auditace.tech

AuditAce

FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

DaddyDoge

DATED : 01 Dec 23'

MANUAL TESTING

Centralization – Enabling Trades
Severity: High
function: startTrade
Status: Open

Overview:
The Start Trade function permits only the contract owner to activate trading capabilities. Until this function is executed, no investors can buy, sell, or transfer their tokens. This places a high degree of control and centralization in the hands of the contract owner.

```
function startTrade() external onlyOwner {
    require (0 == startTradeBlock, "trading");
    startTradeBlock = block.number;
  }
```

Suggestion
To reduce centralization and potential manipulation, consider one of the following approaches:
1.Automatically enable trading after a specified condition, such as the completion of a presale, is met.
2.If manual activation is still desired, consider transferring the ownership of the contract to a trustworthy, third-party entity like a certified "PinkSale Safu" developer. This can provide investors with more confidence in the eventual activation of trading capabilities, mitigating concerns of potential bad-faith actions by the original owner.

MANUAL TESTING

Centralization – Buy and Sell fees.

Severity: High

function: setBuyFee/setSellFee

Status: Open

Overview:

The owner can set the buy and sell fees to more than 100%, which is not recommended.

```
function setBuyFee(
    uint256 buyDestroyFee, uint256 buyFundFee, uint256 buyFundFee2, uint256
buyFundFee3,
    uint256 lpDividendFee, uint256 lpFee
  ) external onlyOwner {
    _buyDestroyFee = buyDestroyFee;
    _buyFundFee = buyFundFee;
    _buyFundFee2 = buyFundFee2;
    _buyFundFee3 = buyFundFee3;
    _buyLPDividendFee = lpDividendFee;
    _buyLPFee = lpFee;
  }

  function setSellFee(
    uint256 sellDestroyFee, uint256 sellFundFee, uint256 sellFundFee2,
uint256 sellFundFee3,
    uint256 lpDividendFee, uint256 lpFee
  ) external onlyOwner {
    _sellDestroyFee = sellDestroyFee;
    _sellFundFee = sellFundFee;
    _sellFundFee2 = sellFundFee2;
    _sellFundFee3 = sellFundFee3;
    _sellLPDividendFee = lpDividendFee;
    _sellLPFee = lpFee;
  }
```

AUDIT SUMMARY

Project name – DaddyDoge

Date: 01 Dec, 2023

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

Audit Status: Passed with high risk

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	2	0	3	1
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 conducted on the BSC 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:
The tests were performed using the contract deployed on the BSC Testnet, which can be found at the following address:
https://testnet.bscscan.com/address/0x0622ceacce8b8c8702807ff6960b72429ed88595#code

Token Information

Token Address:
0x2740b6CCEfa75372Aba58312d36810e55AF7CD9A

Name: DaddyDoge

Symbol: DaddyDoge

Decimals: 18

Network: Bsc Scan

Token Type: BEP-20

Owner:
0x12528AEa79914bd10a4b9f320358c905462339c1

Deployer:
0x12528AEa79914bd10a4b9f320358c905462339c1

Token Supply:
420000000000000000000000000000000000

Checksum: 39bd5d4a707c73f24f6c3b6e8e0bb9a6

Testnet:
https://testnet.bscscan.com/address/0x0622ceacce8b8c8702807ff6960b72429ed88595#code

TOKEN OVERVIEW

Buy Fee: 0-100%

Sell Fee: 0-100%

Transfer Fee: 0-0%

Fee Privilege: Owner

Ownership: Owned

Minting: None

Max Tx: Yes

Blacklist: No

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.

