AuditAce

FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

Spongebob Inu

DATED : 26 Apr 23'

AUDIT SUMMARY

Project name – Spongebob Inu

Date: 26 April, 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	2	0	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 Test Network:
all tests were done on BSC Test 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/0x316e39a96d489ced3d376ed1c1564d00a323dbe2

Token Information

Token Name : Spongebob Inu

Token Symbol: Spongebob

Decimals: 18

Token Supply:100,000,000,000,000

Token Address:
0xC23796b5De89631bE645e718D7bD2a566FFd378C

Checksum:
a1ba5be6880c7a13a4d7c070bc345c7cae291ae9

Owner:
0x20C9E801491D2DB90Fb1dbc40E8f2c6146f3d4B4

Deployer:
0x583312730e667F72c15667a8706808289094b9d7

TOKEN OVERVIEW

Fees:

Buy Fees: upto 8 %

Sell Fees: upto 8 %

Transfer Fees: 0%

Fees Privilige: none

Ownership : Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Priviliges: modifying swap threshold - toggling internal swap - excluding wallets from fee - including wallets in fee - modifying fees - changing buyback threshold

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	2
◆ Medium-Risk	0
◆ Low-Risk	0
◆ Gas Optimization / Suggestions	0

INHERITANCE TREE
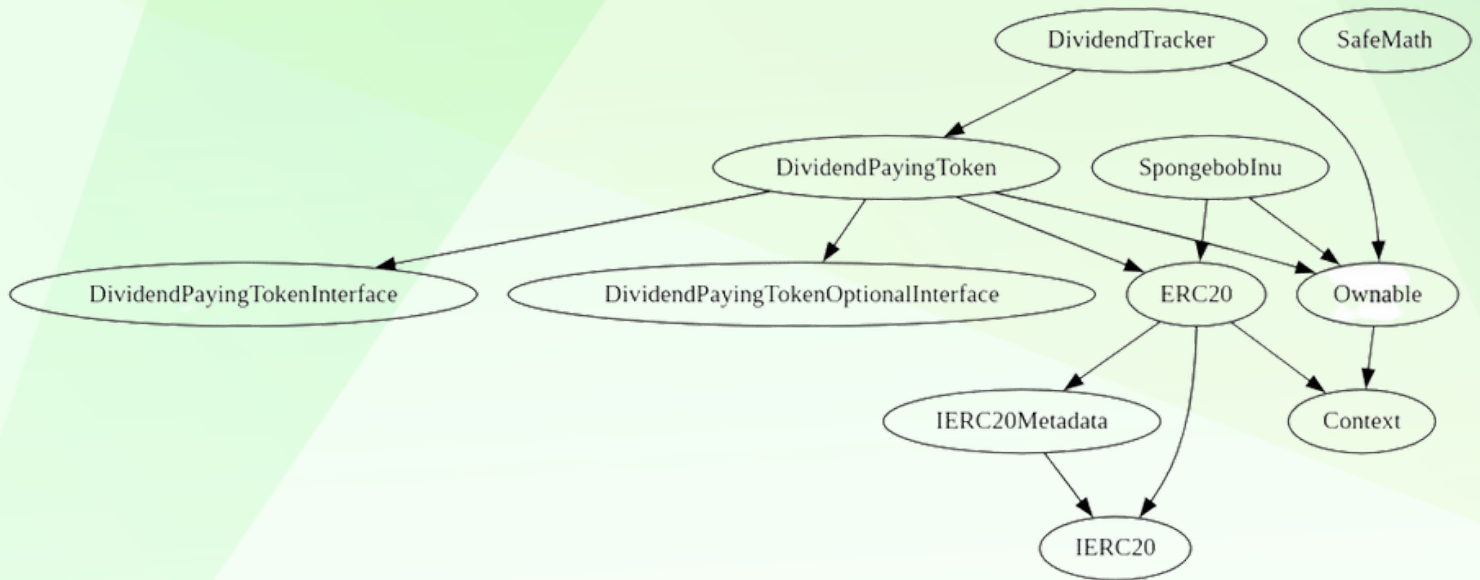
POINTS TO NOTE

- Owner is able to change buy/sell each one up to 8%
- Owner is not ablet o set transfer taxes (0% forever)
- Owner is not able to set max buy/sell/transfer/hold amount
- Owner is not able to blacklist an arbitrary wallet
- Owner is not able to disable trades
- Owner has to enable trades manually for holders
- Owner is not able to mint new tokens
- Token has BNB rewards with auto-distribution
- Fees are used for rewards (BNB), buyback and marketing

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 🔒	🛑	
