auditace.tech

AuditAce

FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

DOGE NEW

DATED : 09 Apr 23'

AUDIT SUMMARY

Project name – Doge New

Date: 09 April, 2023

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

Audit Status: Failed

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	2	3	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/0x8913789ca79505a2365cee8cb3ffc8968d8408a4

Token Information

Token Name : DogeNew

Token Symbol: DOGENEW

Decimals: 18

Token Supply: 420,000,000,000,000,000

Token Address:
0x0c393C25E14ba54ac36A29C08Fa1001fdBf4F433

Checksum:
bc96c68fdc1adacf03efcd6f8468603cbf90a648

Owner:
0xe01F0E7d2e54883adAb08e7113031831CbB5bE1D

TOKEN OVERVIEW

Fees:

Buy Fees: 10%

Sell Fees: 10%

Transfer Fees: 10%

Fees Privilige: None

Ownership : Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Priviliges:including and excluding form fee - changing swap threshold - enabling trades

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

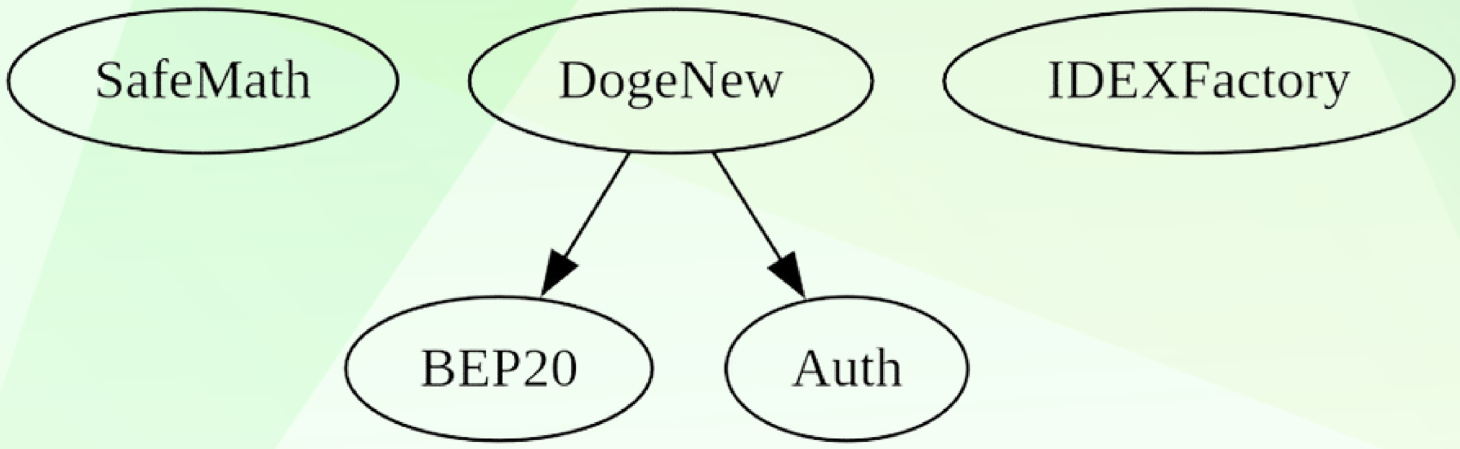
CLASSIFICATION OF RISK

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	2
🟫 High-Risk	3
🟧 Medium-Risk	0
🟨 Low-Risk	0
🟦 Gas Optimization / Suggestions	0

INHERITANCE TREE



POINTS TO NOTE

- Owner is not able to modify buy/sell/transfer fees (10% for each)
- 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 is not able to mint new tokens
- **Owner must enable trading for investors**
- **Owner is able to disable trades**