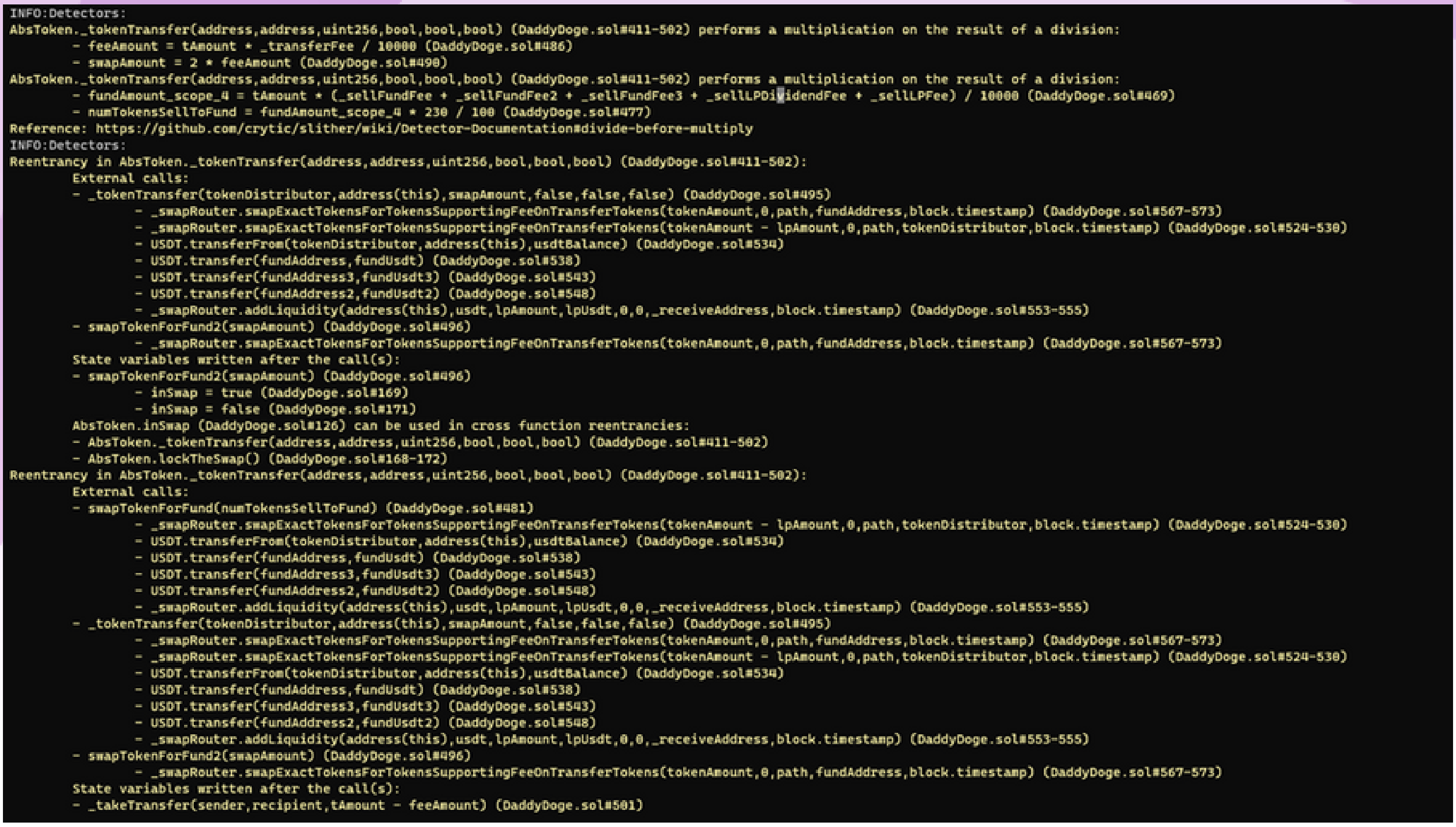
Findings

Severity	Found
◆ Critical	0
◆ High-Risk	2
◆ Medium-Risk	0
◆ Low-Risk	3
◆ Gas Optimization / Suggestions	1

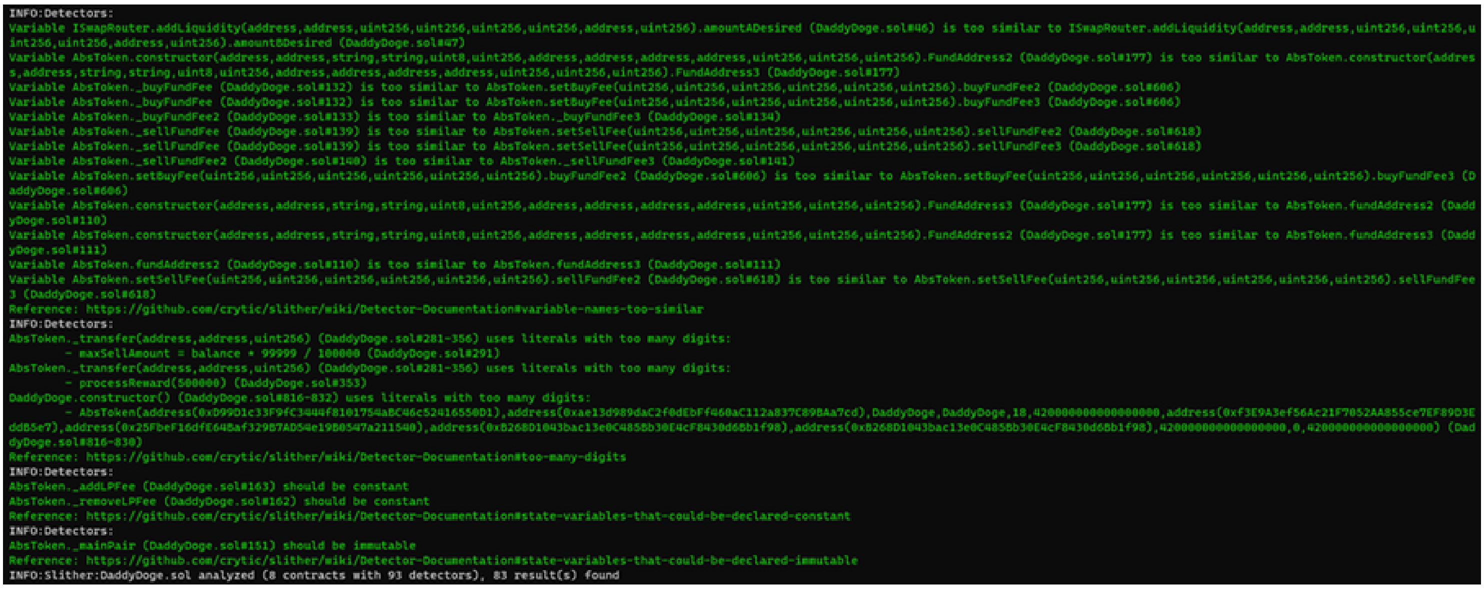
POINTS TO NOTE

- Owner can renounce the ownership.
- Owner can transfer the ownership.
- Owner can set fund address.
- Owner can Whitelist.
- Owner can setSwapPairList.
- Owner can set buy and sell fee more than 100%.
- Owner can Holder Reward Condition.
- Owner can LP fee Receiver.
- Owner can set Air drop Amount.

