



Smart Contract Audit

FOR

Hobbes Dog

DATED : 26 MAR 23'



AUDIT SUMMARY

Project name - Hobbes Dog

Date: 26 March, 2023

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

Audit Status: Passed

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	6	0	1	0	0
Acknowledged	0	0	0	0	0
Resolved	0	0	0	0	0



USED TOOLS

Tools:

1- Manual Review:

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

2- BSC Testnet network:

all tests were done on Bsc Testnet network, each test has its transaction has attached to it.

3- Slither : Static Analysis

Testnet Link: all tests were done using this contract, tests are done on BSC Testnet

<https://testnet.bscscan.com/token/0xDAeF7B109aCcc2B6967a0Cf4bBa9db8C60bEc2b3>



Token Information

Token Name : Hobbes Dog

Token Symbol: Hobbes

Decimals: 18

Token Supply: 10,000,000,000,000

Token Address:

0x799e8D8627073a9017866DAE8Ec471bBB4164F43

Checksum:

94d8a068b67e24a3073ff074c485fff2999b9c17

Owner:

0x9420f84a6038c49A909D933A1cC370b94c77a959

(at time of writing the audit)

Network: ARBITRUM



TOKEN OVERVIEW

Fees:

Buy Fees: 5%

Sell Fees: 5%

Transfer Fees: 5%

Fees Privilege: Owner

Ownership : Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Privileges: disabling trades - changing fee-blacklisting - resetting anti-bot



AUDIT METHODOLOGY

The auditing process will follow a routine as special considerations by Auditace:

- Review of the specifications, sources, and instructions provided to Auditace to make sure the contract logic meets the intentions of the client without exposing the user's funds to risk.
- Manual review of the entire codebase by our experts, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
- Specification comparison is the process of checking whether the code does what the specifications, sources, and instructions provided to Auditace describe.
- Test coverage analysis determines whether the test cases are covering the code and how much code is exercised when we run the test cases.
- Symbolic execution is analysing a program to determine what inputs cause each part of a program to execute.
- Reviewing the codebase to improve maintainability, security, and control based on the established industry and academic practices.

