AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

Siena Chain

DATED : 04 Dec 23'

MANUAL TESTING

Centralization – The owner can lock funds.
Severity: High
function: _transfer
Status: Open

Overview:
In the _transfer function, the owner can set any arbitrary value in the max wallet amount. If it is set to zero, the user cannot hold any amount which is not recommended.

```
function EnableTrading() external onlyOwner {
    require(!swapTokenEnabled, "Cannot re-enable trading");
    swapTokenEnabled = true;
  }
```

Suggestion:
In the _transfer function, there must be a required check.

MANUAL TESTING

Centralization – Buy and Sell fees.

Severity: High

function: setBuyTaxes and setSellTaxes

Status: Open

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

```
function setBuyTaxes(uint256 newLiquidityTax, uint256
newMarketingTax, uint256 newBuyBackTax) external onlyOwner() {
    _buyLiquidityFee = newLiquidityTax;
    _buyMarketingFee = newMarketingTax;
    _buyBuyBackFee = newBuyBackTax;

    _totalTaxIfBuying =
_buyLiquidityFee.add(_buyMarketingFee).add(_buyBuyBackFee);
  }

  function setSellTaxes(uint256 newLiquidityTax, uint256
newMarketingTax, uint256 newBuyBackTax) external onlyOwner() {
    _sellLiquidityFee = newLiquidityTax;
    _sellMarketingFee = newMarketingTax;
    _sellBuyBackFee = newBuyBackTax;

    _totalTaxIfSelling =
_sellLiquidityFee.add(_sellMarketingFee).add(_sellBuyBackFee);
  }
```

Suggestion
It is recommended that no fees in the contract should be more than 25% of the contract.

AUDIT SUMMARY

Project name – Siena Chain

Date: 04 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	1	3	3
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/0x4b6531c9c9cefae19e33918e4453e1fa288f6927#code

Token Information

Token Address:
0x166044b521236d6C0918a1888A39629b10812291

Name: Siena Chain

Symbol: $SIN

Decimals: 9

Network: Binance Smart Chain

Token Type: BEP-20

Owner:
0xc63Ee0b24c874459AD04821Ad4aDD0be4232a0aB

Deployer:
0xc63Ee0b24c874459AD04821Ad4aDD0be4232a0aB

Token Supply: 100000000000000000

Checksum: 2d5d8975b921c928a820860f91bcff4c

Testnet:
https://testnet.bscscan.com/address/0x4b6531c9c9cefae19e33918e4453e1fa288f6927#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

- [x] Return values of low-level calls
- [x] Private modifier
- [x] Multiple Sends
- [x] Using Suicide
- [x] Gas Limitand Loops
- [x] Address hardcoded
- [x] Exception Disorder
- [x] Using inline assembly
- [x] Divide before multiply
- [x] Missing Zero Address Validation
- [x] Compiler version not fixed
- [x] **Gasless Send**
- [x] Using block.timestamp
- [x] Re-entrancy
- [x] Tautology or contradiction
- [x] Timestamp Dependence
- [x] Revert/require functions
- [x] Use of tx.origin
- [x] Integer overflow/underflow
- [x] Dangerous strict equalities
- [x] Using SHA3
- [x] 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	1
Low-Risk	3
Gas Optimization / Suggestions	3

POINTS TO NOTE

- The owner can renounce ownership.
- The owner can transfer ownership.
- The owner can add marketing pair addresses.
- The owner can set buy and sell tax of more than 25%.
- The owner can fee distribution.
- The owner can enable/disable the wallet limit.
- The owner can exempt wallets from the wallet limit.
- The owner can set a marketing wallet address.
- The owner can set buy back wallet address.
- The owner can enable/disable swapping.
- The owner can change the router address

