Ether Authority

www.EtherAuthority.io
audit@etherauthority.io

SMART CONTRACT

Security Audit Report

Customer:	Lottree Token
Website:	https://lottreetoken.com
Platform:	Binance Smart Chain
Language:	Solidity
Date:	June 7th, 2021

Table of contents

Email: audit@EtherAuthority.io

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 PUBLIC AFTER ISSUES ARE RESOLVED.

Email: audit@EtherAuthority.io

Introduction

EtherAuthority was commissioned by LOTTREE TOKEN on June 7th, 2021 to perform a security audit of the Lottree Token smart contract.

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

LOTTREE is a tiered volume based charity lottery. The charity recipient is TREES (https://trees.org/) - the charity wallet address has been generated from the TREES crypto donation widget at https://trees.org/crypto/. The lotteries are triggered based on the volume of transactions and not on a scheduled time interval.

Audit scope

Name	**Code Review and Security Analysis Report for LOTTREE Token Smart Contract**
Platform	**BSC / Solidity**
File	LOTTree.sol
Smart Contract Online Code	https://testnet.bscscan.com/address/0x516082fb7bbEd2e037F3b26f5bb5500797A0cb0a#code
File MD5 Hash	F11FDE703F23947EA1DAD288DB02735F
Audit Date	June 7th, 2021

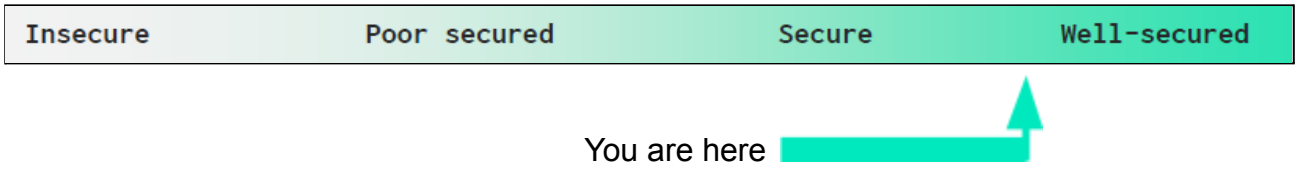
Claimed Smart Contract Features

Claimed Feature Detail	Our Observation
5% total tax on every transaction, which is further broken down as: • 2% lottery pool tax • 1% reflection tax • 1% charity tax • 1% liquidity tax	**YES, Functions Correctly. Owner can change these percentages.**
generated liquidity is automatically locked for 5 years	**YES, Functions Correctly**
contains charity address generated from charity donation widget at https://trees.org/crypto/	**YES, Functions Correctly. Owner can change this anytime.**
lottery draw triggered at 0.3% total supply accumulation	**YES, Functions Correctly. Owner can change this percentage.**
3 tier lottery based on wallet size (Top 33%, Middle 33%, Bottom 33%) • Tier 1 receives 0.14% of 0.3% • Tier 2 receives 0.09% of 0.3% • Tier 3 receives 0.05% of 0.3% • Grand Lottery pool receives 0.02% of 0.3%	**YES, Functions Correctly**

Grand Lottery draw triggered at 0.5% total supply accumulation • All wallets included in Grand Lottery, regardless of size • Grand Lottery pays out 20% every 10 days (40 day total payout)	**YES, Functions Correctly. Owner can change this percentage.**
max transaction limit of 0.5% of total supply.	**YES, Functions Correctly. Owner can change this percentage.**
contains generated 1inch address from charity donation widget	**YES, Functions Correctly. Charity wallet can be changed by the owner.**
retain control to adjust taxes and lottery draw thresholds after launch	**YES, Functions Correctly**

Audit Summary

According to the **standard** audit assessment, Customer`s solidity smart contract is **Well secured**. These contracts also have owner functions (described in the centralization section below), which does not make everything 100% decentralized. Thus, the owner must execute those smart contract functions as per the business plan.



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

We found 0 critical, 0 high, 0 medium, 5 low and some very low level issues which are described in the audit finding section. Those issues are fixed/acknowledged.

Technical Quick Stats:

Main Category	Subcategory	Result
Contract Programming	Solidity version not specified	Passed
	Solidity version too old	Moderated
	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	Passed
	Fallback function misuse	Passed
	Race condition	Passed
	Logical vulnerability	Passed
	Other programming issues	Passed
Code Specification	Function visibility not explicitly declared	Passed
	Var. storage location not explicitly declared	Passed
	Use keywords/functions to be deprecated	Passed
	Other code specification issues	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

Lottree Token smart contract has 1 smart contract. This smart contract also contains Libraries, Smart contract inherits and Interfaces. These are compact and well written contracts.

The libraries in the Lottree token protocol 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 Lottree protocol.

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

Some code parts are **not well** commented on smart contracts.

Documentation

We were given Lottree token smart contracts code in the form of a BscScan web link. The hash of that code and that web link are mentioned above in the table.

As mentioned above, some code parts are **not well** commented. so it is difficult 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://lottreetoken.com 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. And their core code blocks are written well.

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

AS-IS overview

(1) Interface

(a) IERC20I

(b) IUniswapV2Factory

(c) IUniswapV2Pair

(d) IUniswapV2Router01

(e) IUniswapV2Router02

(2) Inherited contracts

(a) Context

(b) Ownable

(c) IERC20

(3) Usages

(a) using SafeMath for uint256;

(b) using Address for address;

(4) Events

(a) event Transfer(address indexed from, address indexed to, uint256 value);

(b) event Approval(address indexed owner, address indexed spender, uint256 value);

(c) event OwnershipTransferred(address indexed previousOwner, address indexed newOwner);

(d) event PairCreated(address indexed token0, address indexed token1, address pair,uint256);

(e) event Mint(address indexed sender, uint256 amount0, uint256 amount1);

(f) event Burn(address indexed sender, uint256 amount0,uint256 amount1,address indexed to);

(g) event Swap(address indexed sender, uint256 amount0In, uint256 amount1In, uint256 amount0Out,uint256 amount1Out, address indexed to);

(h) event Sync(uint112 reserve0, uint112 reserve1);

(i) event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);

(j) event SwapAndLiquifyEnabledUpdated(bool enabled);

(k) event SwapAndLiquify(uint256 tokensSwapped,uint256 ethReceived,uint256 tokensIntoLiqudity);

(5) Functions

Sl.	Functions	Type	Observation	Conclusion
1	name	read	Passed	No Issue
2	symbol	read	Passed	No Issue
3	decimals	read	Passed	No Issue
4	totalSupply	read	Passed	No Issue
5	balanceOf	read	Passed	No Issue
6	transfer	write	Passed	No Issue
7	allowance	read	Passed	No Issue
8	approve	write	Passed	No Issue
9	transferFrom	write	Passed	No Issue
10	increaseAllowance	write	Passed	No Issue
11	decreaseAllowance	write	Passed	No Issue
12	isExcludedFromReward	read	Passed	No Issue
13	totalFees	read	Passed	No Issue
14	deliver	write	Passed	No Issue
15	reflectionFromToken	read	Passed	No Issue
16	tokenFromReflection	read	Passed	No Issue
17	excludeFromReward	write	access only Owner	No Issue
18	includeInReward	external	Infinite loop possibility	**Refer Audit Findings**
19	_transferBothExcluded	write	access only Owner	No Issue
20	excludeFromFee	write	access only Owner	No Issue
21	includeInFee	write	access only Owner	No Issue
22	setTaxFeePercent	external	access only Owner	No Issue
23	setLotteryFee	external	access only Owner	No Issue
24	setLiquidityFeePercent	external	access only Owner	No Issue
25	setMaxTxPercent	external	access only Owner	No Issue
26	setcharityFee	external	access only Owner	No Issue
27	setMinamount	external	access only Owner	No Issue
28	setMaxLotAmount	external	access only Owner	No Issue
29	setMaxgrandLotAmount	external	access only Owner	No Issue
30	setSwapAndLiquifyEnabled	write	access only Owner	No Issue

31	enableTrading	external	access only Owner	No Issue
32	_reflectFee	write	Passed	No Issue
33	_addLottery	private	Passed	No Issue
34	addUser	private	Passed	No Issue
35	addgrandLotteryPool	private	Passed	No Issue
36	_getValues	read	Passed	No Issue
37	_getTValues	read	Passed	No Issue
38	_getRValues	write	Passed	No Issue
39	_getRate	read	Passed	No Issue
40	getLotteryPool	read	Passed	No Issue
41	getGrandLotteryPool	read	Passed	No Issue
42	getWinners	read	Passed	No Issue
43	getgrandWinners	read	Passed	No Issue
44	_getCurrentSupply	read	Passed	No Issue
45	_takeLiquidity	write	Passed	No Issue
46	calculateTaxFee	read	Passed	No Issue
47	calculateLiquidityFee	read	Passed	No Issue
48	removeAllFee	write	Passed	No Issue
49	restoreAllFee	write	Passed	No Issue
50	isExcludedFromFee	read	Passed	No Issue
51	_approve	write	Passed	No Issue
52	_transfer	write	Passed	No Issue
53	swapAndLiquify	write	Passed	No Issue
54	swapTokensForEth	write	Passed	No Issue
55	addLiquidity	write	Centralized risk in addLiquidity	**Refer Audit Findings**
56	_tokenTransfer	write	Passed	No Issue
57	_transferStandard	write	Passed	No Issue
58	_transferToExcluded	write	Passed	No Issue
59	_transferFromExcluded	write	Passed	No Issue
60	random	private	Passed	No Issue
61	_lotteryEvent	write	Passed	No Issue
62	_grandFunds	private	Passed	No Issue
63	_grandlotteryEvent	write	Passed	No Issue
64	getVerifiedWinner	private	Passed	No Issue
65	getUpdatedBuckets	private	Passed	No Issue
66	addBucket	private	Passed	No Issue
67	owner	read	Passed	No Issue
68	renounceOwnership	write	access by only Owner	No Issue
69	transferOwnership	write	access by only Owner	No Issue
70	geUnlockTime	read	Passed	No Issue
71	lock	write	access by only Owner	No Issue
72	unlock	write	Ownership can be regained	**Refer Audit Findings**

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 functions
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

No critical severity vulnerabilities were found.