CONTRACT ASSESMENT

```
| Contract | Type | Bases | | |
|:----------:|:-------------------:|:----------------:|:----------------:|:---------------:|
| └ | **Function Name** | **Visibility** | **Mutability** | **Modifiers** |
||||||
| **SafeMath** | Library | |||
| └ | add | Internal 🔒 | | |
| └ | sub | Internal 🔒 | | |
| └ | sub | Internal 🔒 | | |
| └ | mul | Internal 🔒 | | |
| └ | div | Internal 🔒 | | |
| └ | div | Internal 🔒 | | |
||||||
| **BEP20** | Interface | |||
| └ | getOwner | External ❗ | |NO❗ |
| └ | balanceOf | External ❗ | |NO❗ |
| └ | transfer | External ❗ | 🛑 |NO❗ |
| └ | allowance | External ❗ | |NO❗ |
| └ | approve | External ❗ | 🛑 |NO❗ |
| └ | transferFrom | External ❗ | 🛑 |NO❗ |
||||||
| **Auth** | Implementation | |||
| └ | <Constructor> | Public ❗ | 🛑 |NO❗ |
| └ | authorize | External ❗ | 🛑 | onlyOwner |
| └ | unauthorize | External ❗ | 🛑 | onlyOwner |
| └ | isOwner | Public ❗ | |NO❗ |
| └ | isAuthorized | Public ❗ | |NO❗ |
| └ | transferOwnership | External ❗ | 🛑 | onlyOwner |
| └ | acceptOwnership | External ❗ | 🛑 |NO❗ |
||||||
| **IDEXFactory** | Interface | |||
| └ | createPair | External ❗ | 🛑 |NO❗ |
||||||
| **IDEXRouter** | Interface | |||
| └ | factory | External ❗ | |NO❗ |
| └ | WETH | External ❗ | |NO❗ |
| └ | addLiquidityETH | External ❗ | 💵 |NO❗ |
| └ | swapExactTokensForETHSupportingFeeOnTransferTokens | External ❗ | 🛑 |NO❗ |
||||||
| **DogeNew** | Implementation | BEP20, Auth |||
| └ | <Constructor> | Public ❗ | 🛑 | Auth |
| └ | <Receive Ether> | External ❗ | 💵 |NO❗ |
| └ | getOwner | External ❗ | |NO❗ |
```

CONTRACT ASSESMENT

```
| └ | allowance | External ❗ |   |NO❗ |
| └ | approve | Public ❗ | 🛑  |NO❗ |
| └ | approveMax | External ❗ | 🛑  |NO❗ |
| └ | transfer | External ❗ | 🛑  |NO❗ |
| └ | transferFrom | External ❗ | 🛑  |NO❗ |
| └ | _transferFrom | Internal 🔒 | 🛑  | |
| └ | _basicTransfer | Internal 🔒 | 🛑  | |
| └ | takeFee | Internal 🔒 | 🛑  | |
| └ | shouldSwapBack | Internal 🔒 |   | |
| └ | clearStuckBalance | External ❗ | 🛑  | onlyOwner |
| └ | clearStuckToken | External ❗ | 🛑  | onlyOwner |
| └ | tradingStatus | External ❗ | 🛑  | onlyOwner |
| └ | tradingStatus_launchmode | External ❗ | 🛑  | onlyOwner |
| └ | swapBack | Internal 🔒 | 🛑  | swapping |
| └ | setSwapBackSettings | External ❗ | 🛑  | onlyOwner |
||||||
| **DividendPayingToken** | Implementation | ERC20, DividendPayingTokenInterface, Ownable |||
| └ | <Constructor> | Public ❗ | 🛑  | ERC20 |
| └ | <Receive Ether> | External ❗ |  💵 |NO❗ |
| └ | distributeDividends | Public ❗ |  💵 |NO❗ |
| └ | _withdrawDividendOfUser | Internal 🔒 | 🛑  | |
| └ | setRewardToken | External ❗ | 🛑  | onlyOwner |
| └ | swapBnbForCustomToken | Internal 🔒 | 🛑  | |
| └ | dividendOf | Public ❗ |   |NO❗ |
| └ | withdrawableDividendOf | Public ❗ |   |NO❗ |
| └ | withdrawnDividendOf | Public ❗ |   |NO❗ |
| └ | accumulativeDividendOf | Public ❗ |   |NO❗ |
| └ | _transfer | Internal 🔒 | 🛑  | |
| └ | _tokengeneration | Internal 🔒 | 🛑  | |
| └ | _burn | Internal 🔒 | 🛑  | |
| └ | _setBalance | 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❗ |
```

