabilities are difficult to exploit; however, they also have significant impact on smart contract execution, e.g. public access to crucial
Medium	Medium-level vulnerabilities are important to fix; however, they can't lead to tokens lose
Low	Low-level vulnerabilities are mostly related to outdated, unused etc. code snippets, that can't have significant impact on execution
Lowest / Code Style / Best Practice	Lowest-level vulnerabilities, code style violations and info statements can't affect smart contract execution and can be ignored.

Audit Findings

Critical Severity

No Critical severity vulnerabilities were found.

High Severity

No High severity vulnerabilities were found.

Medium

No Medium severity vulnerabilities were found.

Low

(1) Division before multiplication:

```
function swapTokens(uint256 contractTokenBalance) private lockTheSwap {

    uint256 initialBalance = address(this).balance;
    swapTokensForEth(contractTokenBalance);
    uint256 transferredBalance = address(this).balance.sub(initialBalance);

    //Send to Marketing address

    transferToAddressETH(dappbuilderAddress, transferredBalance.div(_liquidityFee).mul(dappbuilderFee)
    transferToAddressETH(marketingAddress, transferredBalance.div(_liquidityFee).mul(marketingFee.sub(

}
```

Solidity being resource constraint language, dividing any amount and then multiplying will cause discrepancy in the outcome. Therefore always multiply the amount first and then divide it.

Resolution: Consider ordering multiplication before division.
Status: Acknowledged

(2) Critical operation lacks event log:

Missing event log for: deliver

Resolution: Write an event log for listed events.
Status: Acknowledged

(3) Infinite loops possibility:

In below functions, for loops do not have _excluded length limit, which costs more gas:

- _getCurrentSupply
- includeInReward

Resolution: Upper limit should have a certain limit in for loops.

Status: Acknowledged

(4) Regain ownership:

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

The unlock() function can be used to regain ownership after being renounced, if lock() was previously called.

Resolution: We suggest adding this line _previousOwner= address(0) after _owner = _previousOwner;

Status: Acknowledged

(5) Owner can set fee high:

setMarketingFee, setTaxFee, setBuybackFee are used to set fees by the owner. But there is no maximum limit set hence the owner can set any number for fees.

Resolution: We suggest setting some maximum limit for fees.

Status: Acknowledged

Very Low / Informational / Best practices:

(1) Different error message for marketing fee setting:

Validation for marketing fee is done for >= 5. But the error message is not the same at both places.

Resolution: We suggest setting proper error messages for marketing fee validation.

Status: Acknowledged

Centralization

This smart contract has some functions which can be executed by the Admin (Owner) only. If the admin wallet private key would be compromised, then it would create trouble. Following are Admin functions:

- excludeFromReward: Owner can exclude account from reward
- includeInReward: Owner can include account in reward.
- excludeFromFee: Owner can exclude account from fee.
- includeInFee: Owner can include account in fee .
- setTaxFee: Owner is allowed to set tax fee.
- setBuybackFee: Owner is allowed to set buyback fee.
- setMaxTxAmount: Owner can set maximum transaction amount.
- setMarketingFee: Owner can set marketing fee.
- setNumTokensSellToAddToLiquidity: Owner can set number of tokens sold to add to liquidity swap value.
- setBuybackUpperLimit: Owner can set buyback upper limit values.
- setMarketingAddress: Owner can set marketing wallet address.
- setSwapAndLiquifyEnabled: Owner can set swap and liquify enabled status.
- setBuyBackEnabled: Owner can update the status of the buyback.
- presale: Owner can set presale status.

Conclusion

We were given a contract code. And we have used all possible tests based on given objects as files. We have not observed any major issues. So, **it's good to go to production**.

Since possible test cases can be unlimited for such smart contracts protocol, we provide no such guarantee of future outcomes. We have used all the latest static tools and manual observations to cover maximum possible test cases to scan everything.

Smart contracts within the scope were manually reviewed and analyzed with static analysis tools. Smart Contract's high-level description of functionality was presented in the As-is overview section of the report.

Audit report contains all found security vulnerabilities and other issues in the reviewed code.

Security state of the reviewed contract, based on standard audit procedure scope, is **"Secured".**

Our Methodology

We like to work with a transparent process and make our reviews a collaborative effort. The goals of our security audits are to improve the quality of systems we review and aim for sufficient remediation to help protect users. The following is the methodology we use in our security audit process.

Manual Code Review:

In manually reviewing all of the code, we look for any potential issues with code logic, error handling, protocol and header parsing, cryptographic errors, and random number generators. We also watch for areas where more defensive programming could reduce the risk of future mistakes and speed up future audits. Although our primary focus is on the in-scope code, we examine dependency code and behavior when it is relevant to a particular line of investigation.

Vulnerability Analysis:

Our audit techniques included manual code analysis, user interface interaction, and whitebox penetration testing. We look at the project's web site to get a high level understanding of what functionality the software under review provides. We then meet with the developers to gain an appreciation of their vision of the software. We install and use the relevant software, exploring the user interactions and roles. While we do this, we brainstorm threat models and attack surfaces. We read design documentation, review other audit results, search for similar projects, examine source code dependencies, skim open issue tickets, and generally investigate details other than the implementation.

