auditace.tech

AuditAce

FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

EMPR Token

DATED : 16 JAN 23'

EXECUTIVE SUMMARY

Project name – EMPR Token

TimeLine- 16 January , 2023

Method- Manual Review ,Functional Testing, Automated Testing etc.

Scope of Audit- Audit Ace was consulted to conduct the smart contract audit of the solidity source codes. The audit scope of work is strictly limited to mentioned solidity file(s) only: *ElonMuskPampRocketShip.sol*

Audit Status: Failed

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	1	4	2	0	4
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- Goerli:
all tests were done on Goerli network, each test has its transaction has attached to it.

3- UniswapV2

4- Slither : Static Analysis

TESTNET LINKS

1- Deployment:

https://goerli.etherscan.io/tx/0xc8f85da7f2e9ebe8ba64cc22e1a9cebdf2cb37908f7694a5636a177c1d29566d

2- Adding liquidity on UniswapV2 (1ETH and 10% of token supply) (passed)

https://goerli.etherscan.io/tx/0x233fffd2bce070eeaf7ef890631a5a10c115175be75bba792e95ac36d9d61b6f

TOKEN OVERVIEW

Fees:

Buy Fees: 3%

Sell Fees: 3%

Transfer Fees: 0%

Fees Privilige: None

Ownership : Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount:

can change max tx amount and max wallet amount (no threshold)

Blacklist: has blacklist function

Other Priviliges:can set taxes up to 100%

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

- ✓ Re-entrancy
- ✓ Timestamp Dependence
- ✓ Gas Limit and Loops
- ✓ Exception Disorder
- ✓ Gasless Send
- ✓ Use of tx.origin
- ✗ Compiler version not fixed
- ✓ Address hardcoded
- ✓ Divide before multiply
- ✓ Integer overflow/underflow
- ✓ Dangerous strict equalities
- ✓ Tautology or contradiction
- ✓ Return values of low-level calls
- ✗ Missing Zero Address Validation
- ✓ Private modifier
- ✓ Revert/require functions
- ✓ Using block.timestamp
- ✓ Multiple Sends
- ✓ Using SHA3
- ✓ Using suicide
- ✓ Using throw
- ✓ Using inline assembly

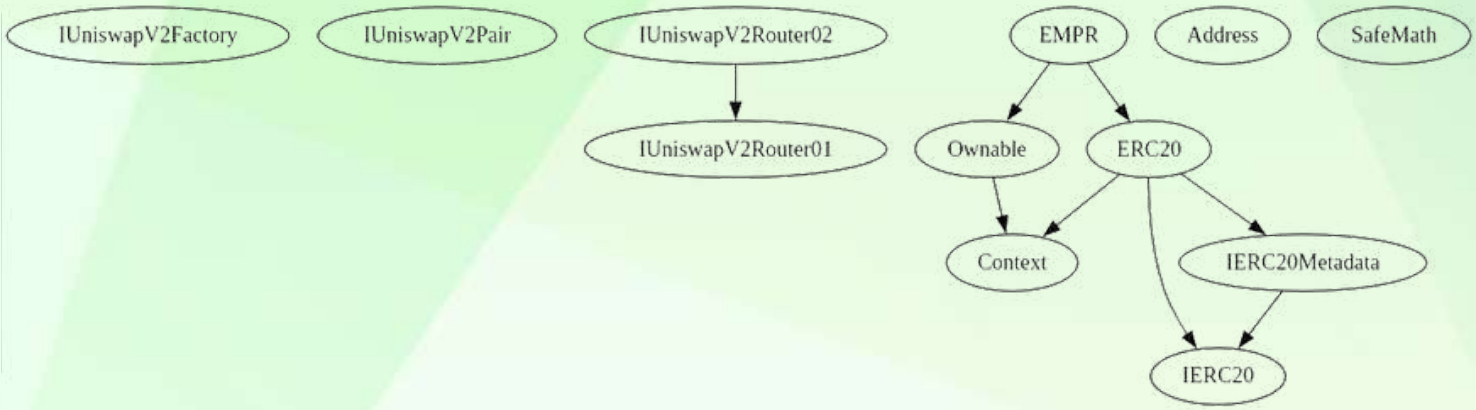
CLASSIFICATION OF RISK

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	1
High-Risk	4
Medium-Risk	2
Low-Risk	0
Gas Optimization / Suggestions	4

