CONTRACT WOLF

Blockchain Security - Smart Contract Audits

Security Assessment

June 4, 2022

Disclaimer

ContractWolf.io audits and reports should not be considered as a form of project's "advertisement" and does not cover any interaction and assessment from "project's contract" to "external contracts" such as Pancakeswap or similar.

ContractWolf does not provide any warranty on its released reports.

ContractWolf should not be used as a decision to invest into an audited project and is not affiliated nor partners to its audited contract projects.

ContractWolf provides transparent report to all its "clients" and to its "clients participants" and will not claim any guarantee of bug-free code within it's **SMART CONTRACT**.

ContractWolf presence is to analyze, audit and assess the client's smart contract's code.

Each company or projects should be liable to its security flaws and functionalities.

Scope of Work

Mimos Coin's team agreed and provided us with the files that needs to be tested (Github, Bscscan, Etherscan, files, etc.). The scope of the audit is the main contract.

The goal of this engagement was to identify if there is a possibility of security flaws in the implementation of the contract or system.

ContractWolf will be focusing on contract issues and functionalities along with the projects claims from smart contract to their website, whitepaper and repository which has been provided by **Mimos Coin.**

Network

Binance Smart Chain (BEP20)

Contract link

https://bscscan.com/address/0x9E354B618d472b76cE35a4ab76916B4B51355428

Website

https://www.mimoscoin.io/

Telegram

https://t.me/mimoscoininfo

Twitter

https://twitter.com/mimoscoin

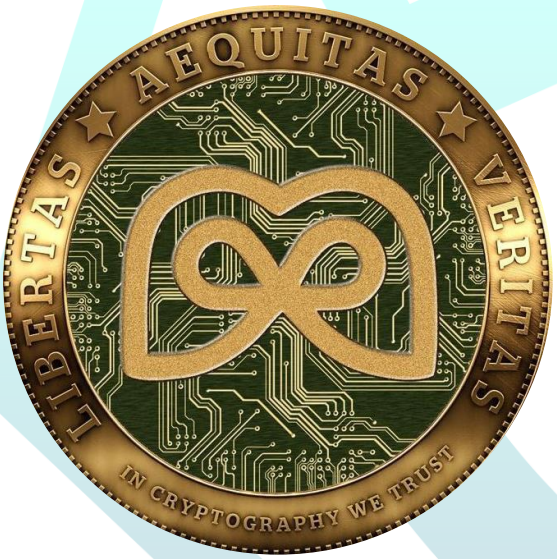
Discord

https://discord.com/invite/kGMRVcNty2

Description

MimosCoin is a peer-to-peer token, hosted on the Binance Smart chain blockchain network, with the intention of facilitating and enabling value transactions by feeding a pre-existing ecosystem to use the asset. It behaves like a commodity like gold and the like, having its value defined by demand and supply for it.

Logo



Risk Level Classification

Risk Level represents the classification or the probability that a certain function or threat that can exploit vulnerability and have an impact within the system or contract.
Risk Level is computed based on CVSS Version 3.0

Level	Value	Vulnerability
Critical	9 - 10	An Exposure that can affect the contract functions in several events that can risk and disrupt the contract
High	7 - 8.9	An Exposure that can affect the outcome when using the contract that can serve as an opening in manipulating the contract in an unwanted manner
Medium	4 - 6.9	An opening that could affect the outcome in executing the contract in a specific situation
Low	0.1 - 3.9	An opening but doesn't have an impact on the functionality of the contract
Informational	0	An opening that consists of information's but will not risk or affect the contract

Auditing Approach

Every line of code along with its functionalities will undergo manual review to check its security issues, quality, and contract scope of inheritance. The manual review will be done by our team that will document any issues that there were discovered.

Methodology

The auditing process follows a routine series of steps:

1. Code review that includes the following:

    - Review of the specifications, sources, and instructions provided to ContractWolf to make sure we understand the size, scope, and functionality of the smart contract.
    - Manual review of code, our team will have a process of reading the code line-by-line with the intention of identifying potential vulnerabilities and security flaws.

2. Testing and automated analysis that includes:

    - Testing the smart contract functions with common test cases and scenarios, to ensure that it returns the expected results.

3. Best practices review, the team will review the contract with the aim to improve efficiency, effectiveness, clarifications, maintainability, security, and control within the smart contract.