Documenting Results:

We follow a conservative, transparent process for analyzing potential security vulnerabilities and seeing them through successful remediation. Whenever a potential issue is discovered, we immediately create an Issue entry for it in this document, even though we have not yet verified the feasibility and impact of the issue. This process is conservative because we document our suspicions early even if they are later shown to not represent exploitable vulnerabilities. We generally follow a process of first documenting the suspicion with unresolved questions, then confirming the issue through code analysis, live experimentation, or automated tests. Code analysis is the most tentative, and we strive to provide test code, log captures, or screenshots demonstrating our confirmation. After this we analyze the feasibility of an attack in a live system.

Suggested Solutions:

We search for immediate mitigations that live deployments can take, and finally we suggest the requirements for remediation engineering for future releases. The mitigation and remediation recommendations should be scrutinized by the developers and deployment engineers, and successful mitigation and remediation is an ongoing collaborative process after we deliver our report, and before the details are made public.

Disclaimers

EtherAuthority.io Disclaimer

EtherAuthority team has analyzed this smart contract in accordance with the best industry practices at the date of this report, in relation to: cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report, (Source Code); the Source Code compilation, deployment and functionality (performing the intended functions).

Due to the fact that the total number of test cases are unlimited, the audit makes no statements or warranties on security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bugfree status or any other statements of the contract. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only. We also suggest conducting a bug bounty program to confirm the high level of security of this smart contract.

Technical Disclaimer

Smart contracts are deployed and executed on the blockchain platform. The platform, its programming language, and other software related to the smart contract can have their own vulnerabilities that can lead to hacks. Thus, the audit can't guarantee explicit security of the audited smart contracts.

Appendix

Code Flow Diagram - Blue bird Token

BuyBackToken (C)

Context
IERC20
Ownable

SafeMath for uint256
Address for address

- address dappbuilderAddress
- address marketingAddress
- address deadAddress
- address=>uint256 _rOwned
- address=>uint256 _tOwned
- address=>mapping address=>uint256 _allowances
- address=>bool _isExcludedFromFee
- address=>bool _isExcluded
- address _excluded
- uint256 MAX
- uint256 _tTotal
- uint256 _rTotal
- uint256 _tFeeTotal
- string _name
- string _symbol
- uint8 _decimals
- uint256 _taxFee
- uint256 _previousTaxFee
- uint256 _liquidityFee
- uint256 _previousLiquidityFee
- uint256 buybackFee
- uint256 previousBuybackFee
- uint256 marketingFee
- uint256 previousMarketingFee
- uint dappbuilderFee
- uint256 _maxTxAmount
- uint256 _previousMaxTxAmount
- uint256 minimumTokensBeforeSwap
- uint256 buyBackUpperLimit
- IUniswapV2Router02 uniswapV2Router
- address uniswapV2Pair
- bool inSwapAndLiquify
- bool swapAndLiquifyEnabled
- bool buyBackEnabled

- __constructor__()
- name()
- symbol()
- decimals()
- totalSupply()
- balanceOf()
- transfer()
- allowance()
- approve()
- transferFrom()
- increaseAllowance()
- decreaseAllowance()
- isExcludedFromReward()
- totalFees()
- minimumTokensBeforeSwapAmount()
- buyBackUpperLimitAmount()
- deliver()
- reflectionFromToken()
- tokenFromReflection()
- excludeFromReward()
- includeInReward()
- _approve()
- _transfer()
- swapTokens()
- buyBackTokens()
- swapTokensForEth()
- swapETHForTokens()
- addLiquidity()
- _tokenTransfer()
- _transferStandard()
- _transferToExcluded()
- _transferFromExcluded()
- _transferBothExcluded()
- _reflectFee()
- _getValues()
- _getTValues()
- _getRValues()
- _getRate()
- _getCurrentSupply()
- _takeLiquidity()
- _calculateTaxFee()
- _calculateLiquidityFee()
- removeAllFee()
- restoreAllFee()
- isExcludedFromFee()
- excludeFromFee()
- includeInFee()
- setTaxFee()
- setBuybackFee()
- setMaxTxAmount()
- setMarketingFee()
- setNumTokensSellToAddToLiquidity()
- setBuybackUpperLimit()
- setMarketingAddress()
- setSwapAndLiquifyEnabled()
- setBuyBackEnabled()
- presale()
- transferToAddressETH()

IUniswapV2Factory (I)

- feeTo()
- feeToSetter()
- getPair()
- allPairs()
- allPairsLength()
- createPair()
- setFeeTo()
- setFeeToSetter()

IUniswapV2Pair (I)

- name()
- symbol()
- decimals()
- totalSupply()
- balanceOf()
- allowance()
- approve()
- transfer()
- transferFrom()
- DOMAIN_SEPARATOR()
- PERMIT_TYPEHASH()
- nonces()
- permit()
- MINIMUM_LIQUIDITY()
- factory()
- token0()
- token1()
- getReserves()
- price0CumulativeLast()
- price1CumulativeLast()
- kLast()
- burn()
- swap()
- skim()
- sync()
- initialize()

IUniswapV2Router02 (I)

IUniswapV2Router01

- removeLiquidityETHSupportingFeeOnTransferTokens()
- removeLiquidityETHWithPermitSupportingFeeOnTransferTokens()
- swapExactTokensForTokensSupportingFeeOnTransferTokens()
- swapExactETHForTokensSupportingFeeOnTransferTokens()
- swapExactTokensForETHSupportingFeeOnTransferTokens()

for uint256   for address

IERC20 (I)

- totalSupply()
- balanceOf()
- transfer()
- allowance()
- approve()
- transferFrom()

SafeMath (A)

- add()
- sub()
- mul()
- div()
- mod()

Address (A)

- isContract()
- sendValue()
- functionCall()
- functionCallWithValue()
- _functionCallWithValue()