INHERITANCE TREE



POINTS TO NOTE

- Owner is able to set taxes up to 100%
- Owner is able to blacklist an arbitrary wallet
- Owner is able to set max buy/sell/holding amounts
- Owner is able to disable trades
- Owner is not able to mint new tokens

CONTRACT ASSESMENT

Symbol	Meaning
●	Function can modify state
💶	Function is payable

functions:|Contract | Type |Bases | | |
|:----------:|:----------:|:----------:|:----------:|:----------:|
| └ | **Function Name** |**Visibility** | **Mutability** |**Modifiers** |
||||||
| **IUniswapV2Factory** | Interface | ||
| └ | feeTo | External ! | |NO ! |
| └ | feeToSetter | External ! | |NO ! |
| └ | getPair | External ! | |NO ! |
| └ | allPairs | External ! | |NO ! |
| └ | allPairsLength | External ! | |NO ! |
| └ | createPair | External ! | ● |NO ! |
| └ | setFeeTo | External ! | ● |NO ! |
| └ | setFeeToSetter | External ! | ● |NO ! |
||||||
| **IUniswapV2Pair** | Interface | ||
| └ | name | External ! | |NO ! |
| └ | symbol | External ! | |NO ! |
| └ | decimals | External ! | |NO ! |
| └ | totalSupply | External ! | |NO ! |
| └ | balanceOf | External ! | |NO ! |
| └ | allowance | External ! | |NO ! |
| └ | approve | External ! | ● |NO ! |

└	transfer	External ❗	🔴	NO❗
└	transferFrom	External ❗	🔴	NO❗
└	DOMAIN_SEPARATOR	External ❗		NO❗
└	PERMIT_TYPEHASH	External ❗		NO❗
└	nonces	External ❗		NO❗
└	permit	External ❗	🔴	NO❗
└	MINIMUM_LIQUIDITY	External ❗		NO❗
└	factory	External ❗		NO❗
└	token0	External ❗		NO❗
└	token1	External ❗		NO❗
└	getReserves	External ❗		NO❗
└	price0CumulativeLast	External ❗		NO❗
└	price1CumulativeLast	External ❗		NO❗
└	kLast	External ❗		NO❗
└	mint	External ❗	🔴	NO❗
└	burn	External ❗	🔴	NO❗
└	swap	External ❗	🔴	NO❗
└	skim	External ❗	🔴	NO❗
└	sync	External ❗	🔴	NO❗
└	initialize	External ❗	🔴	NO❗
IUniswapV2Router01	Interface				
└	factory	External ❗		NO❗
└	WETH	External ❗		NO❗
└	addLiquidity	External ❗	🔴	NO❗
└	addLiquidityETH	External ❗	💵	NO❗
└	removeLiquidity	External ❗	🔴	NO❗
└	removeLiquidityETH	External ❗	🔴	NO❗
└	removeLiquidityWithPermit	External ❗	🔴	NO❗
└	removeLiquidityETHWithPermit	External ❗	🔴	NO❗

└	swapExactTokensForTokens	External ❗	🔴	NO❗
└	swapTokensForExactTokens	External ❗	🔴	NO❗
└	swapExactETHForTokens	External ❗	💵	NO❗
└	swapTokensForExactETH	External ❗	🔴	NO❗
└	swapExactTokensForETH	External ❗	🔴	NO❗
└	swapETHForExactTokens	External ❗	💵	NO❗
└	quote	External ❗		NO❗
└	getAmountOut	External ❗		NO❗
└	getAmountIn	External ❗		NO❗
└	getAmountsOut	External ❗		NO❗
└	getAmountsIn	External ❗		NO❗
IUniswapV2Router02	Interface	IUniswapV2Router01		
└	removeLiquidityETHSupportingFeeOnTransferTokens	External ❗	🔴	NO❗
└	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External ❗	🔴	NO❗
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🔴	NO❗
└	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💵	NO❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🔴	NO❗
IERC20	Interface			
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	transfer	External ❗	🔴	NO❗
└	allowance	External ❗		NO❗

