auditace.tech

AuditAce

FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

MafiaPepe

DATED : 25 May 23'

AUDIT SUMMARY

Project name – MafiaPepe

Date: 25 May, 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	1	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/0x11F0A094BBF9b57aA44E1f094542b71F6341426b

Token Information

Token Name : MafiaPepe

Token Symbol: MFP

Decimals: 18

Token Supply:420,690,000,000,000

Token Address:
0xD768dAD1FeeDaC5A1c2627AAbA711710bEd208d7

Checksum:
d1f7a9c7f9129f0fb998642fa62f27e5a6991245

Owner:
0xe11d0Ea7e24DCDaB70225beFA94E42c6574D354A

Network: Ethereum

Token Type: ERC-20

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.

auditace.tech

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

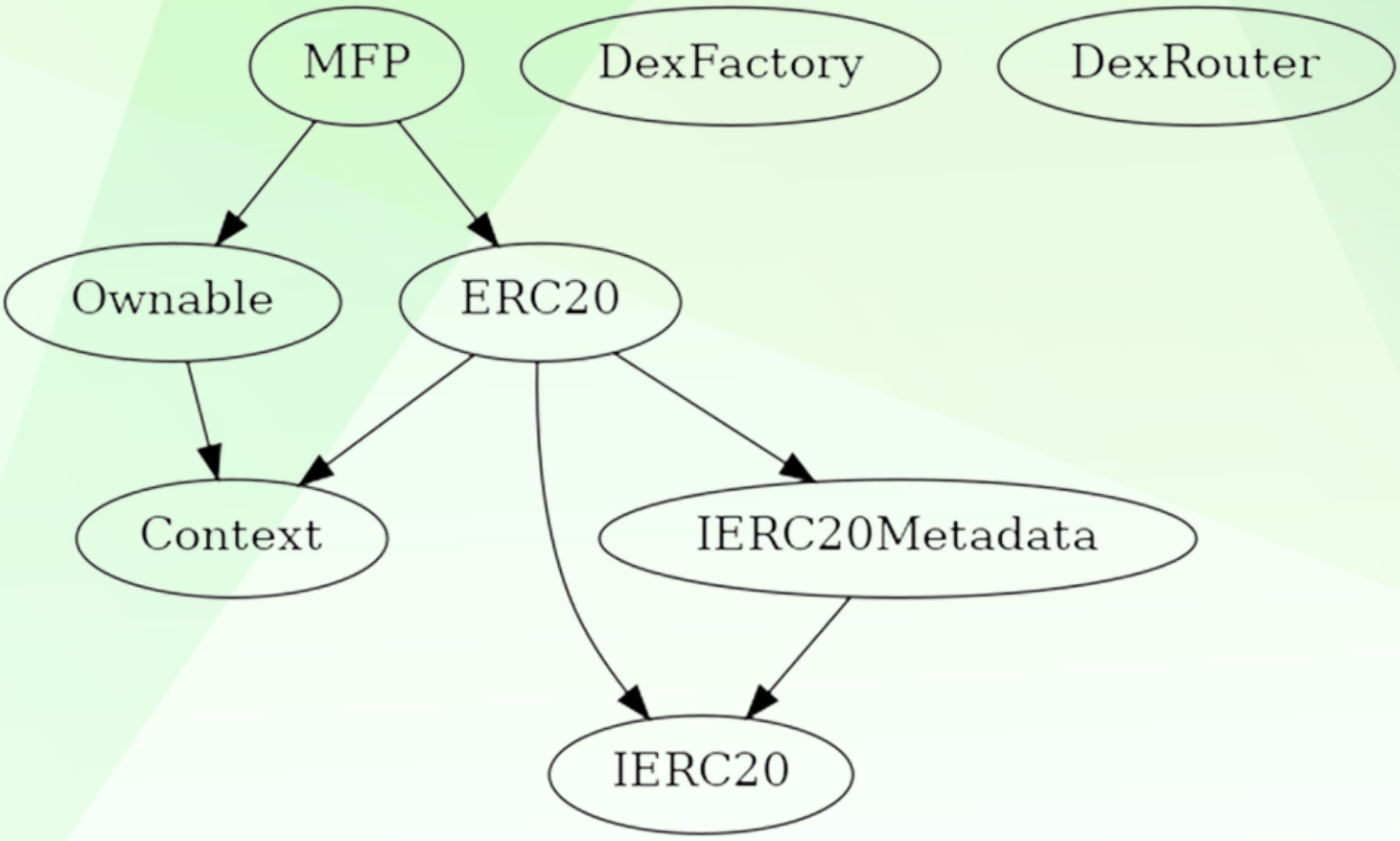
CLASSIFICATION OF RISK

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

INHERITANCE TREE



POINTS TO NOTE

- Owner is not able to set sell/buy/transfer fees (0% static)
- Owner is not able to set max buy/sell/transfer/hold amount
- Owner is not able to blacklist an arbitrary wallet
- Owner is not able able to limit buys/transfers/sells by a max amount as limit
- Owner is not able to mint new tokens
- Owner must enabel trades manually for holders