Ownable (C)

Context

- address _owner
- address _previousOwner
- uint256 _lockTime

- __constructor__()
- owner()
- renounceOwnership()
- transferOwnership()
- getUnlockTime()
- getTime()
- lock()
- unlock()

IUniswapV2Router01 (I)

- factory()
- WETH()
- addLiquidity()
- addLiquidityETH()
- removeLiquidity()
- removeLiquidityETH()
- removeLiquidityWithPermit()
- removeLiquidityETHWithPermit()
- swapExactTokensForTokens()
- swapTokensForExactTokens()
- swapExactETHForTokens()
- swapTokensForExactETH()
- swapExactTokensForETH()
- swapETHForExactTokens()
- quote()
- getAmountOut()
- getAmountIn()
- getAmountsOut()
- getAmountsIn()

Context (C)

- _msgSender()
- _msgData()

Slither Results Log

Slither log >> BuyBackToken.sol

```
INFO:Detectors:
BuyBackToken.allowance(address,address).owner (BuyBackToken.sol#554) shadows:
    - Ownable.owner() (BuyBackToken.sol#161-163) (function)
BuyBackToken._approve(address,address,uint256).owner (BuyBackToken.sol#645) shadows:
    - Ownable.owner() (BuyBackToken.sol#161-163) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
BuyBackToken.constructor(string,string,uint8,uint256,uint256,uint256,uint256,address)._ma (BuyBackToken.sol#485) lacks a zer
o-check on :
        - marketingAddress = address(_ma) (BuyBackToken.sol#495)
BuyBackToken.setMarketingAddress(address)._marketingAddress (BuyBackToken.sol#946) lacks a zero-check on :
        - marketingAddress = address(_marketingAddress) (BuyBackToken.sol#947)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in BuyBackToken._transfer(address,address,uint256) (BuyBackToken.sol#653-691):
        External calls:
        - swapTokens(contractTokenBalance) (BuyBackToken.sol#671)
            - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.
timestamp) (BuyBackToken.sol#722-728)
        - buyBackTokens(balance.div(100)) (BuyBackToken.sol#679)
            - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block
.timestamp.add(300)) (BuyBackToken.sol#740-745)
        External calls sending eth:
        - swapTokens(contractTokenBalance) (BuyBackToken.sol#671)
            - recipient.transfer(amount) (BuyBackToken.sol#975)
        - buyBackTokens(balance.div(100)) (BuyBackToken.sol#679)
            - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block
.timestamp.add(300)) (BuyBackToken.sol#740-745)
        State variables written after the call(s):
        - _tokenTransfer(from,to,amount,takeFee) (BuyBackToken.sol#690)
            - _previousLiquidityFee = _liquidityFee (BuyBackToken.sol#890)
        - _tokenTransfer(from,to,amount,takeFee) (BuyBackToken.sol#690)
            - _previousTaxFee = _taxFee (BuyBackToken.sol#889)
        - _tokenTransfer(from,to,amount,takeFee) (BuyBackToken.sol#690)
            - tFeeTotal = _tFeeTotal.add(tFee) (BuyBackToken.sol#825)
        - _tokenTransfer(from,to,amount,takeFee) (BuyBackToken.sol#690)
            - _taxFee = _previousTaxFee (BuyBackToken.sol#901)
```

```
Reentrancy in BuyBackToken.transferFrom(address,address,uint256) (BuyBackToken.sol#563-567):
        External calls:
        - _transfer(sender,recipient,amount) (BuyBackToken.sol#564)
            - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block
.timestamp.add(300)) (BuyBackToken.sol#740-745)
            - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.
timestamp) (BuyBackToken.sol#722-728)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (BuyBackToken.sol#564)
            - recipient.transfer(amount) (BuyBackToken.sol#975)
            - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block
.timestamp.add(300)) (BuyBackToken.sol#740-745)
        State variables written after the call(s):
        - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance
)) (BuyBackToken.sol#565)
            - _allowances[owner][spender] = amount (BuyBackToken.sol#649)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
Reentrancy in BuyBackToken._transfer(address,address,uint256) (BuyBackToken.sol#653-691):
        External calls:
        - swapTokens(contractTokenBalance) (BuyBackToken.sol#671)
            - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.
timestamp) (BuyBackToken.sol#722-728)
        - buyBackTokens(balance.div(100)) (BuyBackToken.sol#679)
            - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block
.timestamp.add(300)) (BuyBackToken.sol#740-745)
        External calls sending eth:
        - swapTokens(contractTokenBalance) (BuyBackToken.sol#671)
            - recipient.transfer(amount) (BuyBackToken.sol#975)
        - buyBackTokens(balance.div(100)) (BuyBackToken.sol#679)
            - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block
```

```
.timestamp.add(300)) (BuyBackToken.sol#740-745)
        Event emitted after the call(s):
        - SwapETHForTokens(amount,path) (BuyBackToken.sol#747)
            - buyBackTokens(balance.div(100)) (BuyBackToken.sol#679)
        - Transfer(sender,recipient,tTransferAmount) (BuyBackToken.sol#789)
            - _tokenTransfer(from,to,amount,takeFee) (BuyBackToken.sol#690)
        - Transfer(sender,recipient,tTransferAmount) (BuyBackToken.sol#809)
            - _tokenTransfer(from,to,amount,takeFee) (BuyBackToken.sol#690)
        - Transfer(sender,recipient,tTransferAmount) (BuyBackToken.sol#799)
            - _tokenTransfer(from,to,amount,takeFee) (BuyBackToken.sol#690)
        - Transfer(sender,recipient,tTransferAmount) (BuyBackToken.sol#820)
            - _tokenTransfer(from,to,amount,takeFee) (BuyBackToken.sol#690)
Reentrancy in BuyBackToken.constructor(string,string,uint8,uint256,uint256,uint256,uint256,address) (BuyBackToken.sol#485-52
6):
        External calls:
        - uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (B
uyBackToken.sol#516-517)
        Event emitted after the call(s):