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 🔒		
└	abs	Internal 🔒		
└	toUint256Safe	Internal 🔒		
SafeMathUint	Library			
└	toInt256Safe	Internal 🔒		
IterableMapping	Library			
└	get	Internal 🔒		
└	getIndexOfKey	Internal 🔒		
└	getKeyAtIndex	Internal 🔒		
└	size	Internal 🔒		
└	set	Internal 🔒	🛑	
└	remove	Internal 🔒	🛑	

CONTRACT ASSESMENT

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❗
└	token0	External ❗		|NO❗
└	token1	External ❗		|NO❗
└	getReserves	External ❗		|NO❗
└	price0CumulativeLast	External ❗		|NO❗
└	price1CumulativeLast	External ❗		|NO❗
└	kLast	External ❗		|NO❗
└	mint	External ❗	🛑	|NO❗
└	burn	External ❗	🛑	|NO❗
└	swap	External ❗	🛑	|NO❗
└	skim	External ❗	🛑	|NO❗
└	sync	External ❗	🛑	|NO❗
└	initialize	External ❗	🛑	|NO❗
IUniswapV2Router01	Interface			

CONTRACT ASSESMENT

└	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❗
└	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❗
IERC20	Interface			
└	totalSupply	External ❗			NO❗
└	balanceOf	External ❗			NO❗
└	allowance	External ❗			NO❗
└	transfer	External ❗		🛑	NO❗
└	approve	External ❗		🛑	NO❗
└	transferFrom	External ❗		🛑	NO❗
Address	Library			
└	isContract	Internal 🔒			
└	sendValue	Internal 🔒		🛑	
└	functionCall	Internal 🔒		🛑	
└	functionCall	Internal 🔒		🛑	
└	functionCallWithValue	Internal 🔒		🛑	

CONTRACT ASSESMENT

└	functionCallWithValue	Internal 🔒	🛑	
└	_functionCallWithValue	Private 🔐	🛑	
IERC20Metadata	Interface	IERC20		
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	decimals	External ❗		NO❗
IPinkAntiBot	Interface			
└	setTokenOwner	External ❗	🛑	NO❗
└	onPreTransferCheck	External ❗	🛑	NO❗
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 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_approve	Internal 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
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		