VULNERABILITY CHECKLIST



Return values of low-level calls



Gasless Send



Private modifier



Using block.timestamp



Multiple Sends



Re-entrancy



Using Suicide



Tautology or contradiction



Gas Limit and Loops



Timestamp Dependence



Address hardcoded



Revert/require functions



Exception Disorder



Use of tx.origin



Using inline assembly



Integer overflow/underflow



Divide before multiply



Dangerous strict equalities



Missing Zero Address Validation



Using SHA3



Compiler version not fixed



Using throw



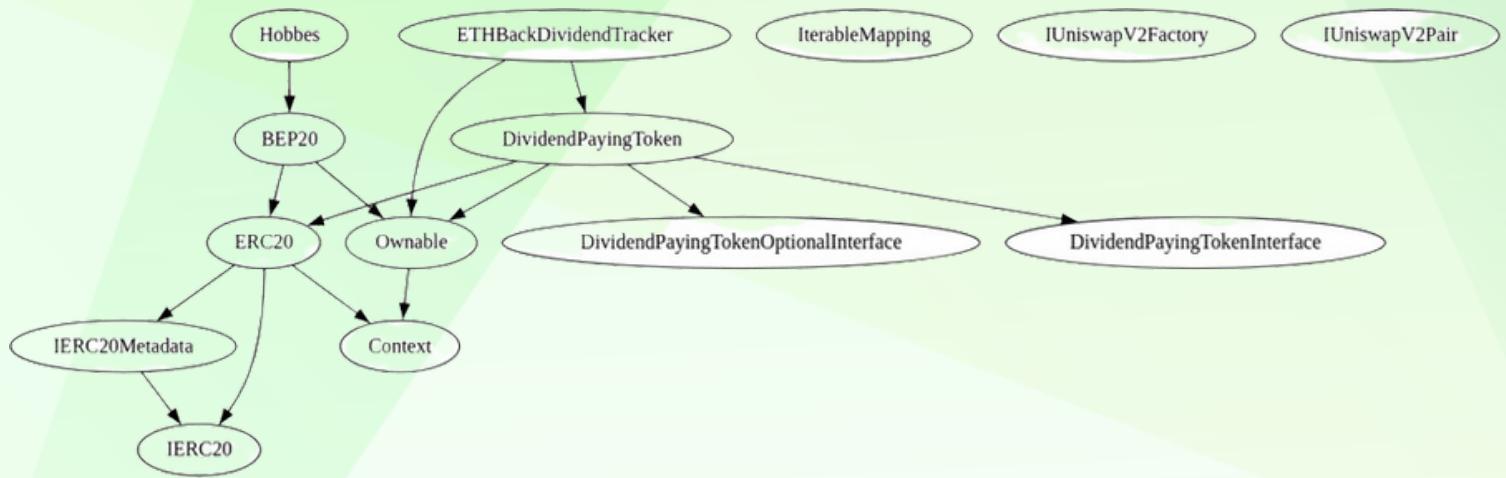
CLASSIFICATION OF RISK

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

Findings

Severity	Found
◆ Critical	6
◆ High-Risk	0
◆ Medium-Risk	1
◆ Low-Risk	0
◆ Gas Optimization / Suggestions	0

INHERITANCE TREE





POINTS TO NOTE

- -Owner is able to set buy/sell/transfer fee each one up to 100%
- -Owner must enable trading for investors to be able to trade
- -Owner is able to disable trades
- -Owner is able to blacklist arbitrary wallets
- -Owner is not able to mint new tokens



TOKEN DISTRIBUTION

It should be noted that the owner currently holds 100% of the total supply. However, information about the distribution of these tokens is not available, and it is recommended that investors exercise caution when considering this aspect.



CONTRACT ASSESSMENT

Contract	Type	Bases			
			Function Name	**Visibility**	**Mutability**
					Modifiers
	IERC20	Interface			
	totalSupply	External !	NO !		
	balanceOf	External !	NO !		
	transfer	External !		NO !	
	allowance	External !	NO !		
	approve	External !		NO !	
	transferFrom	External !		NO !	
	IERC20Metadata	Interface	IERC20		
	name	External !	NO !		
	symbol	External !	NO !		
	decimals	External !	NO !		
	Context	Implementation			
	_msgSender	Internal			
	_msgData	Internal			
	Ownable	Implementation	Context		
	<Constructor>	Public !		NO !	
	owner	Public !	NO !		
	renounceOwnership	Public !		onlyOwner	
	transferOwnership	Public !		onlyOwner	
	ERC20	Implementation	Context, IERC20, IERC20Metadata		
	<Constructor>	Public !		NO !	
	name	Public !	NO !		
	symbol	Public !	NO !		
	decimals	Public !	NO !		
	totalSupply	Public !	NO !		
	balanceOf	Public !	NO !		
	transfer	Public !		NO !	
	allowance	Public !	NO !		
	approve	Public !		NO !	
	transferFrom	Public !		NO !	
	increaseAllowance	Public !		NO !	
	decreaseAllowance	Public !		NO !	
	_transfer	Internal			
	_mint	Internal			