CONTRACT ASSESMENT

```
| └ | approve | Public ❗ | 🛑  |NO❗ |
| └ | transferFrom | Public ❗ | 🛑  |NO❗ |
| └ | increaseAllowance | Public ❗ | 🛑  |NO❗ |
| └ | decreaseAllowance | Public ❗ | 🛑  |NO❗ |
| └ | _transfer | Internal 🔒 | 🛑  | |
| └ | _tokengeneration | Internal 🔒 | 🛑  | |
| └ | _burn | Internal 🔒 | 🛑  | |
| └ | _approve | Internal 🔒 | 🛑  | |
| └ | _beforeTokenTransfer | 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❗ |
||||||
| **Context** | Implementation |  |||
| └ | _msgSender | Internal 🔒 |   | |
| └ | _msgData | 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 🔒 |   | |
```

CONTRACT ASSESMENT

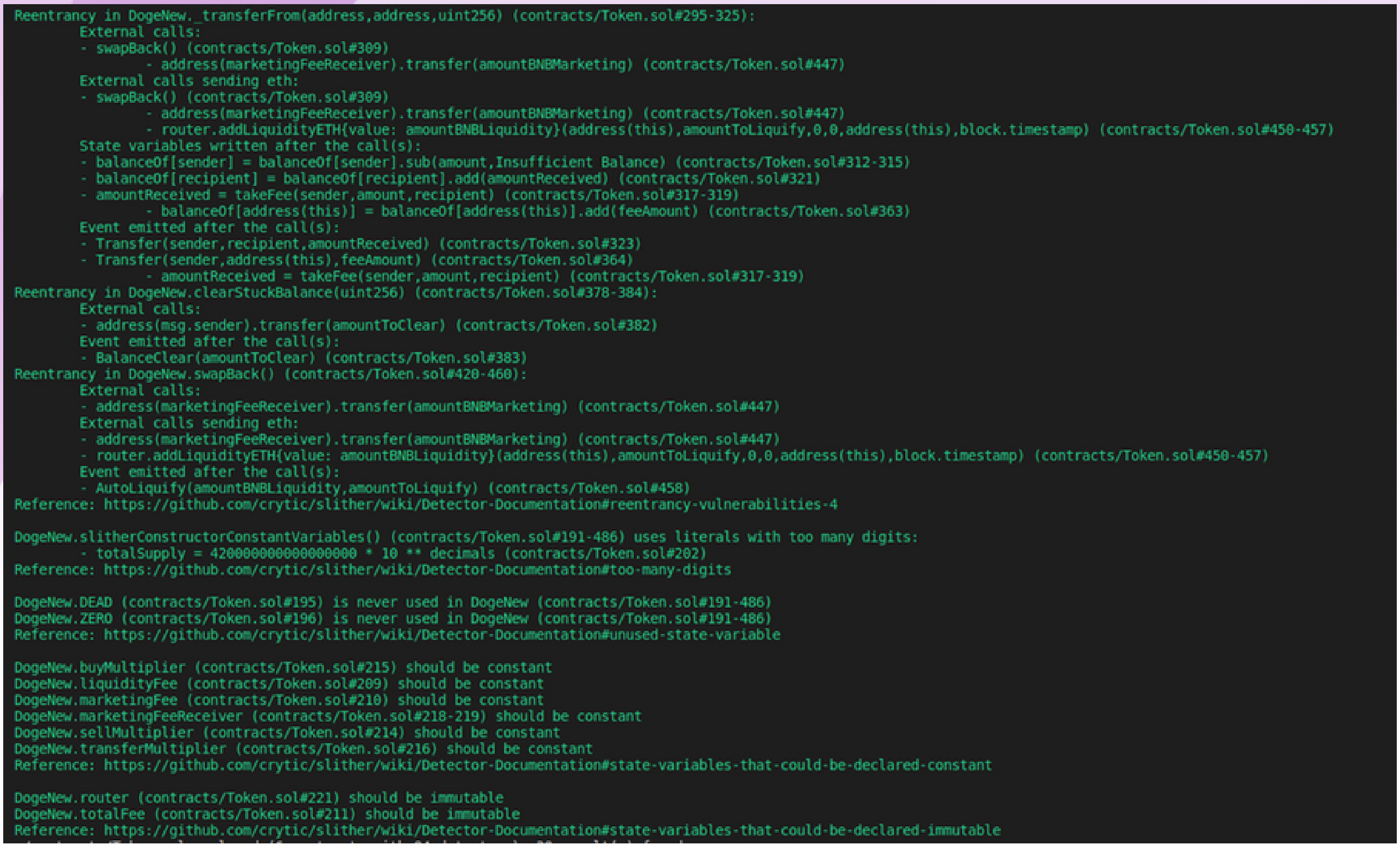
```
| └ | abs | Internal 🔒 |   | |
| └ | toUint256Safe | Internal 🔒 |   | |
||||||
| **SafeMathUint** | Library |  |||
| └ | toInt256Safe | Internal 🔒 |   | |
||||||
| **DividendPayingTokenInterface** | Interface |  |||
| └ | dividendOf | External ❗ |   |NO❗ |
| └ | distributeDividends | External ❗ |  💵 |NO❗ |
| └ | withdrawableDividendOf | External ❗ |   |NO❗ |
| └ | withdrawnDividendOf | External ❗ |   |NO❗ |