STATIC ANALYSIS

```
INFO:Detectors:
SienaCHAIN.addLiquidity(uint256,uint256) (SienaCHAIN.sol#739-752) ignores return value by uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (SienaCHAIN.so
l#744-751)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
INFO:Detectors:
SienaCHAIN.allowance(address,address).owner (SienaCHAIN.sol#506) shadows:
        - Ownable.owner() (SienaCHAIN.sol#163-165) (function)
SienaCHAIN._approve(address,address,uint256).owner (SienaCHAIN.sol#525) shadows:
        - Ownable.owner() (SienaCHAIN.sol#163-165) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
SienaCHAIN.setBuyTaxes(uint256,uint256,uint256) (SienaCHAIN.sol#545-551) should emit an event for:
        - _totalTaxIfBuying = _buyLiquidityFee.add(_buyMarketingFee).add(_buyBuyBackFee) (SienaCHAIN.sol#550)
SienaCHAIN.setSellTaxes(uint256,uint256,uint256) (SienaCHAIN.sol#553-559) should emit an event for:
        - _totalTaxIfSelling = _sellLiquidityFee.add(_sellMarketingFee).add(_sellBuyBackFee) (SienaCHAIN.sol#558)
SienaCHAIN.setDistributionSettings(uint256,uint256,uint256) (SienaCHAIN.sol#561-567) should emit an event for:
        - _liquidityShare = newLiquidityShare (SienaCHAIN.sol#562)
        - _BuyBackShare = newBuyBackShare (SienaCHAIN.sol#564)
        - _totalDistributionShares = _liquidityShare.add(_marketingShare).add(_BuyBackShare) (SienaCHAIN.sol#566)
SienaCHAIN.setMaxTxAmount(uint256) (SienaCHAIN.sol#569-572) should emit an event for:
        - _maxTxAmount = maxTxAmount (SienaCHAIN.sol#571)
SienaCHAIN.setWalletLimit(uint256) (SienaCHAIN.sol#582-584) should emit an event for:
        - _walletMax = newLimit (SienaCHAIN.sol#583)
SienaCHAIN.setNumTokensBeforeSwap(uint256) (SienaCHAIN.sol#586-588) should emit an event for:
        - minimumTokensBeforeSwap = newLimit (SienaCHAIN.sol#587)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
SienaCHAIN.setMarketingWalletAddress(address).newAddress (SienaCHAIN.sol#590) lacks a zero-check on :
        - marketingWalletAddress = address(newAddress) (SienaCHAIN.sol#591)
SienaCHAIN.setBuyBackWalletAddress(address).newAddress (SienaCHAIN.sol#594) lacks a zero-check on :
        - BuyBackWalletAddress = address(newAddress) (SienaCHAIN.sol#595)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in SienaCHAIN.changeRouterVersion(address) (SienaCHAIN.sol#615-632):
        External calls:
        - newPairAddress = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (SienaCHAIN.sol#623-624)
        State variables written after the call(s):
        - isMarketPair[address(uniswapPair)] = true (SienaCHAIN.sol#631)
        - isWalletLimitExempt[address(uniswapPair)] = true (SienaCHAIN.sol#630)
        - uniswapPair = newPairAddress (SienaCHAIN.sol#627)
        - uniswapV2Router = _uniswapV2Router (SienaCHAIN.sol#628)
Reentrancy in SienaCHAIN.swapAndLiquify(uint256) (SienaCHAIN.sol#695-717):
        External calls:
        - swapTokensForEth(tokensForSwap) (SienaCHAIN.sol#700)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (SienaCHAIN.sol#728-734)
        - addLiquidity(tokensForLP,amountBNBLiquidity) (SienaCHAIN.sol#716)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (SienaCHAIN.sol#744-751)
        External calls sending eth:
        - transferToAddressETH(marketingWalletAddress,amountBNBMarketing) (SienaCHAIN.sol#710)
```

```
INFO:Detectors:
Reentrancy in SienaCHAIN._transfer(address,address,uint256) (SienaCHAIN.sol#648-686):
        External calls:
        - swapAndLiquify(contractTokenBalance) (SienaCHAIN.sol#670)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (SienaCHAIN.sol#744-751)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (SienaCHAIN.sol#728-734)
        External calls sending eth:
        - swapAndLiquify(contractTokenBalance) (SienaCHAIN.sol#670)
                - recipient.transfer(amount) (SienaCHAIN.sol#612)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (SienaCHAIN.sol#744-751)
        Event emitted after the call(s):
        - Transfer(sender,address(this),feeAmount) (SienaCHAIN.sol#767)
                - finalAmount = takeFee(sender,recipient,amount) (SienaCHAIN.sol#675-676)
        - Transfer(sender,recipient,finalAmount) (SienaCHAIN.sol#683)
Reentrancy in SienaCHAIN.swapAndLiquify(uint256) (SienaCHAIN.sol#695-717):
        External calls:
        - swapTokensForEth(tokensForSwap) (SienaCHAIN.sol#700)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (SienaCHAIN.sol#728-734)
        - addLiquidity(tokensForLP,amountBNBLiquidity) (SienaCHAIN.sol#716)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (SienaCHAIN.sol#744-751)
        External calls sending eth:
        - transferToAddressETH(marketingWalletAddress,amountBNBMarketing) (SienaCHAIN.sol#710)
                - recipient.transfer(amount) (SienaCHAIN.sol#612)
        - transferToAddressETH(BuyBackWalletAddress,amountBNBBuyBack) (SienaCHAIN.sol#713)
                - recipient.transfer(amount) (SienaCHAIN.sol#612)
        - addLiquidity(tokensForLP,amountBNBLiquidity) (SienaCHAIN.sol#716)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (SienaCHAIN.sol#744-751)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (SienaCHAIN.sol#530)
                - addLiquidity(tokensForLP,amountBNBLiquidity) (SienaCHAIN.sol#716)
Reentrancy in SienaCHAIN.swapTokensForEth(uint256) (SienaCHAIN.sol#719-737):
        External calls:
        - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (SienaCHAIN.sol#728-734)
        Event emitted after the call(s):
        - SwapTokensForETH(tokenAmount,path) (SienaCHAIN.sol#736)
Reentrancy in SienaCHAIN.transferFrom(address,address,uint256) (SienaCHAIN.sol#642-646):
        External calls:
        - _transfer(sender,recipient,amount) (SienaCHAIN.sol#643)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (SienaCHAIN.sol#744-751)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (SienaCHAIN.sol#728-734)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (SienaCHAIN.sol#643)
                - recipient.transfer(amount) (SienaCHAIN.sol#612)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (SienaCHAIN.sol#744-751)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (SienaCHAIN.sol#530)
                - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (SienaCHAIN.sol#644)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
```