        - Transfer(address(0),_msgSender(),_tTotal) (BuyBackToken.sol#525)
Reentrancy in BuyBackToken.swapETHForTokens(uint256) (BuyBackToken.sol#733-748):
        External calls:
        - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timesta
mp.add(300)) (BuyBackToken.sol#740-745)
        Event emitted after the call(s):
        - SwapETHForTokens(amount,path) (BuyBackToken.sol#747)
Reentrancy in BuyBackToken.swapTokensForEth(uint256) (BuyBackToken.sol#713-731):
```

```
              - recipient.transfer(amount) (BuyBackToken.sol#975)
              - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block
.timestamp.add(300)) (BuyBackToken.sol#740-745)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (BuyBackToken.sol#650)
              - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds a
llowance)) (BuyBackToken.sol#565)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
Ownable.unlock() (BuyBackToken.sol#196-201) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(block.timestamp > _lockTime,Contract is locked until 7 days) (BuyBackToken.sol#198)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Address.isContract(address) (BuyBackToken.sol#91-100) uses assembly
        - INLINE ASM (BuyBackToken.sol#98)
Address._functionCallWithValue(address,bytes,uint256,string) (BuyBackToken.sol#128-145) uses assembly
        - INLINE ASM (BuyBackToken.sol#137-140)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address._functionCallWithValue(address,bytes,uint256,string) (BuyBackToken.sol#128-145) is never used and should be removed
Address.functionCall(address,bytes) (BuyBackToken.sol#111-113) is never used and should be removed
Address.functionCall(address,bytes,string) (BuyBackToken.sol#115-117) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (BuyBackToken.sol#119-121) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (BuyBackToken.sol#123-126) is never used and should be removed
Address.isContract(address) (BuyBackToken.sol#91-100) is never used and should be removed
Address.sendValue(address,uint256) (BuyBackToken.sol#102-108) is never used and should be removed
BuyBackToken.addLiquidity(uint256,uint256) (BuyBackToken.sol#750-763) is never used and should be removed
Context._msgData() (BuyBackToken.sol#14-17) is never used and should be removed
SafeMath.mod(uint256,uint256) (BuyBackToken.sol#79-81) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (BuyBackToken.sol#83-86) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
```

```
INFO:Detectors:
Pragma version^0.8.4 (BuyBackToken.sol#7) 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
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (BuyBackToken.sol#102-108):
        - (success) = recipient.call{value: amount}() (BuyBackToken.sol#106)
Low level call in Address._functionCallWithValue(address,bytes,uint256,string) (BuyBackToken.sol#128-145):
        - (success,returndata) = target.call{value: weiValue}(data) (BuyBackToken.sol#131)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (BuyBackToken.sol#236) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (BuyBackToken.sol#237) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (BuyBackToken.sol#253) is not in mixedCase
Function IUniswapV2Router01.WETH() (BuyBackToken.sol#273) is not in mixedCase
Parameter BuyBackToken.calculateTaxFee(uint256)._amount (BuyBackToken.sol#874) is not in mixedCase
Parameter BuyBackToken.calculateLiquidityFee(uint256)._amount (BuyBackToken.sol#880) is not in mixedCase
Parameter BuyBackToken.setBuybackFee(uint256)._buybackFee (BuyBackToken.sol#923) is not in mixedCase
Parameter BuyBackToken.setMarketingFee(uint256)._marketingFee (BuyBackToken.sol#932) is not in mixedCase
Parameter BuyBackToken.setNumTokensSellToAddToLiquidity(uint256)._minimumTokensBeforeSwap (BuyBackToken.sol#938) is not in m
ixedCase
Parameter BuyBackToken.setMarketingAddress(address)._marketingAddress (BuyBackToken.sol#946) is not in mixedCase
Parameter BuyBackToken.setSwapAndLiquifyEnabled(bool)._enabled (BuyBackToken.sol#950) is not in mixedCase
Parameter BuyBackToken.setBuyBackEnabled(bool)._enabled (BuyBackToken.sol#955) is not in mixedCase
Parameter BuyBackToken.presale(bool)._presale (BuyBackToken.sol#960) is not in mixedCase
Variable BuyBackToken._taxFee (BuyBackToken.sol#433) is not in mixedCase
Variable BuyBackToken._maxTxAmount (BuyBackToken.sol#447) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (BuyBackToken.sol#15)" inContext (BuyBackToken.sol#9-18)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
```

```
INFO:Detectors:
Reentrancy in BuyBackToken._transfer(address,address,uint256) (BuyBackToken.sol#653-691):
        External calls:
        - swapTokens(contractTokenBalance) (BuyBackToken.sol#671)
              - recipient.transfer(amount) (BuyBackToken.sol#975)
        External calls sending eth:
        - swapTokens(contractTokenBalance) (BuyBackToken.sol#671)
              - recipient.transfer(amount) (BuyBackToken.sol#975)
        - buyBackTokens(balance.div(100)) (BuyBackToken.sol#679)
              - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block
.timestamp.add(300)) (BuyBackToken.sol#740-745)
        State variables written after the call(s):
        - _tokenTransfer(from,to,amount,takeFee) (BuyBackToken.sol#690)
              - _liquidityFee = _previousLiquidityFee (BuyBackToken.sol#902)
              - _liquidityFee = 0 (BuyBackToken.sol#895)
        - _tokenTransfer(from,to,amount,takeFee) (BuyBackToken.sol#690)
              - _previousLiquidityFee = _liquidityFee (BuyBackToken.sol#890)
        - _tokenTransfer(from,to,amount,takeFee) (BuyBackToken.sol#690)
              - _previousTaxFee = _taxFee (BuyBackToken.sol#889)
        - _tokenTransfer(from,to,amount,takeFee) (BuyBackToken.sol#690)
