auditace.tech

AuditAce

FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

MADAPE

DATED : 31 August 23'

MANUAL TESTING

Centralization – Maximum buy/sell/transfer wallet
Severity: High
function: AntiWhaleSetThreshold
Status: Open
Overview:
Owner is able to set antiWhaleThreshold to zero making buy/sell/transfers impossible for non-privileged wallets

```
function AntiWhaleSetThreshold(uint256 newThreshold)
external onlyOwner {
antiWhaleThreshold = newThreshold * 10 ** 18;
  }
```

Suggestion
Put an upper bound for antiWhaleThreshold:

```
function AntiWhaleSetThreshold(uint256 newThreshold)
external onlyOwner {
require(antiWhaleThreshold >= totalSupply() / 1000, “Can’t
set anti whale threshold lower than 0.1% of total supply”);
antiWhaleThreshold = newThreshold;
  }
```

MANUAL TESTING

Logical – Liquidity pool size limited by anti-whale threshold
Severity: High
function: _transfer
Status: Open
Overview:
non-privileged wallets are not able to hold more tokens than antiWhaleThreshold. The below condition (from _transfer function) doesn’t check if “to” (receiver address) is equal to liquidity pair or not. If “to” is liquidity pair, this condition must be skipped in order to prevent selling/adding liquidity issues.

```
if (AntiWhaleEnabled && !isAntiWhaleExempt[to]) {
require(
            balance[to] <= antiWhaleThreshold,
            "New balance exceeds the max tokens allowed per
wallet."
);
}
```

Suggestion
make sure to skip this condition if “to” is liquidity pool:

```
if (AntiWhaleEnabled && !isAntiWhaleExempt[to] &&
!is_sell(from, to)) {
require(
            balance[to] <= antiWhaleThreshold,
            "New balance exceeds the max tokens allowed per
wallet."
);
}
```

MANUAL TESTING

Logical – Rejecting ETH could disable internal swap
Severity: Medium
function: internalSwap
Status: Open
Overview:
Owner is able to change MarketingAddress, BAddress and DdvAddress to any address. This addresses receive BNB after each internal swap, if any of this addresses rejects receiving BNB the transaction (in which internal swap is going to be performed) will be reverted.

```
bool success;
(success, ) = MarketingAddress.call{value: marketingShare, gas: 35000}("");
require(success, "Transfer to Marketing failed");

(success, ) = DevAddress.call{value: devShare, gas: 35000}("");
require(success, "Transfer to DevAddress failed");

(success, ) = BAddress.call{value: bShare, gas: 35000}("");
require(success, "Transfer to BAddress failed");
```

Example of a contract that rejects receiving BNB:

```
contract BNBRejector {

        //this function is called upon receiving BNB from the token contract
        receive() external payable {
                revert();
        }

}
```

Suggestion
By ignoring "success" we can ensure that none of this low-level calls will revert the transaction. This solution can be achieved by removing require statements which require success to be true.

```
bool success;
(success, ) = MarketingAddress.call{value: marketingShare, gas: 35000}("");
(success, ) = DevAddress.call{value: devShare, gas: 35000}("");
(success, ) = BAddress.call{value: bShare, gas: 35000}("");
```

AUDIT SUMMARY

Project name – MADAPE

Date: 31 August 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	0	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/0xefE618738Af3235e3CFEA1D3d4c4885ab63924b0

Token Information

Token Address :
0x842FD31eAd327AE42D19B7BFbdB045d3A219c662

Name: MADAPE

Symbol: MADAPE

Decimals: 18

Network: Ethereum

Token Type: ERC20

Owner: 0xf6f50011dF50e1180Dc78f4a8237f293F30a3b27

Deployer: 0xf6f50011dF50e1180Dc78f4a8237f293F30a3b27

Token Supply: 1,000,000

Checksum:
a2a3077e5ff005d476698a5904e310392cc7611d

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/0xefE618738Af3235e3CFEA1D3d4c4885ab63924b0

TOKEN OVERVIEW

buy fee: 0-20%

Sell fee: 0-20%

transfer fee: 0%

Fee Privilege: Owner

Ownership: Owned

Minting: None

Max Tx: No

Blacklist: No

Other Privileges:

- Initial distribution of the tokens
- Setting maximum buy/sell/transfer/wallet
- Modifying fees