4. Recommendations to help the project take steps to secure the smart contract.

Used Code from other Frameworks/Smart Contracts (Direct Imports)

Imported Packages

- Context
- IERC20
- IERC20Metadata
- PancakeFactory
- PancakeRouter
- Ownable
- Strings
- ERC20
- SwapHelper
- Authorized
- MimosContract

Description

Optimization enabled: Yes

Decimal: 18

Symbol: MMC

Max / Total supply: 100,000,000

Capabilities

Components

Version	Contracts	Libraries	Interfaces	Abstract
1.0	4	1	4	2

Exposed Functions

Version	Public	Private	External	Internal
1.0	12	4	27	21

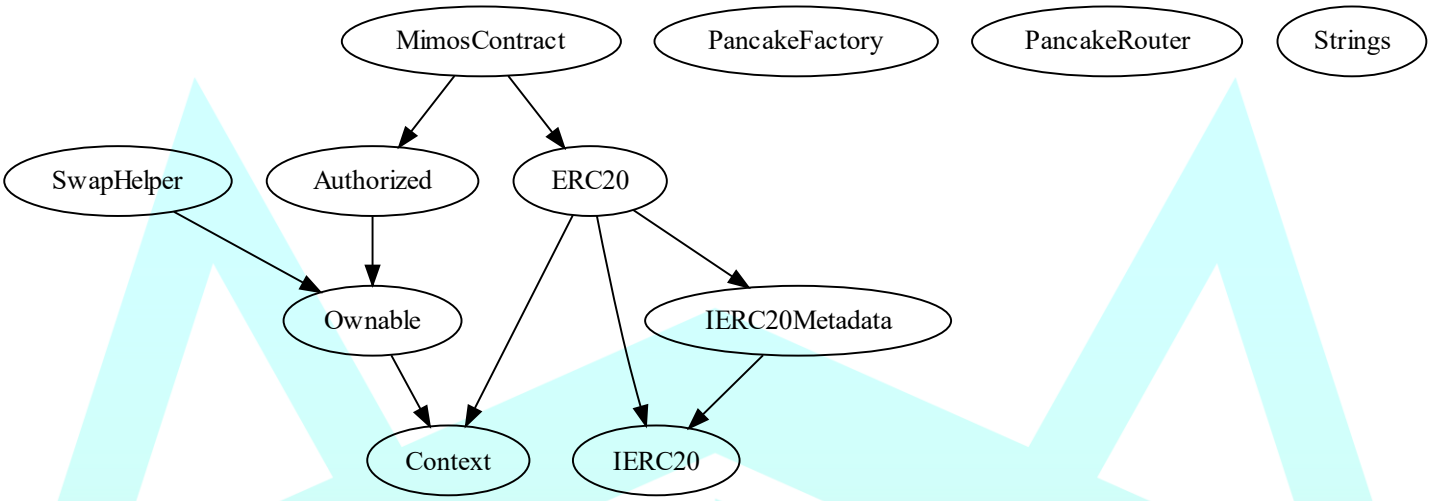
State Variables

Version	Total	Public
1.0	32	12

Capabilities

Version	Solidity Versions Observed	Experimental Features	Can Receive Funds	Uses Assembly	Has Destroyable Contracts
1.0	v0.8.7		Yes	Yes	No

Inheritance Graph



Correct implementation of Token Standard

Tested	Verified
✓	✓

Overall Checkup (Smart Contract Security)

Tested	Verified
✓	✓

Function	Description	Exist	Tested	Verified
TotalSupply	Information about the total coin or token supply	✓	✓	✓
BalanceOf	Details on the account balance from a specified address	✓	✓	✓
Transfer	An action that transfers a specified amount of coin or token to a specified address	✓	✓	✓
TransferFrom	An action that transfers a specified amount of coin or token from a specified address	✓	✓	✓
Approve	Provides permission to withdraw specified number of coin or token from a specified address	✓	✓	✓

Verify Claims

Statement	Exist	Tested	Deployer
Renounce Ownership	✓	✓	✓
Mint	✓	✓	✗
Burn	✓	✓	✓
Block	–	–	–
Pause	–	–	–