              - _rOwned[address(this)] = _rOwned[address(this)].add(rLiquidity) (BuyBackToken.sol#869)
              - _rOwned[sender] = _rOwned[sender].sub(rAmount) (BuyBackToken.sol#785)
              - _rOwned[sender] = _rOwned[sender].sub(rAmount) (BuyBackToken.sol#794)
              - _rOwned[sender] = _rOwned[sender].sub(rAmount) (BuyBackToken.sol#805)
              - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (BuyBackToken.sol#786)
              - _rOwned[sender] = _rOwned[sender].sub(rAmount) (BuyBackToken.sol#815)
              - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (BuyBackToken.sol#806)
              - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (BuyBackToken.sol#796)
              - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (BuyBackToken.sol#817)
        - _tokenTransfer(from,to,amount,takeFee) (BuyBackToken.sol#690)
              - _rTotal = _rTotal.sub(rFee) (BuyBackToken.sol#824)
        - _tokenTransfer(from,to,amount,takeFee) (BuyBackToken.sol#690)
              - _tFeeTotal = _tFeeTotal.add(tFee) (BuyBackToken.sol#825)
        - _tokenTransfer(from,to,amount,takeFee) (BuyBackToken.sol#690)
              - _tOwned[address(this)] = _tOwned[address(this)].add(tLiquidity) (BuyBackToken.sol#871)
              - _tOwned[sender] = _tOwned[sender].sub(tAmount) (BuyBackToken.sol#804)
              - _tOwned[sender] = _tOwned[sender].sub(tAmount) (BuyBackToken.sol#814)
```

```
            - _tokenTransfer(from,to,amount,takeFee) (BuyBackToken.sol#690)
                - previousBuybackFee = buybackFee (BuyBackToken.sol#891)
            - _tokenTransfer(from,to,amount,takeFee) (BuyBackToken.sol#690)
                - previousMarketingFee = marketingFee (BuyBackToken.sol#892)
        Event emitted after the call(s):
        - SwapETHForTokens(amount,path) (BuyBackToken.sol#747)
            - buyBackTokens(balance.div(100)) (BuyBackToken.sol#679)
        - Transfer(sender,recipient,tTransferAmount) (BuyBackToken.sol#789)
            - _tokenTransfer(from,to,amount,takeFee) (BuyBackToken.sol#690)
        - Transfer(sender,recipient,tTransferAmount) (BuyBackToken.sol#799)
            - _tokenTransfer(from,to,amount,takeFee) (BuyBackToken.sol#690)
        - Transfer(sender,recipient,tTransferAmount) (BuyBackToken.sol#809)
            - _tokenTransfer(from,to,amount,takeFee) (BuyBackToken.sol#690)
        - Transfer(sender,recipient,tTransferAmount) (BuyBackToken.sol#820)
            - _tokenTransfer(from,to,amount,takeFee) (BuyBackToken.sol#690)
Reentrancy in BuyBackToken.transferFrom(address,address,uint256) (BuyBackToken.sol#563-567):
        External calls:
        - _transfer(sender,recipient,amount) (BuyBackToken.sol#564)
            - recipient.transfer(amount) (BuyBackToken.sol#975)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (BuyBackToken.sol#564)
            - recipient.transfer(amount) (BuyBackToken.sol#975)
            - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block
.timestamp.add(300)) (BuyBackToken.sol#740-745)
        State variables written after the call(s):
        - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance
)) (BuyBackToken.sol#565)
            - _allowances[owner][spender] = amount (BuyBackToken.sol#649)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (BuyBackToken.sol#650)
            - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds a
llowance)) (BuyBackToken.sol#565)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-4
```

```
INFO:Detectors:
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (Bu
yBackToken.sol#278) is too similar to IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,addres
s,uint256).amountBDesired (BuyBackToken.sol#279)
Variable BuyBackToken._transferFromExcluded(address,address,uint256).rTransferAmount (BuyBackToken.sol#803) is too similar t
o BuyBackToken._transferFromExcluded(address,address,uint256).tTransferAmount (BuyBackToken.sol#803)
Variable BuyBackToken._transferStandard(address,address,uint256).rTransferAmount (BuyBackToken.sol#784) is too similar to Bu
yBackToken._transferStandard(address,address,uint256).tTransferAmount (BuyBackToken.sol#784)
Variable BuyBackToken._transferFromExcluded(address,address,uint256).rTransferAmount (BuyBackToken.sol#803) is too similar t
o BuyBackToken._transferBothExcluded(address,address,uint256).tTransferAmount (BuyBackToken.sol#813)
Variable BuyBackToken.reflectionFromToken(uint256,bool).rTransferAmount (BuyBackToken.sol#611) is too similar to BuyBackToke
n._transferToExcluded(address,address,uint256).tTransferAmount (BuyBackToken.sol#793)
Variable BuyBackToken._transferBothExcluded(address,address,uint256).rTransferAmount (BuyBackToken.sol#813) is too similar t
o BuyBackToken._transferBothExcluded(address,address,uint256).tTransferAmount (BuyBackToken.sol#813)
Variable BuyBackToken._transferToExcluded(address,address,uint256).rTransferAmount (BuyBackToken.sol#793) is too similar to
BuyBackToken._transferToExcluded(address,address,uint256).tTransferAmount (BuyBackToken.sol#793)
Variable BuyBackToken._transferBothExcluded(address,address,uint256).rTransferAmount (BuyBackToken.sol#813) is too similar t
o BuyBackToken._getValues(uint256).tTransferAmount (BuyBackToken.sol#829)
Variable BuyBackToken._transferStandard(address,address,uint256).rTransferAmount (BuyBackToken.sol#784) is too similar to Bu
yBackToken._getTValues(uint256).tTransferAmount (BuyBackToken.sol#837)
Variable BuyBackToken._transferStandard(address,address,uint256).rTransferAmount (BuyBackToken.sol#784) is too similar to Bu
yBackToken._transferToExcluded(address,address,uint256).tTransferAmount (BuyBackToken.sol#793)
Variable BuyBackToken.reflectionFromToken(uint256,bool).rTransferAmount (BuyBackToken.sol#611) is too similar to BuyBackToke
n._getValues(uint256).tTransferAmount (BuyBackToken.sol#829)