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

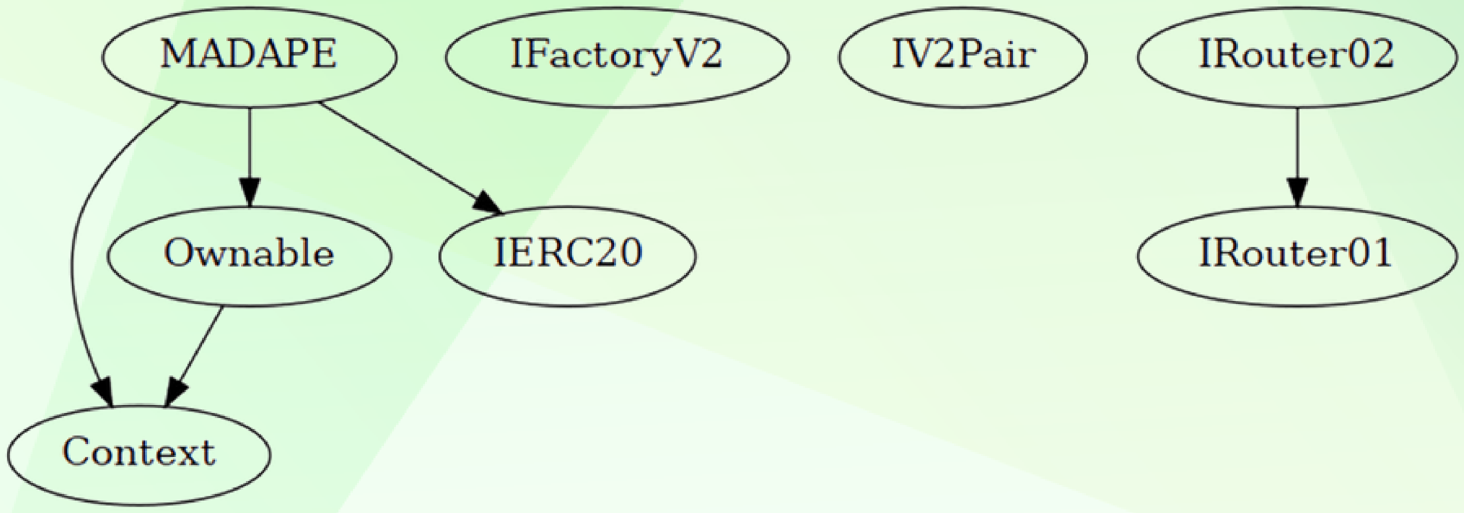
CLASSIFICATION OF RISK

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	1
◆ Low-Risk	0
◆ Gas Optimization / Suggestions	1

