expelee

Building the Futuristic Blockchain Ecosystem

SECURITY AUDIT REPORT

Oggy Floki

TOKEN OVERVIEW

Risk Findings

Severity	Found
High	0
Medium	0
Low	0
Informational	0

Centralization Risks

Owner Privileges	Description
Can Owner Set Taxes >25% ?	Not Detected
Owner needs to enable trading ?	Not Detected
Can Owner Disable Trades ?	Not Detected
Can Owner Mint ?	Not Detected
Can Owner Blacklist ?	Not Detected
Can Owner set Max Wallet amount ?	Not Detected
Can Owner Set Max TX amount ?	Not Detected

expelee

TABLE OF CONTENTS

OVERVIEW

The Expelee team has performed a line-by-line manual analysis and automated review of the smart contract. The smart contract was analysed mainly for common smart contract vulnerabilities, exploits, and manipulation hacks. According to the smart contract audit:

Audit Result	**Passed**
KYC Verification	**-**
Audit Date	**03 December 2023**

CONTRACT DETAILS

Token Address: 0x03002eb4de8Ff52018da7f65020da66094166bb6

Name: Oggy Floki

Symbol: $OGF

Decimals: 18

Network: Binance smart chain

Token Type: BEP20

Owner: 0xD4545DdBC582e5d80bE8d11e9e6B46871Ea91566

Deployer: 0xEA687D0144FD445bA695f96FFe76429163171A3f

Token Supply: 469,690,000,000,000

Checksum: 2a2c977ecdacb1071fd43102a9961979

Testnet version:
The tests were performed using the contract deployed on the Binance smart chain Testnet, which can be found at the following address:
https://testnet.bscscan.com/address/0xd7931113d3f8ded5c00559e753988bcbc450c498#code

AUDIT METHODOLOGY

Audit Details

Our comprehensive audit report provides a full overview of the audited system's architecture, smart contract codebase, and details on any vulnerabilities found within the system.

Audit Goals

The audit goal is to ensure that the project is built to protect investors and users, preventing potentially catastrophic vulnerabilities after launch , that lead to scams and rugpulls.

Code Quality

Our analysis includes both automatic tests and manual code analysis for the following aspects:

- Exploits
- Back-doors
- Vulnerability
- Accuracy
- Readability

Tools

- DE
- Open Zeppelin
- Code Analyzer
- Solidity Code
- Compiler
- Hardhat

VULNERABILITY CHECKS

Design Logic	Passed
Compiler warnings	Passed
Private user data leaks	Passed
Timestamps dependence	Passed
Integer overflow and underflow	Passed
Race conditions & reentrancy. Cross-function race conditions	Passed
Possible delays in data delivery	Passed
Oracle calls	Passed
Front Running	Passed
DoS with Revert	Passed
DoS with block gas limit	Passed
Methods execution permissions	Passed
Economy model	Passed
Impact of the exchange rate on the logic	Passed
Malicious event log	Passed
Scoping and declarations	Passed
Uninitialized storage pointers	Passed
Arithmetic accuracy	Passed
Cross-function race conditions	Passed
Safe Zepplin module	Passed

RISK CLASSIFICATION

When performing smart contract audits, our specialists look for known vulnerabilities as well as logical and acces control issues within the code. The exploitation of these issues by malicious actors may cause serious financial damage to projects that failed to get an audit in time. We categorize these vulnerabilities by the following levels:

High Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

Medium Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

Low Risk

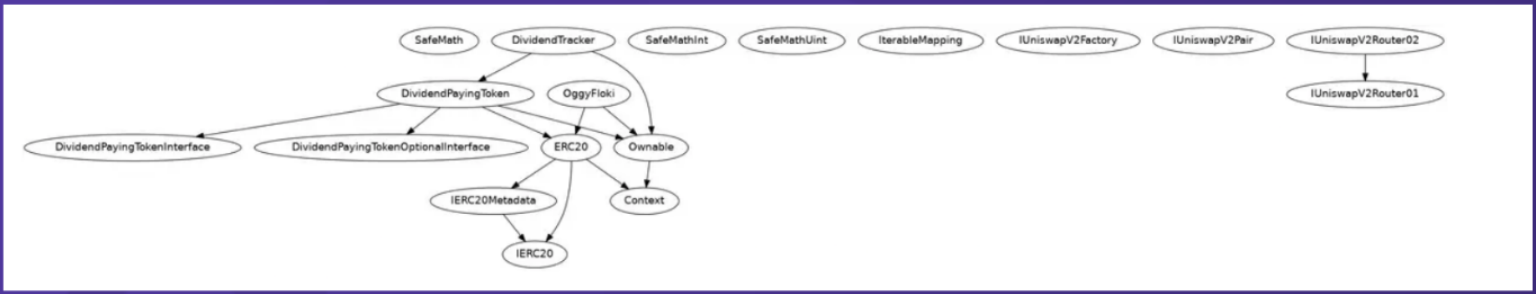
Issues on this level are minor details and warning that can remain unfixed.

Informational

Issues on this level are minor details and warning that can remain unfixed.