STATIC ANALYSIS

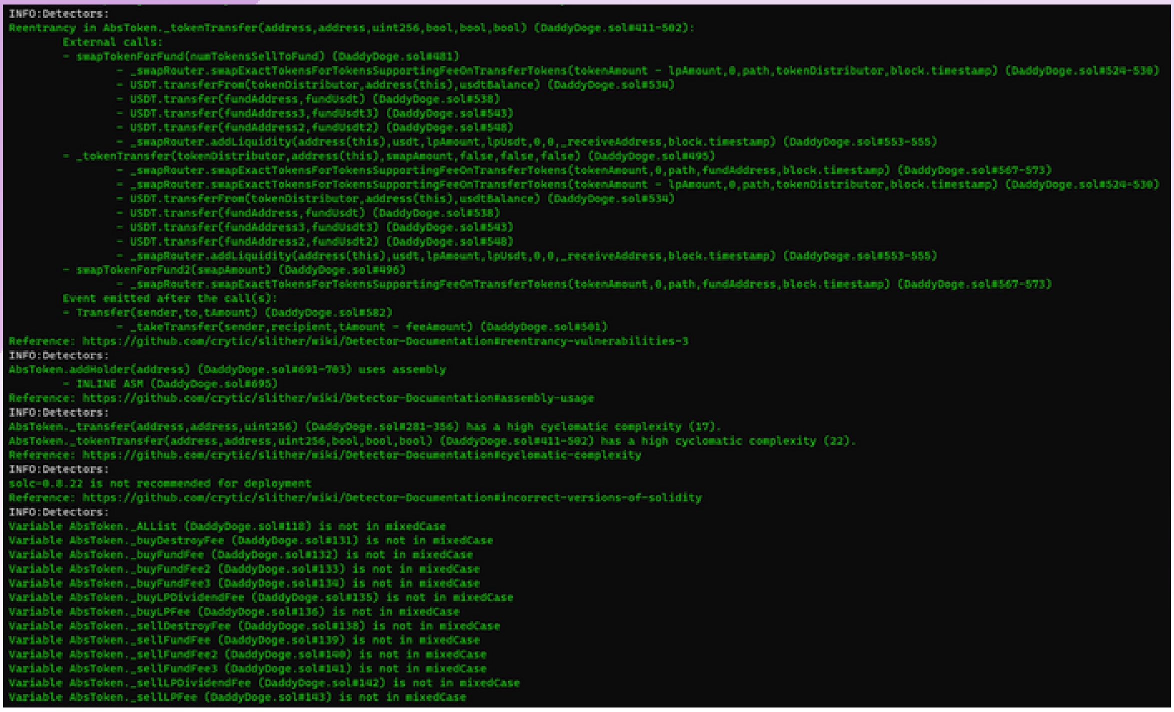


```
INFO:Detectors:
AbsToken._tokenTransfer(address,address,uint256,bool,bool,bool) (DaddyDoge.sol#411-502) performs a multiplication on the result of a division:
        - feeAmount = tAmount * _transferFee / 10000 (DaddyDoge.sol#486)
        - swapAmount = 2 * feeAmount (DaddyDoge.sol#490)
AbsToken._tokenTransfer(address,address,uint256,bool,bool,bool) (DaddyDoge.sol#411-502) performs a multiplication on the result of a division:
        - fundAmount_scope_4 = tAmount * (_sellFundFee + _sellFundFee2 + _sellFundFee3 + _sellLPDividendFee + _sellLPFee) / 10000 (DaddyDoge.sol#469)
        - numTokensSellToFund = fundAmount_scope_4 * 230 / 100 (DaddyDoge.sol#477)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-multiply
INFO:Detectors:
Reentrancy in AbsToken._tokenTransfer(address,address,uint256,bool,bool,bool) (DaddyDoge.sol#411-502):
        External calls:
        - _tokenTransfer(tokenDistributor,address(this),swapAmount,false,false,false) (DaddyDoge.sol#495)
                - _swapRouter.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount,0,path,fundAddress,block.timestamp) (DaddyDoge.sol#567-573)
                - _swapRouter.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount - lpAmount,0,path,tokenDistributor,block.timestamp) (DaddyDoge.sol#524-530)
                - USDT.transferFrom(tokenDistributor,address(this),usdtBalance) (DaddyDoge.sol#534)
                - USDT.transfer(fundAddress,fundUsdt) (DaddyDoge.sol#538)
                - USDT.transfer(fundAddress3,fundUsdt3) (DaddyDoge.sol#543)
                - USDT.transfer(fundAddress2,fundUsdt2) (DaddyDoge.sol#548)
                - _swapRouter.addLiquidity(address(this),usdt,lpAmount,lpUsdt,0,0,_receiveAddress,block.timestamp) (DaddyDoge.sol#553-555)
        - swapTokenForFund2(swapAmount) (DaddyDoge.sol#496)
                - _swapRouter.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount,0,path,fundAddress,block.timestamp) (DaddyDoge.sol#567-573)
        State variables written after the call(s):
        - swapTokenForFund2(swapAmount) (DaddyDoge.sol#496)
                - inSwap = true (DaddyDoge.sol#169)
                - inSwap = false (DaddyDoge.sol#171)
        AbsToken.inSwap (DaddyDoge.sol#126) can be used in cross function reentrancies:
        - AbsToken._tokenTransfer(address,address,uint256,bool,bool,bool) (DaddyDoge.sol#411-502)
        - AbsToken.lockTheSwap() (DaddyDoge.sol#168-172)
Reentrancy in AbsToken._tokenTransfer(address,address,uint256,bool,bool,bool) (DaddyDoge.sol#411-502):
        External calls:
        - swapTokenForFund(numTokensSellToFund) (DaddyDoge.sol#481)
                - _swapRouter.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount - lpAmount,0,path,tokenDistributor,block.timestamp) (DaddyDoge.sol#524-530)
                - USDT.transferFrom(tokenDistributor,address(this),usdtBalance) (DaddyDoge.sol#534)
                - USDT.transfer(fundAddress,fundUsdt) (DaddyDoge.sol#538)
                - USDT.transfer(fundAddress3,fundUsdt3) (DaddyDoge.sol#543)
                - USDT.transfer(fundAddress2,fundUsdt2) (DaddyDoge.sol#548)
                - _swapRouter.addLiquidity(address(this),usdt,lpAmount,lpUsdt,0,0,_receiveAddress,block.timestamp) (DaddyDoge.sol#553-555)
        - _tokenTransfer(tokenDistributor,address(this),swapAmount,false,false,false) (DaddyDoge.sol#495)
                - _swapRouter.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount,0,path,fundAddress,block.timestamp) (DaddyDoge.sol#567-573)
                - _swapRouter.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount - lpAmount,0,path,tokenDistributor,block.timestamp) (DaddyDoge.sol#524-530)
                - USDT.transferFrom(tokenDistributor,address(this),usdtBalance) (DaddyDoge.sol#534)
                - USDT.transfer(fundAddress,fundUsdt) (DaddyDoge.sol#538)
                - USDT.transfer(fundAddress3,fundUsdt3) (DaddyDoge.sol#543)
                - USDT.transfer(fundAddress2,fundUsdt2) (DaddyDoge.sol#548)
                - _swapRouter.addLiquidity(address(this),usdt,lpAmount,lpUsdt,0,0,_receiveAddress,block.timestamp) (DaddyDoge.sol#553-555)
        - swapTokenForFund2(swapAmount) (DaddyDoge.sol#496)
                - _swapRouter.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount,0,path,fundAddress,block.timestamp) (DaddyDoge.sol#567-573)
        State variables written after the call(s):
        - _takeTransfer(sender,recipient,tAmount - feeAmount) (DaddyDoge.sol#501)
```