High

No high severity vulnerabilities were found.

Medium

No Medium severity vulnerabilities were found.

Low

(1) Possible to gain ownership after renouncing the contract ownership. Owner can renounce ownership and make contract without owner but here is a catch, owner can misuse it by performing the following operations:

- Owner calls the lock function in contract to set the current owner as _previousOwner.
- Owner calls unlock to unlock contract and sets _owner = _previousOwner.
- Owner called renounceOwnership to leave the contract without the owner.
- Owner calls unlock to regain ownership.

Solution: We advise updating/removing lock and unlock functions in the contract or call renounceOwnership function first before calling lock/unlock functions.

(2) Centralized risk in addLiquidity

```
function addLiquidity(uint256 tokenAmount, uint256 ethAmount) private {
    // approve token transfer to cover all possible scenarios
    _approve(address(this), address(uniswapV2Router), tokenAmount);

    // add the liquidity
    uniswapV2Router.addLiquidityETH{value: ethAmount}(
        address(this),
        tokenAmount,
        0, // slippage is unavoidable
        0, // slippage is unavoidable
        owner(),
        block.timestamp
    );
}
```

In addLiquidityETH function, owner() gets LP Tokens from the Pool. At some time, The owner will accumulate significant LP tokens. If the _owner is an EOA (Externally Owned Account), mishandling of its private key can have devastating consequences to the project.

Solution: uniswapV2Router.addLiquidityETH function call to be replaced by address (this), and to restrict the management of the LP tokens. This will protect the LP tokens from being stolen if the _owner account is compromised.

(3) Infinite loop possibility

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

If there are so many excluded wallets, then this logic will fail, as it might hit the block's gas limit. If there are very limited exceptions, then this will work, but will cost more gas.

Solution: We got confirmation from Lottree token team to exclude limited wallets only.

(4) Variable could be declared as constant

```
string private _name = "LOTTree";
string private _symbol = "LOTTREE";
uint8 private _decimals = 9;
```

States variables that never change need to be declared as constants.
Variables Like:

- _name
- _symbol
- _decimals
- _tTotal
- numTokensSellToAddToLiquidity

(5) Missing Events: Following state changing functions should emit an event.

- excludeFromFee
- excludeFromReward
- includeInFee
- includeInReward
- setLiquidityFeePercent
- setMaxTxPercent
- setTaxFeePercent
- setLotteryFee
- setcharityFee
- setMinamount
- setMaxLotAmount
- setMaxgrandLotAmount
- setSwapAndLiquifyEnabled
- enableTrading

Very Low / Discussion / Best practices:

(1) Solidity version

```
pragma solidity ^0.7.0;
```

Use the latest solidity version while contract deployment to prevent any compiler version level bugs.

Solution: This issue is acknowledged.

(2) Consider specifying function visibility to “external” instead of “public”, if that function is not being called internally. It will save some gas as well.
https://ethereum.stackexchange.com/questions/32353/what-is-the-difference-between-an-internal-external-and-public-private-function/32464

Centralization

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

- Owner has permission to change the owner address and receive LP Tokens.
- Owner can lock the contract.
- Owner can unlock the contract and regain the rounced ownership.
- Owner can enable/disable swapAndLiquifyEnabled.
- Owner can set Tax Fee Percent, Max Tx Percent, Liquidity fee percent, etc.
- Owner can include/exclude any wallet from reward and fees.

For better decentralized user experience, it's better to renounce the ownership if not needed, or delegate to a decentralized governance voting smart contract.

Conclusion

We were given a contract code. And we have used all possible tests based on given objects as files. We observed some issues in the smart contracts and those are fixed/acknowledged in the smart contracts. **So it is good to go for the 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 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 "**Well 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).

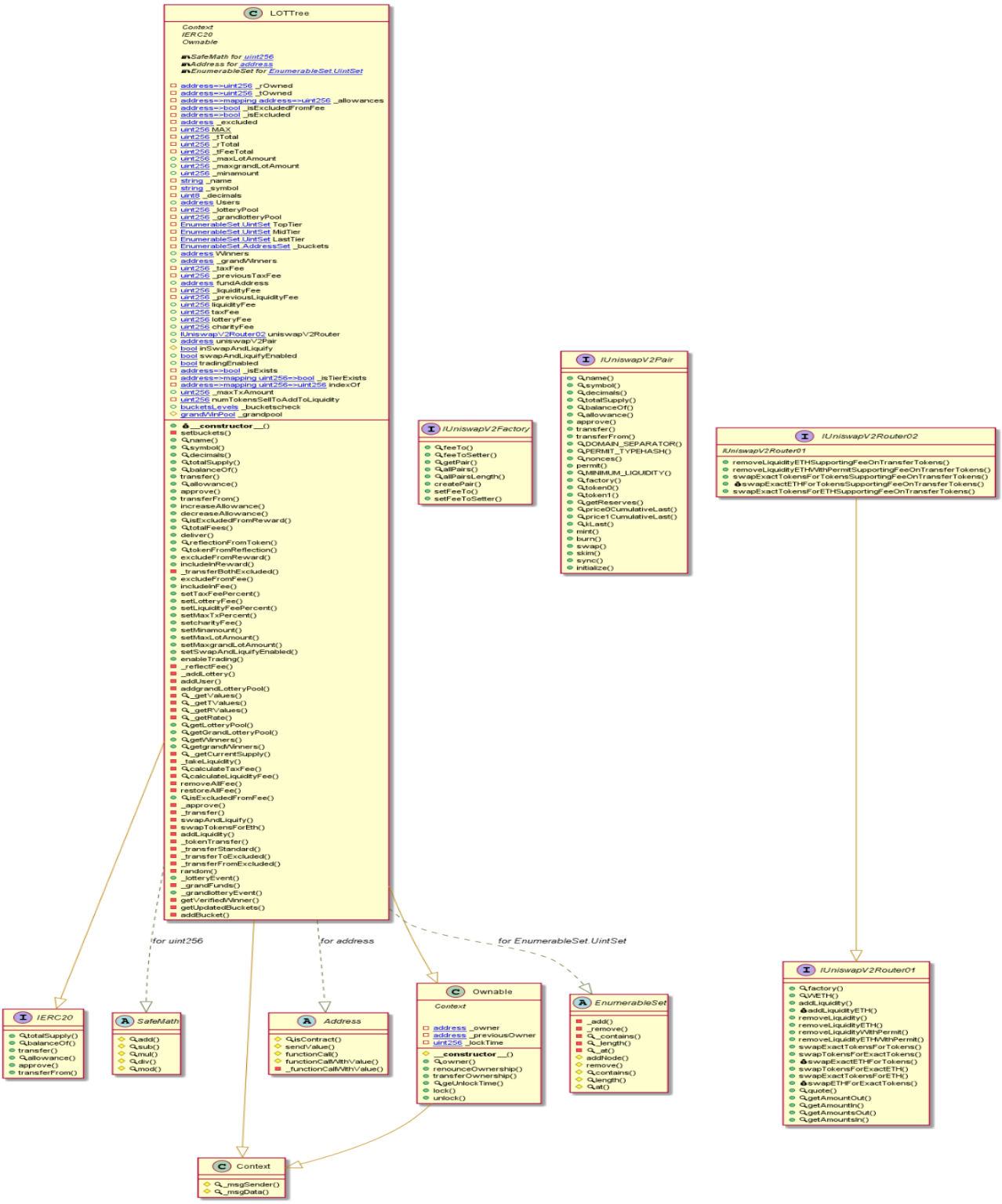
Due to the fact that the total number of test cases are unlimited, the audit makes no statements or warranties on security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bugfree status or any other statements of the contract. While we 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 - Lottree Token