expelee

INHERITANCE TREES



STATIC ANALYSIS

```
INFO:Detectors:
Reentrancy in DividendPayingToken._withdrawDividendOfUser(address) (OggyFloki.sol#1182-1197):
        External calls:
        - success = IERC20(rewardToken).transfer(user,_withdrawableDividend) (OggyFloki.sol#1187)
        State variables written after the call(s):
        - withdrawnDividends[user] = withdrawnDividends[user].sub(_withdrawableDividend) (OggyFloki.sol#1190)
        DividendPayingToken.withdrawnDividends (OggyFloki.sol#1159) can be used in cross function reentrancies:
        - DividendPayingToken._withdrawDividendOfUser(address) (OggyFloki.sol#1182-1197)
        - DividendPayingToken.withdrawableDividendOf(address) (OggyFloki.sol#1203-1205)
        - DividendPayingToken.withdrawnDividendOf(address) (OggyFloki.sol#1207-1209)
Reentrancy in DividendTracker.process(uint256) (OggyFloki.sol#1406-1451):
        External calls:
        - processAccount(address(account),true) (OggyFloki.sol#1432)
                - success = IERC20(rewardToken).transfer(user,_withdrawableDividend) (OggyFloki.sol#1187)
        State variables written after the call(s):
        - lastProcessedIndex = _lastProcessedIndex (OggyFloki.sol#1448)
        DividendTracker.lastProcessedIndex (OggyFloki.sol#1257) can be used in cross function reentrancies:
        - DividendTracker.getAccount(address) (OggyFloki.sol#1317-1360)
        - DividendTracker.getLastProcessedIndex() (OggyFloki.sol#1309-1311)
        - DividendTracker.lastProcessedIndex (OggyFloki.sol#1257)
        - DividendTracker.process(uint256) (OggyFloki.sol#1406-1451)
        - DividendTracker.setLastProcessedIndex(uint256) (OggyFloki.sol#1305-1307)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1
INFO:Detectors:
OggyFloki._transfer(address,address,uint256).liquidityTokens (OggyFloki.sol#1701) is a local variable never initialized
OggyFloki._transfer(address,address,uint256).burnTokens (OggyFloki.sol#1710) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
OggyFloki.swapAndLiquify(uint256) (OggyFloki.sol#1791-1820) ignores return value by uniswapV2Router.addLiquidityETH{value: newBalance}(address(this),otherHalf,0,0,DEAD,block.timestamp) (OggyFloki.sol#1810-1817
)
OggyFloki.getAccountDividendsInfo(address) (OggyFloki.sol#1893-1904) ignores return value by dividendTracker.getAccount(account) (OggyFloki.sol#1903)
OggyFloki.getAccountDividendsInfoAtIndex(uint256) (OggyFloki.sol#1906-1917) ignores return value by dividendTracker.getAccountAtIndex(index) (OggyFloki.sol#1916)
OggyFloki.claim() (OggyFloki.sol#1920-1926) ignores return value by dividendTracker.processAccount(address(msg.sender),false) (OggyFloki.sol#1925)
OggyFloki.claimAddress(address) (OggyFloki.sol#1928-1938) ignores return value by dividendTracker.processAccount(address(claimee),false) (OggyFloki.sol#1929)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
INFO:Detectors:
DividendPayingToken.constructor(string,string,address)._name (OggyFloki.sol#1161) shadows:
        - ERC20._name (OggyFloki.sol#489) (state variable)
DividendPayingToken.constructor(string,string,address)._symbol (OggyFloki.sol#1161) shadows:
        - ERC20._symbol (OggyFloki.sol#490) (state variable)
DividendPayingToken.dividendOf(address)._owner (OggyFloki.sol#1199) shadows:
        - Ownable._owner (OggyFloki.sol#269) (state variable)
DividendPayingToken.withdrawableDividendOf(address)._owner (OggyFloki.sol#1203) shadows:
        - Ownable._owner (OggyFloki.sol#269) (state variable)
DividendPayingToken.withdrawnDividendOf(address)._owner (OggyFloki.sol#1207) shadows:
        - Ownable._owner (OggyFloki.sol#269) (state variable)
DividendPayingToken.accumulativeDividendOf(address)._owner (OggyFloki.sol#1211) shadows:
        - Ownable._owner (OggyFloki.sol#269) (state variable)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
```

```
INFO:Detectors:
DividendPayingToken.constructor(string,string,address)._name (OggyFloki.sol#1161) shadows:
        - ERC20._name (OggyFloki.sol#489) (state variable)
DividendPayingToken.constructor(string,string,address)._symbol (OggyFloki.sol#1161) shadows:
        - ERC20._symbol (OggyFloki.sol#490) (state variable)
DividendPayingToken.dividendOf(address)._owner (OggyFloki.sol#1199) shadows:
        - Ownable._owner (OggyFloki.sol#269) (state variable)
DividendPayingToken.withdrawableDividendOf(address)._owner (OggyFloki.sol#1203) shadows:
        - Ownable._owner (OggyFloki.sol#269) (state variable)
DividendPayingToken.withdrawnDividendOf(address)._owner (OggyFloki.sol#1207) shadows:
        - Ownable._owner (OggyFloki.sol#269) (state variable)
DividendPayingToken.accumulativeDividendOf(address)._owner (OggyFloki.sol#1211) shadows:
        - Ownable._owner (OggyFloki.sol#269) (state variable)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
DividendTracker.setLastProcessedIndex(uint256) (OggyFloki.sol#1305-1307) should emit an event for:
        - lastProcessedIndex = index (OggyFloki.sol#1306)
OggyFloki.setSwapTokensAtAmount(uint256) (OggyFloki.sol#1847-1850) should emit an event for:
        - swapTokensAtAmount = newAmount (OggyFloki.sol#1849)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
DividendPayingToken.constructor(string,string,address)._rewardToken (OggyFloki.sol#1161) lacks a zero-check on :
        - rewardToken = _rewardToken (OggyFloki.sol#1162)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
DividendPayingToken._withdrawDividendOfUser(address) (OggyFloki.sol#1182-1197) has external calls inside a loop: success = IERC20(rewardToken).transfer(user,_withdrawableDividend) (OggyFloki.sol#1187)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-a-loop
INFO:Detectors:
Reentrancy in OggyFloki._transfer(address,address,uint256) (OggyFloki.sol#1670-1789):
        External calls:
        - swapAndLiquify(liquidityTokens) (OggyFloki.sol#1705)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(half,0,path,address(this),block.timestamp) (OggyFloki.sol#1801-1806)
                - uniswapV2Router.addLiquidityETH{value: newBalance}(address(this),otherHalf,0,0,DEAD,block.timestamp) (OggyFloki.sol#1810-1817)
        External calls sending eth:
        - swapAndLiquify(liquidityTokens) (OggyFloki.sol#1705)
                - uniswapV2Router.addLiquidityETH{value: newBalance}(address(this),otherHalf,0,0,DEAD,block.timestamp) (OggyFloki.sol#1810-1817)
        State variables written after the call(s):
        - _burn(address(this),burnTokens) (OggyFloki.sol#1714)
                - _totalSupply -= amount (OggyFloki.sol#731)
Reentrancy in DividendTracker.processAccount(address,bool) (OggyFloki.sol#1453-1463):
        External calls:
        - amount = _withdrawDividendOfUser(account) (OggyFloki.sol#1454)
                - success = IERC20(rewardToken).transfer(user,_withdrawableDividend) (OggyFloki.sol#1187)
        State variables written after the call(s):
        - lastClaimTimes[account] = block.timestamp (OggyFloki.sol#1457)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
```