| └ | accumulativeDividendOf | External ❗ |   |NO❗ |
||||||
| **Ownable** | Implementation | Context |||
| └ | <Constructor> | Public ❗ | 🛑  |NO❗ |
| └ | owner | Public ❗ |   |NO❗ |
| └ | renounceOwnership | Public ❗ | 🛑  | onlyOwner |
| └ | transferOwnership | Public ❗ | 🛑  | onlyOwner |
||||||
| **IPair** | Interface |  |||
| └ | sync | External ❗ | 🛑  |NO❗ |
||||||
| **IFactory** | Interface |  |||
| └ | createPair | External ❗ | 🛑  |NO❗ |
| └ | getPair | External ❗ |   |NO❗ |
||||||
| **IRouter** | Interface |  |||
| └ | factory | External ❗ |   |NO❗ |
| └ | WETH | External ❗ |   |NO❗ |
| └ | addLiquidityETH | External ❗ |  💵 |NO❗ |
| └ | swapExactTokensForTokensSupportingFeeOnTransferTokens | External ❗ | 🛑  |NO❗ |
| └ | swapExactETHForTokens | External ❗ |  💵 |NO❗ |
| └ | swapExactTokensForETHSupportingFeeOnTransferTokens | External ❗ | 🛑  |NO❗ |
||||||
| **IterableMapping** | Library |  |||
| └ | get | Internal 🔒 |   | |
| └ | getIndexOfKey | Internal 🔒 |   | |
| └ | getKeyAtIndex | Internal 🔒 |   | |
| └ | size | Internal 🔒 |   | |
| └ | set | Internal 🔒 | 🛑  | |
| └ | remove | Internal 🔒 | 🛑  | |
```