CONTRACT ASSESSMENT

```
| L | _burn | Internal 🔒 | ○○ | | |
| L | _approve | Internal 🔒 | ○○ | |
| L | _beforeTokenTransfer | Internal 🔒 | ○○ | |
|||||||
| **BEP20** | Implementation | ERC20, Ownable |||
| L | <Constructor> | Public ! | ○○ | ERC20 |
| L | setSwapTokensAtAmount | Public ! | ○○ | onlyOwner |
| L | <Receive Ether> | External ! | 💸 | NO! |
| L | L | Public ! | ○○ | onlyOwner |
| L | updateDividendTracker | Public ! | ○○ | onlyOwner |
| L | updateUniswapV2Router | Public ! | ○○ | onlyOwner |
| L | excludeFromFees | Public ! | ○○ | onlyOwner |
| L | excludeMultipleAccountsFromFees | Public ! | ○○ | onlyOwner |
| L | setMarketingWallet_1 | External ! | ○○ | onlyOwner |
| L | setMarketingWallet_2 | External ! | ○○ | onlyOwner |
| L | setMarketingWallet_3 | External ! | ○○ | onlyOwner |
| L | setETHRewardsFee | External ! | ○○ | onlyOwner |
| L | setLiquiditFee | External ! | ○○ | onlyOwner |
| L | setMarketingFee | External ! | ○○ | onlyOwner |
| L | setMarketingFee_2 | External ! | ○○ | onlyOwner |
| L | setETHRewardsFee_2 | External ! | ○○ | onlyOwner |
| L | setLiquiditFee_2 | External ! | ○○ | onlyOwner |
| L | setAutomatedMarketMakerPair | Public ! | ○○ | onlyOwner |
| L | bclistAddress | Public ! | ○○ | onlyOwner |
| L | multi_bclist | Public ! | ○○ | onlyOwner |
| L | _setAutomatedMarketMakerPair | Private 🔒 | ○○ | |
| L | updateGasForProcessing | Public ! | ○○ | onlyOwner |
| L | updateClaimWait | External ! | ○○ | onlyOwner |
| L | getClaimWait | External ! | | NO! |
| L | getTotalDividendsDistributed | External ! | | NO! |
| L | isExcludedFromFees | Public ! | | NO! |
| L | withdrawableDividendOf | Public ! | | NO! |
| L | dividendTokenBalanceOf | Public ! | | NO! |
| L | excludeFromDividends | External ! | ○○ | onlyOwner |
| L | getAccountDividendsInfo | External ! | | NO! |
| L | getAccountDividendsInfoAtIndex | External ! | | NO! |
| L | processDividendTracker | External ! | ○○ | NO! |
| L | claim | External ! | ○○ | NO! |
| L | getLastProcessedIndex | External ! | | NO! |
| L | getNumberOfDividendTokenHolders | External ! | | NO! |
| L | setSwapAndLiquifyEnabled | Public ! | ○○ | onlyOwner |
```



CONTRACT ASSESSMENT

```
| L | setBurnEnable | Public ! | 🔒 | onlyOwner | |
| L | setTransferFree | Public ! | 🔒 | onlyOwner |
| L | _transfer | Internal 🔒 | 🔒 |||
| L | swapAndSendToFee | Private 🔒 | 🔒 |||
| L | swapAndLiquify | Private 🔒 | 🔒 |||
| L | swapTokensForEth | Private 🔒 | 🔒 |||
| L | swapTokensForETH | Private 🔒 | 🔒 |||
| L | addLiquidity | Private 🔒 | 🔒 |||
| L | swapAndSendDividends | Private 🔒 | 🔒 |||
|||||||
| **DividendPayingTokenOptionalInterface** | Interface | |||
| L | withdrawableDividendOf | External ! | | NO! |
| L | withdrawnDividendOf | External ! | | NO! |
| L | accumulativeDividendOf | External ! | | NO! |
|||||||
| **DividendPayingTokenInterface** | Interface | |||
| L | dividendOf | External ! | | NO! |
| L | withdrawDividend | External ! | 🔒 | NO! |
|||||||
| **DividendPayingToken** | Implementation | ERC20, Ownable, DividendPayingTokenInterface,
DividendPayingTokenOptionalInterface |||
| L | <Constructor> | Public ! | 🔒 | ERC20 | |
| L | distributeETHDividends | Public ! | 🔒 | onlyOwner |
| L | withdrawDividend | Public ! | 🔒 | NO! |
| L | _withdrawDividendOfUser | Internal 🔒 | 🔒 |||
| L | dividendOf | Public ! | | NO! |
| L | withdrawableDividendOf | Public ! | | NO! |
| L | withdrawnDividendOf | Public ! | | NO! |
| L | accumulativeDividendOf | Public ! | | NO! |
| L | _transfer | Internal 🔒 | 🔒 |||
| L | _mint | Internal 🔒 | 🔒 |||
| L | _burn | Internal 🔒 | 🔒 |||
| L | _setBalance | Internal 🔒 | 🔒 |||
|||||||
| **ETHBackDividendTracker** | Implementation | Ownable, DividendPayingToken |||
| L | <Constructor> | Public ! | 🔒 | DividendPayingToken |
| L | _transfer | Internal 🔒 | 🔒 |||
| L | withdrawDividend | Public ! | 🔒 | NO! |
| L | excludeFromDividends | External ! | 🔒 | onlyOwner |
| L | updateClaimWait | External ! | 🔒 | onlyOwner |
| L | getLastProcessedIndex | External ! | | NO! |
| L | getNumberOfTokenHolders | External ! | | NO! |
```