Slither Results Log

Slither Log >> Lottree.sol

INFO:Detectors:
LOTTree.random(uint256) (LOTTree.sol#2116-2119) uses a weak PRNG: "randomnumber = uint256(keccak256(bytes)(abi.encodePacked(block.timestamp,block.difficulty,msg.sender))) % max (LOTTree.sol#2117)"
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#weak-PRNG
INFO:Detectors:
Reentrancy in LOTTree._transfer(address,address,uint256) (LOTTree.sol#1901-1965):
External calls:
- swapAndLiquify(contractTokenBalance) (LOTTree.sol#1941)
- uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (LOTTree.sol#2017-2024)
- uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (LOTTree.sol#2003-2009)
External calls sending eth:
- swapAndLiquify(contractTokenBalance) (LOTTree.sol#1941)
- uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (LOTTree.sol#2017-2024)
- fundAddress.transfer(newBalance.sub(toaddLiquidty)) (LOTTree.sol#1989)
State variables written after the call(s):
- _grandlotteryEvent() (LOTTree.sol#1955)
- _grandlotteryPool = 0 (LOTTree.sol#2203)
- _lotteryEvent() (LOTTree.sol#1958)
- _grandlotteryPool = _grandlotteryPool.add(amount) (LOTTree.sol#1745)
- _lotteryEvent() (LOTTree.sol#1958)
- _liquidityFee = _previousLiquidityFee (LOTTree.sol#1882)
- _liquidityFee = 0 (LOTTree.sol#1877)
- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
- _liquidityFee = _previousLiquidityFee (LOTTree.sol#1882)
- _liquidityFee = 0 (LOTTree.sol#1877)
- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
- _liquidityFee = _previousLiquidityFee (LOTTree.sol#1882)
- _liquidityFee = 0 (LOTTree.sol#1877)
- _grandlotteryEvent() (LOTTree.sol#1955)
- _lotteryPool = _lotteryPool.add(lotteryAmount) (LOTTree.sol#1726)
- _lotteryEvent() (LOTTree.sol#1958)
- _lotteryPool = _lotteryPool.add(lotteryAmount) (LOTTree.sol#1726)
- _lotteryPool = _lotteryPools.mul(_getRate()) (LOTTree.sol#2156)
- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
- _lotteryPool = _lotteryPool.add(lotteryAmount) (LOTTree.sol#1726)
- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
- _lotteryPool = _lotteryPool.add(lotteryAmount) (LOTTree.sol#1726)
- _grandlotteryEvent() (LOTTree.sol#1955)
- _rOwned[address(this)] = _rOwned[address(this)].add(rLiquidity) (LOTTree.sol#1853)
- _rOwned[address(this)] = _rOwned[address(this)].add(lotteryAmount) (LOTTree.sol#1728)
- _rOwned[sender] = _rOwned[sender].sub(rAmount) (LOTTree.sol#2064)
- _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (LOTTree.sol#2065)
- _lotteryEvent() (LOTTree.sol#1958)
- _rOwned[address(this)] = (_rOwned[address(this)]).add(amount) (LOTTree.sol#1746)
- _rOwned[address(this)] = _rOwned[address(this)].add(rLiquidity) (LOTTree.sol#1853)
- _rOwned[address(this)] = _rOwned[address(this)].add(lotteryAmount) (LOTTree.sol#1728)
- _rOwned[sender] = _rOwned[sender].sub(rAmount) (LOTTree.sol#2064)
- _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (LOTTree.sol#2065)
- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)

- _rOwned[address(this)] = _rOwned[address(this)].add(rLiquidity) (LOTTree.sol#1853)
- _rOwned[address(this)] = _rOwned[address(this)].add(lotteryAmount) (LOTTree.sol#1728)
- _rOwned[sender] = _rOwned[sender].sub(rAmount) (LOTTree.sol#2064)
- _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (LOTTree.sol#2065)

- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
- _rOwned[address(this)] = _rOwned[address(this)].add(rLiquidity) (LOTTree.sol#1853)
- _rOwned[address(this)] = _rOwned[address(this)].add(lotteryAmount) (LOTTree.sol#1728)
- _rOwned[sender] = _rOwned[sender].sub(rAmount) (LOTTree.sol#2085)
- _rOwned[sender] = _rOwned[sender].sub(rAmount) (LOTTree.sol#2064)
- _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (LOTTree.sol#2065)
- _rOwned[sender] = _rOwned[sender].sub(rAmount) (LOTTree.sol#2108)
- _rOwned[sender] = _rOwned[sender].sub(rAmount) (LOTTree.sol#1647)
- _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (LOTTree.sol#2109)
- _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (LOTTree.sol#2087)
- _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (LOTTree.sol#1649)

- _grandlotteryEvent() (LOTTree.sol#1955)
- _rTotal = _rTotal.sub(roneThird.mul(taxFee)) (LOTTree.sol#1718)

- _lotteryEvent() (LOTTree.sol#1958)
- _rTotal = _rTotal.sub(roneThird.mul(taxFee)) (LOTTree.sol#1718)

- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
- _rTotal = _rTotal.sub(roneThird.mul(taxFee)) (LOTTree.sol#1718)

- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
- _rTotal = _rTotal.sub(roneThird.mul(taxFee)) (LOTTree.sol#1718)

- _grandlotteryEvent() (LOTTree.sol#1955)
- _tOwned[address(this)] = _tOwned[address(this)].add(tLiquidity) (LOTTree.sol#1855)
- _tOwned[address(this)] = _tOwned[address(this)].add(lotteryAmount) (LOTTree.sol#1730)

- _lotteryEvent() (LOTTree.sol#1958)
- _tOwned[address(this)] = _tOwned[address(this)].add(tLiquidity) (LOTTree.sol#1855)
- _tOwned[address(this)] = (_tOwned[address(this)]).add(amount) (LOTTree.sol#1748)
- _tOwned[address(this)] = _tOwned[address(this)].add(lotteryAmount) (LOTTree.sol#1730)

- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
- _tOwned[address(this)] = _tOwned[address(this)].add(tLiquidity) (LOTTree.sol#1855)
- _tOwned[address(this)] = _tOwned[address(this)].add(lotteryAmount) (LOTTree.sol#1730)

- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
- _tOwned[address(this)] = _tOwned[address(this)].add(tLiquidity) (LOTTree.sol#1855)
- _tOwned[address(this)] = _tOwned[address(this)].add(lotteryAmount) (LOTTree.sol#1730)
- _tOwned[sender] = _tOwned[sender].sub(tAmount) (LOTTree.sol#2107)
- _tOwned[sender] = _tOwned[sender].sub(tAmount) (LOTTree.sol#1646)
- _tOwned[recipient] = _tOwned[recipient].add(tTransferAmount) (LOTTree.sol#2086)
- _tOwned[recipient] = _tOwned[recipient].add(tTransferAmount) (LOTTree.sol#1648)

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities
INFO:Detectors:
LOTTree._reflectFee(uint256,uint256) (LOTTree.sol#1714-1722) performs a multiplication on the result of a division:
-roneThird = rFee.div(_taxFee) (LOTTree.sol#1716)
-_rTotal = _rTotal.sub(roneThird.mul(taxFee)) (LOTTree.sol#1718)
LOTTree._reflectFee(uint256,uint256) (LOTTree.sol#1714-1722) performs a multiplication on the result of a division:
-toneThird = tFee.div(_taxFee) (LOTTree.sol#1717)
-_tFeeTotal = _tFeeTotal.add(toneThird.mul(taxFee)) (LOTTree.sol#1719)
LOTTree._reflectFee(uint256,uint256) (LOTTree.sol#1714-1722) performs a multiplication on the result of a division:
-roneThird = rFee.div(_taxFee) (LOTTree.sol#1716)
-_addLottery(roneThird.mul(lotteryFee)) (LOTTree.sol#1720)
LOTTree._lotteryEvent() (LOTTree.sol#2122-2159) performs a multiplication on the result of a division:
-addgrandLotteryPool((_lotteryPools.mul(8).div(100)).mul(_getRate())) (LOTTree.sol#2153)
LOTTree.getUpdatedBuckets() (LOTTree.sol#2235-2273) performs a multiplication on the result of a division:
-(sum.div(Users.length)).mul(100) <= 33 && low && j <= 33 (LOTTree.sol#2246)
LOTTree.getUpdatedBuckets() (LOTTree.sol#2235-2273) performs a multiplication on the result of a division:
-(sum.div(Users.length)).mul(100) <= 33 && middle && j <= 66 (LOTTree.sol#2256)
LOTTree.getUpdatedBuckets() (LOTTree.sol#2235-2273) performs a multiplication on the result of a division:
-(sum.div(Users.length)).mul(100) >= 33 || j >= 66 (LOTTree.sol#2260)