expelee

STATIC ANALYSIS

```
INFO:Detectors:
Reentrancy in OggyFloki._setAutomatedMarketMakerPair(address,bool) (OggyFloki.sol#1595-1604):
        External calls:
        - dividendTracker.excludeFromDividends(pair) (OggyFloki.sol#1600)
        Event emitted after the call(s):
        - SetAutomatedMarketMakerPair(pair,value) (OggyFloki.sol#1603)
Reentrancy in OggyFloki._transfer(address,address,uint256) (OggyFloki.sol#1670-1789):
        External calls:
        - swapAndLiquify(liquidityTokens) (OggyFloki.sol#1705)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(half,0,path,address(this),block.timestamp) (OggyFloki.sol#1801-1806)
                - uniswapV2Router.addLiquidityETH{value: newBalance}(address(this),otherHalf,0,0,DEAD,block.timestamp) (OggyFloki.sol#1810-1817)
        External calls sending eth:
        - swapAndLiquify(liquidityTokens) (OggyFloki.sol#1705)
                - uniswapV2Router.addLiquidityETH{value: newBalance}(address(this),otherHalf,0,0,DEAD,block.timestamp) (OggyFloki.sol#1810-1817)
        Event emitted after the call(s):
        - Transfer(account,address(0),amount) (OggyFloki.sol#734)
                - _burn(address(this),burnTokens) (OggyFloki.sol#1714)
Reentrancy in OggyFloki._transfer(address,address,uint256) (OggyFloki.sol#1670-1789):
        External calls:
        - swapAndLiquify(liquidityTokens) (OggyFloki.sol#1705)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(half,0,path,address(this),block.timestamp) (OggyFloki.sol#1801-1806)
                - uniswapV2Router.addLiquidityETH{value: newBalance}(address(this),otherHalf,0,0,DEAD,block.timestamp) (OggyFloki.sol#1810-1817)
        - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(contractTokenBalance,0,path,address(this),block.timestamp) (OggyFloki.sol#1729-1734)
        - (success) = marketingWallet.call{value: marketingshare}() (OggyFloki.sol#1740)
        - swapAndSendDividends(address(this).balance - initialBalance) (OggyFloki.sol#1747)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,address(this),block.timestamp) (OggyFloki.sol#1829-1834)
                - success = IERC20(rewardToken).transfer(address(dividendTracker),balanceRewardToken) (OggyFloki.sol#1837)
                - dividendTracker.distributeDividends(balanceRewardToken) (OggyFloki.sol#1840)
        External calls sending eth:
        - swapAndLiquify(liquidityTokens) (OggyFloki.sol#1705)
                - uniswapV2Router.addLiquidityETH{value: newBalance}(address(this),otherHalf,0,0,DEAD,block.timestamp) (OggyFloki.sol#1810-1817)
        - (success) = marketingWallet.call{value: marketingshare}() (OggyFloki.sol#1740)
        - swapAndSendDividends(address(this).balance - initialBalance) (OggyFloki.sol#1747)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,address(this),block.timestamp) (OggyFloki.sol#1829-1834)
        Event emitted after the call(s):
        - SendDividends(balanceRewardToken) (OggyFloki.sol#1841)
                - swapAndSendDividends(address(this).balance - initialBalance) (OggyFloki.sol#1747)
        - Transfer(from,to,amount) (OggyFloki.sol#681)
                - super._transfer(from,address(this),fees) (OggyFloki.sol#1771)
        - Transfer(from,to,amount) (OggyFloki.sol#681)
                - super._transfer(from,to,amount) (OggyFloki.sol#1774)
Reentrancy in OggyFloki._transfer(address,address,uint256) (OggyFloki.sol#1670-1789):
        External calls:
        - swapAndLiquify(liquidityTokens) (OggyFloki.sol#1705)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(half,0,path,address(this),block.timestamp) (OggyFloki.sol#1801-1806)
                - uniswapV2Router.addLiquidityETH{value: newBalance}(address(this),otherHalf,0,0,DEAD,block.timestamp) (OggyFloki.sol#1810-1817)
        - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(contractTokenBalance,0,path,address(this),block.timestamp) (OggyFloki.sol#1729-1734)
        - (success) = marketingWallet.call{value: marketingshare}() (OggyFloki.sol#1740)
        - swapAndSendDividends(address(this).balance - initialBalance) (OggyFloki.sol#1747)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,address(this),block.timestamp) (OggyFloki.sol#1829-1834)
