auditace.tech

AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

PARTY

DATED : 22 MAY 23'

AUDIT SUMMARY

Project name – PARTY

Date: 22 May, 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	0	0	0	0	0
Acknowledged	0	0	0	0	0
Resolved	0	0	0	0	0

USED TOOLS

Tools:

1.Manual Review: The code has undergone a line-by-line review by the **Ace** team.

2.ETH Test Network: All tests were conducted on the ETH Test network, and each test has a corresponding transaction attached to it. These tests can be found in the "Functional Tests" section of the report.

3.Slither: The code has undergone static analysis using Slither.

Testnet version:
Contract has been tested on binance smart chain testnet which can be found in below link:
https://testnet.bscscan.com/token/0x11bDC05dd64Ee6BBBfB6D8bFF8d711aF3dC2c7F2

Token Information

Name : Party Parrot

Symbol : PARTY

Decimals: 18

Network: Binance smart chain

Token Type:BEP20

Token Address:
0x1487e69F42C755cD4048BfC96dA0658DF7C2cEB2

Owner:
0x15B43A006e8d3cD5a459CE2d603f4c2B22a0b816
(at time of writing the audit)

Deployer:0x5dD1721Bb020118d5A0602e5dD83631E3A50F698

Token Information

Fees:

Buy Fees: 0-12%

Sell Fees: 0-12%

Transfer Fees: 0%

Fees Privilige: Owner

Ownership :

0x56d7db7cE1128Cf8cB288650Cf614b8EBa03256b

Minting: None

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Priviliges:- - Enabling trades

- initial distribution of the tokens

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 isexercised 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
- Private modifier
- Multiple Sends
- Using Suicide
- Gas Limitand Loops
- Address hardcoded
- Exception Disorder
- Using inline assembly
- Divide before multiply
- Missing Zero Address Validation
- Compiler version not fixed
- **Gasless Send**
- Using block.timestamp
- Re-entrancy
- Tautology or contradiction
- Timestamp Dependence
- Revert/require functions
- Use of tx.origin
- Integer overflow/underflow
- Dangerous strict equalities
- Using SHA3
- Using throw

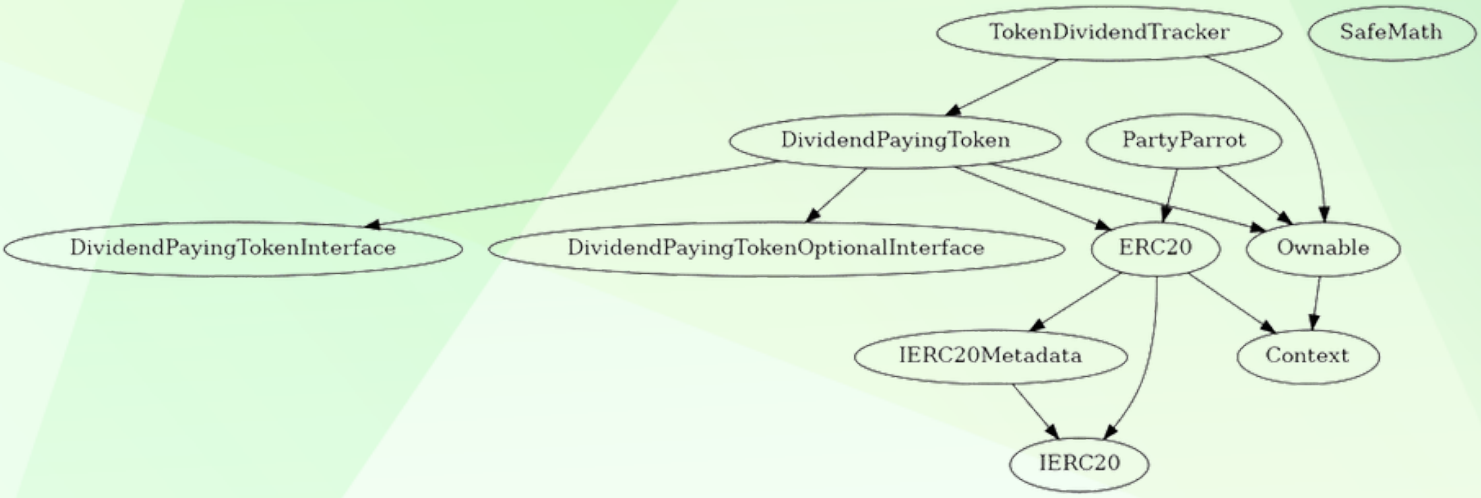
CLASSIFICATION OF RISK

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	0
High-Risk	0
Medium-Risk	0
Low-Risk	0
Gas Optimization / Suggestions	0