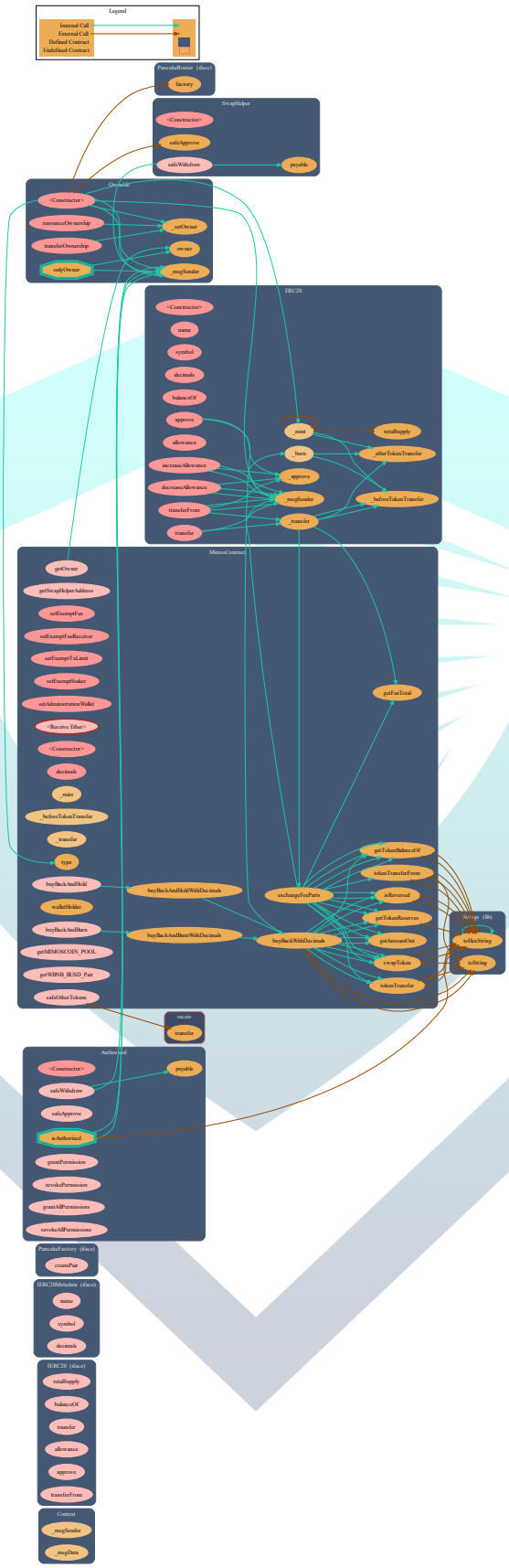
Legend

Attribute	Symbol
Verified / Can	✓
Verified / Cannot	✗
Unverified / Not checked	⚑
Not Available	–

Write Functions of Contract

1. approve
2. buyBackAndBurn
3. buyBackAndBurnWithDecimals
4. buyBackAndHold
5. buyBackAndHoldWithDecimals
6. decreaseAllowance
7. grantAllPermissions
8. grantPermission
9. increaseAllowance
10. renounceOwnership
11. revokeAllPermissions
12. revokePermission
13. safeApprove
14. safeOtherTokens
15. safeWithdraw
16. setAdministrationWallet
17. setExemptFee
18. setExemptFeeReceiver
19. setExemptStaker
20. setExemptTxLimit
21. transfer
22. transferFrom
23. transferOwnership

Call Graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
PancakeRouter (iface)
factory
SwapHelper
<Constructor>
safeApprove
safeWithdraw
payable
Ownable
<Constructor>
renounceOwnership
transferOwnership
onlyOwner
_setOwner
owner
_msgSender
ERC20
<Constructor>
name
symbol
decimals
balanceOf
approve
allowance
increaseAllowance
decreaseAllowance
transferFrom
transfer
_mint
_burn
_approve
_msgSender
_transfer
totalSupply
_afterTokenTransfer
_beforeTokenTransfer
MimosoContract
getOwner
getSwapHelperAddress
setExemptFee
setExemptFeeReceiver
setExemptTxLimit
setExemptStaker
setAdministrationWallet
<Receive Ether>
<Constructor>
decimals
mint
_beforeTokenTransfer
_transfer
type
buyBackAndHold
walletHolder
buyBackAndBurn
getMIMOSCOIN_POOL
getWBNB_BUSD_Pair
safeOtherTokens
buyBackAndHoldWithDecimals
buyBackAndBurnWithDecimals
getFeeTotal
exchangeFeeParts
buyBackWithDecimals
getTokenBalanceOf
tokenTransferFrom
isReversed
getTokenReserves
getAmountOut
swapToken
tokenTransfer
Strings (lib)
toHexString
toString
rescue
_transfer
Authorized
<Constructor>
payable
safeWithdraw
safeApprove
isAuthorized
grantPermission
revokePermission
grantAllPermissions
revokeAllPermissions
PancakeFactory (iface)
createPair
IERC20Metadata (iface)
name
symbol
decimals
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom
Context
_msgSender
_msgData