CONTRACT ASSESSMENT

```
| L | getAccount | Public ! | | NO! | |
| L | getAccountAtIndex | Public ! | | NO! |
| L | canAutoClaim | Private 🔒 | | |
| L | setBalance | External ! | ⚡ | onlyOwner |
| L | process | Public ! | ⚡ | NO! |
| L | processAccount | Public ! | ⚡ | onlyOwner |
|||||||
| **IterableMapping** | Library | |||
| L | get | Internal 🔒 | | |
| L | getIndexOfKey | Internal 🔒 | | |
| L | getKeyAtIndex | Internal 🔒 | | |
| L | size | Internal 🔒 | | |
| L | set | Internal 🔒 | ⚡ | |
| L | remove | Internal 🔒 | ⚡ | |
|||||||
| **IUniswapV2Factory** | Interface | |||
| L | feeTo | External ! | | NO! |
| L | feeToSetter | External ! | | NO! |
| L | getPair | External ! | | NO! |
| L | allPairs | External ! | | NO! |
| L | allPairsLength | External ! | | NO! |
| L | createPair | External ! | ⚡ | NO! |
| L | setFeeTo | External ! | ⚡ | NO! |
| L | setFeeToSetter | External ! | ⚡ | NO! |
|||||||
| **IUniswapV2Pair** | Interface | |||
| L | name | External ! | | NO! |
| L | symbol | External ! | | NO! |
| L | decimals | External ! | | NO! |
| L | totalSupply | External ! | | NO! |
| L | balanceOf | External ! | | NO! |
| L | allowance | External ! | | NO! |
| L | approve | External ! | ⚡ | NO! |
| L | transfer | External ! | ⚡ | NO! |
| L | transferFrom | External ! | ⚡ | NO! |
| L | DOMAIN_SEPARATOR | External ! | | NO! |
| L | PERMIT_TYPEHASH | External ! | | NO! |
| L | nonces | External ! | | NO! |
| L | permit | External ! | ⚡ | NO! |
| L | MINIMUM_LIQUIDITY | External ! | | NO! |
| L | factory | External ! | | NO! |
```



CONTRACT ASSESSMENT

L token0 External ! NO!
L token1 External ! NO!
L getReserves External ! NO!
L price0CumulativeLast External ! NO!
L price1CumulativeLast External ! NO!
L kLast External ! NO!
L mint External ! NO!
L burn External ! NO!
L swap External ! NO!
L skim External ! NO!
L sync External ! NO!
L initialize External ! NO!
IUniswapV2Router01 Interface
L factory External ! NO!
L WETH External ! NO!
L addLiquidity External ! NO!
L addLiquidityETH External ! NO!
L removeLiquidity External ! NO!
L removeLiquidityETH External ! NO!
L removeLiquidityWithPermit External ! NO!
L removeLiquidityETHWithPermit External ! NO!
L swapExactTokensForTokens External ! NO!
L swapTokensForExactTokens External ! NO!
L swapExactETHForTokens External ! NO!
L swapTokensForExactETH External ! NO!
L swapExactTokensForETH External ! NO!
L swapETHForExactTokens External ! NO!
L quote External ! NO!
L getAmountOut External ! NO!
L getAmountIn External ! NO!
L getAmountsOut External ! NO!
L getAmountsIn External ! NO!
IUniswapV2Router02 Interface IUniswapV2Router01
L removeLiquidityETHSupportingFeeOnTransferTokens External ! NO!
L removeLiquidityETHWithPermitSupportingFeeOnTransferTokens External ! NO!
L swapExactTokensForTokensSupportingFeeOnTransferTokens External ! NO!
L swapExactETHForTokensSupportingFeeOnTransferTokens External ! NO!
L swapExactTokensForETHSupportingFeeOnTransferTokens External ! NO!