INHERITANCE TREE



POINTS TO NOTE

- Owner is not able to set buy/sell fees over 12% each
- Owner is not able to set fee on transfers
- Owner is not able to blacklist an arbitrary address.
- Owner is able to disable trades but not directly
- Owner is able to burn tokens from any wallet
- Owner is not able to limit buy/sell/transfer/wallet amounts
- Owner is not able to mint new tokens

CONTRACT ASSESMENT

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO ❗
└	owner	Public ❗		NO ❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_transferOwnership	Internal 🔒	🛑	
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 ❗
SafeMath	Library			
└	add	Internal 🔒		
└	sub	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
└	mod	Internal 🔒		
SafeMathInt	Library			
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	sub	Internal 🔒		
└	add	Internal 🔒		

CONTRACT ASSESMENT

└	abs	Internal 🔒			
└	toUint256Safe	Internal 🔒			
SafeMathUint	Library			
└	toInt256Safe	Internal 🔒			
Clones	Library			
└	clone	Internal 🔒		🛑	
└	cloneDeterministic	Internal 🔒		🛑	
└	predictDeterministicAddress	Internal 🔒			
└	predictDeterministicAddress	Internal 🔒			
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 🔒	🛑	
└	_cast	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_approve	Internal 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
IUniswapV2Router01	Interface			
└	factory	External ❗		NO ❗
└	WETH	External ❗		NO ❗
└	addLiquidity	External ❗	🛑	NO ❗
└	addLiquidityETH	External ❗	💵	NO ❗
└	removeLiquidity	External ❗	🛑	NO ❗
└	removeLiquidityETH	External ❗	🛑	NO ❗
└	removeLiquidityWithPermit	External ❗	🛑	NO ❗
└	removeLiquidityETHWithPermit	External ❗	🛑	NO ❗
└	swapExactTokensForTokens	External ❗	🛑	NO ❗
└	swapTokensForExactTokens	External ❗	🛑	NO ❗
└	swapExactETHForTokens	External ❗	💵	NO ❗

CONTRACT ASSESMENT

└	swapTokensForExactETH	External ❗	🛑	NO ❗
└	swapExactTokensForETH	External ❗	🛑	NO ❗
└	swapETHForExactTokens	External ❗	💵	NO ❗
└	quote	External ❗		NO ❗
└	getAmountOut	External ❗		NO ❗
└	getAmountIn	External ❗		NO ❗
└	getAmountsOut	External ❗		NO ❗
└	getAmountsIn	External ❗		NO ❗
IUniswapV2Router02	Interface	IUniswapV2Router01		
└	removeLiquidityETHSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
└	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
└	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💵	NO ❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
IUniswapV2Factory	Interface			
└	feeTo	External ❗		NO ❗
└	feeToSetter	External ❗		NO ❗
└	getPair	External ❗		NO ❗
└	allPairs	External ❗		NO ❗
└	allPairsLength	External ❗		NO ❗
└	createPair	External ❗	🛑	NO ❗
└	setFeeTo	External ❗	🛑	NO ❗
└	setFeeToSetter	External ❗	🛑	NO ❗
IUniswapV2Pair	Interface			
└	name	External ❗		NO ❗
└	symbol	External ❗		NO ❗
└	decimals	External ❗		NO ❗
└	totalSupply	External ❗		NO ❗
└	balanceOf	External ❗		NO ❗
└	allowance	External ❗		NO ❗
└	approve	External ❗	🛑	NO ❗
└	transfer	External ❗	🛑	NO ❗
└	transferFrom	External ❗	🛑	NO ❗
└	DOMAIN_SEPARATOR	External ❗		NO ❗
└	PERMIT_TYPEHASH	External ❗		NO ❗
└	nonces	External ❗		NO ❗
└	permit	External ❗	🛑	NO ❗
└	MINIMUM_LIQUIDITY	External ❗		NO ❗
└	factory	External ❗		NO ❗