STATIC ANALYSIS

```
INFO:Detectors:
Address._functionCallWithValue(address,bytes,uint256,string) (SienaCHAIN.sol#131-148) is never used and should be removed
Address.functionCall(address,bytes) (SienaCHAIN.sol#114-116) is never used and should be removed
Address.functionCall(address,bytes,string) (SienaCHAIN.sol#118-120) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (SienaCHAIN.sol#122-124) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (SienaCHAIN.sol#126-129) is never used and should be removed
Address.isContract(address) (SienaCHAIN.sol#95-104) is never used and should be removed
Address.sendValue(address,uint256) (SienaCHAIN.sol#106-112) is never used and should be removed
Context._msgData() (SienaCHAIN.sol#22-25) is never used and should be removed
SafeMath.mod(uint256,uint256) (SienaCHAIN.sol#83-85) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (SienaCHAIN.sol#87-90) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.4 (SienaCHAIN.sol#12) allows old versions
solc-0.8.22 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (SienaCHAIN.sol#106-112):
        - (success) = recipient.call{value: amount}() (SienaCHAIN.sol#110)
Low level call in Address._functionCallWithValue(address,bytes,uint256,string) (SienaCHAIN.sol#131-148):
        - (success,returndata) = target.call{value: weiValue}(data) (SienaCHAIN.sol#134)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (SienaCHAIN.sol#215) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (SienaCHAIN.sol#216) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (SienaCHAIN.sol#232) is not in mixedCase
Function IUniswapV2Router01.WETH() (SienaCHAIN.sol#251) is not in mixedCase
Parameter SienaCHAIN.setSwapAndLiquifyEnabled(bool)._enabled (SienaCHAIN.sol#598) is not in mixedCase
Variable SienaCHAIN.BuyBackWalletAddress (SienaCHAIN.sol#394) is not in mixedCase
Variable SienaCHAIN._buyLiquidityFee (SienaCHAIN.sol#405) is not in mixedCase
Variable SienaCHAIN._buyMarketingFee (SienaCHAIN.sol#406) is not in mixedCase
Variable SienaCHAIN._buyBuyBackFee (SienaCHAIN.sol#407) is not in mixedCase
Variable SienaCHAIN._sellLiquidityFee (SienaCHAIN.sol#408) is not in mixedCase
Variable SienaCHAIN._sellMarketingFee (SienaCHAIN.sol#409) is not in mixedCase
Variable SienaCHAIN._sellBuyBackFee (SienaCHAIN.sol#410) is not in mixedCase
Variable SienaCHAIN._liquidityShare (SienaCHAIN.sol#412) is not in mixedCase
Variable SienaCHAIN._marketingShare (SienaCHAIN.sol#413) is not in mixedCase
Variable SienaCHAIN._BuyBackShare (SienaCHAIN.sol#414) is not in mixedCase
Variable SienaCHAIN._totalTaxIfBuying (SienaCHAIN.sol#416) is not in mixedCase
Variable SienaCHAIN._totalTaxIfSelling (SienaCHAIN.sol#417) is not in mixedCase
Variable SienaCHAIN._totalDistributionShares (SienaCHAIN.sol#418) is not in mixedCase
Variable SienaCHAIN._maxTxAmount (SienaCHAIN.sol#421) is not in mixedCase
Variable SienaCHAIN._walletMax (SienaCHAIN.sol#422) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