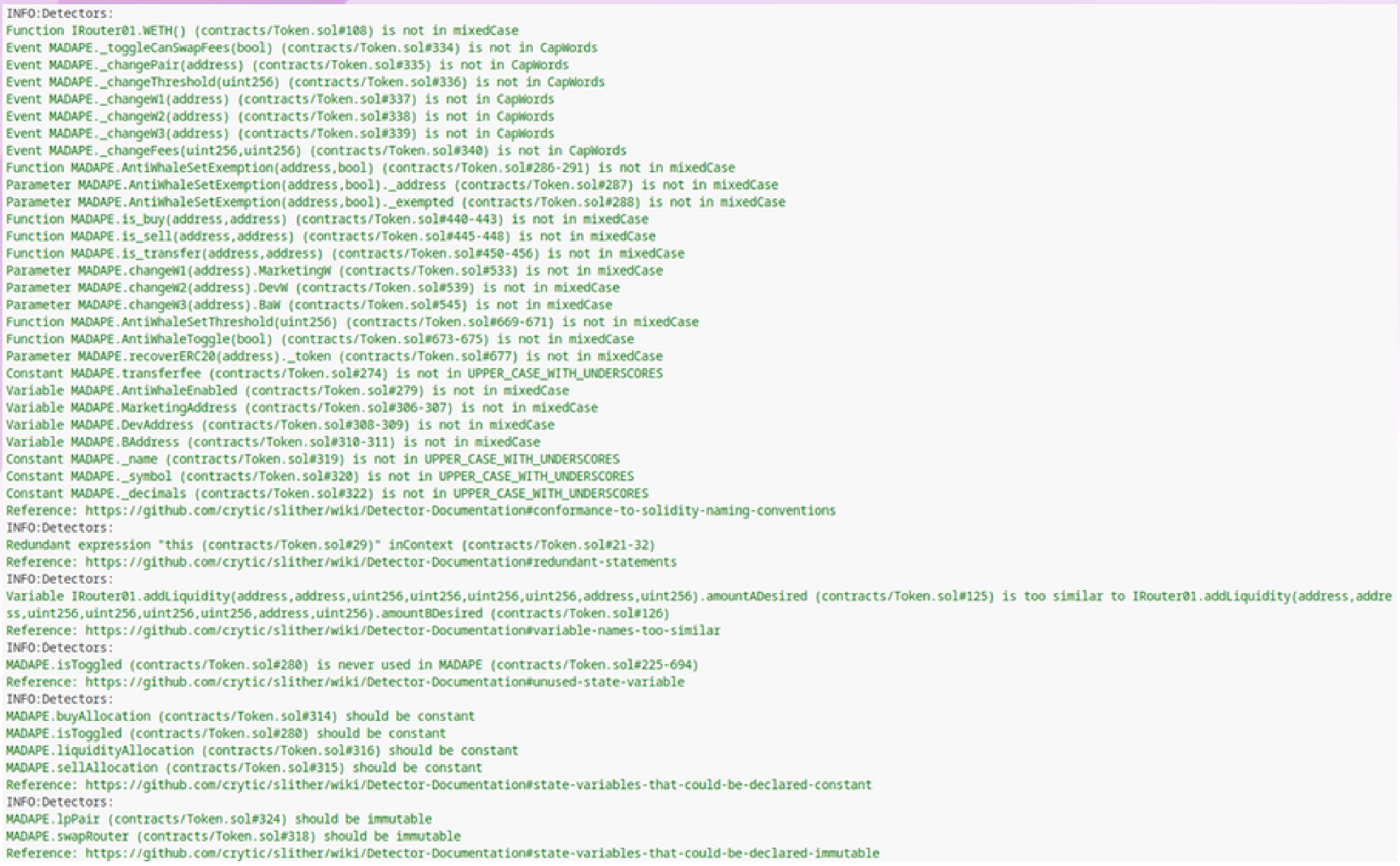
INHERITANCE TREE



POINTS TO NOTE

- **Owner is able to adjust buy/sell fees within 0-20% (0% transfer fee)**
- Owner is not able to blacklist an arbitrary wallet
- **Owner is able to disable trades**
- Owner is not able to mint new tokens
- **Owner is able to set maximum wallet and maximum buy/sell/transfer limits (unbounded)**
- **Owner must enable trades manually**

STATIC ANALYSIS



```
INFO:Detectors:
Function IRouter01.WETH() (contracts/Token.sol#108) is not in mixedCase
Event MADAPE._toggleCanSwapFees(bool) (contracts/Token.sol#334) is not in CapWords
Event MADAPE._changePair(address) (contracts/Token.sol#335) is not in CapWords
Event MADAPE._changeThreshold(uint256) (contracts/Token.sol#336) is not in CapWords
Event MADAPE._changeW1(address) (contracts/Token.sol#337) is not in CapWords
Event MADAPE._changeW2(address) (contracts/Token.sol#338) is not in CapWords
Event MADAPE._changeW3(address) (contracts/Token.sol#339) is not in CapWords
Event MADAPE._changeFees(uint256,uint256) (contracts/Token.sol#340) is not in CapWords
Function MADAPE.AntiWhaleSetExemption(address,bool) (contracts/Token.sol#286-291) is not in mixedCase
Parameter MADAPE.AntiWhaleSetExemption(address,bool)._address (contracts/Token.sol#287) is not in mixedCase
Parameter MADAPE.AntiWhaleSetExemption(address,bool)._exempted (contracts/Token.sol#288) is not in mixedCase
Function MADAPE.is_buy(address,address) (contracts/Token.sol#440-443) is not in mixedCase
Function MADAPE.is_sell(address,address) (contracts/Token.sol#445-448) is not in mixedCase
Function MADAPE.is_transfer(address,address) (contracts/Token.sol#450-456) is not in mixedCase
Parameter MADAPE.changeW1(address).MarketingW (contracts/Token.sol#533) is not in mixedCase
Parameter MADAPE.changeW2(address).DevW (contracts/Token.sol#539) is not in mixedCase
Parameter MADAPE.changeW3(address).BaW (contracts/Token.sol#545) is not in mixedCase
Function MADAPE.AntiWhaleSetThreshold(uint256) (contracts/Token.sol#669-671) is not in mixedCase
Function MADAPE.AntiWhaleToggle(bool) (contracts/Token.sol#673-675) is not in mixedCase
Parameter MADAPE.recoverERC20(address)._token (contracts/Token.sol#677) is not in mixedCase
Constant MADAPE.transferfee (contracts/Token.sol#274) is not in UPPER_CASE_WITH_UNDERSCORES
Variable MADAPE.AntiWhaleEnabled (contracts/Token.sol#279) is not in mixedCase
Variable MADAPE.MarketingAddress (contracts/Token.sol#306-307) is not in mixedCase
Variable MADAPE.DevAddress (contracts/Token.sol#308-309) is not in mixedCase
Variable MADAPE.BAddress (contracts/Token.sol#310-311) is not in mixedCase
Constant MADAPE._name (contracts/Token.sol#319) is not in UPPER_CASE_WITH_UNDERSCORES
Constant MADAPE._symbol (contracts/Token.sol#320) is not in UPPER_CASE_WITH_UNDERSCORES
Constant MADAPE._decimals (contracts/Token.sol#322) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (contracts/Token.sol#29)" inContext (contracts/Token.sol#21-32)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
Variable IRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (contracts/Token.sol#125) is too similar to IRouter01.addLiquidity(address,addre
ss,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (contracts/Token.sol#126)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
MADAPE.isToggled (contracts/Token.sol#280) is never used in MADAPE (contracts/Token.sol#225-694)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable
INFO:Detectors:
MADAPE.buyAllocation (contracts/Token.sol#314) should be constant
MADAPE.isToggled (contracts/Token.sol#280) should be constant
MADAPE.liquidityAllocation (contracts/Token.sol#316) should be constant
MADAPE.sellAllocation (contracts/Token.sol#315) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
MADAPE.lpPair (contracts/Token.sol#324) should be immutable
MADAPE.swapRouter (contracts/Token.sol#318) 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

CONTRACT ASSESMENT