CONTRACT ASSESSMENT

```
| **SafeMath** | Library | ||| |
| L | add | Internal 🔒 | |||
| L | sub | Internal 🔒 | |||
| L | sub | Internal 🔒 | |||
| L | mul | Internal 🔒 | |||
| L | div | Internal 🔒 | |||
| L | div | Internal 🔒 | |||
| L | mod | Internal 🔒 | |||
| L | mod | Internal 🔒 | |||
|||||||
| **SafeMathInt** | Library | |||
| L | mul | Internal 🔒 | |||
| L | div | Internal 🔒 | |||
| L | sub | Internal 🔒 | |||
| L | add | Internal 🔒 | |||
| L | abs | Internal 🔒 | |||
| L | toUint256Safe | Internal 🔒 | |||
|||||||
| **SafeMathUint** | Library | |||
| L | toInt256Safe | Internal 🔒 | |||
|||||||
| **Hobbes** | Implementation | BEP20 | ||
| L | <Constructor> | Public ! | ⚡ | BEP20 |
```

Symbol	Meaning
⚡	Function can modify state
💵	Function is payable



STATIC ANALYSIS

```
Variable BEP20.sell_ETHRewardsFee (contracts/Token.sol#569) is not in mixedCase
Variable BEP20.sell_totalFees (contracts/Token.sol#570) is not in mixedCase
Variable BEP20.marketingWalletAddress_1 (contracts/Token.sol#576) is not in mixedCase
Variable BEP20._marketingWalletAddress_2 (contracts/Token.sol#577) is not in mixedCase
Variable BEP20._marketingWalletAddress_3 (contracts/Token.sol#578) is not in mixedCase
Parameter DividendPayingToken.dividendOf(address), owner (contracts/Token.sol#1403) is not in mixedCase
Parameter DividendPayingToken.withdrawableDividendOf(address), _owner (contracts/Token.sol#1411) is not in mixedCase
Parameter DividendPayingToken.withdrawnDividendOf(address), owner (contracts/Token.sol#1420) is not in mixedCase
Parameter DividendPayingToken.accumulativeDividendOf(address), _owner (contracts/Token.sol#1431) is not in mixedCase
Variable DividendPayingToken.ETH (contracts/Token.sol#1322) is not in mixedCase
Constant DividendPayingToken.magnitude (contracts/Token.sol#1327) is not in UPPER_CASE_WITH_UNDERSCORES
Parameter ETHBackDividendTracker.getAccount(address), account (contracts/Token.sol#1581) is not in mixedCase
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (contracts/Token.sol#1880) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (contracts/Token.sol#1882) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (contracts/Token.sol#1913) is not in mixedCase
Function IUniswapV2Router01.WETH() (contracts/Token.sol#1953) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Redundant expression "this (contracts/Token.sol#140)" inContext (contracts/Token.sol#134-143)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

Variable BEP20._marketingWalletAddress_1 (contracts/Token.sol#576) is too similar to BEP20._marketingWalletAddress_2 (contracts/Token.sol#577)
Variable BEP20._marketingWalletAddress_1 (contracts/Token.sol#576) is too similar to BEP20._marketingWalletAddress_3 (contracts/Token.sol#578)
Variable BEP20._marketingWalletAddress_2 (contracts/Token.sol#577) is too similar to BEP20._marketingWalletAddress_3 (contracts/Token.sol#578)
Variable BEP20.constructor(string,string,uint256,uint256[3],uint256[3],uint256,address,address,address,address).marketWallet_1 (contracts/Token.sol#639) is too similar to BEP20.constructor(string ,string,uint256,uint256[3],uint256[3],uint256,address,address,address,address).marketWallet_2 (contracts/Token.sol#640)
Variable BEP20.constructor(string,string,uint256,uint256[3],uint256[3],uint256,address,address,address,address).marketWallet_1 (contracts/Token.sol#639) is too similar to BEP20.constructor(string ,string,uint256,uint256[3],uint256[3],uint256,address,address,address,address).marketWallet_3 (contracts/Token.sol#641)
Variable BEP20.constructor(string,string,uint256,uint256[3],uint256[3],uint256,address,address,address,address).marketWallet_2 (contracts/Token.sol#640) is too similar to BEP20.constructor(string ,string,uint256,uint256[3],uint256[3],uint256,address,address,address,address).marketWallet_3 (contracts/Token.sol#641)
Variable BEP20.swapAndSendToFee(uint256), marketingFee_1 (contracts/Token.sol#1137) is too similar to BEP20.swapAndSendToFee(uint256).marketingFee_2 (contracts/Token.sol#1138)
Variable BEP20.swapAndSendToFee(uint256), marketingFee_1 (contracts/Token.sol#1137) is too similar to BEP20.swapAndSendToFee(uint256).marketingFee_3 (contracts/Token.sol#1139)
Variable BEP20.swapAndSendToFee(uint256), marketingFee_2 (contracts/Token.sol#1138) is too similar to BEP20.swapAndSendToFee(uint256).marketingFee_3 (contracts/Token.sol#1139)
Variable DividendPayingToken._withdrawDividendOfUser(address), _withdrawableDividend (contracts/Token.sol#1379) is too similar to ETHBackDividendTracker.getAccount(address).withdrawableDividends (contracts/Token.sol#1589)
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (contracts/Token.sol#1958) is too similar to IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (contracts/Token.sol#1959)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

BEP20.updateGasForProcessing(uint256) (contracts/Token.sol#866-877) uses literals with too many digits:
- require(bool,string)(newValue >= 200000 && newValue <= 500000,ETHback: gasForProcessing must be between 200,000 and 500,000) (contracts/Token.sol#867-870)
Hobbes.constructor() (contracts/Token.sol#2404-2417) uses literals with too many digits:
- BEP20(Hobbes Dog,Hobbes,1000000000000000,(uint256(40),uint256(0),uint256(10)),(uint256(40),uint256(0),uint256(10)),50000,address(msg.sender),address(msg.sender),address(msg.sender),0xe024 FC36d5Ee211Ea25A80239Fb8C4CfdB0f12Ee) (contracts/Token.sol#2405-2416)
Hobbes.slitherConstructorVariables() (contracts/Token.sol#2403-2418) uses literals with too many digits:
- gasForProcessing = 300000 (contracts/Token.sol#581)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

SafeMathInt.MAX_INT256 (contracts/Token.sol#2334) is never used in SafeMathInt (contracts/Token.sol#2332-2389)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

BEP20.deadWallet (contracts/Token.sol#554) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
```