```

```
INFO:Detectors:
Low level call in OggyFloki._transfer(address,address,uint256) (OggyFloki.sol#1670-1789):
        - (success) = marketingWallet.call{value: marketingshare}() (OggyFloki.sol#1740)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (OggyFloki.sol#954) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (OggyFloki.sol#955) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (OggyFloki.sol#972) is not in mixedCase
Function IUniswapV2Router01.WETH() (OggyFloki.sol#992) is not in mixedCase
Parameter DividendPayingToken.dividendOf(address)._owner (OggyFloki.sol#1199) is not in mixedCase
Parameter DividendPayingToken.withdrawableDividendOf(address)._owner (OggyFloki.sol#1203) is not in mixedCase
Parameter DividendPayingToken.withdrawnDividendOf(address)._owner (OggyFloki.sol#1207) is not in mixedCase
Parameter DividendPayingToken.accumulativeDividendOf(address)._owner (OggyFloki.sol#1211) is not in mixedCase
Constant DividendPayingToken.magnitude (OggyFloki.sol#1152) is not in UPPER_CASE_WITH_UNDERSCORES
Parameter DividendTracker.updateMinimumTokenBalanceForDividends(uint256)._newMinimumBalance (OggyFloki.sol#1283) is not in mixedCase
Parameter DividendTracker.getAccount(address)._account (OggyFloki.sol#1317) is not in mixedCase
Parameter OggyFloki.setWhitelistStatus(address,bool)._wallet (OggyFloki.sol#1687) is not in mixedCase
Parameter OggyFloki.setWhitelistStatus(address,bool)._status (OggyFloki.sol#1688) is not in mixedCase
Parameter OggyFloki.updateBuyFees(uint256,uint256,uint256,uint256)._liquidityFeeOnBuy (OggyFloki.sol#1618) is not in mixedCase
Parameter OggyFloki.updateBuyFees(uint256,uint256,uint256,uint256)._marketingFeeOnBuy (OggyFloki.sol#1618) is not in mixedCase
Parameter OggyFloki.updateBuyFees(uint256,uint256,uint256,uint256)._rewardsFeeOnBuy (OggyFloki.sol#1618) is not in mixedCase
Parameter OggyFloki.updateBuyFees(uint256,uint256,uint256,uint256)._trueBurnFeeOnBuy (OggyFloki.sol#1618) is not in mixedCase
Parameter OggyFloki.updateSellFees(uint256,uint256,uint256,uint256)._liquidityFeeOnSell (OggyFloki.sol#1632) is not in mixedCase
Parameter OggyFloki.updateSellFees(uint256,uint256,uint256,uint256)._marketingFeeOnSell (OggyFloki.sol#1632) is not in mixedCase
Parameter OggyFloki.updateSellFees(uint256,uint256,uint256,uint256)._rewardsFeeOnSell (OggyFloki.sol#1632) is not in mixedCase
Parameter OggyFloki.updateSellFees(uint256,uint256,uint256,uint256)._trueBurnFeeOnSell (OggyFloki.sol#1632) is not in mixedCase
Parameter OggyFloki.changeMarketingWallet(address)._marketingWallet (OggyFloki.sol#1645) is not in mixedCase
Parameter OggyFloki.setMaxWalletPercentage(uint256)._percentage (OggyFloki.sol#1654) is not in mixedCase
Constant OggyFloki._totalSupply (OggyFloki.sol#1489) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (OggyFloki.sol#997) is too similar to IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (OggyFloki.sol#998)
Variable DividendPayingToken._withdrawDividendOfUser(address)._withdrawableDividend (OggyFloki.sol#1183) is too similar to DividendTracker.getAccount(address).withdrawableDividends (OggyFloki.sol#1322)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
OggyFloki.constructor() (OggyFloki.sol#1527-1573) uses literals with too many digits:
        - swapTokensAtAmount = totalSupply() / 100000 (OggyFloki.sol#1572)
OggyFloki.slitherConstructorConstantVariables() (OggyFloki.sol#1466-1944) uses literals with too many digits:
        - _totalSupply = 469690000000000 * 1e18 (OggyFloki.sol#1489)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
SafeMathInt.MAX_INT256 (OggyFloki.sol#823) is never used in SafeMathInt (OggyFloki.sol#821-858)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable
INFO:Detectors:
OggyFloki.dividendTracker (OggyFloki.sol#1498) should be immutable
OggyFloki.uniswapV2Pair (OggyFloki.sol#1485) should be immutable
OggyFloki.uniswapV2Router (OggyFloki.sol#1484) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
INFO:Slither:OggyFloki.sol analyzed (18 contracts with 93 detectors), 92 result(s) found
```