CONTRACT ASSESMENT

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS



```
Reentrancy in DogeNew._transferFrom(address,address,uint256) (contracts/Token.sol#295-325):
        External calls:
        - swapBack() (contracts/Token.sol#309)
                - address(marketingFeeReceiver).transfer(amountBNBMarketing) (contracts/Token.sol#447)
        External calls sending eth:
        - swapBack() (contracts/Token.sol#309)
                - address(marketingFeeReceiver).transfer(amountBNBMarketing) (contracts/Token.sol#447)
                - router.addLiquidityETH{value: amountBNBLiquidity}(address(this),amountToLiquify,0,0,address(this),block.timestamp) (contracts/Token.sol#450-457)
        State variables written after the call(s):
        - balanceOf[sender] = balanceOf[sender].sub(amount,Insufficient Balance) (contracts/Token.sol#312-315)
        - balanceOf[recipient] = balanceOf[recipient].add(amountReceived) (contracts/Token.sol#321)
        - amountReceived = takeFee(sender,amount,recipient) (contracts/Token.sol#317-319)
                - balanceOf[address(this)] = balanceOf[address(this)].add(feeAmount) (contracts/Token.sol#363)
        Event emitted after the call(s):
        - Transfer(sender,recipient,amountReceived) (contracts/Token.sol#323)
        - Transfer(sender,address(this),feeAmount) (contracts/Token.sol#364)
                - amountReceived = takeFee(sender,amount,recipient) (contracts/Token.sol#317-319)
Reentrancy in DogeNew.clearStuckBalance(uint256) (contracts/Token.sol#378-384):
        External calls:
        - address(msg.sender).transfer(amountToClear) (contracts/Token.sol#382)
        Event emitted after the call(s):
        - BalanceClear(amountToClear) (contracts/Token.sol#383)
Reentrancy in DogeNew.swapBack() (contracts/Token.sol#420-460):
        External calls:
        - address(marketingFeeReceiver).transfer(amountBNBMarketing) (contracts/Token.sol#447)
        External calls sending eth:
        - address(marketingFeeReceiver).transfer(amountBNBMarketing) (contracts/Token.sol#447)
        - router.addLiquidityETH{value: amountBNBLiquidity}(address(this),amountToLiquify,0,0,address(this),block.timestamp) (contracts/Token.sol#450-457)
        Event emitted after the call(s):
        - AutoLiquify(amountBNBLiquidity,amountToLiquify) (contracts/Token.sol#458)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-4

DogeNew.slitherConstructorConstantVariables() (contracts/Token.sol#191-486) uses literals with too many digits:
        - totalSupply = 420000000000000000 * 10 ** decimals (contracts/Token.sol#202)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

DogeNew.DEAD (contracts/Token.sol#195) is never used in DogeNew (contracts/Token.sol#191-486)
DogeNew.ZERO (contracts/Token.sol#196) is never used in DogeNew (contracts/Token.sol#191-486)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

DogeNew.buyMultiplier (contracts/Token.sol#215) should be constant
DogeNew.liquidityFee (contracts/Token.sol#209) should be constant
DogeNew.marketingFee (contracts/Token.sol#210) should be constant
DogeNew.marketingFeeReceiver (contracts/Token.sol#218-219) should be constant
DogeNew.sellMultiplier (contracts/Token.sol#214) should be constant
DogeNew.transferMultiplier (contracts/Token.sol#216) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

DogeNew.router (contracts/Token.sol#221) should be immutable
DogeNew.totalFee (contracts/Token.sol#211) 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/0x2741fd9da33a1c55ba43ab4afbc5cf2ba5721e55200c4ebd090f6206d5ef7fb9

2- Buying when excluded (0% tax) **(passed):**
https://testnet.bscscan.com/tx/0xd56f55404761a5e87275c55b5479be28dda1ef945bb4aef1453c068b6eb51c20

3- Selling when excluded (0% tax) **(passed):**
https://testnet.bscscan.com/tx/0x98186f00b3e244a43aba9b57c00fae0a2c691f38eec9111834b7227fab995561

4- Transferring when excluded (0% tax) **(passed):**
https://testnet.bscscan.com/tx/0x485c09e78312f64c0f60e5872df4aede15e9aee2682911006a9c43d8b0248e1b

5- Buying when not excluded (10% tax) **(passed):**
https://testnet.bscscan.com/tx/0x43c0754d40dc55a0c49d552944d2c68ef00066f0445f40a1b04517e68ff62d9a

6- Selling when not excluded (10% tax) **(passed):**
https://testnet.bscscan.com/tx/0xf19eb9b0b51b3fd1bb0c5fa88a2929b0acffbff7c4eb531ac305229f4cfbb25d

FUNCTIONAL TESTING

7- Transferring when not excluded (10% tax) (passed):
https://testnet.bscscan.com/tx/0xa938710911de08c4062a15972bf85c20a55e2d25b8080c4cc0990a00d18e457a

8- Internal swap (passed):
Marketing wallet received BNB
https://testnet.bscscan.com/address/0x7a212db86e1cd807bdd4e53776ac1525b35787a9#internaltx

9- Auto-liquidty (passed):
https://testnet.bscscan.com/token/0x98d763a4cd7dcbba889ac9acf30ef4cacb6a47ff?a=0x8913789ca79505a2365cee8cb3ffc8968d8408a4

MANUAL TESTING

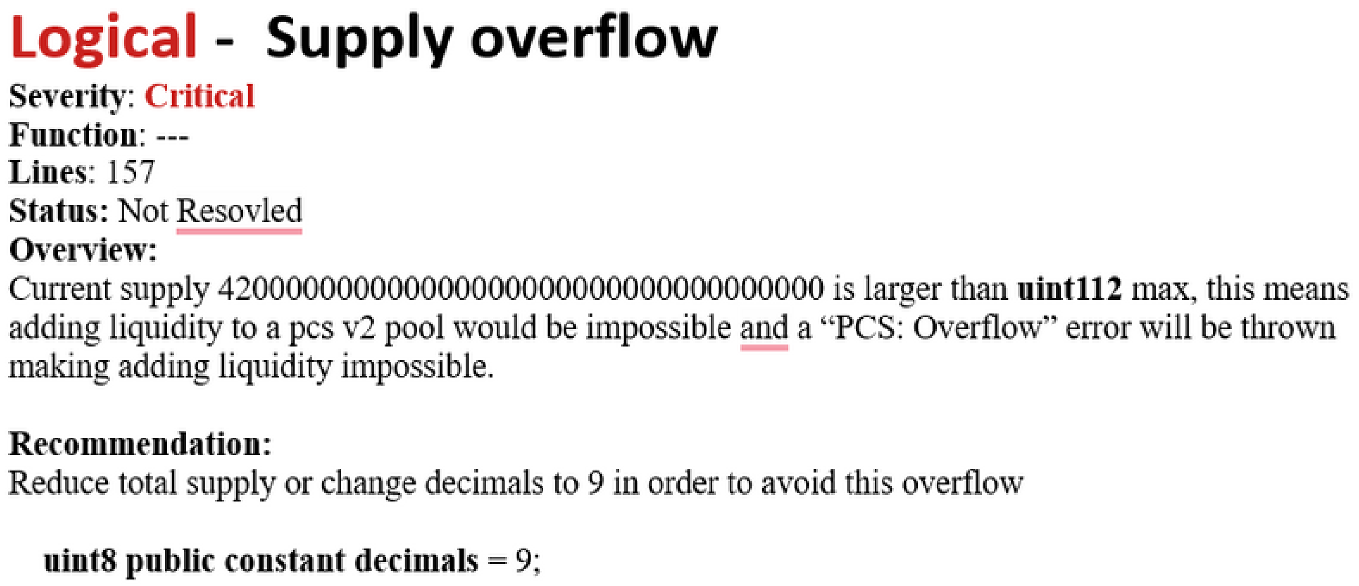
Logical – Owner is able to disable trades

Severity: Critical
Function: tradingStatus
Lines: 306
Status: Not Resolved
as long as the **launchMode** variable is set to true, owner is able to enable/disable trades. launchMode is only set to false if tradingStatus_launchmode funciton is called by owner

