AuditAce

FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

MatrixGPT

DATED : 12 FEB 23'

AUDIT SUMMARY

Project name – MatrixGPT

Date: 12 February, 2023

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

Audit Status: Passed

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	0	0	0	0
Acknowledged	0	0	0	0	0
Resolved	0	0	0	0	0

USED TOOLS

Tools:

1- Manual Review:
a line by line code review has been performed by audit ace team.

2- BSC Test Network:
all tests were done on BSC Test network, each test has its transaction has attached to it.

3- Slither : Static Analysis

Testnet Link: all tests were done using this contract, tests are done on BSC Testnet

https://testnet.bscscan.com/token/0x26135d9b8d22e6d67a77e797907d9cc63a674e6e

Token Information

Token Name : MatrixGPT

Token Symbol: MAI

Decimals: 18

Token Supply: 1,000,000,000

Token Address:
0xE045fc25581cfdC3CFb5c282501f3CD1A133A7ec

Checksum:
e880433f4bc0ef1faa57c0f9c1a4f09fdcadde77f90cf
f67cbe23b8000289cafc

Owner: Not deployed

Deployer:
Not deployed

TOKEN OVERVIEW

Fees:

Buy Fees: 1%

Sell Fees: 1%

Transfer Fees: 0%

Fees Privilige: None

Ownership : Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Priviliges: 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

◆ **Critical**

◆ **High-Risk**

◆ **Medium-Risk**

◆ **Low-Risk**

◆ **Gas Optimization /Suggestion**

Description

These vulnerabilities could be exploited easily and can lead to asset loss, data loss, asset, or data manipulation. They should be fixed right away.

A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.

A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.

A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.

A vulnerability that has an informational character but is not affecting any of the code.

Findings

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

INHERITANCE TREE

POINTS TO NOTE

- Owner is not able to change buy/sell/transfer taxes (1% buy and 1% sell)

- Owner is not able to blacklist an arbitrary wallet

- Owner is not able to set max buy/sell/transfer amounts

- Owner is not able to disable trades

- Owner is not able to mint new tokens

CONTRACT ASSESMENT

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
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 🔒		
SafeMathInt	Library			
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	sub	Internal 🔒		
└	add	Internal 🔒		
└	abs	Internal 🔒		
└	toUint256Safe	Internal 🔒		
SafeMathUint	Library			
└	toInt256Safe	Internal 🔒		
IterableMapping	Library			
└	get	Internal 🔒		
└	getIndexOfKey	Internal 🔒		
└	getKeyAtIndex	Internal 🔒		
└	size	Internal 🔒		
└	set	Internal 🔒	🛑	

CONTRACT ASSESMENT

└	remove	Internal 🔒	🛑		
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❗

CONTRACT ASSESMENT

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❗️
└	allowance	External ❗️		NO❗️
└	transfer	External ❗️	🛑	NO❗️
└	approve	External ❗️	🛑	NO❗️
└	transferFrom	External ❗️	🛑	NO❗️
IERC20Metadata	Interface	IERC20		
└	name	External ❗️		NO❗️
└	symbol	External ❗️		NO❗️
└	decimals	External ❗️		NO❗️
ERC20	Implementation	Context, IERC20, IERC20Metadata		

CONTRACT ASSESMENT

└	<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 🔒	🛑	
DividendPayingTokenInterface	Interface			
└	dividendOf	External ❗			NO❗
└	withdrawDividend	External ❗	🛑		NO❗
DividendPayingTokenOptionalInterface	Interface			
└	withdrawableDividendOf	External ❗			NO❗
└	withdrawnDividendOf	External ❗			NO❗
└	accumulativeDividendOf	External ❗			NO❗
DividendPayingToken	Implementation	ERC20, Ownable, DividendPayingTokenInterface, DividendPayingTokenOptionalInterface		
└	<Constructor>	Public ❗	🛑	ERC20
└	distributeDividends	Public ❗	🛑	onlyOwner
└	withdrawDividend	Public ❗	🛑		NO❗
└	_withdrawDividendOfUser	Internal 🔒	🛑	
└	dividendOf	Public ❗			NO❗
└	withdrawableDividendOf	Public ❗			NO❗
└	withdrawnDividendOf	Public ❗			NO❗
└	accumulativeDividendOf	Public ❗			NO❗
└	_transfer	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_setBalance	Internal 🔒	🛑	