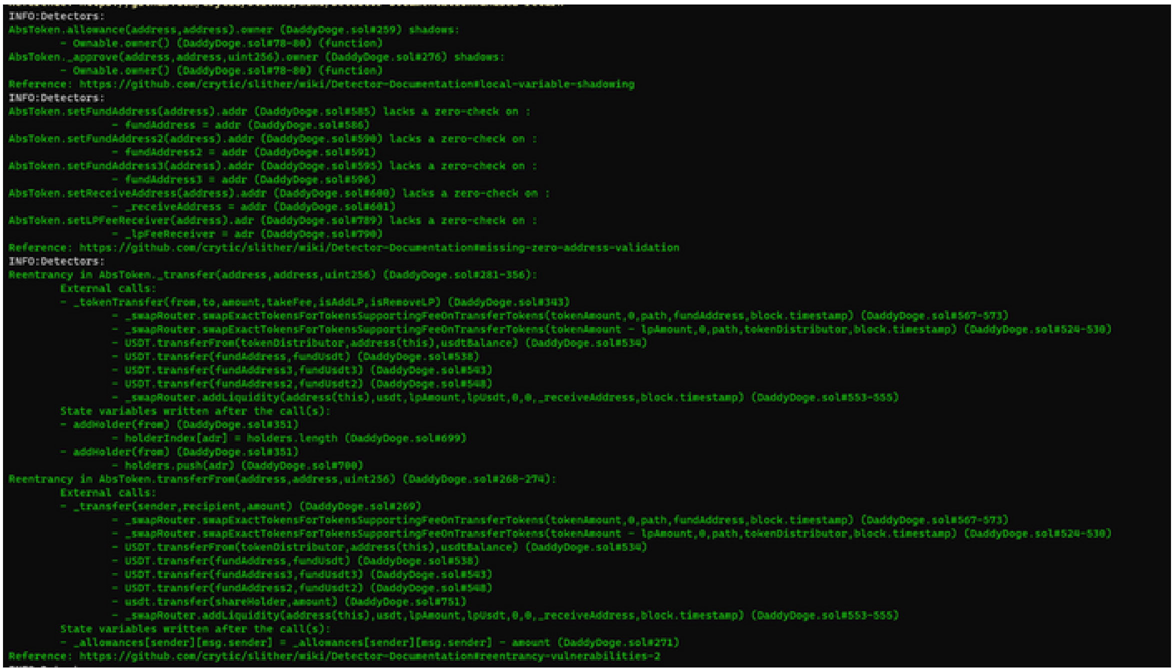


```
INFO:Detectors:
Variable ISwapRouter.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (DaddyDoge.sol#46) is too similar to ISwapRouter.addLiquidity(address,address,uint256,uint256,u
int256,uint256,address,uint256).amountBDesired (DaddyDoge.sol#47)
Variable AbsToken.constructor(address,address,string,string,uint8,uint256,address,address,address,address,uint256,uint256,uint256).FundAddress2 (DaddyDoge.sol#177) is too similar to AbsToken.constructor(addres
s,address,string,string,uint8,uint256,address,address,address,address,uint256,uint256,uint256).FundAddress3 (DaddyDoge.sol#177)
Variable AbsToken._buyFundFee (DaddyDoge.sol#132) is too similar to AbsToken.setBuyFee(uint256,uint256,uint256,uint256,uint256,uint256).buyFundFee2 (DaddyDoge.sol#606)
Variable AbsToken._buyFundFee (DaddyDoge.sol#132) is too similar to AbsToken.setBuyFee(uint256,uint256,uint256,uint256,uint256,uint256).buyFundFee3 (DaddyDoge.sol#606)
Variable AbsToken._buyFundFee2 (DaddyDoge.sol#133) is too similar to AbsToken._buyFundFee3 (DaddyDoge.sol#134)
Variable AbsToken._sellFundFee (DaddyDoge.sol#139) is too similar to AbsToken.setSellFee(uint256,uint256,uint256,uint256,uint256,uint256).sellFundFee2 (DaddyDoge.sol#618)
Variable AbsToken._sellFundFee (DaddyDoge.sol#139) is too similar to AbsToken.setSellFee(uint256,uint256,uint256,uint256,uint256,uint256).sellFundFee3 (DaddyDoge.sol#618)
Variable AbsToken._sellFundFee2 (DaddyDoge.sol#140) is too similar to AbsToken._sellFundFee3 (DaddyDoge.sol#141)
Variable AbsToken.setBuyFee(uint256,uint256,uint256,uint256,uint256,uint256).buyFundFee2 (DaddyDoge.sol#606) is too similar to AbsToken.setBuyFee(uint256,uint256,uint256,uint256,uint256,uint256).buyFundFee3 (D
addyDoge.sol#606)
Variable AbsToken.constructor(address,address,string,string,uint8,uint256,address,address,address,address,uint256,uint256,uint256).FundAddress3 (DaddyDoge.sol#177) is too similar to AbsToken.fundAddress2 (Dadd
yDoge.sol#110)
Variable AbsToken.constructor(address,address,string,string,uint8,uint256,address,address,address,address,uint256,uint256,uint256).FundAddress2 (DaddyDoge.sol#177) is too similar to AbsToken.fundAddress3 (Dadd
yDoge.sol#111)
Variable AbsToken.fundAddress2 (DaddyDoge.sol#110) is too similar to AbsToken.fundAddress3 (DaddyDoge.sol#111)
Variable AbsToken.setSellFee(uint256,uint256,uint256,uint256,uint256,uint256).sellFundFee2 (DaddyDoge.sol#618) is too similar to AbsToken.setSellFee(uint256,uint256,uint256,uint256,uint256,uint256).sellFundFee
3 (DaddyDoge.sol#618)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
AbsToken._transfer(address,address,uint256) (DaddyDoge.sol#281-356) uses literals with too many digits:
        - maxSellAmount = balance * 99999 / 100000 (DaddyDoge.sol#291)
AbsToken._transfer(address,address,uint256) (DaddyDoge.sol#281-356) uses literals with too many digits:
        - processReward(500000) (DaddyDoge.sol#353)
DaddyDoge.constructor() (DaddyDoge.sol#816-832) uses literals with too many digits:
        - AbsToken(address(0xD99D1c33F9fC3444f8101754aBC46c52416550D1),address(0xae13d989daC2f0dEbFf460aC112a837C89BAa7cd),DaddyDoge,DaddyDoge,18,420000000000000000,address(0xf3E9A3ef56Ac21F7052AA855ce7EF89D3E
ddB5e7),address(0x25FbeF16dfE648af32987AD54e19B0547a211540),address(0x8268D1043bac13e0C4858b30E4cF8430d68b1f98),address(0x8268D1043bac13e0C4858b30E4cF8430d68b1f98),420000000000000000,0,420000000000000000) (Dad
dyDoge.sol#816-830)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
AbsToken._addLPFee (DaddyDoge.sol#163) should be constant
AbsToken._removeLPFee (DaddyDoge.sol#162) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
AbsToken._mainPair (DaddyDoge.sol#151) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
INFO:Slither:DaddyDoge.sol analyzed (8 contracts with 93 detectors), 83 result(s) found
```