CONTRACT ASSESMENT

└	\<Constructor\>	Public ❗	⛔	ERC20
└	distributeDividends	Public ❗	⛔	onlyOwner
└	withdrawDividend	Public ❗	⛔	NO❗
└	_withdrawDividendOfUser	Internal 🔒	⛔	
└	dividendOf	Public ❗		NO❗
└	withdrawableDividendOf	Public ❗		NO❗
└	withdrawnDividendOf	Public ❗		NO❗
└	accumulativeDividendOf	Public ❗		NO❗
└	_transfer	Internal 🔒	⛔	
└	_mint	Internal 🔒	⛔	
└	_burn	Internal 🔒	⛔	
└	_setBalance	Internal 🔒	⛔	
DividendTracker	Implementation	Ownable, DividendPayingToken		
└	\<Constructor\>	Public ❗	⛔	DividendPayingToken
└	\<Receive Ether\>	External ❗	💵	NO❗
└	_transfer	Internal 🔒		
└	withdrawDividend	Public ❗		NO❗
└	updateMinimumTokenBalanceForDividends	External ❗	⛔	onlyOwner
└	excludeFromDividends	External ❗	⛔	onlyOwner
└	updateClaimWait	External ❗	⛔	onlyOwner
└	setLastProcessedIndex	External ❗	⛔	onlyOwner
└	getLastProcessedIndex	External ❗		NO❗
└	getNumberOfTokenHolders	External ❗		NO❗
└	getAccount	Public ❗		NO❗
└	getAccountAtIndex	Public ❗		NO❗
└	canAutoClaim	Private 🔐		
└	setBalance	External ❗	⛔	onlyOwner
└	process	Public ❗	⛔	NO❗
└	processAccount	Public ❗	⛔	onlyOwner
SpongebobInu	Implementation	ERC20, Ownable		
└	\<Constructor\>	Public ❗	⛔	ERC20
└	\<Receive Ether\>	External ❗	💵	NO❗
└	claimStuckTokens	External ❗	⛔	onlyOwner
└	excludeFromFees	External ❗	⛔	onlyOwner
└	isExcludedFromFees	Public ❗		NO❗
└	updateBuyFees	External ❗	⛔	onlyOwner
└	updateSellFees	External ❗	⛔	onlyOwner
└	changeMarketingWallet	External ❗	⛔	onlyOwner
└	enableTrading	External ❗	⛔	onlyOwner

CONTRACT ASSESMENT

└	_transfer	Internal 🔒	🛑	
└	swapAndSendDividends	Private 🔐	🛑	
└	setSwapEnabled	External ❗	🛑	onlyOwner
└	setBNBValueForBuyBurn	Public ❗	🛑	onlyOwner
└	buyBack	Internal 🔒	🛑	
└	setSwapTokensAtAmount	External ❗	🛑	onlyOwner
└	updateDividendTracker	Public ❗	🛑	onlyOwner
└	updateGasForProcessing	Public ❗	🛑	onlyOwner
└	updateMinimumBalanceForDividends	External ❗	🛑	onlyOwner
└	updateClaimWait	External ❗	🛑	onlyOwner
└	getClaimWait	External ❗		NO❗
└	getTotalDividendsDistributed	External ❗		NO❗
└	withdrawableDividendOf	Public ❗		NO❗
└	dividendTokenBalanceOf	Public ❗		NO❗
└	totalRewardsEarned	Public ❗		NO❗
└	excludeFromDividends	External ❗	🛑	onlyOwner
└	isExcludedFromDividends	Public ❗		NO❗
└	getAccountDividendsInfo	External ❗		NO❗
└	getAccountDividendsInfoAtIndex	External ❗		NO❗
└	processDividendTracker	External ❗	🛑	NO❗
└	claim	External ❗	🛑	NO❗
└	claimAddress	External ❗	🛑	onlyOwner
└	getLastProcessedIndex	External ❗		NO❗
└	setLastProcessedIndex	External ❗	🛑	onlyOwner
└	getNumberOfDividendTokenHolders	External ❗		NO❗