```
function tradingStatus(bool _status) external onlyOwner {
    if (!_status) {
        require(launchMode, "Cannot stop trading after launch is done");
    }
    tradingOpen = _status;
    emit config_TradingStatus(tradingOpen);
}
```

Recommendation:
Ensure that owner is not able to disable trades once enabled

MANUAL TESTING



Logical - Supply overflow

Severity: Critical
Function: ---
Lines: 157
Status: Not Resovled
Overview:
Current supply 420000000000000000000000000000000000 is larger than **uint112** max, this means adding liquidity to a pcs v2 pool would be impossible and a “PCS: Overflow” error will be thrown making adding liquidity impossible.

Recommendation:
Reduce total supply or change decimals to 9 in order to avoid this overflow

```
uint8 public constant decimals = 9;
```

MANUAL TESTING

Centralization - Owner must enable trading

Severity: High
Function: tradingStatus_launchmode - tradingStatus
Lines: 314, 306
Status: Not Resovled
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 tradingStatus(bool _status) external onlyOwner {
    if (!_status) {
        require(launchMode, "Cannot stop trading after launch is done");
    }
    tradingOpen = _status;
    emit config_TradingStatus(tradingOpen);
}

function tradingStatus_launchmode(uint256 confirm) external onlyOwner {
    require(confirm == 123123, "Accidental Press");
    require(tradingOpen, "Cant close launch mode when trading is disabled");
    launchMode = false;
    emit config_LaunchMode(launchMode);
}
```

Recommendation:
Incorporate a safety mechanism that allows investors to activate trading if a specified duration has elapsed since the conclusion of the presale or consider alternative ways such as allowing trades ater investors claimed their presale tokens.

MANUAL TESTING

Logical - Setting internal swap threshold to 0 can disable sells

Severity: High
Function: setSwapBackSettings
Lines: 362
Status: Not Resolved
If the **swapThreshold** is set to 0, sell transactions will fail at the _transfer function. This occurs because the checks for performing a swapAndLiquify will still pass even if the swapThreshold is set to 0 and the contract has 0 tokens. Consequently, the transaction will fail while attempting to swap 0 tokens (i.e., **swapThreshold**) to BNB.

```
function setSwapBackSettings(
    bool _enabled,
    uint256 _amount
) external onlyOwner {
    require(_amount < (totalSupply / 10), "Amount too high");

    swapEnabled = _enabled;
    swapThreshold = _amount;

    emit config_SwapSettings(swapThreshold, swapEnabled);
}
```

Recommendation:
Ensure that the swapThreshold is set to a value greater than a reasonable minimum.

```
function setSwapBackSettings(
    bool _enabled,
    uint256 _amount
) external onlyOwner {
    require(_amount < (totalSupply / 10), "Amount too high");
    require(_amount > 0 , "Amount can't be zero");
    swapEnabled = _enabled;
    swapThreshold = _amount;

    emit config_SwapSettings(swapThreshold, swapEnabled);
}
```

MANUAL TESTING

Logical – Lack of whitelisting function

Severity: High
Function: ---
Lines: ---
Status: Not Resolved
At current implementation of the contract there are no functions to whitelist a wallet from fees, this means some wallets (like presale contract) still have to pay a fixed 10% fee

Recommendation:
add a function to be able to exclude wallets from fees.
Example:

```
function setExcludedFromFees(
    address _wallet,
    bool _status
) public onlyOwner {
    isFeeExempt[_wallet] = _status;
}
```

DISCLAIMER

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