└	approve	External ❗	🔴	NO ❗
└	transferFrom	External ❗	🔴	NO ❗
IERC20Metadata	Interface	IERC20		
└	name	External ❗		NO ❗
└	symbol	External ❗		NO ❗
└	decimals	External ❗		NO ❗
Address	Library			
└	isContract	Internal 🔒		
└	sendValue	Internal 🔒	🔴	
└	functionCall	Internal 🔒	🔴	
└	functionCall	Internal 🔒	🔴	
└	functionCallWithValue	Internal 🔒	🔴	
└	functionCallWithValue	Internal 🔒	🔴	
└	functionStaticCall	Internal 🔒		
└	functionStaticCall	Internal 🔒		
└	functionDelegateCall	Internal 🔒	🔴	
└	functionDelegateCall	Internal 🔒	🔴	
└	verifyCallResultFromTarget	Internal 🔒		
└	verifyCallResult	Internal 🔒		
└	_revert	Private 🔐		
Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🔴	NO ❗
└	owner	Public ❗		NO ❗

└	renounceOwnership	Public ❗	🔴	onlyOwner
└	transferOwnership	Public ❗	🔴	onlyOwner
└	_transferOwnership	Internal 🔒	🔴	
SafeMath	Library			
└	add	Internal 🔒		
└	sub	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
└	mod	Internal 🔒		
ERC20	Implementation	Context, IERC20, IERC20Metadata		
└	<Constructor>	Public ❗	🔴	NO ❗
└	name	Public ❗		NO ❗
└	symbol	Public ❗		NO ❗
└	decimals	Public ❗		NO ❗
└	totalSupply	Public ❗		NO ❗
└	balanceOf	Public ❗		NO ❗
└	transfer	Public ❗	🔴	NO ❗
└	allowance	Public ❗		NO ❗
└	approve	Public ❗	🔴	NO ❗
└	transferFrom	Public ❗	🔴	NO ❗
└	increaseAllowance	Public ❗	🔴	NO ❗
└	decreaseAllowance	Public ❗	🔴	NO ❗
└	_transfer	Internal 🔒	🔴	
└	_mint	Internal 🔒	🔴	
└	_burn	Internal 🔒	🔴	
└	_approve	Internal 🔒	🔴	

└	_beforeTokenTransfer	Internal 🔒	🔴	
EMPR	Implementation	ERC20, Ownable		
└	<Constructor>	Public ❗	🔴	ERC20
└	<Receive Ether>	External ❗	💵	NO ❗
└	claimStuckTokens	External ❗	🔴	onlyOwner
└	burn	External ❗	🔴	onlyOwner
└	excludeFromFees	External ❗	🔴	onlyOwner
└	isExcludedFromFees	Public ❗		NO ❗
└	updateBuyFees	External ❗	🔴	onlyOwner
└	updateSellFees	External ❗	🔴	onlyOwner
└	changeMarketingWallet	External ❗	🔴	onlyOwner
└	enableTrading	External ❗	🔴	onlyOwner
└	setBlackListEnabled	Public ❗	🔴	onlyOwner
└	addBlacklist	Public ❗	🔴	onlyOwner
└	isBlacklisted	Public ❗		NO ❗
└	_transfer	Internal 🔒	🔴	
└	setSwapEnabled	External ❗	🔴	onlyOwner
└	setSwapTokensAtAmount	External ❗	🔴	onlyOwner
└	swapAndLiquify	Private 🔐	🔴	
└	swapAndSendMarketing	Private 🔐	🔴	
└	setEnableMaxWalletLimit	External ❗	🔴	onlyOwner
└	setMaxWalletAmount	External ❗	🔴	onlyOwner
└	excludeFromMaxWallet	External ❗	🔴	onlyOwner
└	isExcludedFromMaxWalletLimit	Public ❗		NO ❗
└	setEnableMaxTransactionLimit	External ❗	🔴	onlyOwner
└	setMaxTransactionAmounts	External ❗	🔴	onlyOwner
└	excludeFromMaxTransactionLimit	External ❗	🔴	onlyOwner
└	isExcludedFromMaxTransaction	Public ❗		NO ❗