```
| Contract|      Type      |Bases     |            |           |
|:----------:|:-------------------:|:----------------:|:----------------:|:---------------:|
|    └| **Function Name** |**Visibility** | **Mutability** |**Modifiers** |
||||||
| **Context** | Implementation | |||
| └ | <Constructor> | Public ❗ | 🛑|NO ❗ |
| └ | _msgSender | Internal 🔒 | | |
| └ | _msgData | Internal 🔒 | | |
||||||
| **Ownable** | Implementation | Context |||
| └ | <Constructor> | Public ❗ | 🛑|NO ❗ |
| └ | owner | Public ❗ | |NO ❗ |
| └ | renounceOwnership | Public ❗ | 🛑| onlyOwner |
| └ | transferOwnership | Public ❗ | 🛑| onlyOwner |
| └ | _setOwner | Private 🔐 | 🛑| |
||||||
| **IFactoryV2** | Interface | |||
| └ | getPair | External ❗ | |NO ❗ |
| └ | createPair | External ❗ | 🛑|NO ❗ |
||||||
| **IV2Pair** | Interface | |||
| └ | factory | External ❗ | |NO ❗ |
| └ | getReserves | External ❗ | |NO ❗ |
| └ | sync | External ❗ | 🛑 |NO ❗ |
||||||
| **IRouter01** | Interface | |||
| └ | factory | External ❗ | |NO ❗ |
| └ | WETH | External ❗ | |NO ❗ |
| └ | addLiquidityETH | External ❗ | 💵 |NO ❗ |
| └ | addLiquidity | External ❗ | 🛑|NO ❗ |
| └ | swapExactETHForTokens | External ❗ | 💵 |NO ❗ |
| └ | getAmountsOut | External ❗ | |NO ❗ |
| └ | getAmountsIn | External ❗ | |NO ❗ |
```

CONTRACT ASSESMENT