CONTRACT ASSESMENT

DividendTracker	Implementation	Ownable, DividendPayingToken		
└	\<Constructor\>	Public ❗	🛑	DividendPayingToken
└	_transfer	Internal 🔒		
└	withdrawDividend	Public ❗		NO❗
└	updateMinimumTokenBalanceForDividends	External ❗	🛑	onlyOwner
└	excludeFromDividends	External ❗	🛑	onlyOwner
└	updateClaimWait	External ❗	🛑	onlyOwner
└	setLastProcessedIndex	External ❗	🛑	onlyOwner
└	getLastProcessedIndex	External ❗		NO❗
└	getNumberOfTokenHolders	External ❗		NO❗
└	getAccount	Public ❗		NO❗
└	getAccountAtIndex	Public ❗		NO❗
└	canAutoClaim	Private 🔐		
└	setBalance	External ❗	🛑	onlyOwner
└	process	Public ❗	🛑	NO❗
└	processAccount	Public ❗	🛑	onlyOwner
MAI	Implementation	ERC20, Ownable		
└	\<Constructor\>	Public ❗	💵	ERC20
└	\<Receive Ether\>	External ❗	💵	NO❗
└	claimStuckTokens	External ❗	🛑	onlyOwner
└	isContract	Internal 🔒		
└	sendBNB	Internal 🔒	🛑	
└	_setAutomatedMarketMakerPair	Private 🔐	🛑	
└	excludeFromFees	External ❗	🛑	onlyOwner
└	isExcludedFromFees	Public ❗		NO❗
└	changeTreasuryWallet	External ❗	🛑	onlyOwner
└	_transfer	Internal 🔒	🛑	
└	swapAndSendDividends	Private 🔐	🛑	
└	setSwapTokensAtAmount	External ❗	🛑	onlyOwner
└	updateGasForProcessing	Public ❗	🛑	onlyOwner
└	updateMinimumBalanceForDividends	External ❗	🛑	onlyOwner
└	updateClaimWait	External ❗	🛑	onlyOwner
└	getClaimWait	External ❗		NO❗
└	getTotalDividendsDistributed	External ❗		NO❗
└	withdrawableDividendOf	Public ❗		NO❗
└	dividendTokenBalanceOf	Public ❗		NO❗
└	totalRewardsEarned	Public ❗		NO❗
└	excludeFromDividends	External ❗	🛑	onlyOwner
└	getAccountDividendsInfo	External ❗		NO❗
└	getAccountDividendsInfoAtIndex	External ❗		NO❗

CONTRACT ASSESMENT

└	processDividendTracker	External ❗	🛑	NO❗
└	claim	External ❗	🛑	NO❗
└	claimAddress	External ❗	🛑	onlyOwner
└	getLastProcessedIndex	External ❗		NO❗
└	setLastProcessedIndex	External ❗	🛑	onlyOwner
└	getNumberOfDividendTokenHolders	External ❗		NO❗