AUTOMATED TESTING

```
EMPR.updateBuyFees(uint256,uint256) (tests/test.sol#891-904) should emit an event for:
        - liquidityFeeOnBuy = _liquidityFeeOnBuy (tests/test.sol#895)
        - marketingFeeOnBuy = _marketingFeeOnBuy (tests/test.sol#896)
        - _totalFeesOnBuy = liquidityFeeOnBuy + marketingFeeOnBuy (tests/test.sol#898)
EMPR.updateSellFees(uint256,uint256) (tests/test.sol#906-919) should emit an event for:
        - liquidityFeeOnSell = _liquidityFeeOnSell (tests/test.sol#910)
        - marketingFeeOnSell = _marketingFeeOnSell (tests/test.sol#911)
        - _totalFeesOnSell = liquidityFeeOnSell + marketingFeeOnSell (tests/test.sol#913)
EMPR.setSwapTokensAtAmount(uint256) (tests/test.sol#1075-1081) should emit an event for:
        - swapTokensAtAmount = newAmount (tests/test.sol#1080)
EMPR.setMaxWalletAmount(uint256) (tests/test.sol#1145-1151) should emit an event for:
        - maxWalletAmount = _maxWalletAmount * (10 ** 18) (tests/test.sol#1150)
EMPR.setMaxTransactionAmounts(uint256,uint256) (tests/test.sol#1193-1205) should emit an event for:
        - maxTransactionAmountBuy = _maxTransactionAmountBuy * (10 ** 18) (tests/test.sol#1203)
        - maxTransactionAmountSell = _maxTransactionAmountSell * (10 ** 18) (tests/test.sol#1204)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic

Reentrancy in EMPR._transfer(address,address,uint256) (tests/test.sol#962-1068):
        External calls:
        - swapAndLiquify(liquidityTokens) (tests/test.sol#1024)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(half,0,path,address(this),block.timestamp) (tests/test.sol#1093-1099)
                - uniswapV2Router.addLiquidityETH{value: newBalance}(address(this),otherHalf,0,0,address(0xdead),block.timestamp) (tests/test.sol#1103-1110)
        - swapAndSendMarketing(marketingTokens) (tests/test.sol#1030)
                - (success) = recipient.call{value: amount}() (tests/test.sol#383)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (tests/test.sol#1120-1126)
                - address(marketingWallet).sendValue(newBalance) (tests/test.sol#1130)
        External calls sending eth:
        - swapAndLiquify(liquidityTokens) (tests/test.sol#1024)
                - uniswapV2Router.addLiquidityETH{value: newBalance}(address(this),otherHalf,0,0,address(0xdead),block.timestamp) (tests/test.sol#1103-1110)
        - swapAndSendMarketing(marketingTokens) (tests/test.sol#1030)
                - (success) = recipient.call{value: amount}() (tests/test.sol#383)
        Event emitted after the call(s):
        - Transfer(sender,recipient,amount) (tests/test.sol#785)
                - super._transfer(from,address(this),fees) (tests/test.sol#1050)
        - Transfer(sender,recipient,amount) (tests/test.sol#785)
                - super._transfer(from,to,amount) (tests/test.sol#1067)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Address._revert(bytes,string) (tests/test.sol#531-546) uses assembly
        - INLINE ASM (tests/test.sol#539-542)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

EMPR.isBlacklisted(address) (tests/test.sol#951-960) compares to a boolean constant:
        - _isBlackListed[_account] == true (tests/test.sol#954)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#boolean-equality

Address._revert(bytes,string) (tests/test.sol#531-546) is never used and should be removed
Address.functionCall(address,bytes) (tests/test.sol#390-401) is never used and should be removed
Address.functionCall(address,bytes,string) (tests/test.sol#403-409) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (tests/test.sol#411-423) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (tests/test.sol#425-445) is never used and should be removed
Address.functionDelegateCall(address,bytes) (tests/test.sol#474-484) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (tests/test.sol#486-499) is never used and should be removed
```


```
                - address(marketingWallet).sendValue(newBalance) (tests/test.sol#1130)
        External calls sending eth:
        - swapAndLiquify(liquidityTokens) (tests/test.sol#1024)
                - uniswapV2Router.addLiquidityETH{value: newBalance}(address(this),otherHalf,0,0,address(0xdead),block.timestamp) (tests/test.sol#1103-1110)
        - swapAndSendMarketing(marketingTokens) (tests/test.sol#1030)
                - (success) = recipient.call{value: amount}() (tests/test.sol#383)
        Event emitted after the call(s):
        - Transfer(sender,recipient,amount) (tests/test.sol#785)
                - super._transfer(from,address(this),fees) (tests/test.sol#1050)
        - Transfer(sender,recipient,amount) (tests/test.sol#785)
                - super._transfer(from,to,amount) (tests/test.sol#1067)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Address._revert(bytes,string) (tests/test.sol#531-546) uses assembly
        - INLINE ASM (tests/test.sol#539-542)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