TESTNET VERSION

1- Increase Allowance (passed):
https://testnet.bscscan.com/tx/0x4d675d9ca5b8515953faefdd1df5dd08a775235915233574efd8acaebf7f91ba

2- Decrease Allowance (passed):
https://testnet.bscscan.com/tx/0x39ff3b3b6d8340979f536f93383b73571945b727b2fc79628c800b90f14f0b8e

3- Set Whitelist Status (passed):
https://testnet.bscscan.com/tx/0x0b6c5c6e367ca4050afd585e7947dd7321ac72018d293b978bc3958a824bea5e

4- Transfer (passed):
https://testnet.bscscan.com/tx/0x0da536569bbc57a1c69d2286c5d36b5544bae36b48c20bea85c1c0ae4b7e5bc7

5- Transfer From (passed):
https://testnet.bscscan.com/tx/0xe517fabace222ae1ba611cb61cbef87b54f46aaa4c61e7f044ffd71dd87038c7

6- Process Dividend Tracker (passed):
https://testnet.bscscan.com/tx/0x4f279ed3e3456a839311534e16c9d2c4e4e6083b1a2b90d2169971530274d9fb

7- Approve (passed):
https://testnet.bscscan.com/tx/0x5868eda314ca6dd8c54885203e4ba58aa0ffa2b739e5af805b33e079c9f539c3

8- Set Max Wallet Percentage (passed):
https://testnet.bscscan.com/tx/0xd6a4915877a152dde8b3315269f382543ebf642fac2231e605eeeca90fb740a4

9- Change Marketing Wallet (passed):
https://testnet.bscscan.com/tx/0x3a9d38e5d8df75687ef97c6078bc1bff0eaab7b005ac691b78b46686d7bf7220

MANUAL REVIEW

Severity Criteria

Expelee assesses the severity of disclosed vulnerabilities according to methodology based on OWASP standarts.

Vulnerabilities are dividend into three primary risk categroies:
High
Medium
Low

High-level considerations for vulnerabilities span the following key areas when conducting assessments:

- Malicious input handling
- Escalation of privileges
- Arithmetic
- Gas use

Overall Risk Severity				
Impact	HIGH	Medium	High	Critical
	MEDIUM	Low	Medium	High
	LOW	Note	Low	Medium
		LOW	MEDIUM	HIGH
	Likelihood			

ABOUT EXPELEE

Expelee is a product-based aspirational Web3 start-up. Coping up with numerous solutions for blockchain security and constructing a Web3 ecosystem from deal making platform to developer hosting open platform, while also developing our own commercial and sustainable blockchain.

www.expelee.com

expeleeofficial
expelee
Expelee
expelee
expelee_official
expelee-co

expelee

Building the Futuristic Blockchain Ecosystem

DISCLAIMER

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment. Team provides no guarantess against the sale of team tokens or the removal of liquidity by the project audited in this document.

Always do your own research and project yourselves from being scammed. The Expelee team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools.

Under no circumstances did Expelee receive a payment to manipulate those results or change the awarding badge that we will be adding in our website. Alway do your own research and protect yourselves from scams.

This document should not be presented as a reason to buy or not buy any particular token. The Expelee team disclaims any liability for the resulting losses.