Symbol	Meaning
🛑	Function can modify state
💹	Function is payable

STATIC ANALYSIS

```
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (contracts/Token.sol#303) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (contracts/Token.sol#305) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (contracts/Token.sol#336) is not in mixedCase
Function IUniswapV2Router01.WETH() (contracts/Token.sol#378) is not in mixedCase
Parameter DividendPayingToken.dividendOf(address)._owner (contracts/Token.sol#969) is not in mixedCase
Parameter DividendPayingToken.withdrawableDividendOf(address)._owner (contracts/Token.sol#974) is not in mixedCase
Parameter DividendPayingToken.withdrawnDividendOf(address)._owner (contracts/Token.sol#980) is not in mixedCase
Parameter DividendPayingToken.accumulativeDividendOf(address)._owner (contracts/Token.sol#986) is not in mixedCase
Constant DividendPayingToken.magnitude (contracts/Token.sol#911) is not in UPPER_CASE_WITH_UNDERSCORES
Parameter DividendTracker.updateMinimumTokenBalanceForDividends(uint256)._newMinimumBalance (contracts/Token.sol#1087) is not in mixedCase
Parameter DividendTracker.getAccount(address)._account (contracts/Token.sol#1132) is not in mixedCase
Parameter SpongebobInu.updateBuyFees(uint256,uint256,uint256)._autoBuybackBurnFeeOnBuy (contracts/Token.sol#1467) is not in mixedCase
Parameter SpongebobInu.updateBuyFees(uint256,uint256,uint256)._marketingFeeOnBuy (contracts/Token.sol#1468) is not in mixedCase
Parameter SpongebobInu.updateBuyFees(uint256,uint256,uint256)._rewardFeeOnBuy (contracts/Token.sol#1469) is not in mixedCase
Parameter SpongebobInu.updateSellFees(uint256,uint256,uint256)._autoBuybackBurnFeeOnSell (contracts/Token.sol#1491) is not in mixedCase
Parameter SpongebobInu.updateSellFees(uint256,uint256,uint256)._marketingFeeOnSell (contracts/Token.sol#1492) is not in mixedCase
Parameter SpongebobInu.updateSellFees(uint256,uint256,uint256)._rewardFeeOnSell (contracts/Token.sol#1493) is not in mixedCase
Parameter SpongebobInu.changeMarketingWallet(address)._marketingWallet (contracts/Token.sol#1517) is not in mixedCase
Parameter SpongebobInu.setSwapEnabled(bool)._enabled (contracts/Token.sol#1683) is not in mixedCase
Variable SpongebobInu.DEAD (contracts/Token.sol#1332) 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#383) is too similar to IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (contracts/Token.sol#384)
Variable DividendPayingToken._withdrawDividendOfUser(address)._withdrawableDividend (contracts/Token.sol#948) is too similar to DividendTracker.getAccount(address).withdrawableDividends (contracts/Token.sol#1140)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

SpongebobInu.setSwapTokensAtAmount(uint256) (contracts/Token.sol#1706-1713) uses literals with too many digits:
        - require(bool,string)(newAmount > totalSupply() / 100000,SwapTokensAtAmount must be greater than 0.001% of total supply) (contracts/Token.sol#1707-1710)
SpongebobInu.updateGasForProcessing(uint256) (contracts/Token.sol#1748-1759) 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#1749-1752)
SpongebobInu.slitherConstructorVariables() (contracts/Token.sol#1306-1880) uses literals with too many digits:
        - gasForProcessing = 300000 (contracts/Token.sol#1345)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

SafeMathInt.MAX_INT256 (contracts/Token.sol#125) is never used in SafeMathInt (contracts/Token.sol#123-165)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

ERC20.antiBotEnabled (contracts/Token.sol#722) should be constant
ERC20.pinkAntiBot (contracts/Token.sol#721) should be constant
SpongebobInu.DEAD (contracts/Token.sol#1332) should be constant
SpongebobInu.rewardToken (contracts/Token.sol#1344) should be constant
SpongebobInu.swapWithLimit (contracts/Token.sol#1337) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

ERC20.antiBotEnabled (contracts/Token.sol#722) should be immutable
ERC20.pinkAntiBot (contracts/Token.sol#721) should be immutable
SpongebobInu.uniswapV2Pair (contracts/Token.sol#1330) should be immutable
SpongebobInu.uniswapV2Router (contracts/Token.sol#1329) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
```

Result => A static analysis of contract's source code has been performed using slither,
No major issues were found in the output

FUNCTIONAL TESTING

Router (PCS V2):
0xD99D1c33F9fC3444f8101754aBC46c52416550D1

All the functionalities have been tested, no issues were found

1- Adding liquidity (passed):
https://testnet.bscscan.com/tx/0x08a5d23aeb5db77acf55151350
5d65c57d29baf4041508654f384a18281e1c57