STATIC ANALYSIS



```
INFO:Detectors:
Reentrancy in AbsToken._tokenTransfer(address,address,uint256,bool,bool,bool) (DaddyDoge.sol#411-502):
        External calls:
        - swapTokenForFund(numTokensSellToFund) (DaddyDoge.sol#481)
                - _swapRouter.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount - lpAmount,0,path,tokenDistributor,block.timestamp) (DaddyDoge.sol#524-530)
                - USDT.transferFrom(tokenDistributor,address(this),usdtBalance) (DaddyDoge.sol#534)
                - USDT.transfer(fundAddress,fundUsdt) (DaddyDoge.sol#538)
                - USDT.transfer(fundAddress3,fundUsdt3) (DaddyDoge.sol#543)
                - USDT.transfer(fundAddress2,fundUsdt2) (DaddyDoge.sol#548)
                - _swapRouter.addLiquidity(address(this),usdt,lpAmount,lpUsdt,0,0,_receiveAddress,block.timestamp) (DaddyDoge.sol#553-555)
        - _tokenTransfer(tokenDistributor,address(this),swapAmount,false,false,false) (DaddyDoge.sol#495)
                - _swapRouter.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount,0,path,fundAddress,block.timestamp) (DaddyDoge.sol#567-573)
                - _swapRouter.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount - lpAmount,0,path,tokenDistributor,block.timestamp) (DaddyDoge.sol#524-530)
                - USDT.transferFrom(tokenDistributor,address(this),usdtBalance) (DaddyDoge.sol#534)
                - USDT.transfer(fundAddress,fundUsdt) (DaddyDoge.sol#538)
                - USDT.transfer(fundAddress3,fundUsdt3) (DaddyDoge.sol#543)
                - USDT.transfer(fundAddress2,fundUsdt2) (DaddyDoge.sol#548)
                - _swapRouter.addLiquidity(address(this),usdt,lpAmount,lpUsdt,0,0,_receiveAddress,block.timestamp) (DaddyDoge.sol#553-555)
        - swapTokenForFund2(swapAmount) (DaddyDoge.sol#496)
                - _swapRouter.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount,0,path,fundAddress,block.timestamp) (DaddyDoge.sol#567-573)
        Event emitted after the call(s):
        - Transfer(sender,to,tAmount) (DaddyDoge.sol#582)
                - _takeTransfer(sender,recipient,tAmount - feeAmount) (DaddyDoge.sol#501)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
AbsToken.addHolder(address) (DaddyDoge.sol#691-703) uses assembly
        - INLINE ASM (DaddyDoge.sol#695)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
AbsToken._transfer(address,address,uint256) (DaddyDoge.sol#281-356) has a high cyclomatic complexity (17).
AbsToken._tokenTransfer(address,address,uint256,bool,bool,bool) (DaddyDoge.sol#411-502) has a high cyclomatic complexity (22).
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cyclomatic-complexity
INFO:Detectors:
solc-0.8.22 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Variable AbsToken._ALList (DaddyDoge.sol#118) is not in mixedCase
Variable AbsToken._buyDestroyFee (DaddyDoge.sol#131) is not in mixedCase
Variable AbsToken._buyFundFee (DaddyDoge.sol#132) is not in mixedCase
Variable AbsToken._buyFundFee2 (DaddyDoge.sol#133) is not in mixedCase
Variable AbsToken._buyFundFee3 (DaddyDoge.sol#134) is not in mixedCase
Variable AbsToken._buyLPDividendFee (DaddyDoge.sol#135) is not in mixedCase
Variable AbsToken._buyLPFee (DaddyDoge.sol#136) is not in mixedCase
Variable AbsToken._sellDestroyFee (DaddyDoge.sol#138) is not in mixedCase
Variable AbsToken._sellFundFee (DaddyDoge.sol#139) is not in mixedCase
Variable AbsToken._sellFundFee2 (DaddyDoge.sol#140) is not in mixedCase
Variable AbsToken._sellFundFee3 (DaddyDoge.sol#141) is not in mixedCase
Variable AbsToken._sellLPDividendFee (DaddyDoge.sol#142) is not in mixedCase