LOTTree.getUpdatedBuckets() (LOTTree.sol#2235-2273) performs a multiplication on the result of a division:
-(sum.div(Users.length)).mul(100) >= 33 || j >= 33 (LOTTree.sol#2250)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-multiply
INFO:Detectors:
LOTTree.addBucket(address).eligibleBucket (LOTTree.sol#2277) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
LOTTree.addLiquidity(uint256,uint256) (LOTTree.sol#2012-2025) ignores return value by uniswapV2Router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (LOTTree.sol#2017-2024)
LOTTree.getUpdatedBuckets() (LOTTree.sol#2235-2273) ignores return value by LastTier.addNode(j) (LOTTree.sol#2247)
LOTTree.getUpdatedBuckets() (LOTTree.sol#2235-2273) ignores return value by TopTier.remove(j) (LOTTree.sol#2248)
LOTTree.getUpdatedBuckets() (LOTTree.sol#2235-2273) ignores return value by MidTier.remove(j) (LOTTree.sol#2249)
LOTTree.getUpdatedBuckets() (LOTTree.sol#2235-2273) ignores return value by MidTier.addNode(j) (LOTTree.sol#2257)
LOTTree.getUpdatedBuckets() (LOTTree.sol#2235-2273) ignores return value by TopTier.remove(j) (LOTTree.sol#2258)
LOTTree.getUpdatedBuckets() (LOTTree.sol#2235-2273) ignores return value by LastTier.remove(j) (LOTTree.sol#2259)
LOTTree.getUpdatedBuckets() (LOTTree.sol#2235-2273) ignores return value by TopTier.addNode(j) (LOTTree.sol#2266)
LOTTree.getUpdatedBuckets() (LOTTree.sol#2235-2273) ignores return value by MidTier.remove(j) (LOTTree.sol#2267)
LOTTree.getUpdatedBuckets() (LOTTree.sol#2235-2273) ignores return value by LastTier.remove(j) (LOTTree.sol#2268)
LOTTree.addBucket(address) (LOTTree.sol#2275-2293) ignores return value by _buckets[j].addNode(user) (LOTTree.sol#2281)
LOTTree.addBucket(address) (LOTTree.sol#2275-2293) ignores return value by _buckets[j].remove(user) (LOTTree.sol#2285)
LOTTree.addBucket(address) (LOTTree.sol#2275-2293) ignores return value by _buckets[j].remove(user) (LOTTree.sol#2288)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
INFO:Detectors:
LOTTree.allowance(address,address).owner (LOTTree.sol#1499) shadows:
- Ownable.owner() (LOTTree.sol#451-453) (function)
LOTTree._approve(address,address,uint256).owner (LOTTree.sol#1890) shadows:
- Ownable.owner() (LOTTree.sol#451-453) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
Reentrancy in LOTTree._transfer(address,address,uint256) (LOTTree.sol#1901-1965):
External calls:
- swapAndLiquify(contractTokenBalance) (LOTTree.sol#1941)
- uniswapV2Router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (LOTTree.sol#2017-2024)
- uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (LOTTree.sol#2003-2009)
External calls sending eth:
- swapAndLiquify(contractTokenBalance) (LOTTree.sol#1941)
- uniswapV2Router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (LOTTree.sol#2017-2024)
- fundAddress.transfer(newBalance.sub(toaddLiquidty)) (LOTTree.sol#1989)
State variables written after the call(s):
- _grandlotteryEvent() (LOTTree.sol#1955)
- Users.push(user) (LOTTree.sol#1736)
- _lotteryEvent() (LOTTree.sol#1958)
- Users.push(user) (LOTTree.sol#1736)
- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
- Users.push(user) (LOTTree.sol#1736)
- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
- Users.push(user) (LOTTree.sol#1736)

- _lotteryEvent() (LOTTree.sol#1958)
  - Winners.push(top) (LOTTree.sol#2136)
  - Winners.push(mid) (LOTTree.sol#2142)
  - Winners.push(last) (LOTTree.sol#2148)
- _grandlotteryEvent() (LOTTree.sol#1955)
  - _grandWinners.push(Users[index]) (LOTTree.sol#2202)
- _grandlotteryEvent() (LOTTree.sol#1955)
  - _grandpool.push(grandWinPool(Users[index],grandPoolamount,block.timestamp,grandPoolamount.sub(toTransfer))) (LOTTree.sol#2194-2201)
- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
  - _grandpool[_index].balance = (_grandpool[_index].amount).sub(toTransfer) (LOTTree.sol#2173-2175)
  - _grandpool[_index].lastReward = block.timestamp (LOTTree.sol#2176)
- _grandlotteryEvent() (LOTTree.sol#1955)
  - _isExists[user] = true (LOTTree.sol#1737)
- _lotteryEvent() (LOTTree.sol#1958)
  - _isExists[user] = true (LOTTree.sol#1737)
- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
  - _isExists[user] = true (LOTTree.sol#1737)
- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
  - _isExists[user] = true (LOTTree.sol#1737)
- _lotteryEvent() (LOTTree.sol#1958)
  - _previousLiquidityFee = _liquidityFee (LOTTree.sol#1874)
- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
  - _previousLiquidityFee = _liquidityFee (LOTTree.sol#1874)
- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
  - _previousLiquidityFee = _liquidityFee (LOTTree.sol#1874)
- _lotteryEvent() (LOTTree.sol#1958)
  - _previousTaxFee = _taxFee (LOTTree.sol#1873)
- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
  - _previousTaxFee = _taxFee (LOTTree.sol#1873)
- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
  - _previousTaxFee = _taxFee (LOTTree.sol#1873)
- _grandlotteryEvent() (LOTTree.sol#1955)
  - _tFeeTotal = _tFeeTotal.add(toneThird.mul(taxFee)) (LOTTree.sol#1719)
- _lotteryEvent() (LOTTree.sol#1958)
  - _tFeeTotal = _tFeeTotal.add(toneThird.mul(taxFee)) (LOTTree.sol#1719)
- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
  - _tFeeTotal = _tFeeTotal.add(toneThird.mul(taxFee)) (LOTTree.sol#1719)
- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
  - _tFeeTotal = _tFeeTotal.add(toneThird.mul(taxFee)) (LOTTree.sol#1719)
- _lotteryEvent() (LOTTree.sol#1958)
  - _taxFee = _previousTaxFee (LOTTree.sol#1881)
  - _taxFee = 0 (LOTTree.sol#1876)
- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
  - _taxFee = _previousTaxFee (LOTTree.sol#1881)
  - _taxFee = 0 (LOTTree.sol#1876)
- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
  - _taxFee = _previousTaxFee (LOTTree.sol#1881)
  - _taxFee = 0 (LOTTree.sol#1876)

Reentrancy in LOTTree.constructor() (LOTTree.sol#1223-1433):

External calls:

- uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (LOTTree.sol#1421-1422)

State variables written after the call(s):

- setbuckets() (LOTTree.sol#1430)
  - _bucketscheck.push(bucketsLevels(min,val)) (LOTTree.sol#1460-1465)
- _isExcludedFromFee[owner()] = true (LOTTree.sol#1428)
- _isExcludedFromFee[address(this)] = true (LOTTree.sol#1429)
- uniswapV2Router = _uniswapV2Router (LOTTree.sol#1425)

Reentrancy in LOTTree.swapAndLiquify(uint256) (LOTTree.sol#1967-1992):

External calls:

- swapTokensForEth(half.add(forCharity)) (LOTTree.sol#1981)

- uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (LOTTree.sol#2003-2009)
- addLiquidity(otherHalf,toaddLiquidty) (LOTTree.sol#1988)
- uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (LOTTree.sol#2017-2024)

External calls sending eth:
- addLiquidity(otherHalf,toaddLiquidty) (LOTTree.sol#1988)
- uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (LOTTree.sol#2017-2024)

State variables written after the call(s):
- addLiquidity(otherHalf,toaddLiquidty) (LOTTree.sol#1988)
  - _allowances[owner][spender] = amount (LOTTree.sol#1897)

Reentrancy in LOTTree.transferFrom(address,address,uint256) (LOTTree.sol#1517-1532):

External calls:
- _transfer(sender,recipient,amount) (LOTTree.sol#1522)
- uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (LOTTree.sol#2017-2024)
- uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (LOTTree.sol#2003-2009)

External calls sending eth:
- _transfer(sender,recipient,amount) (LOTTree.sol#1522)
- uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (LOTTree.sol#2017-2024)
  - fundAddress.transfer(newBalance.sub(toaddLiquidty)) (LOTTree.sol#1989)

State variables written after the call(s):
- _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (LOTTree.sol#1523-1530)
  - _allowances[owner][spender] = amount (LOTTree.sol#1897)

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2

INFO:Detectors:

Reentrancy in LOTTree._transfer(address,address,uint256) (LOTTree.sol#1901-1965):

External calls:
- swapAndLiquify(contractTokenBalance) (LOTTree.sol#1941)
- uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (LOTTree.sol#2017-2024)
- uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (LOTTree.sol#2003-2009)

External calls sending eth:
- swapAndLiquify(contractTokenBalance) (LOTTree.sol#1941)
- uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (LOTTree.sol#2017-2024)
  - fundAddress.transfer(newBalance.sub(toaddLiquidty)) (LOTTree.sol#1989)

Event emitted after the call(s):
- Transfer(sender,recipient,tTransferAmount) (LOTTree.sol#2069)
  - _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
- Transfer(sender,recipient,tTransferAmount) (LOTTree.sol#2069)
  - _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
- Transfer(sender,recipient,tTransferAmount) (LOTTree.sol#2069)
  - _lotteryEvent() (LOTTree.sol#1958)
- Transfer(sender,recipient,tTransferAmount) (LOTTree.sol#2069)
  - _grandlotteryEvent() (LOTTree.sol#1955)
- Transfer(sender,recipient,tTransferAmount) (LOTTree.sol#1652)
  - _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
- Transfer(sender,recipient,tTransferAmount) (LOTTree.sol#2113)
  - _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
- Transfer(sender,recipient,tTransferAmount) (LOTTree.sol#2091)
  - _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)

Reentrancy in LOTTree.constructor() (LOTTree.sol#1223-1433):

External calls:
- uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (LOTTree.sol#1421-1422)