```
||||||
| **IRouter02** | Interface | IRouter01 |||
| └ | swapExactTokensForETHSupportingFeeOnTransferTokens | External ❗ | 🛑 |NO ❗ |
| └ | swapExactETHForTokensSupportingFeeOnTransferTokens | External ❗ | 💵 |NO ❗ |
| └ | swapExactTokensForTokensSupportingFeeOnTransferTokens | External ❗ | 🛑 |NO ❗ |
| └ | swapExactTokensForTokens | External ❗ | 🛑 |NO ❗ |
||||||
| **IERC20** | Interface | |||
| └ | totalSupply | External ❗ | |NO ❗ |
| └ | decimals | External ❗ | |NO ❗ |
| └ | symbol | External ❗ | |NO ❗ |
| └ | name | External ❗ | |NO ❗ |
| └ | getOwner | External ❗ | |NO ❗ |
| └ | balanceOf | External ❗ | |NO ❗ |
| └ | transfer | External ❗ | 🛑 |NO ❗ |
| └ | allowance | External ❗ | |NO ❗ |
| └ | approve | External ❗ | 🛑 |NO ❗ |
| └ | transferFrom | External ❗ | 🛑|NO ❗ |
||||||
| **MADAPE** | Implementation | Context, Ownable, IERC20 |||
| └ | totalSupply | External ❗ | |NO ❗ |
| └ | decimals | External ❗ | |NO ❗ |
| └ | symbol | External ❗ | |NO ❗ |
| └ | name | External ❗ | |NO ❗ |
| └ | getOwner | External ❗ | |NO ❗ |
| └ | allowance | External ❗ | |NO ❗ |
| └ | balanceOf | Public ❗ | |NO ❗ |
| └ | AntiWhaleSetExemption | External ❗ | 🛑| onlyOwner |
| └ | viewTaxes | External ❗ | |NO ❗ |
| └ | <Constructor> | Public ❗ | 🛑|NO ❗ |
| └ | <Receive Ether> | External ❗ | 💵 |NO ❗ |
| └ | transfer | Public ❗ | 🛑 |NO ❗ |
| └ | approve | External ❗ | 🛑 |NO ❗ |
| └ | _approve | Internal 🔒 | 🛑| |
| └ | transferFrom | External ❗ | 🛑 |NO ❗ |
| └ | isNoFeeWallet | External ❗ | |NO ❗ |
| └ | setNoFeeWallet | Public ❗ | 🛑 | onlyOwner |
```

CONTRACT ASSESMENT

```
| └ | isLimitedAddress | Internal 🔒 || |
| └ | is_buy | Internal 🔒 | | |
| └ | is_sell | Internal 🔒 | | |
| └ | is_transfer | Internal 🔒 | | |
| └ | canSwap | Internal 🔒 | | |
| └ | changeLpPair | External ❗ | 🛑 | onlyOwner |
| └ | toggleCanSwapFees | External ❗ | 🛑| onlyOwner |
| └ | _transfer | Internal 🔒 | 🛑| |
| └ | changeW1 | External ❗ | 🛑 | onlyOwner |
| └ | changeW2 | External ❗ | 🛑 | onlyOwner |
| └ | changeW3 | External ❗ | 🛑 | onlyOwner |
| └ | setBuyFee | External ❗ | 🛑 | onlyOwner |
| └ | setSellFee | External ❗ | 🛑 | onlyOwner |
| └ | takeTaxes | Internal 🔒 | 🛑 | |
| └ | swapAndLiquify | Internal 🔒 | 🛑| inSwapFlag |
| └ | internalSwap | Internal 🔒 | 🛑| inSwapFlag |
| └ | AntiWhaleSetThreshold | External ❗ | 🛑| onlyOwner |
| └ | AntiWhaleToggle | External ❗ | 🛑| onlyOwner |
| └ | recoverERC20 | External ❗ | 🛑|NO ❗ |
| └ | recoverEther | External ❗ | 🛑|NO ❗ |

### Legend

| Symbol| Meaning |
|:---------:|-----------|
| 🛑| Function can modify state |
| 💵| Function is payable |
```

FUNCTIONAL TESTING

1- Adding liquidity (passed):
https://testnet.bscscan.com/tx/0x107b9bf7b7860e29aa93b38aaba33dd0ed2c64b612dc0973fe699c6e22627e54

2- Buying when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0x84293f82ecd5b2b891c41290f3a56143933f1449b2ce1a40be271524f2c72b1c

3- Selling when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0xfdfa800f1f0ad7ad308ffe0dd0b69fe36bf15f45d04c735b1a3f26e2d78d9659

4- Transferring when excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0x3e54f3ba8e0e6acae7991a9501cf425f638cd0bb351470a95486de28cdd6e913

5- Buying when not excluded from fees (tax 0-20%) (passed):
https://testnet.bscscan.com/tx/0x22ad9a96ad2c09797592fd4bfd2f92b2f05b0b4570af0a55aedeeac707432f02

6- Selling when not excluded from fees (tax 0-20%) (passed):
https://testnet.bscscan.com/tx/0x67fca2c066a8a2097cb9ab90a583ae10677a37bb4c9fdccb151715f9ece458a8

7- Transferring when not excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0xc0e55f7f742d18b4746967e7eb7c5ecb03224d6cc874137f4bddb59fd012e1b3

8- Internal swap (BNB set to marketing wallet | Auto-liquidity)(passed):
https://testnet.bscscan.com/tx/0x67fca2c066a8a2097cb9ab90a583ae10677a37bb4c9fdccb151715f9ece458a8

MANUAL TESTING

Centralization – Maximum buy/sell/transfer wallet

Severity: High
function: AntiWhaleSetThreshold
Status: Open
Overview:
Owner is able to set antiWhaleThreshold to zero making buy/sell/transfers impossible for non-privileged wallets