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS

```
Context. msgData() (contracts/TestToken.sol#26-28) is never used and should be removed
DividendPayingToken._transfer(address,address,uint256) (contracts/TestToken.sol#899-905) is never used and should be removed
IterableMapping.get(IterableMapping.Map,address) (contracts/TestToken.sol#195-197) is never used and should be removed
SafeMath.div(uint256,uint256) (contracts/TestToken.sol#105-107) is never used and should be removed
SafeMath.div(uint256,uint256,string) (contracts/TestToken.sol#109-119) is never used and should be removed
SafeMath.mod(uint256,uint256) (contracts/TestToken.sol#121-123) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (contracts/TestToken.sol#125-132) is never used and should be removed
SafeMathInt.abs(int256) (contracts/TestToken.sol#168-171) is never used and should be removed
SafeMathInt.div(int256,int256) (contracts/TestToken.sol#148-154) is never used and should be removed
SafeMathInt.mul(int256,int256) (contracts/TestToken.sol#139-146) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version0.8.17 (contracts/TestToken.sol#19) 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 MAI.sendBNB(address,uint256) (contracts/TestToken.sol#1326-1337):
    - (success) = recipient.call{value: amount}() (contracts/TestToken.sol#1332)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function IUniswapV2Pair.DOMAIN_SEPARATOR() (contracts/TestToken.sol#311) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (contracts/TestToken.sol#313) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (contracts/TestToken.sol#344) is not in mixedCase
Function IUniswapV2Router01.WETH() (contracts/TestToken.sol#384) is not in mixedCase
Parameter DividendPayingToken.dividendOf(address)._owner (contracts/TestToken.sol#863) is not in mixedCase
Parameter DividendPayingToken.withdrawableDividendOf(address)._owner (contracts/TestToken.sol#868) is not in mixedCase
Parameter DividendPayingToken.withdrawnDividendOf(address)._owner (contracts/TestToken.sol#874) is not in mixedCase
Parameter DividendPayingToken.accumulativeDividendOf(address)._owner (contracts/TestToken.sol#880) is not in mixedCase
Constant DividendPayingToken.magnitude (contracts/TestToken.sol#803) is not in UPPER_CASE_WITH_UNDERSCORES
Parameter DividendTracker.updateMinimumTokenBalanceForDividends(uint256)._newMinimumBalance (contracts/TestToken.sol#979) is not in mixedCase
Parameter DividendTracker.getAccount(address)._account (contracts/TestToken.sol#1024) is not in mixedCase
Parameter MAI.changeTreasuryWallet(address)._treasuryWallet (contracts/TestToken.sol#1370) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (contracts/TestToken.sol#389) is too similar to IUniswapV2Router01.ad
dLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (contracts/TestToken.sol#390)
Variable DividendPayingToken._withdrawDividendOfUser(address)._withdrawableDividend (contracts/TestToken.sol#840) is too similar to DividendTracker.getAccount(address).withdrawableDividends
(contracts/TestToken.sol#1032)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

SafeMathInt.MAX_INT256 (contracts/TestToken.sol#137) is never used in SafeMathInt (contracts/TestToken.sol#135-177)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

MAI.dividendTracker (contracts/TestToken.sol#1224) should be immutable
MAI.liquidityFeeOnBuy (contracts/TestToken.sol#1199) should be immutable
MAI.liquidityFeeOnSell (contracts/TestToken.sol#1205) should be immutable
MAI.rewardsFeeOnBuy (contracts/TestToken.sol#1201) should be immutable
MAI.rewardsFeeOnSell (contracts/TestToken.sol#1207) should be immutable
MAI.totalBuyFee (contracts/TestToken.sol#1203) should be immutable
MAI.totalSellFee (contracts/TestToken.sol#1209) should be immutable
MAI.treasuryFeeOnBuy (contracts/TestToken.sol#1200) should be immutable
MAI.treasuryFeeOnSell (contracts/TestToken.sol#1206) should be immutable
MAI.uniswapV2Pair (contracts/TestToken.sol#1214) should be immutable
MAI.uniswapV2Router (contracts/TestToken.sol#1213) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
```

Result => A static analysis of contract's source code has been performed using slither,
No major issues were found in the output

FUNCTIONAL TESTING

Router (PCS V2):
0xD99D1c33F9fC3444f8101754aBC46c52416550D1

1- Adding Liquidity (Passed):
liquidity added on Pancakeswap V2:

https://testnet.bscscan.com/tx/0x1379e0311e1ae3ca5d349bf84d71e86ad72e5799dad927ec8c8d28754a63bb7b

no issue were found on adding liquidity.

2- Buying (treasury + rewards fee | 1% max)(Passed):

https://testnet.bscscan.com/tx/0x58943848c5c656936978f39034c7e326814055167eb0c893d5ee242b302e37cd

3- Selling (treasury fee | 1% max)(Passed):

https://testnet.bscscan.com/tx/0x9e7eeda12433ecd18bfa72178e30ef31996e235e560e5efb823be117e72a964e

4-Transferring (0% tax)(Passed):

https://testnet.bscscan.com/tx/0x1e8f572b7a6d9200fc149ab9c2baeb01f6e4db3ddd4f82ca3673387c91ffb532

FUNCTIONAL TESTING

5-Rewards distributed (Passed):

BUSD token sent to holder as seen in this transactions

https://testnet.bscscan.com/tx/0x9e7eeda12433ecd18bfa72178e3
0ef31996e235e560e5efb823be117e72a964e

https://testnet.bscscan.com/tx/0x459734842c6ffc2714e238c654
abfde3afe582615a9a6b8b46dadc343a302998

6-Internal Swap(Passed):

https://testnet.bscscan.com/tx/0x9e7eeda12433ecd18bfa72178e3
0ef31996e235e560e5efb823be117e72a964e

MANUAL TESTING

NO ISSUES FOUND

Social Media Overview

Here are the Social Media Accounts of

MatrixGPT

https://t.me/MatrixGPT_ann

https://twitter.com/MatrixGPT

https://matrixgpt.ai

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