Variable BuyBackToken.reflectionFromToken(uint256,bool).rTransferAmount (BuyBackToken.sol#611) is too similar to BuyBackToke
n._transferStandard(address,address,uint256).tTransferAmount (BuyBackToken.sol#784)
```

```
Variable BuyBackToken._getValues(uint256).rTransferAmount (BuyBackToken.sol#830) is too similar to BuyBackToken._transferBot
hExcluded(address,address,uint256).tTransferAmount (BuyBackToken.sol#813)
Variable BuyBackToken.reflectionFromToken(uint256,bool).rTransferAmount (BuyBackToken.sol#611) is too similar to BuyBackToke
n._getTValues(uint256).tTransferAmount (BuyBackToken.sol#837)
Variable BuyBackToken._transferBothExcluded(address,address,uint256).rTransferAmount (BuyBackToken.sol#813) is too similar t
o BuyBackToken._transferFromExcluded(address,address,uint256).tTransferAmount (BuyBackToken.sol#803)
Variable BuyBackToken._transferToExcluded(address,address,uint256).rTransferAmount (BuyBackToken.sol#793) is too similar to
BuyBackToken._transferFromExcluded(address,address,uint256).tTransferAmount (BuyBackToken.sol#803)
Variable BuyBackToken._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (BuyBackToken.sol#845) is too similar to
BuyBackToken._getValues(uint256).tTransferAmount (BuyBackToken.sol#829)
Variable BuyBackToken._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (BuyBackToken.sol#845) is too similar to
BuyBackToken._getTValues(uint256).tTransferAmount (BuyBackToken.sol#837)
Variable BuyBackToken._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (BuyBackToken.sol#845) is too similar to
BuyBackToken._transferToExcluded(address,address,uint256).tTransferAmount (BuyBackToken.sol#793)
Variable BuyBackToken._transferStandard(address,address,uint256).rTransferAmount (BuyBackToken.sol#784) is too similar to Bu
yBackToken._getValues(uint256).tTransferAmount (BuyBackToken.sol#829)
Variable BuyBackToken._transferToExcluded(address,address,uint256).rTransferAmount (BuyBackToken.sol#793) is too similar to
BuyBackToken._transferBothExcluded(address,address,uint256).tTransferAmount (BuyBackToken.sol#813)
Variable BuyBackToken._transferToExcluded(address,address,uint256).rTransferAmount (BuyBackToken.sol#793) is too similar to
BuyBackToken._transferStandard(address,address,uint256).tTransferAmount (BuyBackToken.sol#784)
Variable BuyBackToken._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (BuyBackToken.sol#845) is too similar to
BuyBackToken._transferStandard(address,address,uint256).tTransferAmount (BuyBackToken.sol#784)
Variable BuyBackToken._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (BuyBackToken.sol#845) is too similar to
BuyBackToken._transferBothExcluded(address,address,uint256).tTransferAmount (BuyBackToken.sol#813)
Variable BuyBackToken._transferStandard(address,address,uint256).rTransferAmount (BuyBackToken.sol#784) is too similar to Bu
yBackToken._transferBothExcluded(address,address,uint256).tTransferAmount (BuyBackToken.sol#813)
Variable BuyBackToken._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (BuyBackToken.sol#845) is too similar to
BuyBackToken._transferFromExcluded(address,address,uint256).tTransferAmount (BuyBackToken.sol#803)
Variable BuyBackToken._transferToExcluded(address,address,uint256).rTransferAmount (BuyBackToken.sol#793) is too similar to
BuyBackToken._getValues(uint256).tTransferAmount (BuyBackToken.sol#829)
Variable BuyBackToken.reflectionFromToken(uint256,bool).rTransferAmount (BuyBackToken.sol#611) is too similar to BuyBackToke
n._transferFromExcluded(address,address,uint256).tTransferAmount (BuyBackToken.sol#803)
Variable BuyBackToken._transferStandard(address,address,uint256).rTransferAmount (BuyBackToken.sol#784) is too similar to Bu
yBackToken._transferFromExcluded(address,address,uint256).tTransferAmount (BuyBackToken.sol#803)
Variable BuyBackToken._transferStandard(address,address,uint256).rTransferAmount (BuyBackToken.sol#793) is too similar to
BuyBackToken._getTValues(uint256).tTransferAmount (BuyBackToken.sol#837)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
```

```
INFO:Detectors:
BuyBackToken.slitherConstructorVariables() (BuyBackToken.sol#407-981) uses literals with too many digits:
        - deadAddress = 0x000000000000000000000000000000000000dEaD (BuyBackToken.sol#413)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
BuyBackToken.deadAddress (BuyBackToken.sol#413) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (BuyBackToken.sol#170-173)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (BuyBackToken.sol#175-179)
getUnlockTime() should be declared external:
        - Ownable.getUnlockTime() (BuyBackToken.sol#181-183)
getTime() should be declared external:
        - Ownable.getTime() (BuyBackToken.sol#185-187)
lock(uint256) should be declared external:
        - Ownable.lock(uint256) (BuyBackToken.sol#189-194)
unlock() should be declared external:
        - Ownable.unlock() (BuyBackToken.sol#196-201)
name() should be declared external:
        - BuyBackToken.name() (BuyBackToken.sol#528-530)
symbol() should be declared external:
        - BuyBackToken.symbol() (BuyBackToken.sol#532-534)
decimals() should be declared external:
        - BuyBackToken.decimals() (BuyBackToken.sol#536-538)
transfer(address,uint256) should be declared external:
        - BuyBackToken.transfer(address,uint256) (BuyBackToken.sol#549-552)
allowance(address,address) should be declared external:
        - BuyBackToken.allowance(address,address) (BuyBackToken.sol#554-556)
approve(address,uint256) should be declared external:
        - BuyBackToken.approve(address,uint256) (BuyBackToken.sol#558-561)
transferFrom(address,address,uint256) should be declared external:
        - BuyBackToken.transferFrom(address,address,uint256) (BuyBackToken.sol#563-567)
```

```