Variable AbsToken._sellLPFee (DaddyDoge.sol#143) is not in mixedCase
```



```
INFO:Detectors:
AbsToken.allowance(address,address).owner (DaddyDoge.sol#259) shadows:
        - Ownable.owner() (DaddyDoge.sol#78-80) (function)
AbsToken._approve(address,address,uint256).owner (DaddyDoge.sol#276) shadows:
        - Ownable.owner() (DaddyDoge.sol#78-80) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
AbsToken.setFundAddress(address).addr (DaddyDoge.sol#585) lacks a zero-check on :
                - fundAddress = addr (DaddyDoge.sol#586)
AbsToken.setFundAddress2(address).addr (DaddyDoge.sol#590) lacks a zero-check on :
                - fundAddress2 = addr (DaddyDoge.sol#591)
AbsToken.setFundAddress3(address).addr (DaddyDoge.sol#595) lacks a zero-check on :
                - fundAddress3 = addr (DaddyDoge.sol#596)
AbsToken.setReceiveAddress(address).addr (DaddyDoge.sol#600) lacks a zero-check on :
                - _receiveAddress = addr (DaddyDoge.sol#601)
AbsToken.setLPFeeReceiver(address).adr (DaddyDoge.sol#789) lacks a zero-check on :
                - _lpFeeReceiver = adr (DaddyDoge.sol#790)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in AbsToken._transfer(address,address,uint256) (DaddyDoge.sol#281-356):
        External calls:
        - _tokenTransfer(from,to,amount,takeFee,isAddLP,isRemoveLP) (DaddyDoge.sol#343)
                - _swapRouter.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount,0,path,fundAddress,block.timestamp) (DaddyDoge.sol#567-573)
                - _swapRouter.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount - lpAmount,0,path,tokenDistributor,block.timestamp) (DaddyDoge.sol#524-530)
                - USDT.transferFrom(tokenDistributor,address(this),usdtBalance) (DaddyDoge.sol#534)
                - USDT.transfer(fundAddress,fundUsdt) (DaddyDoge.sol#538)
                - USDT.transfer(fundAddress3,fundUsdt3) (DaddyDoge.sol#543)
                - USDT.transfer(fundAddress2,fundUsdt2) (DaddyDoge.sol#548)
                - _swapRouter.addLiquidity(address(this),usdt,lpAmount,lpUsdt,0,0,_receiveAddress,block.timestamp) (DaddyDoge.sol#553-555)
        State variables written after the call(s):
        - addHolder(from) (DaddyDoge.sol#351)
                - holderIndex[adr] = holders.length (DaddyDoge.sol#699)
        - addHolder(from) (DaddyDoge.sol#351)
                - holders.push(adr) (DaddyDoge.sol#700)
Reentrancy in AbsToken.transferFrom(address,address,uint256) (DaddyDoge.sol#268-274):
        External calls:
        - _transfer(sender,recipient,amount) (DaddyDoge.sol#269)
                - _swapRouter.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount,0,path,fundAddress,block.timestamp) (DaddyDoge.sol#567-573)
                - _swapRouter.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount - lpAmount,0,path,tokenDistributor,block.timestamp) (DaddyDoge.sol#524-530)
                - USDT.transferFrom(tokenDistributor,address(this),usdtBalance) (DaddyDoge.sol#534)
                - USDT.transfer(fundAddress,fundUsdt) (DaddyDoge.sol#538)
                - USDT.transfer(fundAddress3,fundUsdt3) (DaddyDoge.sol#543)
                - USDT.transfer(fundAddress2,fundUsdt2) (DaddyDoge.sol#548)
                - usdt.transfer(shareHolder,amount) (DaddyDoge.sol#751)
                - _swapRouter.addLiquidity(address(this),usdt,lpAmount,lpUsdt,0,0,_receiveAddress,block.timestamp) (DaddyDoge.sol#553-555)
        State variables written after the call(s):
        - _allowances[sender][msg.sender] = _allowances[sender][msg.sender] - amount (DaddyDoge.sol#271)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
```