Result => A static analysis of contract's source code has been performed using slither,

No issues found



FUNCTIONAL TESTING

Router (PCS V2):

0xD99D1c33F9fC3444f8101754aBC46c52416550D1

1- Adding Liquidity (Passed):

liquidity added on Pancakeswap V2:

<https://testnet.bscscan.com/tx/0xff3d727b6a975dd9ef48653bccebb725a122f3a2cab0168babdf5b8bfb8e37cb>

2- Buying when trading not enabled (owner%) (Passed):

<https://testnet.bscscan.com/tx/0x94f68f89c304838ac748e46ae2594ac4268efaae55cf691b4950b7f0455561b9>

3- Selling when trading not enabled (0%) (Passed):

<https://testnet.bscscan.com/tx/0xeafc4f6c56542421465475b445b83b4ec0ef6c7611c589fcba265e5d73c00db9f>

4- Transferring when trading not enabled (0% tax) (passed):

<https://testnet.bscscan.com/tx/0x4fdcb2c83d52933a0d457c6f5268e1aeb1f81127ceb65146e90985c7c2747ab5>

5- Buying when trading enabled (up to 100% tax) (passed):

<https://testnet.bscscan.com/tx/0xcb0ad894084e14ec1a967e9101ae7fe4e0984c9ddbaed32e6eec845d268a6a67>



FUNCTIONAL TESTING

6- Selling when trading enabled (up to 100% tax) (passed):