Event emitted after the call(s):
- Transfer(address(0),_msgSender(),_tTotal) (LOTTree.sol#1432)
Reentrancy in LOTTree.swapAndLiquify(uint256) (LOTTree.sol#1967-1992):
External calls:
- swapTokensForEth(half.add(forCharity)) (LOTTree.sol#1981)
- uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (LOTTree.sol#2003-2009)
- addLiquidity(otherHalf,toaddLiquidty) (LOTTree.sol#1988)
- uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (LOTTree.sol#2017-2024)
External calls sending eth:
- addLiquidity(otherHalf,toaddLiquidty) (LOTTree.sol#1988)
- uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (LOTTree.sol#2017-2024)
Event emitted after the call(s):
- Approval(owner,spender,amount) (LOTTree.sol#1898)
- addLiquidity(otherHalf,toaddLiquidty) (LOTTree.sol#1988)
Reentrancy in LOTTree.swapAndLiquify(uint256) (LOTTree.sol#1967-1992):
External calls:
- swapTokensForEth(half.add(forCharity)) (LOTTree.sol#1981)
- uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (LOTTree.sol#2003-2009)
- addLiquidity(otherHalf,toaddLiquidty) (LOTTree.sol#1988)
- uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (LOTTree.sol#2017-2024)
External calls sending eth:
- addLiquidity(otherHalf,toaddLiquidty) (LOTTree.sol#1988)
- uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (LOTTree.sol#2017-2024)
- fundAddress.transfer(newBalance.sub(toaddLiquidty)) (LOTTree.sol#1989)
Event emitted after the call(s):
- SwapAndLiquify(half,toaddLiquidty,otherHalf) (LOTTree.sol#1991)
Reentrancy in LOTTree.transferFrom(address,address,uint256) (LOTTree.sol#1517-1532):
External calls:
- _transfer(sender,recipient,amount) (LOTTree.sol#1522)
- uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (LOTTree.sol#2017-2024)
- uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (LOTTree.sol#2003-2009)
External calls sending eth:
- _transfer(sender,recipient,amount) (LOTTree.sol#1522)
- uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (LOTTree.sol#2017-2024)
- fundAddress.transfer(newBalance.sub(toaddLiquidty)) (LOTTree.sol#1989)
Event emitted after the call(s):
- Approval(owner,spender,amount) (LOTTree.sol#1898)
- _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (LOTTree.sol#1523-1530)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
Ownable.unlock() (LOTTree.sol#501-509) uses timestamp for comparisons
Dangerous comparisons:
- require(bool,string)(now > _lockTime,Contract is locked until 7 days) (LOTTree.sol#506)
LOTTree.addUser(address) (LOTTree.sol#1734-1740) uses timestamp for comparisons
Dangerous comparisons:
- ! _isExists[user] && user != owner() && balanceOf(user) > _minamount (LOTTree.sol#1735)
LOTTree._lotteryEvent() (LOTTree.sol#2122-2159) uses timestamp for comparisons
Dangerous comparisons:
- require(bool,string)(Users.length >= 3,Only Minimum Users Found) (LOTTree.sol#2125)
- top != address(0) (LOTTree.sol#2135)
- mid != address(0) (LOTTree.sol#2141)
- last != address(0) (LOTTree.sol#2147)

LOTTree._grandFunds(uint256) (LOTTree.sol#2161-2179) uses timestamp for comparisons
Dangerous comparisons:
- _grandpool[_index].balance > 0 && block.timestamp > (_grandpool[_index].lastReward + 777600) (LOTTree.sol#2163-2164)
LOTTree._grandlotteryEvent() (LOTTree.sol#2181-2204) uses timestamp for comparisons
Dangerous comparisons:
- require(bool,string)(Users.length >= 3,Only Minimum Users Found) (LOTTree.sol#2183)
- require(bool,string)(_grandpool[_grandpool.length - 1].balance == 0,Grand Lottery is Already Running) (LOTTree.sol#2185)
LOTTree.addBucket(address) (LOTTree.sol#2275-2293) uses timestamp for comparisons
Dangerous comparisons:
- isAdded && _buckets[j].contains(user) && j != eligibleBucket (LOTTree.sol#2287)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Address.isContract(address) (LOTTree.sol#272-284) uses assembly
- INLINE ASM (LOTTree.sol#280-282)
Address._functionCallWithValue(address,bytes,uint256,string) (LOTTree.sol#399-426) uses assembly
- INLINE ASM (LOTTree.sol#418-421)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address._functionCallWithValue(address,bytes,uint256,string) (LOTTree.sol#399-426) is never used and should be removed
Address.functionCall(address,bytes) (LOTTree.sol#334-339) is never used and should be removed
Address.functionCall(address,bytes,string) (LOTTree.sol#347-353) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (LOTTree.sol#366-378) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (LOTTree.sol#386-397) is never used and should be removed
Address.isContract(address) (LOTTree.sol#272-284) is never used and should be removed
Address.sendValue(address,uint256) (LOTTree.sol#302-314) is never used and should be removed
Context._msgData() (LOTTree.sol#248-251) is never used and should be removed
EnumerableSet.addNode(EnumerableSet.Bytes32Set,bytes32) (LOTTree.sol#974-976) is never used and should be removed
EnumerableSet.at(EnumerableSet.Bytes32Set,uint256) (LOTTree.sol#1012-1014) is never used and should be removed
EnumerableSet.contains(EnumerableSet.Bytes32Set,bytes32) (LOTTree.sol#991-993) is never used and should be removed
EnumerableSet.contains(EnumerableSet.UintSet,uint256) (LOTTree.sol#1100-1102) is never used and should be removed
EnumerableSet.length(EnumerableSet.Bytes32Set) (LOTTree.sol#998-1000) is never used and should be removed
EnumerableSet.remove(EnumerableSet.Bytes32Set,bytes32) (LOTTree.sol#984-986) is never used and should be removed
SafeMath.mod(uint256,uint256) (LOTTree.sol#217-219) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (LOTTree.sol#233-240) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
LOTTree._rTotal (LOTTree.sol#1144) is set pre-construction with a non-constant function or state variable:
- (MAX - (MAX % _tTotal))
LOTTree._previousTaxFee (LOTTree.sol#1168) is set pre-construction with a non-constant function or state variable:
- _taxFee
LOTTree._previousLiquidityFee (LOTTree.sol#1172) 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-variables
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (LOTTree.sol#302-314):
- (success) = recipient.call{value: amount}() (LOTTree.sol#309)
Low level call in Address._functionCallWithValue(address,bytes,uint256,string) (LOTTree.sol#399-426):
- (success,returndata) = target.call{value: weiValue}(data) (LOTTree.sol#408-409)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

INFO:Detectors:
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (LOTTree.sol#575) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (LOTTree.sol#577) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (LOTTree.sol#608) is not in mixedCase
Function IUniswapV2Router01.WETH() (LOTTree.sol#654) is not in mixedCase
Struct LOTTree.grandWinPool (LOTTree.sol#1193-1198) is not in CapWords
Struct LOTTree.bucketsLevels (LOTTree.sol#1200-1203) is not in CapWords
Parameter LOTTree.setMaxLotAmount(uint256).LotAmount (LOTTree.sol#1695) is not in mixedCase
Parameter LOTTree.setSwapAndLiquifyEnabled(bool)._enabled (LOTTree.sol#1703) is not in mixedCase
Parameter LOTTree.calculateTaxFee(uint256)._amount (LOTTree.sol#1858) is not in mixedCase
Parameter LOTTree.calculateLiquidityFee(uint256)._amount (LOTTree.sol#1862) is not in mixedCase
Function LOTTree._lotteryEvent() (LOTTree.sol#2122-2159) is not in mixedCase
Function LOTTree._grandlotteryEvent() (LOTTree.sol#2181-2204) is not in mixedCase
Parameter LOTTree.getVerifiedWinner(uint256,uint256).Tier (LOTTree.sol#2206) is not in mixedCase
Variable LOTTree._maxLotAmount (LOTTree.sol#1147) is not in mixedCase
Variable LOTTree._maxgrandLotAmount (LOTTree.sol#1149) is not in mixedCase
Variable LOTTree._minamount (LOTTree.sol#1151) is not in mixedCase
Variable LOTTree.Users (LOTTree.sol#1157) is not in mixedCase
Variable LOTTree.TopTier (LOTTree.sol#1161) is not in mixedCase
Variable LOTTree.MidTier (LOTTree.sol#1162) is not in mixedCase
Variable LOTTree.LastTier (LOTTree.sol#1163) is not in mixedCase
Variable LOTTree.Winners (LOTTree.sol#1165) is not in mixedCase
Variable LOTTree._grandWinners (LOTTree.sol#1166) is not in mixedCase
Variable LOTTree._maxTxAmount (LOTTree.sol#1190) is not in mixedCase
Variable LOTTree._bucketscheck (LOTTree.sol#1205) is not in mixedCase
Variable LOTTree._grandpool (LOTTree.sol#1207) is not in mixedCase
Reference:
https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (LOTTree.sol#249)" inContext (LOTTree.sol#243-252)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
Reentrancy in LOTTree._transfer(address,address,uint256) (LOTTree.sol#1901-1965):
External calls:
- swapAndLiquify(contractTokenBalance) (LOTTree.sol#1941)
- fundAddress.transfer(newBalance.sub(toaddLiquidty)) (LOTTree.sol#1989)
External calls sending eth:
- swapAndLiquify(contractTokenBalance) (LOTTree.sol#1941)
- uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (LOTTree.sol#2017-2024)
- fundAddress.transfer(newBalance.sub(toaddLiquidty)) (LOTTree.sol#1989)
State variables written after the call(s):
- _grandlotteryEvent() (LOTTree.sol#1955)
- Users.push(user) (LOTTree.sol#1736)
- _lotteryEvent() (LOTTree.sol#1958)
- Users.push(user) (LOTTree.sol#1736)
- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
- Users.push(user) (LOTTree.sol#1736)
- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
- Users.push(user) (LOTTree.sol#1736)
- _lotteryEvent() (LOTTree.sol#1958)
- Winners.push(top) (LOTTree.sol#2136)
- Winners.push(mid) (LOTTree.sol#2142)
- Winners.push(last) (LOTTree.sol#2148)
- _grandlotteryEvent() (LOTTree.sol#1955)
- _grandWinners.push(Users[index]) (LOTTree.sol#2202)
- _grandlotteryEvent() (LOTTree.sol#1955)
- _grandlotteryPool = 0 (LOTTree.sol#2203)
- _lotteryEvent() (LOTTree.sol#1958)
- _grandlotteryPool = _grandlotteryPool.add(amount) (LOTTree.sol#1745)
- _grandlotteryEvent() (LOTTree.sol#1955)
- _grandpool.push(grandWinPool(Users[index],grandPoolamount,block.timestamp,grandPoolamount.sub