CONTRACT ASSESMENT

└	token0	External ❗	|NO ❗
└	token1	External ❗	|NO ❗
└	getReserves	External ❗	|NO ❗
└	price0CumulativeLast	External ❗	|NO ❗
└	price1CumulativeLast	External ❗	|NO ❗
└	kLast	External ❗	|NO ❗
└	burn	External ❗	🛑 |NO ❗
└	swap	External ❗	🛑 |NO ❗
└	skim	External ❗	🛑 |NO ❗
└	sync	External ❗	🛑 |NO ❗
└	initialize	External ❗	🛑 |NO ❗

DividendPayingTokenInterface	Interface	|||
└	dividendOf	External ❗	|NO ❗
└	withdrawDividend	External ❗	🛑 |NO ❗

DividendPayingTokenOptionalInterface	Interface	|||
└	withdrawableDividendOf	External ❗	|NO ❗
└	withdrawnDividendOf	External ❗	|NO ❗
└	accumulativeDividendOf	External ❗	|NO ❗

DividendPayingToken	Implementation	ERC20, Ownable, DividendPayingTokenInterface, DividendPayingTokenOptionalInterface			
└	<Constructor>	Public ❗	🛑	ERC20
└	distributeCAKEDividends	Public ❗	🛑	onlyOwner
└	withdrawDividend	Public ❗	🛑 |NO ❗
└	_withdrawDividendOfUser	Internal 🔒	🛑	
└	dividendOf	Public ❗	|NO ❗
└	withdrawableDividendOf	Public ❗	|NO ❗
└	withdrawnDividendOf	Public ❗	|NO ❗
└	accumulativeDividendOf	Public ❗	|NO ❗
└	_transfer	Internal 🔒	🛑	
└	_cast	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_setBalance	Internal 🔒	🛑	

TokenDividendTracker	Implementation	Ownable, DividendPayingToken			
└	<Constructor>	Public ❗	🛑	DividendPayingToken
└	_transfer	Internal 🔒		
└	withdrawDividend	Public ❗	|NO ❗
└	setMinimumTokenBalanceForDividends	External ❗	🛑	onlyOwner
└	excludeFromDividends	External ❗	🛑	onlyOwner
└	updateClaimWait	External ❗	🛑	onlyOwner

CONTRACT ASSESMENT

└	getLastProcessedIndex	External ❗		NO ❗
└	getNumberOfTokenHolders	External ❗		NO ❗
└	isExcludedFromDividends	Public ❗		NO ❗
└	getAccount	Public ❗		NO ❗
└	getAccountAtIndex	Public ❗		NO ❗
└	canAutoClaim	Private 🔐		
└	setBalance	External ❗	🛑	onlyOwner
└	process	Public ❗	🛑	NO ❗
└	processAccount	Public ❗	🛑	onlyOwner
└	MAPGet	Public ❗		NO ❗
└	MAPGetIndexOfKey	Public ❗		NO ❗
└	MAPGetKeyAtIndex	Public ❗		NO ❗
└	MAPSize	Public ❗		NO ❗
└	MAPSet	Public ❗	🛑	NO ❗
└	MAPRemove	Public ❗	🛑	NO ❗
PartyParrot	Implementation	ERC20, Ownable	
└	<Constructor>	Public ❗	💵	ERC20
└	<Receive Ether>	External ❗	💵	NO ❗
└	updateMinimumTokenBalanceForDividends	Public ❗	🛑	onlyOwner
└	updateUniswapV2Router	Public ❗	🛑	onlyOwner
└	excludeFromFees	Public ❗	🛑	onlyOwner
└	excludeMultipleAccountsFromFees	Public ❗	🛑	onlyOwner
└	setMarketingWallet	External ❗	🛑	onlyOwner
└	setAutomatedMarketMakerPair	Public ❗	🛑	onlyOwner
└	burn	External ❗	🛑	onlyOwner
└	_setAutomatedMarketMakerPair	Private 🔐	🛑	
└	updateGasForProcessing	Public ❗	🛑	onlyOwner
└	updateClaimWait	External ❗	🛑	onlyOwner
└	getClaimWait	External ❗		NO ❗
└	getTotalDividendsDistributed	External ❗		NO ❗
└	isExcludedFromFees	Public ❗		NO ❗
└	withdrawableDividendOf	Public ❗		NO ❗
└	dividendTokenBalanceOf	Public ❗		NO ❗
└	excludeFromDividends	External ❗	🛑	onlyOwner
└	isExcludedFromDividends	Public ❗		NO ❗
└	getAccountDividendsInfo	External ❗		NO ❗
└	getAccountDividendsInfoAtIndex	External ❗		NO ❗
└	processDividendTracker	External ❗	🛑	NO ❗
└	claim	External ❗	🛑	NO ❗
└	getLastProcessedIndex	External ❗		NO ❗