```
INFO:Detectors:
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (SienaCHAIN.sol#256) is too similar to IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (SienaCHAIN.sol#257)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
SienaCHAIN._decimals (SienaCHAIN.sol#391) should be constant
SienaCHAIN._name (SienaCHAIN.sol#389) should be constant
SienaCHAIN._symbol (SienaCHAIN.sol#390) should be constant
SienaCHAIN._totalSupply (SienaCHAIN.sol#420) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Slither:SienaCHAIN.sol analyzed (10 contracts with 93 detectors), 65 result(s) found
```

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/0xf35ef93743fb8b8747e613c7e987a4cac45897c8192ffa03b2809cf781c97994

2- Add Market Pair (passed):

https://testnet.bscscan.com/tx/0xf09d21ecffa94627f12740203a5faa40eefeeff712ce0e39076a374963489ac6

3- Change Router Version (passed):

https://testnet.bscscan.com/tx/0xc6c6f76554677e560fc69adc2a30d3d5b8ba0dc39834318550044445d853e227c

4- Increase Allowance (passed):

https://testnet.bscscan.com/tx/0xff4b203f0951e0e359c6c241b2614f087603658b6d562fcc56d6625bb9143c8b

5- Decrease Allowance (passed):

https://testnet.bscscan.com/tx/0x641a5f692fe2dd5a5fbb6e11befefe8d1a1abc2ca1f2a6c23f5ce181fb1008b9

6- Set Buy Back Wallet Address (passed):

https://testnet.bscscan.com/tx/0x93af9e223d11a49d2108cc5130fbf1c3e9f30947acdfc9cae2d3c943281eca90

FUNCTIONAL TESTING

7- Set Buy Taxes(passed):

https://testnet.bscscan.com/tx/0x6eb5c8176590b69702098bc38064f4e0920ed140e786185ef9d76ca74aa78c15

8- Set Transfer Buy Fee (passed):

https://testnet.bscscan.com/tx/0xbe1fb4a6e73317a29026acd6bc9aec96f7d3662c133496de65c33917072ab074

9- Set Distribution Settings (passed):

https://testnet.bscscan.com/tx/0x90aaa6c0dae8dc2564b09a3ec43d4cf1189624ef8a04541464f54c19aee6a656

10- Set Is Excluded From Fee (passed):

https://testnet.bscscan.com/tx/0xb72a3fec922e012a22cf3783d0b48e2ae85e287700f71debc3501cdce1448aca

MANUAL TESTING

Centralization – The owner can lock funds.
Severity: High
function: _transfer
Status: Open

Overview:
In the _transfer function, the owner can set any arbitrary value in the max wallet amount. If it is set to zero, the user cannot hold any amount which is not recommended.

```
function EnableTrading() external onlyOwner {
    require(!swapTokenEnabled, "Cannot re-enable trading");
    swapTokenEnabled = true;
  }
```

Suggestion:
In the _transfer function, there must be a required check.

MANUAL TESTING

Centralization – Buy and Sell fees.

Severity: High

function: setBuyTaxes and setSellTaxes

Status: Open

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

```
function setBuyTaxes(uint256 newLiquidityTax, uint256
newMarketingTax, uint256 newBuyBackTax) external onlyOwner() {
    _buyLiquidityFee = newLiquidityTax;
    _buyMarketingFee = newMarketingTax;
    _buyBuyBackFee = newBuyBackTax;

    _totalTaxIfBuying =
_buyLiquidityFee.add(_buyMarketingFee).add(_buyBuyBackFee);
  }

  function setSellTaxes(uint256 newLiquidityTax, uint256
newMarketingTax, uint256 newBuyBackTax) external onlyOwner() {
    _sellLiquidityFee = newLiquidityTax;
    _sellMarketingFee = newMarketingTax;
    _sellBuyBackFee = newBuyBackTax;

    _totalTaxIfSelling =
_sellLiquidityFee.add(_sellMarketingFee).add(_sellBuyBackFee);
  }
```

Suggestion
It is recommended that no fees in the contract should be more than 25% of the contract.

MANUAL TESTING

Centralization – Liquidity is added to EOA.

Severity: Medium

function: addLiquidity

Status: Open

Overview:

Liquidity is added to EOA. It may be drained by the addLiquidityETH.

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

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

MANUAL TESTING

Centralization – Missing Visibility
Severity: Low
subject: Missing Visibility
Status: Open

Overview:
It's simply saying that no visibility was specified, so it's going with the default. This has been related to security issues in contracts.