(toTransfer))) (LOTTree.sol#2194-2201)
- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
- _grandpool[_index].balance = (_grandpool[_index].amount).sub(toTransfer) (LOTTree.sol#2173-2175)
- _grandpool[_index].lastReward = block.timestamp (LOTTree.sol#2176)
- _grandlotteryEvent() (LOTTree.sol#1955)
- _isExists[user] = true (LOTTree.sol#1737)
- _lotteryEvent() (LOTTree.sol#1958)
- _isExists[user] = true (LOTTree.sol#1737)
- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
- _isExists[user] = true (LOTTree.sol#1737)
- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
- _isExists[user] = true (LOTTree.sol#1737)
- _lotteryEvent() (LOTTree.sol#1958)
- _liquidityFee = _previousLiquidityFee (LOTTree.sol#1882)
- _liquidityFee = 0 (LOTTree.sol#1877)
- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
- _liquidityFee = _previousLiquidityFee (LOTTree.sol#1882)
- _liquidityFee = 0 (LOTTree.sol#1877)
- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
- _liquidityFee = _previousLiquidityFee (LOTTree.sol#1882)
- _liquidityFee = 0 (LOTTree.sol#1877)
- _grandlotteryEvent() (LOTTree.sol#1955)
- _lotteryPool = _lotteryPool.add(lotteryAmount) (LOTTree.sol#1726)
- _lotteryEvent() (LOTTree.sol#1958)
- _lotteryPool = _lotteryPool.add(lotteryAmount) (LOTTree.sol#1726)
- _lotteryPool = _lotteryPools.mul(_getRate()) (LOTTree.sol#2156)
- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
- _lotteryPool = _lotteryPool.add(lotteryAmount) (LOTTree.sol#1726)
- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
- _lotteryPool = _lotteryPool.add(lotteryAmount) (LOTTree.sol#1726)
- _lotteryEvent() (LOTTree.sol#1958)
- _previousLiquidityFee = _liquidityFee (LOTTree.sol#1874)
- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
- _previousLiquidityFee = _liquidityFee (LOTTree.sol#1874)
- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
- _previousLiquidityFee = _liquidityFee (LOTTree.sol#1874)
- _lotteryEvent() (LOTTree.sol#1958)
- _previousTaxFee = _taxFee (LOTTree.sol#1873)
- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
- _previousTaxFee = _taxFee (LOTTree.sol#1873)
- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
- _previousTaxFee = _taxFee (LOTTree.sol#1873)
- _grandlotteryEvent() (LOTTree.sol#1955)
- _rOwned[address(this)] = _rOwned[address(this)].add(rLiquidity) (LOTTree.sol#1853)
- _rOwned[address(this)] = _rOwned[address(this)].add(lotteryAmount) (LOTTree.sol#1728)
- _rOwned[sender] = _rOwned[sender].sub(rAmount) (LOTTree.sol#2064)
- _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (LOTTree.sol#2065)
- _lotteryEvent() (LOTTree.sol#1958)
- _rOwned[address(this)] = (_rOwned[address(this)]).add(amount) (LOTTree.sol#1746)
- _rOwned[address(this)] = _rOwned[address(this)].add(rLiquidity) (LOTTree.sol#1853)
- _rOwned[address(this)] = _rOwned[address(this)].add(lotteryAmount) (LOTTree.sol#1728)
- _rOwned[sender] = _rOwned[sender].sub(rAmount) (LOTTree.sol#2064)
- _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (LOTTree.sol#2065)
- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
- _rOwned[address(this)] = _rOwned[address(this)].add(rLiquidity) (LOTTree.sol#1853)
- _rOwned[address(this)] = _rOwned[address(this)].add(lotteryAmount) (LOTTree.sol#1728)
- _rOwned[sender] = _rOwned[sender].sub(rAmount) (LOTTree.sol#2064)
- _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (LOTTree.sol#2065)
- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
- _rOwned[address(this)] = _rOwned[address(this)].add(rLiquidity) (LOTTree.sol#1853)
- _rOwned[address(this)] = _rOwned[address(this)].add(lotteryAmount) (LOTTree.sol#1728)
- _rOwned[sender] = _rOwned[sender].sub(rAmount) (LOTTree.sol#2085)
- _rOwned[sender] = _rOwned[sender].sub(rAmount) (LOTTree.sol#2064)
- _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (LOTTree.sol#2065)

- _rOwned[sender] = _rOwned[sender].sub(rAmount) (LOTTree.sol#2108)
- _rOwned[sender] = _rOwned[sender].sub(rAmount) (LOTTree.sol#1647)
- _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (LOTTree.sol#2109)
- _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (LOTTree.sol#2087)
- _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (LOTTree.sol#1649)

- _grandlotteryEvent() (LOTTree.sol#1955)
- _rTotal = _rTotal.sub(roneThird.mul(taxFee)) (LOTTree.sol#1718)

- _lotteryEvent() (LOTTree.sol#1958)
- _rTotal = _rTotal.sub(roneThird.mul(taxFee)) (LOTTree.sol#1718)

- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
- _rTotal = _rTotal.sub(roneThird.mul(taxFee)) (LOTTree.sol#1718)

- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
- _rTotal = _rTotal.sub(roneThird.mul(taxFee)) (LOTTree.sol#1718)

- _grandlotteryEvent() (LOTTree.sol#1955)
- _tFeeTotal = _tFeeTotal.add(toneThird.mul(taxFee)) (LOTTree.sol#1719)

- _lotteryEvent() (LOTTree.sol#1958)
- _tFeeTotal = _tFeeTotal.add(toneThird.mul(taxFee)) (LOTTree.sol#1719)

- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
- _tFeeTotal = _tFeeTotal.add(toneThird.mul(taxFee)) (LOTTree.sol#1719)

- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
- _tFeeTotal = _tFeeTotal.add(toneThird.mul(taxFee)) (LOTTree.sol#1719)

- _grandlotteryEvent() (LOTTree.sol#1955)
- _tOwned[address(this)] = _tOwned[address(this)].add(tLiquidity) (LOTTree.sol#1855)
- _tOwned[address(this)] = _tOwned[address(this)].add(lotteryAmount) (LOTTree.sol#1730)

- _lotteryEvent() (LOTTree.sol#1958)
- _tOwned[address(this)] = _tOwned[address(this)].add(tLiquidity) (LOTTree.sol#1855)
- _tOwned[address(this)] = (_tOwned[address(this)]).add(amount) (LOTTree.sol#1748)
- _tOwned[address(this)] = _tOwned[address(this)].add(lotteryAmount) (LOTTree.sol#1730)

- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
- _tOwned[address(this)] = _tOwned[address(this)].add(tLiquidity) (LOTTree.sol#1855)
- _tOwned[address(this)] = _tOwned[address(this)].add(lotteryAmount) (LOTTree.sol#1730)

- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
- _tOwned[address(this)] = _tOwned[address(this)].add(tLiquidity) (LOTTree.sol#1855)
- _tOwned[address(this)] = _tOwned[address(this)].add(lotteryAmount) (LOTTree.sol#1730)
- _tOwned[sender] = _tOwned[sender].sub(tAmount) (LOTTree.sol#2107)
- _tOwned[sender] = _tOwned[sender].sub(tAmount) (LOTTree.sol#1646)
- _tOwned[recipient] = _tOwned[recipient].add(tTransferAmount) (LOTTree.sol#2086)
- _tOwned[recipient] = _tOwned[recipient].add(tTransferAmount) (LOTTree.sol#1648)

- _lotteryEvent() (LOTTree.sol#1958)
- _taxFee = _previousTaxFee (LOTTree.sol#1881)
- _taxFee = 0 (LOTTree.sol#1876)

- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)
- _taxFee = _previousTaxFee (LOTTree.sol#1881)
- _taxFee = 0 (LOTTree.sol#1876)

- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)
- _taxFee = _previousTaxFee (LOTTree.sol#1881)
- _taxFee = 0 (LOTTree.sol#1876)

Event emitted after the call(s):

- Transfer(sender,recipient,tTransferAmount) (LOTTree.sol#2069)
- _grandlotteryEvent() (LOTTree.sol#1955)

- Transfer(sender,recipient,tTransferAmount) (LOTTree.sol#2069)
- _grandFunds(_grandpool.length - 1) (LOTTree.sol#1961)

- Transfer(sender,recipient,tTransferAmount) (LOTTree.sol#2069)
- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)

- Transfer(sender,recipient,tTransferAmount) (LOTTree.sol#2069)
- _lotteryEvent() (LOTTree.sol#1958)

- Transfer(sender,recipient,tTransferAmount) (LOTTree.sol#2091)
- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)

- Transfer(sender,recipient,tTransferAmount) (LOTTree.sol#1652)
- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)

- Transfer(sender,recipient,tTransferAmount) (LOTTree.sol#2113)
- _tokenTransfer(from,to,amount,takeFee) (LOTTree.sol#1964)