CONTRACT ASSESMENT

└	getNumberOfDividendTokenHolders	External ❗		NO ❗
└	swapManual	Public ❗	🛑	onlyOwner
└	setSwapTokensAtAmount	Public ❗	🛑	onlyOwner
└	setDeadWallet	Public ❗	🛑	onlyOwner
└	setBuyTaxes	External ❗	🛑	onlyOwner
└	setSelTaxes	External ❗	🛑	onlyOwner
└	_transfer	Internal 🔒	🛑	
└	swapAndSendToFee	Private 🔐	🛑	
└	swapAndLiquify	Private 🔐	🛑	
└	swapTokensForEth	Private 🔐	🛑	
└	swapTokensForToken	Private 🔐	🛑	
└	addLiquidity	Private 🔐	🛑	
└	swapAndSendDividends	Private 🔐	🛑	

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS

```
Parameter DividendPayingToken.dividendOf(address)._owner (contracts/Token.sol#1088) is not in mixedCase
Parameter DividendPayingToken.withdrawableDividendOf(address)._owner (contracts/Token.sol#1096) is not in mixedCase
Parameter DividendPayingToken.withdrawnDividendOf(address)._owner (contracts/Token.sol#1105) is not in mixedCase
Parameter DividendPayingToken.accumulativeDividendOf(address)._owner (contracts/Token.sol#1116) is not in mixedCase
Variable DividendPayingToken.REWARD_TOKEN (contracts/Token.sol#1004) is not in mixedCase
Constant DividendPayingToken.magnitude (contracts/Token.sol#1009) is not in UPPER_CASE_WITH_UNDERSCORES
Parameter TokenDividendTracker.getAccount(address)._account (contracts/Token.sol#1284) is not in mixedCase
Function TokenDividendTracker.MAPGet(address) (contracts/Token.sol#1457-1459) is not in mixedCase
Function TokenDividendTracker.MAPGetIndexOfKey(address) (contracts/Token.sol#1461-1466) is not in mixedCase
Function TokenDividendTracker.MAPGetKeyAtIndex(uint256) (contracts/Token.sol#1468-1470) is not in mixedCase
Function TokenDividendTracker.MAPSize() (contracts/Token.sol#1472-1474) is not in mixedCase
Function TokenDividendTracker.MAPSet(address,uint256) (contracts/Token.sol#1476-1485) is not in mixedCase
Function TokenDividendTracker.MAPRemove(address) (contracts/Token.sol#1487-1504) is not in mixedCase
Variable PartyParrot.AmountLiquidityFee (contracts/Token.sol#1529) is not in mixedCase
Variable PartyParrot.AmountTokenRewardsFee (contracts/Token.sol#1530) is not in mixedCase
Variable PartyParrot.AmountMarketingFee (contracts/Token.sol#1531) is not in mixedCase
Variable PartyParrot._marketingWalletAddress (contracts/Token.sol#1533) is not in mixedCase
Variable PartyParrot.Optimization (contracts/Token.sol#1537) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (contracts/Token.sol#632) is too simil
ar to IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (contracts/Token.sol#633)
Variable DividendPayingToken._withdrawDividendOfUser(address)._withdrawableDividend (contracts/Token.sol#1061) is too similar to TokenDividendTracker.getAccoun
t(address).withdrawableDividends (contracts/Token.sol#1292)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

Clones.clone(address) (contracts/Token.sol#238-253) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_clone_asm_0,0x3d602d80600a3d3981f3363d3d373d3d3d363d73000000000000000000000000) (contracts/Token.sol#241-244)
Clones.clone(address) (contracts/Token.sol#238-253) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_clone_asm_0 + 0x28,0x5af43d82803e903d91602b57fd5bf3000000000000000000000000000000000000) (contracts/Token.sol#246-249)
Clones.cloneDeterministic(address,bytes32) (contracts/Token.sol#262-280) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_cloneDeterministic_asm_0,0x3d602d80600a3d3981f3363d3d373d3d3d363d73000000000000000000000000) (contracts/Token.sol#268-271