transferFrom(address,address,uint256) should be declared external:
        - BuyBackToken.transferFrom(address,address,uint256) (BuyBackToken.sol#563-567)
increaseAllowance(address,uint256) should be declared external:
        - BuyBackToken.increaseAllowance(address,uint256) (BuyBackToken.sol#569-572)
decreaseAllowance(address,uint256) should be declared external:
        - BuyBackToken.decreaseAllowance(address,uint256) (BuyBackToken.sol#574-577)
isExcludedFromReward(address) should be declared external:
        - BuyBackToken.isExcludedFromReward(address) (BuyBackToken.sol#579-581)
totalFees() should be declared external:
        - BuyBackToken.totalFees() (BuyBackToken.sol#583-585)
minimumTokensBeforeSwapAmount() should be declared external:
        - BuyBackToken.minimumTokensBeforeSwapAmount() (BuyBackToken.sol#587-589)
buyBackUpperLimitAmount() should be declared external:
        - BuyBackToken.buyBackUpperLimitAmount() (BuyBackToken.sol#591-593)
deliver(uint256) should be declared external:
        - BuyBackToken.deliver(uint256) (BuyBackToken.sol#595-602)
reflectionFromToken(uint256,bool) should be declared external:
        - BuyBackToken.reflectionFromToken(uint256,bool) (BuyBackToken.sol#605-614)
excludeFromReward(address) should be declared external:
        - BuyBackToken.excludeFromReward(address) (BuyBackToken.sol#622-630)
isExcludedFromFee(address) should be declared external:
        - BuyBackToken.isExcludedFromFee(address) (BuyBackToken.sol#907-909)
excludeFromFee(address) should be declared external:
        - BuyBackToken.excludeFromFee(address) (BuyBackToken.sol#911-913)
includeInFee(address) should be declared external:
        - BuyBackToken.includeInFee(address) (BuyBackToken.sol#915-917)
setBuyBackEnabled(bool) should be declared external:
        - BuyBackToken.setBuyBackEnabled(bool) (BuyBackToken.sol#955-958)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:BuyBackToken.sol analyzed (10 contracts with 75 detectors), 125 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```
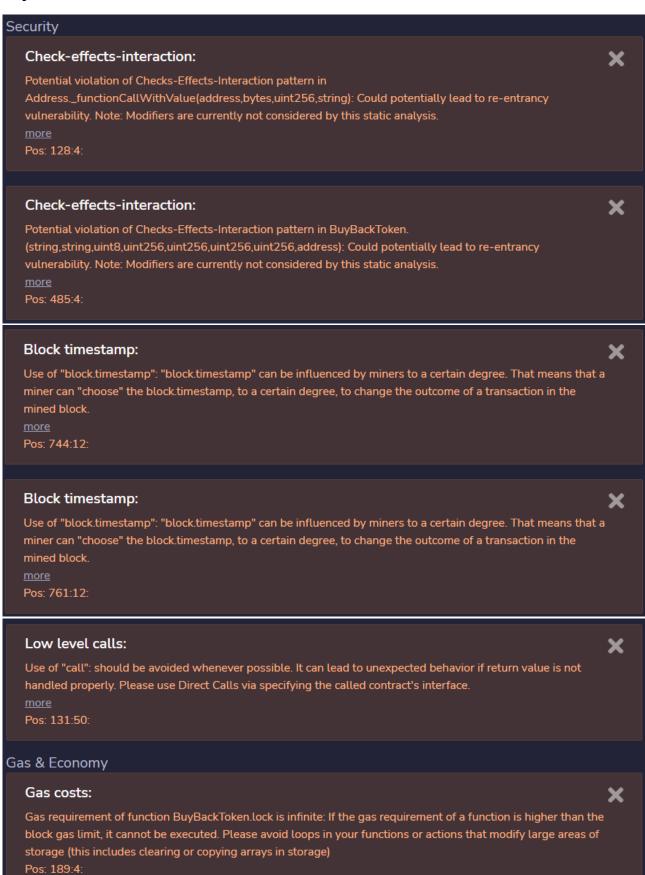
Solidity Static Analysis

BuyBackToken.sol

Security

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in Address._functionCallWithValue(address,bytes,uint256,string): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 128:4:

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in BuyBackToken. (string,string,uint8,uint256,uint256,uint256,uint256,address): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 485:4:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 744:12:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 761:12:

Low level calls:

Use of "call": should be avoided whenever possible. It can lead to unexpected behavior if return value is not handled properly. Please use Direct Calls via specifying the called contract's interface.
more
Pos: 131:50:

Gas & Economy

Gas costs:

Gas requirement of function BuyBackToken.lock is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 189:4:

Gas costs:

Gas requirement of function BuyBackToken.includeInReward is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)

Pos: 632:4:

Gas costs:

Gas requirement of function BuyBackToken.setBuybackFee is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)

Pos: 923:4:

For loop over dynamic array:

Loops that do not have a fixed number of iterations, for example, loops that depend on storage values, have to be used carefully. Due to the block gas limit, transactions can only consume a certain amount of gas. The number of iterations in a loop can grow beyond the block gas limit which can cause the complete contract to be stalled at a certain point. Additionally, using unbounded loops incurs in a lot of avoidable gas costs. Carefully test how many items at maximum you can pass to such functions to make it successful.

more

Pos: 857:8:

ERC

ERC20:

ERC20 contract's "decimals" function should have "uint8" as return type

more

Pos: 227:4:

Miscellaneous

Constant/View/Pure functions:

IERC20.transfer(address,uint256) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 25:4:

Constant/View/Pure functions:

BuyBackToken.reflectionFromToken(uint256,bool) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 605:4:

Similar variable names:

BuyBackToken._transferStandard(address,address,uint256) : Variables have very similar names "rAmount" and "tAmount". Note: Modifiers are currently not considered by this static analysis.

Pos: 784:9:

Similar variable names:

BuyBackToken._transferStandard(address,address,uint256) : Variables have very similar names "rAmount" and "tAmount". Note: Modifiers are currently not considered by this static analysis.

Pos: 784:137:

Similar variable names:

BuyBackToken._transferFromExcluded(address,address,uint256) : Variables have very similar names "_rOwned" and "_tOwned". Note: Modifiers are currently not considered by this static analysis.
Pos: 804:23:

Similar variable names:

BuyBackToken._transferFromExcluded(address,address,uint256) : Variables have very similar names "_rOwned" and "_tOwned". Note: Modifiers are currently not considered by this static analysis.
Pos: 805:8:

No return:

IUniswapV2Router02.removeLiquidityETHSupportingFeeOnTransferTokens(address,uint256,uint256,uint256,address,ui
Defines a return type but never explicitly returns a value.
Pos: 367:4:

No return:

IUniswapV2Router02.removeLiquidityETHWithPermitSupportingFeeOnTransferTokens(address,uint256,uint256,uint256
Defines a return type but never explicitly returns a value.
Pos: 375:4:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 646:8:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 647:8:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 61:16:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 73:20:

Solhint Linter

BuyBackToken.sol