Reentrancy in LOTTree.swapAndLiquify(uint256) (LOTTree.sol#1967-1992):

External calls:

- fundAddress.transfer(newBalance.sub(toaddLiquidty)) (LOTTree.sol#1989)

External calls sending eth:
- addLiquidity(otherHalf,toaddLiquidty) (LOTTree.sol#1988)
- uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (LOTTree.sol#2017-2024)
- fundAddress.transfer(newBalance.sub(toaddLiquidty)) (LOTTree.sol#1989)
Event emitted after the call(s):
- SwapAndLiquify(half,toaddLiquidty,otherHalf) (LOTTree.sol#1991)
Reentrancy in LOTTree.transferFrom(address,address,uint256) (LOTTree.sol#1517-1532):
External calls:
- _transfer(sender,recipient,amount) (LOTTree.sol#1522)
- fundAddress.transfer(newBalance.sub(toaddLiquidty)) (LOTTree.sol#1989)
External calls sending eth:
- _transfer(sender,recipient,amount) (LOTTree.sol#1522)
- uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (LOTTree.sol#2017-2024)
- fundAddress.transfer(newBalance.sub(toaddLiquidty)) (LOTTree.sol#1989)
State variables written after the call(s):
- _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (LOTTree.sol#1523-1530)
- _allowances[owner][spender] = amount (LOTTree.sol#1897)
Event emitted after the call(s):
- Approval(owner,spender,amount) (LOTTree.sol#1898)
- _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (LOTTree.sol#1523-1530)
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 (LOTTree.sol#659) is too similar to IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (LOTTree.sol#660)
Variable LOTTree.getVerifiedWinner(uint256,uint256).lengthOf_scope_2 (LOTTree.sol#2218) is too similar to LOTTree.getVerifiedWinner(uint256,uint256).lengthOf_scope_7 (LOTTree.sol#2225)
Variable LOTTree._getValues(uint256).rTransferAmount (LOTTree.sol#1766) is too similar to LOTTree._transferToExcluded(address,address,uint256).tTransferAmount (LOTTree.sol#2081)
Variable LOTTree._transferFromExcluded(address,address,uint256).rTransferAmount (LOTTree.sol#2101) is too similar to LOTTree._transferFromExcluded(address,address,uint256).tTransferAmount (LOTTree.sol#2103)
Variable LOTTree._transferToExcluded(address,address,uint256).rTransferAmount (LOTTree.sol#2079) is too similar to LOTTree._transferToExcluded(address,address,uint256).tTransferAmount (LOTTree.sol#2081)
Variable LOTTree._transferBothExcluded(address,address,uint256).rTransferAmount (LOTTree.sol#1640) is too similar to LOTTree._getValues(uint256).tTransferAmount (LOTTree.sol#1764)
Variable LOTTree._transferStandard(address,address,uint256).rTransferAmount (LOTTree.sol#2058) is too similar to LOTTree._transferStandard(address,address,uint256).tTransferAmount (LOTTree.sol#2060)
Variable LOTTree._getValues(uint256).rTransferAmount (LOTTree.sol#1766) is too similar to LOTTree._transferBothExcluded(address,address,uint256).tTransferAmount (LOTTree.sol#1642)
Variable LOTTree._transferStandard(address,address,uint256).rTransferAmount (LOTTree.sol#2058) is too similar to LOTTree._getTValues(uint256).tTransferAmount (LOTTree.sol#1789)
Variable LOTTree._transferFromExcluded(address,address,uint256).rTransferAmount (LOTTree.sol#2101) is too similar to LOTTree._transferBothExcluded(address,address,uint256).tTransferAmount (LOTTree.sol#1642)
Variable LOTTree._transferBothExcluded(address,address,uint256).rTransferAmount (LOTTree.sol#1640) is too similar to LOTTree._transferToExcluded(address,address,uint256).tTransferAmount (LOTTree.sol#2081)
Variable LOTTree._transferFromExcluded(address,address,uint256).rTransferAmount (LOTTree.sol#2101) is too similar to LOTTree._getValues(uint256).tTransferAmount (LOTTree.sol#1764)
Variable LOTTree._transferStandard(address,address,uint256).rTransferAmount (LOTTree.sol#2058) is too similar to LOTTree._transferToExcluded(address,address,uint256).tTransferAmount (LOTTree.sol#2081)
Variable LOTTree._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (LOTTree.sol#1810) is too similar to LOTTree._getValues(uint256).tTransferAmount (LOTTree.sol#1764)

Variable LOTTree.reflectionFromToken(uint256,bool).rTransferAmount (LOTTree.sol#1593) is too similar to LOTTree._transferToExcluded(address,address,uint256).tTransferAmount (LOTTree.sol#2081)
Variable LOTTree._transferStandard(address,address,uint256).rTransferAmount (LOTTree.sol#2058) is too similar to LOTTree._getValues(uint256).tTransferAmount (LOTTree.sol#1764)
Variable LOTTree._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (LOTTree.sol#1810) is too similar to LOTTree._transferToExcluded(address,address,uint256).tTransferAmount (LOTTree.sol#2081)
Variable LOTTree.reflectionFromToken(uint256,bool).rTransferAmount (LOTTree.sol#1593) is too similar to LOTTree._getValues(uint256).tTransferAmount (LOTTree.sol#1764)
Variable LOTTree._transferFromExcluded(address,address,uint256).rTransferAmount (LOTTree.sol#2101) is too similar to LOTTree._getTValues(uint256).tTransferAmount (LOTTree.sol#1789)
Variable LOTTree._getValues(uint256).rTransferAmount (LOTTree.sol#1766) is too similar to LOTTree._transferFromExcluded(address,address,uint256).tTransferAmount (LOTTree.sol#2103)
Variable LOTTree._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (LOTTree.sol#1810) is too similar to LOTTree._getTValues(uint256).tTransferAmount (LOTTree.sol#1789)
Variable LOTTree._transferFromExcluded(address,address,uint256).rTransferAmount (LOTTree.sol#2101) is too similar to LOTTree._transferToExcluded(address,address,uint256).tTransferAmount (LOTTree.sol#2081)
Variable LOTTree._transferBothExcluded(address,address,uint256).rTransferAmount (LOTTree.sol#1640) is too similar to LOTTree._transferBothExcluded(address,address,uint256).tTransferAmount (LOTTree.sol#1642)
Variable LOTTree._transferStandard(address,address,uint256).rTransferAmount (LOTTree.sol#2058) is too similar to LOTTree._transferBothExcluded(address,address,uint256).tTransferAmount (LOTTree.sol#1642)
Variable LOTTree.reflectionFromToken(uint256,bool).rTransferAmount (LOTTree.sol#1593) is too similar to LOTTree._transferBothExcluded(address,address,uint256).tTransferAmount (LOTTree.sol#1642)
Variable LOTTree._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (LOTTree.sol#1810) is too similar to LOTTree._transferBothExcluded(address,address,uint256).tTransferAmount (LOTTree.sol#1642)
Variable LOTTree._getValues(uint256).rTransferAmount (LOTTree.sol#1766) is too similar to LOTTree._getValues(uint256).tTransferAmount (LOTTree.sol#1764)
Variable LOTTree._getValues(uint256).rTransferAmount (LOTTree.sol#1766) is too similar to LOTTree._transferStandard(address,address,uint256).tTransferAmount (LOTTree.sol#2060)
Variable LOTTree._transferStandard(address,address,uint256).rTransferAmount (LOTTree.sol#2058) is too similar to LOTTree._transferFromExcluded(address,address,uint256).tTransferAmount (LOTTree.sol#2103)
Variable LOTTree._getValues(uint256).rTransferAmount (LOTTree.sol#1766) is too similar to LOTTree._getTValues(uint256).tTransferAmount (LOTTree.sol#1789)
Variable LOTTree.getVerifiedWinner(uint256,uint256).selectedBucket_scope_1 (LOTTree.sol#2217) is too similar to LOTTree.getVerifiedWinner(uint256,uint256).selectedBucket_scope_6 (LOTTree.sol#2224)
Variable LOTTree.getVerifiedWinner(uint256,uint256).selectedTierBucket_scope_0 (LOTTree.sol#2216) is too similar to LOTTree.getVerifiedWinner(uint256,uint256).selectedTierBucket_scope_5 (LOTTree.sol#2223)
Variable LOTTree.getVerifiedWinner(uint256,uint256).selectedUser_scope_3 (LOTTree.sol#2219) is too similar to LOTTree.getVerifiedWinner(uint256,uint256).selectedUser_scope_8 (LOTTree.sol#2226)
Variable LOTTree.reflectionFromToken(uint256,bool).rTransferAmount (LOTTree.sol#1593) is too similar to LOTTree._getTValues(uint256).tTransferAmount (LOTTree.sol#1789)
Variable LOTTree._transferToExcluded(address,address,uint256).rTransferAmount (LOTTree.sol#2079) is too similar to LOTTree._getTValues(uint256).tTransferAmount (LOTTree.sol#1789)
Variable LOTTree._transferFromExcluded(address,address,uint256).rTransferAmount (LOTTree.sol#2101) is too similar to LOTTree._transferStandard(address,address,uint256).tTransferAmount (LOTTree.sol#2060)
Variable LOTTree._transferBothExcluded(address,address,uint256).rTransferAmount (LOTTree.sol#1640) is too similar to LOTTree._getTValues(uint256).tTransferAmount (LOTTree.sol#1789)
Variable LOTTree._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (LOTTree.sol#1810) is too similar to LOTTree._transferStandard(address,address,uint256).tTransferAmount (LOTTree.sol#2060)
Variable LOTTree._transferToExcluded(address,address,uint256).rTransferAmount (LOTTree.sol#2079) is too similar to LOTTree._transferFromExcluded(address,address,uint256).tTransferAmount (LOTTree.sol#2103)