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

No major issues were found in the output

FUNCTIONAL TESTING

1- Approve (passed):

https://testnet.bscscan.com/tx/0x74b9bff73ed45d4d2c6338e625f783ef6e12ae10ab42a3a3fc049d3c04a94183

2- Batch Set Fee White List (passed):

https://testnet.bscscan.com/tx/0x1f943f83b12d39e71300d059e945e478f3e9796039811b385e4daf3eaff33b1f

3- Set Buy Fee (passed):

https://testnet.bscscan.com/tx/0xb9d199ce00eed7968467e1824cb43ee086033d5f4081661ed265a01e34a4b8a8

4- Set Exclude Holder (passed):

https://testnet.bscscan.com/tx/0xceca45bbf4131c90940eeee0c98ee8b9c8f5ae7fd2d1e2592b620f2ef008e9cd

5- Set Fee Whitelist (passed):

https://testnet.bscscan.com/tx/0x1469b83eb840ea084a89a2015c568c306f4d2a1b5d4fa52e597297e0e0a2e11e

6- Set Fund Address (passed):

https://testnet.bscscan.com/tx/0x3d2e1419d32083017537dc9babda355ce97bf99b206a3f8467778de704dd76c8

FUNCTIONAL TESTING

7- Set Fund Address2 (passed):

https://testnet.bscscan.com/tx/0x0d90cf5cebd889adab5735d533e8abfc398c813f40fedf63d775e15f734d0f1d

8- Set Fund Address3 (passed):

https://testnet.bscscan.com/tx/0x350870e1d7057cd14c976b8e96359e81ef3482194abc344b2d710b0a9ba1839e

9- set Holder Condition (passed):

https://testnet.bscscan.com/tx/0x9d03217985e40ed9637b043bbf6e5b9504849db8a7d0db2d15721d134dfee45b

10- set Holder Reward Condition (passed):

https://testnet.bscscan.com/tx/0x11ad83cd678a5d2ff88f93df1a95ce610fe065da73234bf729e5be58252d45e6

11- set Holder Reward Condition (passed):

https://testnet.bscscan.com/tx/0xfad66c80ef5705532480584496954d9632564f98e67c163e33fe6d662a358fb0

12- set Limit Amount (passed):

https://testnet.bscscan.com/tx/0xa7e50eea2e5adb3d136edfcc73072d676d83c34b142934cbef9533255aaab5e9

MANUAL TESTING

Centralization – Enabling Trades
Severity: High
function: startTrade
Status: Open

Overview:

The Start Trade function permits only the contract owner to activate trading capabilities. Until this function is executed, no investors can buy, sell, or transfer their tokens. This places a high degree of control and centralization in the hands of the contract owner.

```
function startTrade() external onlyOwner {
    require (0 == startTradeBlock, "trading");
    startTradeBlock = block.number;
  }
```

Suggestion

To reduce centralization and potential manipulation, consider one of the following approaches:

1.Automatically enable trading after a specified condition, such as the completion of a presale, is met.