2- Buying when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0xcfbe71234b3e16c4dc9065a76f0
d6a3d71e50f07037779ae941eb6afd2d73f17

3- Selling when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0x718aa8fb8c4bf8a527ae9326df2
a16d91336f764015d4d1f45a1d7cae1d856f4

4- Transferring when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0x40536a087e022bc584fe0345a5
bf45a3a18d423dcc5c8658efdc4e6a101ed352

5- Buying when not excluded (8% tax) (passed):
https://testnet.bscscan.com/tx/0x8245330da5b777e01539075d26
1974ab362647aafd9408cde92976a90bd01daf

6- Selling when not excluded (8% tax) (passed):
https://testnet.bscscan.com/tx/0xb6b0e88cb5e7d02880a545725
d48703f8b51e7605b72e3e5fbf5b39c8ad4f80c

FUNCTIONAL TESTING

7- Transferring when not excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0xddb52521e87a8c0a50aa2f19471
9eec5702afaffef37dd48d3128e31bdc0b6b9

8- Internal swap (passed):
Marketing wallet received BNB
https://testnet.bscscan.com/address/0xc106ad08f4d436bdb69a14
88c628994968998aff#internaltx

9- Buyback(passed):
Buyback happened after reaching the threshold, and tokens were
sent to dead wallet
https://testnet.bscscan.com/token/0x316e39a96d489ced3d376ed
1c1564d00a323dbe2?
a=0x000000000000000000000000000000000000dead

MANUAL TESTING

Logical – Upgradeable dividend tracker

Severity: High

Function: enableTrading

Lines: 1531

Status: Not Resolved

Overview:

Owner of the contract is able to change dividend tracker to another address. If new dividend tracker has issues (spending lots of gas, not compatible with current token implementation etc), buy/sell/transfers would be disabled.

```
function updateDividendTracker(address newAddress) public onlyOwner {
    require(
        newAddress != address(dividendTracker),
        "The dividend tracker already has that address"
    );

    DividendTracker newDividendTracker = DividendTracker(
        payable(newAddress)
    );

    require(
        newDividendTracker.owner() == address(this),
        "The new dividend tracker must be owned by the token contract"
    );

    try
        newDividendTracker.excludeFromDividends(address(newDividendTracker))
    {} catch {}
    try newDividendTracker.excludeFromDividends(address(this)) {} catch {}
    try newDividendTracker.excludeFromDividends(DEAD) {} catch {}
    try
        newDividendTracker.excludeFromDividends(address(uniswapV2Router))
    {} catch {}
    try
        newDividendTracker.excludeFromDividends(address(uniswapV2Pair))
    {} catch {}

    emit UpdateDividendTracker(newAddress, address(dividendTracker));

    dividendTracker = newDividendTracker;
}
```

Recommendation:

There are several options to mitigate this issue
- renounce the ownership
- Ensure that dividend tracker is not upgradeable
- Add additional functions to current dividend tracker so be able to change reward token, router address etc. this means we don't need to create a new dividend tracker later in order to update current one.

MANUAL TESTING

Centralization - Owner Must Enable Trades

Severity: High / Informational
Function: enableTrading
Lines: 1531
Status: Not Resolved

Overview:

The owner is required to enable trading for investors. If trading remains disabled, token holders will not have the ability to buy, sell, or transfer their tokens.

```
function enableTrading() external onlyOwner {
    tradingEnabled = true;
}
```

Recommendation:

While the presence of this function is considered a feature rather than a flaw, it is crucial to highlight the centralization risk it inherently poses. To address this issue and ensure the enablement of trades, one possible solution would be to transfer the contract's ownership to a trusted third party, such as a Pinksale Safu developer. This would help mitigate the centralization risk associated with this function.

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.

Social Media

Social Media Details for

Spongebob Inu

https://spongebobinu.world/

https://t.me/OfficialSpongebobInu

https://twitter.com/spongebobinubsc

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