SWC Attacks

ID	Title	Status
SWC-136	Unencrypted Private Data On-Chain	PASSED
SWC-135	Code With No Effects	PASSED
SWC-134	Message call with hardcoded gas amount	PASSED
SWC-133	Hash Collisions with Multiple Variable Length Arguments	PASSED
SWC-132	Unexpected Ether balance	PASSED
SWC-131	Presence of unused variables	PASSED
SWC-130	Right-To Left Override control character (U+202E)	PASSED
SWC-129	Typographical Error	PASSED
SWC-128	DoS With Block Gas Limit	PASSED
SWC-127	Arbitrary Jump with Function Type Variable	PASSED
SWC-126	Insufficient Gas Griefing	PASSED
SWC-125	Incorrect Inheritance Order	PASSED
SWC-124	Write to Arbitrary Storage Location	PASSED
SWC-123	Requirement Violation	PASSED
SWC-122	Lack of Proper Signature Verification	PASSED
SWC-121	Missing Protection against Signature Replay Attacks	PASSED
SWC-120	Weak Sources of Randomness from Chain Attributes	PASSED
SWC-119	Shadowing State Variables	PASSED
SWC-118	Incorrect Constructor Name	PASSED
SWC-117	Signature Malleability	PASSED
SWC-116	Block values as a proxy for time	PASSED
SWC-115	Authorization through tx.origin	PASSED
SWC-114	Transaction Order Dependence	PASSED
SWC-113	DoS with Failed Call	PASSED
SWC-112	Delegate call to Untrusted Callee	PASSED
SWC-111	Use of Deprecated Solidity Functions	PASSED

SWC-110	Assert Violation	**PASSED**
SWC-109	Uninitialized Storage Pointer	**PASSED**
SWC-108	State Variable Default Visibility	**LOW ISSUE**
SWC-107	Reentrancy	**PASSED**
SWC-106	Unprotected SELFDESTRUCT Instruction	**PASSED**
SWC-105	Unprotected Ether Withdrawal	**PASSED**
SWC-104	Unchecked Call Return Value	**PASSED**
SWC-103	Floating Pragma	**LOW ISSUE**
SWC-102	Outdated Compiler Version	**PASSED**
SWC-101	Integer Overflow and Underflow	**PASSED**
SWC-100	Function Default Visibility	**PASSED**

AUDIT PASSED

Low Issues

A floating pragma is set (SWC-103)	L: 2	MimosContract.sol
A floating pragma is set (SWC-103)	L: 3	Authorized.sol Context.sol ERC20.sol IERC20.sol IERC20Metadata.sol IPancake.sol Ownable.sol Strings.sol SwapHelper.sol
State variable visibility is not set (SWC-108)	L: 66, 67	MimosContract.sol

Audit Comments

- Deployer can renounce ownership
- Deployer can transfer ownership
- Deployer can safe approve addresses
- Deployer can withdraw tokens from contract
- Authorized 0 can safe approve addresses
- Authorized 0 can withdraw tokens from contract
- Authorized 0 can grant permissions to an address
- Authorized 0 can revoke permissions to an address
- Authorized 0 can set administrator wallet address
- Authorized 0 can withdraw tokens from contract to an address
- Authorized 2 can exclude/include addresses from fees
- Authorized 2 can exclude/include addresses from exempt fee receivers
- Authorized 2 can exclude/include addresses from transaction limit
- Authorized 2 can exclude/include addresses from stakers
- Authorized 3 can buy back and hold tokens
- Authorized 3 can buy back and burn tokens
- Deployer cannot mint after initial deployment
- Deployer cannot pause
- Deployer cannot burn
- Deployer cannot block users

CONTRACTWOLF

Blockchain Security - Smart Contract Audits