<https://testnet.bscscan.com/tx/0x4d3c2120b3c3969fde7be4cf0845c5613d836628ea01b4f3046fd1093625a736>

7- Transferring when trading enabled (up to 100% tax) (passed):

<https://testnet.bscscan.com/tx/0x95b7413b033fd00cb81d0d1edd77ba867fd995a267f855bf245eacacff8be3c8>

8- Internal swap (passed):

fee receivers received ETH

<https://testnet.bscscan.com/tx/0x4d3c2120b3c3969fde7be4cf0845c5613d836628ea01b4f3046fd1093625a736>

9- Auto Liquidity (passed):

<https://testnet.bscscan.com/tx/0x4d3c2120b3c3969fde7be4cf0845c5613d836628ea01b4f3046fd1093625a736>

10- Reflections (passed):

BUSD tokens are distributed between holders

<https://testnet.bscscan.com/tx/0x4d3c2120b3c3969fde7be4cf0845c5613d836628ea01b4f3046fd1093625a736>

11- Auto Burning (passed):

Auto burning can be seen in this tx (2 new holders in each burn)

<https://testnet.bscscan.com/tx/0x4d3c2120b3c3969fde7be4cf0845c5613d836628ea01b4f3046fd1093625a736>



MANUAL TESTING

Centralization - Owner must enable trading

Severity: Critical

Function: L

Lines: 640

Status: not resolved

Overview:

The owner must activate trading for investors to buy, sell, or transfer tokens. If trading remains disabled, token holders will be unable to trade their tokens.

```
function L(bool st, uint256 muchBt) public onlyOwner {  
    isL = st;  
    lunachB = block.number;  
    killNum = muchBt;  
}
```

Recommendation:

Incorporate a safety mechanism that allows investors to activate trading if a specified duration has elapsed since the conclusion of the presale.



MANUAL TESTING

Centralization – Owner is able to set up to 100% tax

Severity: Critical

Functions:

- setMarketingWallet_1
- setMarketingWallet_2
- setMarketingWallet_3
- setETHRewardsFee
- setLiquiditFee
- setMarketingFee
- setMarketingFee_2
- setETHRewardsFee_2
- setLiquiditFee_2

Status: not resolved

Overview:

Owner is able to set buy/sell/transfer taxes up to 100%

Recommendation:

set up a max limit for buy/sell/transfer



MANUAL TESTING

Centralization – Blacklisting

Severity: Critical

Functions: bclistAddress - multi_bclist

Status: not resolved

Overview:

The owner has the power to blacklist single or multiple wallets at their discretion. Having a blacklist feature presents a significant centralization risk and may deter potential investors from participating in the token.

```
function bclistAddress(address account, bool value) public onlyOwner {
    isbclisted[account] = value;
}

Ftrace | funcSig
function multi_bclist(
    address[] calldata addresses,
    bool value
) public onlyOwner {
    require(addresses.length < 201);
    for (uint256 i; i < addresses.length; ++i) {
        isbclisted[addresses[i]] = value;
    }
}
```

Recommendation:

- Consider removing the blacklisting functionality to reduce centralization risk and encourage investor participation.
- If the blacklisting feature is necessary, implement proper governance mechanisms to ensure decisions are made collectively by the community, rather than solely by the owner.
- Provide clear and transparent guidelines outlining the circumstances under which an address may be blacklisted to maintain investor trust and confidence.



MANUAL TESTING

Centralization – disabling trades

Severity: Critical

Function: L

Lines: 640

Status: not resolved

Overview:

Owner is able to disable trades by setting `isL` to false.

```
function L(bool s, uint256 muchB) public onlyOwner {
    isL = s;
    lunachB = block.number;
    killNum = muchB;
}
```

Recommendation:

1. Consider removing or modifying the trade disabling functionality to reduce centralization risk and protect investor interests.
2. If trade disabling is necessary for specific situations, establish a transparent governance mechanism that involves the token's community in the decision-making process.
3. Implement a safety mechanism that automatically re-enables trading after a predetermined time period has elapsed, or allows investors to vote on re-enabling trading after a certain duration since the last disabling event. This ensures the protection of investor interests and promotes decentralized control.