```
function AntiWhaleSetThreshold(uint256 newThreshold)
external onlyOwner {
antiWhaleThreshold = newThreshold * 10 ** 18;
  }
```

Suggestion
Put an upper bound for antiWhaleThreshold:

```
function AntiWhaleSetThreshold(uint256 newThreshold)
external onlyOwner {
require(antiWhaleThreshold >= totalSupply() / 1000, “Can’t
set anti whale threshold lower than 0.1% of total supply”);
antiWhaleThreshold = newThreshold;
  }
```

MANUAL TESTING

Logical – Liquidity pool size limited by anti-whale threshold

Severity: High
function: _transfer
Status: Open
Overview:
non-privileged wallets are not able to hold more tokens than antiWhaleThreshold. The below condition (from _transfer function) doesn’t check if “to” (receiver address) is equal to liquidity pair or not. If “to” is liquidity pair, this condition must be skipped in order to prevent selling/adding liquidity issues.

```
if (AntiWhaleEnabled && !isAntiWhaleExempt[to]) {
require(
            balance[to] <= antiWhaleThreshold,
            "New balance exceeds the max tokens allowed per
wallet."
);
}
```

Suggestion
make sure to skip this condition if “to” is liquidity pool:

```
if (AntiWhaleEnabled && !isAntiWhaleExempt[to] &&
!is_sell(from, to)) {
require(
            balance[to] <= antiWhaleThreshold,
            "New balance exceeds the max tokens allowed per
wallet."
);
}
```

MANUAL TESTING

Logical – Rejecting ETH could disable internal swap

Severity: Medium
function: internalSwap
Status: Open
Overview:
Owner is able to change MarketingAddress, BAddress and DdvAddress to any address. This addresses receive BNB after each internal swap, if any of this addresses rejects receiving BNB the transaction (in which internal swap is going to be performed) will be reverted.

```
bool success;
(success, ) = MarketingAddress.call{value: marketingShare, gas: 35000}("");
require(success, "Transfer to Marketing failed");

(success, ) = DevAddress.call{value: devShare, gas: 35000}("");
require(success, "Transfer to DevAddress failed");

(success, ) = BAddress.call{value: bShare, gas: 35000}("");
require(success, "Transfer to BAddress failed");
```

Example of a contract that rejects receiving BNB:

```
contract BNBRejector {

        //this function is called upon receiving BNB from the token contract
        receive() external payable {
                revert();
        }

}
```

Suggestion
By ignoring “success” we can ensure that none of this low-level calls will revert the transaction. This solution can be achieved by removing require statements which require success to be true.

```
bool success;
(success, ) = MarketingAddress.call{value: marketingShare, gas: 35000}("");
(success, ) = DevAddress.call{value: devShare, gas: 35000}("");
(success, ) = BAddress.call{value: bShare, gas: 35000}("");
```

MANUAL TESTING

Centralization – Excessive Fees

Severity: Informational

function: _transfer

Status: Open

Overview:

Owner is able to set up to 20% fee on swaps (token => BNB and BNB => token)

```
function setBuyFee(uint256 newBuyFee) external onlyOwner {
    require(newBuyFee <= MAX_FEE, "Fee is too high!");
    buyFee = newBuyFee;
    emit FeeUpdated("Buy", newBuyFee);
}

function setSellFee(uint256 newSellFee) external onlyOwner {
    require(newSellFee <= MAX_FEE, "Fee is too high!");
    sellFee = newSellFee;
    emit FeeUpdated("Sell", newSellFee);
}
```

Suggestion

Its suggested to keep fees within 0-10% on buy or sells

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