)
Clones.cloneDeterministic(address,bytes32) (contracts/Token.sol#262-280) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_cloneDeterministic_asm_0 + 0x28,0x5af43d82803e903d91602b57fd5bf3000000000000000000000000000000000000) (contracts/Token.sol#
273-276)
Clones.predictDeterministicAddress(address,bytes32,address) (contracts/Token.sol#285-306) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_predictDeterministicAddress_asm_0,0x3d602d80600a3d3981f3363d3d373d3d3d363d73000000000000000000000000) (contracts/Token.so
l#292-295)
Clones.predictDeterministicAddress(address,bytes32,address) (contracts/Token.sol#285-306) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_predictDeterministicAddress_asm_0 + 0x28,0x5af43d82803e903d91602b57fd5bf3ff0000000000000000000000000000000000) (contracts/T
oken.sol#297-300)
PartyParrot.constructor(string,string,uint256,address[4],uint256[4],uint256[4],uint256) (contracts/Token.sol#1590-1658) uses literals with too many digits:
        - gasForProcessing = 300000 (contracts/Token.sol#1629)
PartyParrot.updateGasForProcessing(uint256) (contracts/Token.sol#1730-1741) uses literals with too many digits:
        - require(bool,string)(newValue >= 200000 && newValue <= 500000,GasForProcessing must be between 200,000 and 500,000) (contracts/Token.sol#1731-1734)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

SafeMathInt.MAX_INT256 (contracts/Token.sol#167) is never used in SafeMathInt (contracts/Token.sol#165-222)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

PartyParrot.Optimization (contracts/Token.sol#1537) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

DividendPayingToken.REWARD_TOKEN (contracts/Token.sol#1004) should be immutable
PartyParrot.dividendTracker (contracts/Token.sol#1515) should be immutable
PartyParrot.rewardToken (contracts/Token.sol#1517) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
```

Static Analysis

an static analysis of the code were performed using slither. No issues were found

FUNCTIONAL TESTING

1- Adding liquidity (passed):
https://testnet.bscscan.com/tx/0x1b5336ae33e6330d78026f54035944a42091c68203ddecaee8930b327d54c588

2- Buying when excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0x98762dedff86acac0fed2dab020fcb425f3ba80ca21204844f82d39d71905817

3- Selling when excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0x3248b538e987952a7338695a536a6ef338214a7e5c504973f042bfa2e50dc711

4- Transferring when excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0x9c961b63585da76e0c05db544a94b790b44fc99621298ffb0fd10e0311fd5aad

5- Buying when not excluded from fees (0-12% tax) (passed):
https://testnet.bscscan.com/tx/0x2e139e79d47f6c5251491bd7704352dd1282bc3a77f5b1d8f0300312d239b3be

6- Selling from a regular wallet (0-12% tax) (passed):
https://testnet.bscscan.com/tx/0x58eeca44ad8acc4f28bbec7597f212287ba99444edf61d93ed724fc711c9af17

7- Transferring from a regular wallet (0% tax) (passed):
https://testnet.bscscan.com/tx/0x18f603de450d35f7dda0df904806157b03a586e946e2d67a5c7debf00afdd1fb

FUNCTIONAL TESTING

7- Internal swap + rewards distirbution + auto liquidity + burn (passed):

https://testnet.bscscan.com/tx/0x58eeca44ad8acc4f28bbec7597f212287ba99444edf61d93ed724fc711c9af17

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 specializes 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