STATIC ANALYSIS

```
Context._msgData() (contracts/Token.sol#13-15) is never used and should be removed
ERC20._burn(address,uint256) (contracts/Token.sol#224-235) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.17 (contracts/Token.sol#6) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16
solc-0.8.19 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Function DexRouter.WETH() (contracts/Token.sol#288) is not in mixedCase
Parameter MFP.setWhitelisted(address,bool)._wallet (contracts/Token.sol#338) is not in mixedCase
Parameter MFP.setWhitelisted(address,bool)._status (contracts/Token.sol#338) is not in mixedCase
Parameter MFP.clearERC20(address,address,uint256)._tokenAddr (contracts/Token.sol#349) is not in mixedCase
Parameter MFP.clearERC20(address,address,uint256)._to (contracts/Token.sol#350) is not in mixedCase
Parameter MFP.clearERC20(address,address,uint256)._amount (contracts/Token.sol#351) is not in mixedCase
Parameter MFP.checkWhitelisted(address)._wallet (contracts/Token.sol#357) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

MFP.constructor() (contracts/Token.sol#319-330) uses literals with too many digits:
        - _mint(msg.sender,420690000000000 * 10 ** decimals()) (contracts/Token.sol#329)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
```

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			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO❗
└	owner	Public ❗		NO❗
└	_checkOwner	Internal 🔒		
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_transferOwnership	Internal 🔒	🛑	
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❗
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❗

CONTRACT ASSESMENT

└	_transfer	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_approve	Internal 🔒	🛑	
└	_spendAllowance	Internal 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
└	_afterTokenTransfer	Internal 🔒	🛑	
DexFactory	Interface		
└	createPair	External ❗	🛑	NO❗
DexRouter	Interface		
└	factory	External ❗		NO❗
└	WETH	External ❗		NO❗
└	addLiquidityETH	External ❗	💵	NO❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
MFP	Implementation	ERC20, Ownable	
└	<Constructor>	Public ❗	🛑	ERC20
└	enableTrading	External ❗	🛑	onlyOwner
└	setWhitelisted	External ❗	🛑	onlyOwner
└	clearETH	External ❗	🛑	onlyOwner
└	clearERC20	External ❗	🛑	onlyOwner
└	checkWhitelisted	External ❗		NO❗
└	_transfer	Internal 🔒	🛑	

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

FUNCTIONAL TESTING

Router (PCS V2):
0xD99D1c33F9fC3444f8101754aBC46c52416550D1

All the functionalities have been tested, no issues were found

1- Adding liquidity (passed):
https://testnet.bscscan.com/tx/0xcc74261ae2c45608d48ae0ace935f94afce57187cb3636d35e8297f971648fed

2- Buying (0% tax) (passed):
https://testnet.bscscan.com/tx/0x7348f62ab17bc3897180cfcf9bfd8ac910e5aa41137ccf151569f06acea7ece0

3- Selling (0% tax) (passed):
https://testnet.bscscan.com/tx/0x5a6644ce400ce46ab4621ac7e316e6a3dc353aabd54de2a9cc352ab60fd92df6

4- Transferring (0% tax) (passed):
https://testnet.bscscan.com/tx/0x6a0ff95f25e5154434dce8195317fcbd8a06cf1e91ffc01e35d38dbf76a247bc

Manual Testing

Centralization – Owner must enable trades

Severity: High

function: enableTrading

Status: Not Resolved

Overview:

Owner must enable trades for investors manually. If trades remain disabled, no one would be able to buy/sell/transfer tokens (except owner)

```
function enableTrading() external onlyOwner {
    require(!tradingEnabled, "Trading is already enabled");
    tradingEnabled = true;
    startTradingBlock = block.number;
}
```

Suggestion

To mitigate this issue, there are several options:

- Enable trades before starting the presale

- Transfer ownership of the contract to a trust 3rd party like pinksale (safu dev) in order to guarantee that trades will be enabled

- create a mechanism which will enable trades automatically after a preiod of time

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