```
BuyBackToken.sol:7:1: Error: Compiler version ^0.8.4 does not satisfy
the r semver requirement
BuyBackToken.sol:131:51: Error: Avoid using low level calls.
BuyBackToken.sol:137:17: Error: Avoid using inline assembly. It is
acceptable only in rare cases
BuyBackToken.sol:155:5: Error: Explicitly mark visibility in function
(Set ignoreConstructors to true if using solidity >=0.7.0)
BuyBackToken.sol:186:16: Error: Avoid to make time-based decisions in
your business logic
BuyBackToken.sol:192:21: Error: Avoid to make time-based decisions in
your business logic
BuyBackToken.sol:198:17: Error: Avoid to make time-based decisions in
your business logic
BuyBackToken.sol:236:5: Error: Function name must be in mixedCase
BuyBackToken.sol:237:5: Error: Function name must be in mixedCase
BuyBackToken.sol:253:5: Error: Function name must be in mixedCase
BuyBackToken.sol:273:5: Error: Function name must be in mixedCase
BuyBackToken.sol:407:1: Error: Contract has 30 states declarations
but allowed no more than 15
BuyBackToken.sol:455:5: Error: Explicitly mark visibility of state
BuyBackToken.sol:485:5: Error: Explicitly mark visibility in function
(Set ignoreConstructors to true if using solidity >=0.7.0)
BuyBackToken.sol:487:28: Error: Use double quotes for string literals
BuyBackToken.sol:727:13: Error: Avoid to make time-based decisions in
your business logic
BuyBackToken.sol:744:13: Error: Avoid to make time-based decisions in
your business logic
BuyBackToken.sol:761:13: Error: Avoid to make time-based decisions in
your business logic
BuyBackToken.sol:933:50: Error: Use double quotes for string literals
BuyBackToken.sol:979:32: Error: Code contains empty blocks
```

Software analysis result:

These software reported many false positive results and some are informational issues. So, those issues can be safely ignored.