Variable LOTTree._transferBothExcluded(address,address,uint256).rTransferAmount (LOTTree.sol#1640) is too similar to LOTTree._transferStandard(address,address,uint256).tTransferAmount (LOTTree.sol#2060)
Variable LOTTree.reflectionFromToken(uint256,bool).rTransferAmount (LOTTree.sol#1593) is too similar to LOTTree._transferStandard(address,address,uint256).tTransferAmount (LOTTree.sol#2060)
Variable LOTTree._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (LOTTree.sol#1810) is too similar to LOTTree._transferFromExcluded(address,address,uint256).tTransferAmount (LOTTree.sol#2103)
Variable LOTTree.reflectionFromToken(uint256,bool).rTransferAmount (LOTTree.sol#1593) is too similar to LOTTree._transferFromExcluded(address,address,uint256).tTransferAmount (LOTTree.sol#2103)
Variable LOTTree._transferToExcluded(address,address,uint256).rTransferAmount (LOTTree.sol#2079) is too similar to LOTTree._getValues(uint256).tTransferAmount (LOTTree.sol#1764)
Variable LOTTree._transferBothExcluded(address,address,uint256).rTransferAmount (LOTTree.sol#1640) is too similar to LOTTree._transferFromExcluded(address,address,uint256).tTransferAmount (LOTTree.sol#2103)
Variable LOTTree._transferToExcluded(address,address,uint256).rTransferAmount (LOTTree.sol#2079) is too similar to LOTTree._transferBothExcluded(address,address,uint256).tTransferAmount (LOTTree.sol#1642)
Variable LOTTree._transferToExcluded(address,address,uint256).rTransferAmount (LOTTree.sol#2079) is too similar to LOTTree._transferStandard(address,address,uint256).tTransferAmount (LOTTree.sol#2060)
Variable LOTTree.getVerifiedWinner(uint256,uint256).winner_scope_4 (LOTTree.sol#2220) is too similar to LOTTree.getVerifiedWinner(uint256,uint256).winner_scope_9 (LOTTree.sol#2227)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
LOTTree.constructor() (LOTTree.sol#1223-1433) uses literals with too many digits:
- _rOwned[0x2a3c1136F752E7A2486eea2D77256e2fB22B932f] = uint256(100000e9).mul(_getRate()) (LOTTree.sol#1366-1367)
LOTTree.constructor() (LOTTree.sol#1223-1433) uses literals with too many digits:
- totalDistributed = totalDistributed.add(uint256(100000e9).mul(_getRate())) (LOTTree.sol#1368-1370)
LOTTree.constructor() (LOTTree.sol#1223-1433) uses literals with too many digits:
- _rOwned[0x66626C7e0D3167aa708885F72A844A094D27B4D1] = uint256(100000e9).mul(_getRate()) (LOTTree.sol#1371-1372)
LOTTree.constructor() (LOTTree.sol#1223-1433) uses literals with too many digits:
- totalDistributed = totalDistributed.add(uint256(100000e9).mul(_getRate())) (LOTTree.sol#1373-1375)
LOTTree.constructor() (LOTTree.sol#1223-1433) uses literals with too many digits:
- _rOwned[0x81bBC3D1E95b55DA0B3B72fEeB3eD7bA5626a81d] = uint256(100000e9).mul(_getRate()) (LOTTree.sol#1376-1377)
LOTTree.constructor() (LOTTree.sol#1223-1433) uses literals with too many digits:
- totalDistributed = totalDistributed.add(uint256(100000e9).mul(_getRate())) (LOTTree.sol#1378-1380)
LOTTree.constructor() (LOTTree.sol#1223-1433) uses literals with too many digits:
- _rOwned[0x2b3215Bc47129088E6e9611B0f36fcbD0605bf81] = uint256(100000e9).mul(_getRate()) (LOTTree.sol#1381-1382)
LOTTree.constructor() (LOTTree.sol#1223-1433) uses literals with too many digits:
- totalDistributed = totalDistributed.add(uint256(100000e9).mul(_getRate())) (LOTTree.sol#1383-1385)
LOTTree.slitherConstructorVariables() (LOTTree.sol#1127-2297) uses literals with too many digits:
- _tTotal = 1000000000 * 10 ** 6 * 10 ** 9 (LOTTree.sol#1143)
LOTTree.slitherConstructorVariables() (LOTTree.sol#1127-2297) uses literals with too many digits:
- _maxLotAmount = 3000000000000000000 (LOTTree.sol#1147)
LOTTree.slitherConstructorVariables() (LOTTree.sol#1127-2297) uses literals with too many digits:
- _maxgrandLotAmount = 5000000000000000000 (LOTTree.sol#1149)
LOTTree.slitherConstructorVariables() (LOTTree.sol#1127-2297) uses literals with too many digits:
- _maxTxAmount = 5000000 * 10 ** 6 * 10 ** 9 (LOTTree.sol#1190)
LOTTree.slitherConstructorVariables() (LOTTree.sol#1127-2297) uses literals with too many digits:
- numTokensSellToAddToLiquidity = 500000 * 10 ** 6 * 10 ** 9 (LOTTree.sol#1191)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
LOTTree._isTierExists (LOTTree.sol#1187) is never used in LOTTree (LOTTree.sol#1127-2297)
LOTTree.indexOf (LOTTree.sol#1188) is never used in LOTTree (LOTTree.sol#1127-2297)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variables

INFO:Detectors:
LOTTree._decimals (LOTTree.sol#1155) should be constant
LOTTree._name (LOTTree.sol#1153) should be constant
LOTTree._symbol (LOTTree.sol#1154) should be constant
LOTTree._tTotal (LOTTree.sol#1143) should be constant
LOTTree.fundAddress (LOTTree.sol#1169) should be constant
LOTTree.numTokensSellToAddToLiquidity (LOTTree.sol#1191) should be constant
Reference:
https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-con
stant
INFO:Detectors:
renounceOwnership() should be declared external:
- Ownable.renounceOwnership() (LOTTree.sol#470-473)
transferOwnership(address) should be declared external:
- Ownable.transferOwnership(address) (LOTTree.sol#479-486)
geUnlockTime() should be declared external:
- Ownable.geUnlockTime() (LOTTree.sol#488-490)
lock(uint256) should be declared external:
- Ownable.lock(uint256) (LOTTree.sol#493-498)
unlock() should be declared external:
- Ownable.unlock() (LOTTree.sol#501-509)
name() should be declared external:
- LOTTree.name() (LOTTree.sol#1469-1471)
symbol() should be declared external:
- LOTTree.symbol() (LOTTree.sol#1473-1475)
decimals() should be declared external:
- LOTTree.decimals() (LOTTree.sol#1477-1479)
totalSupply() should be declared external:
- LOTTree.totalSupply() (LOTTree.sol#1481-1483)
transfer(address,uint256) should be declared external:
- LOTTree.transfer(address,uint256) (LOTTree.sol#1490-1497)
allowance(address,address) should be declared external:
- LOTTree.allowance(address,address) (LOTTree.sol#1499-1506)
approve(address,uint256) should be declared external:
- LOTTree.approve(address,uint256) (LOTTree.sol#1508-1515)
transferFrom(address,address,uint256) should be declared external:
- LOTTree.transferFrom(address,address,uint256) (LOTTree.sol#1517-1532)
increaseAllowance(address,uint256) should be declared external:
- LOTTree.increaseAllowance(address,uint256) (LOTTree.sol#1534-1545)
decreaseAllowance(address,uint256) should be declared external:
- LOTTree.decreaseAllowance(address,uint256) (LOTTree.sol#1547-1561)
isExcludedFromReward(address) should be declared external:
- LOTTree.isExcludedFromReward(address) (LOTTree.sol#1563-1565)
totalFees() should be declared external:
- LOTTree.totalFees() (LOTTree.sol#1567-1569)
deliver(uint256) should be declared external:
- LOTTree.deliver(uint256) (LOTTree.sol#1571-1581)
reflectionFromToken(uint256,bool) should be declared external:
- LOTTree.reflectionFromToken(uint256,bool) (LOTTree.sol#1583-1596)
excludeFromReward(address) should be declared external:
- LOTTree.excludeFromReward(address) (LOTTree.sol#1611-1618)
excludeFromFee(address) should be declared external:
- LOTTree.excludeFromFee(address) (LOTTree.sol#1655-1657)
includeInFee(address) should be declared external:
- LOTTree.includeInFee(address) (LOTTree.sol#1659-1661)
setSwapAndLiquifyEnabled(bool) should be declared external:
- LOTTree.setSwapAndLiquifyEnabled(bool) (LOTTree.sol#1703-1706)
getLotteryPool() should be declared external:
- LOTTree.getLotteryPool() (LOTTree.sol#1819-1821)
getGrandLotteryPool() should be declared external:
- LOTTree.getGrandLotteryPool() (LOTTree.sol#1823-1825)
getWinners() should be declared external:
- LOTTree.getWinners() (LOTTree.sol#1827-1829)
getgrandWinners() should be declared external:
- LOTTree.getgrandWinners() (LOTTree.sol#1831-1833)

Email: audit@EtherAuthority.io