EMPR.isBlacklisted(address) (tests/test.sol#951-960) compares to a boolean constant:
        - _isBlackListed[_account] == true (tests/test.sol#954)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#boolean-equality

Address._revert(bytes,string) (tests/test.sol#531-546) is never used and should be removed
Address.functionCall(address,bytes) (tests/test.sol#390-401) is never used and should be removed
Address.functionCall(address,bytes,string) (tests/test.sol#403-409) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (tests/test.sol#411-423) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (tests/test.sol#425-445) is never used and should be removed
Address.functionDelegateCall(address,bytes) (tests/test.sol#474-484) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (tests/test.sol#486-499) is never used and should be removed
```


```
Address.functionStaticCall(address,bytes) (tests/test.sol#447-457) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (tests/test.sol#459-472) is never used and should be removed
Address.isContract(address) (tests/test.sol#373-375) is never used and should be removed
Address.verifyCallResult(bool,bytes,string) (tests/test.sol#519-529) is never used and should be removed
Address.verifyCallResultFromTarget(address,bool,bytes,string) (tests/test.sol#501-517) is never used and should be removed
Context._msgData() (tests/test.sol#554-557) is never used and should be removed
ERC20._mint(address,uint256) (tests/test.sol#788-794) is never used and should be removed
SafeMath.div(uint256,uint256) (tests/test.sol#638-640) is never used and should be removed
SafeMath.div(uint256,uint256,string) (tests/test.sol#642-652) is never used and should be removed
SafeMath.mod(uint256,uint256) (tests/test.sol#654-656) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (tests/test.sol#658-665) is never used and should be removed
SafeMath.mul(uint256,uint256) (tests/test.sol#627-636) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.17 (tests/test.sol#5) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (tests/test.sol#377-388):
        - (success) = recipient.call{value: amount}() (tests/test.sol#383)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (tests/test.sol#425-445):
```

```
evel call in Address.functionStaticCall(address,bytes,string) (tests/test.sol#459-472):
    - (success,returndata) = target.staticcall(data) (tests/test.sol#464)
evel call in Address.functionDelegateCall(address,bytes,string) (tests/test.sol#486-499):
    - (success,returndata) = target.delegatecall(data) (tests/test.sol#491)
ence: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

ion IUniswapV2Pair.DOMAIN_SEPARATOR() (tests/test.sol#67) is not in mixedCase
ion IUniswapV2Pair.PERMIT_TYPEHASH() (tests/test.sol#69) is not in mixedCase
ion IUniswapV2Pair.MINIMUM_LIQUIDITY() (tests/test.sol#100) is not in mixedCase
ion IUniswapV2Router01.WETH() (tests/test.sol#140) is not in mixedCase
eter EMPR.burn(uint256)._amount (tests/test.sol#872) is not in mixedCase
eter EMPR.updateBuyFees(uint256,uint256)._liquidityFeeOnBuy (tests/test.sol#892) is not in mixedCase
eter EMPR.updateBuyFees(uint256,uint256)._marketingFeeOnBuy (tests/test.sol#893) is not in mixedCase
eter EMPR.updateSellFees(uint256,uint256)._liquidityFeeOnSell (tests/test.sol#907) is not in mixedCase
eter EMPR.updateSellFees(uint256,uint256)._marketingFeeOnSell (tests/test.sol#908) is not in mixedCase
eter EMPR.changeMarketingWallet(address)._marketingWallet (tests/test.sol#922) is not in mixedCase
eter EMPR.setBlackListEnabled(bool)._enabled (tests/test.sol#943) is not in mixedCase
eter EMPR.addBlacklist(address,bool)._value (tests/test.sol#947) is not in mixedCase
eter EMPR.isBlacklisted(address)._account (tests/test.sol#951) is not in mixedCase
eter EMPR.setSwapEnabled(bool)._enabled (tests/test.sol#1070) is not in mixedCase
eter EMPR.setMaxWalletAmount(uint256)._maxWalletAmount (tests/test.sol#1145) is not in mixedCase
eter EMPR.setMaxTransactionAmounts(uint256,uint256)._maxTransactionAmountBuy (tests/test.sol#1194) is not in mixedCase
eter EMPR.setMaxTransactionAmounts(uint256,uint256)._maxTransactionAmountSell (tests/test.sol#1195) is not in mixedCase
ence: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

dant expression "this (tests/test.sol#555)" inContext (tests/test.sol#549-558)
ence: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

ble IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (tests/test.sol#145) is too similar to IUniswapV2Router01.addLi
ress,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (tests/test.sol#146)
ence: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

axFee (tests/test.sol#842) is never used in EMPR (tests/test.sol#825-1223)
ence: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

axFee (tests/test.sol#842) should be constant
niswapV2Pair (tests/test.sol#829) should be constant
niswapV2Router (tests/test.sol#828) should be constant
ence: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
```