```
bool inSwapAndLiquify;

mapping (address => uint256) _balances;
```

Suggestion:
You can easily silence the warning by adding the visibility public/private.

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 setIsTxLimitExempt(address holder, bool exempt)
external onlyOwner {
    isTxLimitExempt[holder] = exempt;
  }
function setIsExcludedFromFee(address account, bool newValue)
public onlyOwner {
    isExcludedFromFee[account] = newValue;
  }
function setBuyTaxes(uint256 newLiquidityTax, uint256
newMarketingTax, uint256 newBuyBackTax) external onlyOwner()
{
    _buyLiquidityFee = newLiquidityTax;
    _buyMarketingFee = newMarketingTax;
    _buyBuyBackFee = newBuyBackTax;

    _totalTaxIfBuying =
_buyLiquidityFee.add(_buyMarketingFee).add(_buyBuyBackFee);
  }
```

MANUAL TESTING

```
 function setSellTaxes(uint256 newLiquidityTax, uint256
newMarketingTax, uint256 newBuyBackTax) external onlyOwner()
{
 _sellLiquidityFee = newLiquidityTax;
 _sellMarketingFee = newMarketingTax;
 _sellBuyBackFee = newBuyBackTax;

 _totalTaxIfSelling =
_sellLiquidityFee.add(_sellMarketingFee).add(_sellBuyBackFee);
 }

 function setDistributionSettings(uint256 newLiquidityShare,
uint256 newMarketingShare, uint256 newBuyBackShare) external
onlyOwner() {
 _liquidityShare = newLiquidityShare;
 _marketingShare = newMarketingShare;
 _BuyBackShare = newBuyBackShare;

 _totalDistributionShares =
_liquidityShare.add(_marketingShare).add(_BuyBackShare);
 }
function setMaxTxAmount(uint256 maxTxAmount) external
onlyOwner() {
 require(maxTxAmount <= (40 * 10**6 * 10**9), "Max wallet should
be less or euqal to 4% totalSupply");
 _maxTxAmount = maxTxAmount;

 function setIsWalletLimitExempt(address holder, bool exempt)
external onlyOwner {
 isWalletLimitExempt[holder] = exempt;
 }
```

MANUAL TESTING

```
function setWalletLimit(uint256 newLimit) external onlyOwner {
 _walletMax = newLimit;
 }
 function setNumTokensBeforeSwap(uint256 newLimit) external
onlyOwner() {
 minimumTokensBeforeSwap = newLimit;
 }
function setMarketingWalletAddress(address newAddress)
external onlyOwner() {
 marketingWalletAddress = payable(newAddress);
 }
 function setBuyBackWalletAddress(address newAddress) external
onlyOwner() {
 BuyBackWalletAddress = payable(newAddress);
 }
```

MANUAL TESTING

Centralization – Local variable Shadowing

Severity: Low

Subject: Variable Shadowing

Status: Open

Overview:

```
function _approve(address owner, address spender,
uint256 amount) private {
    require(owner != address(0), "ERC20: approve from the
zero address");
    require(spender != address(0), "ERC20: approve to the
zero address");

    _allowances[owner][spender] = amount;
    emit Approval(owner, spender, amount);
  }

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

Suggestion:

Rename the local variables that shadow another component.

MANUAL TESTING

Optimization

Severity: Informational

subject: Remove Safe Math

Status: Open

Line: 40-91

Overview:

compiler version above 0.8.0 can control arithmetic overflow/underflow, It is recommended to remove the unwanted code to avoid high gas fees.

MANUAL TESTING

Optimization

Severity: Informational

subject: Remove unused code.

Status: Open

Overview:

Unused variables are allowed in Solidity, and they do. not pose a direct security issue. It is the best practice. though to avoid them.

```
event SwapETHForTokens(
    uint256 amountIn,
    address[] path
 );
```

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 ^0.8.4;

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