MANUAL TESTING

Centralization – Auto-LP generated tokens are sent to marketing wallet

Severity: **Critical**

Function: addLiquidity

Lines: 1048

Status: not resolved

Overview:

The marketing wallet receives liquidity tokens generated through the auto-liquidity feature. Over time, these accumulated tokens could be exploited by a malicious actor to remove a substantial amount of tokens from the liquidity pool.

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

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

Recommendation:

1. Consider redirecting the auto-generated liquidity tokens to a dedicated, auditable smart contract or multisig wallet instead of the marketing wallet. This reduces the centralization risk and potential for misuse.
2. Implement a transparent and community-driven governance process for managing the auto-generated liquidity tokens, ensuring that decisions regarding their usage are made collectively by token holders.
3. Provide clear guidelines and a rationale for the allocation of liquidity tokens to ensure transparency and maintain investor trust.
4. Burn generated liquidity tokens



MANUAL TESTING

Centralization – Setting total buy fees and total sell fees up to 0 can disable trades.

Severity: **Critical**

Function: `_transfer`

Lines: 1022-1030

Status: not resolved

Overview:

Setting `buy_totalFees` and `sell_totalFees` to 0 can disable trades if accumulated fees exceed the swap threshold, as this causes a division by zero error.

```
uint256 marketingTokens = contractTokenBalance
    .mul(buy_marketingFee + sell_marketingFee)
    .div(buy_totalFees + sell_totalFees);
if (marketingTokens > 0) swapAndSendToFee(marketingTokens);

uint256 swapTokens = contractTokenBalance
    .mul(buy_liquidityFee + sell_liquidityFee)
    .div(buy_totalFees + sell_totalFees);
if (swapTokens > 0) swapAndLiquify(swapTokens);

uint256 sellTokens = balanceOf(address(this));
if (sellTokens > 0) swapAndSendDividends(sellTokens);

swapping = false;
```

Recommendation:

1. Implement input validation checks to prevent the `buy_totalFees` and `sell_totalFees` values from being set to zero, thus avoiding a division by zero error and ensuring trades remain functional.
2. In case the `buy_totalFees` and `sell_totalFees` need to be set to zero for specific reasons, ensure that the contract logic can handle this scenario without causing a division by zero error or disrupting trades.
3. Provide a transparent and community-driven governance process for managing fee changes to minimize centralization risks and protect investor interests.



MANUAL TESTING

Compiler – Outdated compiler version

Severity: **Medium**

Status: not resolved

Overview:

The smart contract is compiled using an outdated compiler version 0.6.2, which may expose it to potential security vulnerabilities, compatibility issues, or suboptimal performance.

```
uint256 marketingTokens = contractTokenBalance
    .mul([buy marketingFee] + [sell marketingFee])
    .div([buy totalFees] + [sell totalFees]);
if (marketingTokens > 0) swapAndSendToFee(marketingTokens);

uint256 swapTokens = contractTokenBalance
    .mul([buy liquidityFee] + [sell liquidityFee])
    .div([buy totalFees] + [sell totalFees]);
if (swapTokens > 0) swapAndLiquify(swapTokens);

uint256 sellTokens = [balanceOf(address(this))];
if (sellTokens > 0) swapAndSendDividends(sellTokens);

[swapping] = false;
```

Recommendation:

1. Update the compiler version to the latest stable release (e.g., 0.8.x), ensuring that the smart contract benefits from the most recent security fixes, language features, and optimizations.
2. Carefully review the changes between the current compiler version (0.6.2) and the latest stable version to identify any breaking changes or adjustments required in the smart contract code.



DISCLAIMER

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



ABOUT AUDITACE

We specialize in providing thorough and reliable audits for Web3 projects. With a team of experienced professionals, we use cutting-edge technology and rigorous methodologies to evaluate the security and integrity of blockchain systems. We are committed to helping our clients ensure the safety and transparency of their digital assets and transactions.



<https://auditace.tech/>



https://t.me/Audit_Ace



https://twitter.com/auditace_



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