MANUAL TESTING

Critical Risk Findings:

Logical – buys will be disabled after accumulated taxes reaching swap and liquify threshold, swapping contract accumulated fees should be performed on a sell transaction (not buys), this is because on buys, pool contract is the sender of tokens, and because we are also going to swap accumulated taxes to ETH, new constant product wont match this condition due to un-synced pool balances:

new k >= last k (this condition is not met, hence transaction reverts on buys with a swap and liquifiy)

Test tx:
first we used setSwapTokensAtAmount to set threshold at (totalSupply() / 1000000) + 1 (minimum threshold amount), and then we tried to perform a buy action on uniswap, all the transaction were getting rejected by **"TRANSFER_FAILED"** error
https://goerli.etherscan.io/tx/0xd39e8ea67dd26b4ee3d43ae7429c06bf64b966a5c86e36c4398d2d96ee491ff6

Recommendation:
make sure that receiver is pair when performing swap and liquifiy.

High Risk Findings:

Logical – token has 9 decimals while some functions are pefrorming calculations based on 18 decimals, this functions are :

- setMaxTransactionAmounts

- setEnableMaxTransactionLimit

- setMaxWalletAmount

this opens up new centralization issues.

==========

Centralization – Owner is able to set up to 100% tax on buys or 100% tax on sells (buy + sell tax are always less than 100). Require statement is not matching returned error. (tax denominator is 100)

Testnet link:

https://goerli.etherscan.io/tx/0x1ea09887bc302b603d835dc05bc20085367ac03e72070a5742ba4d0b12a2e460

Codes

```
function updateBuyFees(
    uint256 _liquidityFeeOnBuy,
    uint256 _marketingFeeOnBuy
 ) external onlyOwner {
    liquidityFeeOnBuy = _liquidityFeeOnBuy;
    marketingFeeOnBuy = _marketingFeeOnBuy;

    _totalFeesOnBuy = liquidityFeeOnBuy + marketingFeeOnBuy;

    require(
        _totalFeesOnBuy + _totalFeesOnSell <= 100,
        "Total Fees cannot exceed 10%"
    );
 }
```

```
function updateSellFees(
uint256 _liquidityFeeOnSell,
uint256 _marketingFeeOnSell
) external onlyOwner {
liquidityFeeOnSell = _liquidityFeeOnSell;
marketingFeeOnSell = _marketingFeeOnSell;

_totalFeesOnSell = liquidityFeeOnSell + marketingFeeOnSell;

require(
_totalFeesOnBuy + _totalFeesOnSell <= 100,
"Total Fees cannot exceed 10%"
);
}
```

Recommendation:

make sure than new total fees (buy and sells) are less than 10

==========

Centralization/Logical issue – enabling anti-bot will revert all the buys, sells, transfers

```
function isBlacklisted(address _account) public view returns (bool) {
if (blacklistEnabled) {
{
        _isBlackListed[_account] == true;
}
return true;
} else {
return false;
}
  }
```

Centralization – Owner is able to set max wallet, max buy and max sell amount to zero, this is because of incompatibility between token decimals and the decimals that this functions are using.

Test tx:

we could successfully set max wallet, max buy and max sell amount to 0, this means all the trades will be disabled for non excluded wallets.

Setting max wallet to 0:

https://goerli.etherscan.io/tx/0x8b45a44023c2bf3cd6a011f80fdbf127f597b8380bd570af50f7835390a3e954

Setting max buy & sell to 0:

https://goerli.etherscan.io/tx/0xf30a1914ce47164046eaa738b5e0e264756b791c28632ba73d0856d86a16b623

Medium Risk Findings:

Logical – setting marketing wallet to a contract that reverts receiving ether, can disable swaps

```
contract marketingWallet {
        receive() external payable{ revert(); }
```

Suggestion:

make sure that new marketing wallet can not be an address

==========

Logical - Renounce ownership doesn't transfer ownership to zero address, it only emits an event

```
function renounceOwnership() public virtual onlyOwner {
emit OwnershipTransferred(_owner, address(0));
}
```

Suggestion:

change owner variable to address(0)

Gas Optimizations

- declare _name, _symbol, _total supply as constant
- maxFee is never used in the contract

Suggestions

- Use up to 3 indexed event params
- Use fixed solidity version
- Arrange variables and events to be at top of the contract code
- Add comment to functions

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