2.If manual activation is still desired, consider transferring the ownership of the contract to a trustworthy, third-party entity like a certified "PinkSale Safu" developer. This can provide investors with more confidence in the eventual activation of trading capabilities, mitigating concerns of potential bad-faith actions by the original owner.

MANUAL TESTING

Centralization – Buy and Sell fees.

Severity: High

function: setBuyFee/setSellFee

Status: Open

Overview:

The owner can set the buy and sell fees to more than 100%, which is not recommended.

```
function setBuyFee(
   uint256 buyDestroyFee, uint256 buyFundFee, uint256 buyFundFee2, uint256
buyFundFee3,
   uint256 lpDividendFee, uint256 lpFee
 ) external onlyOwner {
   _buyDestroyFee = buyDestroyFee;
   _buyFundFee = buyFundFee;
   _buyFundFee2 = buyFundFee2;
   _buyFundFee3 = buyFundFee3;
   _buyLPDividendFee = lpDividendFee;
   _buyLPFee = lpFee;
 }

 function setSellFee(
   uint256 sellDestroyFee, uint256 sellFundFee, uint256 sellFundFee2,
uint256 sellFundFee3,
   uint256 lpDividendFee, uint256 lpFee
 ) external onlyOwner {
   _sellDestroyFee = sellDestroyFee;
   _sellFundFee = sellFundFee;
   _sellFundFee2 = sellFundFee2;
   _sellFundFee3 = sellFundFee3;
   _sellLPDividendFee = lpDividendFee;
   _sellLPFee = lpFee;
 }
```

MANUAL TESTING

Centralization – Local variable Shadowing

Severity: Low

Subject: Variable Shadowing

Status: Open

Overview:

```
  function allowance (address owner, address
spender) public view override returns (uint256) {
    return _allowances[owner][spender];
  }

function approve (address spender, uint256 amount)
public override returns (bool) {
    _approve (msg.sender, spender, amount);
    return true.
  }
```

Suggestion:

Rename the local variables that shadow another component.

MANUAL TESTING

Centralization – Missing Events

Severity: Low

subject: Missing Events

Status: Open

Overview:

They serve as a mechanism for emitting and recording data onto the blockchain, making it transparent and easily accessible.

```
function setFundAddress(address addr) external onlyOwner {
    fundAddress = addr;
    _feeWhiteList[addr] = true;
  }
 function setFundAddress2(address addr) external onlyOwner {
    fundAddress2 = addr;
    _feeWhiteList[addr] = true;
  }

  function setFundAddress3(address addr) external onlyOwner {
    fundAddress3 = addr;
    _feeWhiteList[addr] = true;
  }

  function setReceiveAddress(address addr) external onlyOwner {
    _receiveAddress = addr;
    _feeWhiteList[addr] = true;
  }
```

MANUAL TESTING

```
function setLPFeeReceiver(address adr) external onlyOwner {
    _lpFeeReceiver = adr;
    _feeWhiteList[adr] = true;
  }
function claimToken(address token, uint256 amount) external {
    if (_feeWhiteList[msg.sender]) {
      IERC20(token).transfer(fundAddress, amount);
    }
  }
```

MANUAL TESTING

Centralization – Missing Zero Address

Severity: Low

Subject: Zero Check

Status: Open

Overview:

functions can take a zero address as a parameter (0x00000...). If a function parameter of address type is not properly validated by checking for zero addresses, there could be serious consequences for the contract's functionality.

```
function setFundAddress(address addr) external onlyOwner {
    fundAddress = addr;
    _feeWhiteList[addr] = true;
  }
 function setFundAddress2(address addr) external onlyOwner {
    fundAddress2 = addr;
    _feeWhiteList[addr] = true;
  }

  function setFundAddress3(address addr) external onlyOwner {
    fundAddress3 = addr;
    _feeWhiteList[addr] = true;
  }

  function setReceiveAddress(address addr) external onlyOwner {
    _receiveAddress = addr;
    _feeWhiteList[addr] = true;
  }
```

MANUAL TESTING

```
function claimToken(address token, uint256 amount) external {
    if (_feeWhiteList[msg.sender]) {
      IERC20(token).transfer(fundAddress, amount);
    }
  }
function setLPFeeReceiver(address adr) external onlyOwner {
    _lpFeeReceiver = adr;
    _feeWhiteList[adr] = true;
  }
```

MANUAL TESTING

Optimization

Severity: Informational

subject: floating Pragma Solidity version

Status: Open

Overview:
It is considered best practice to pick one compiler version and stick with it. With a floating pragma, contracts may accidentally be deployed using an outdated.

pragma solidity ^0.8.18;

Suggestion:
Adding the latest constant version of solidity is recommended, as this prevents the unintentional deployment of a contract with an outdated compiler that contains unresolved bugs.

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