Smart Contract Vulnerability Audit

Metabolic

Dec 1, 2021

auditsfinance auditsfinancegroup

Table of Contents

DISCLAIMER

Audits.finance Inc. is in no way responsible or liable for any legal actions resulting from the use of this presentation. By reading this audit or any part of it, you agree to the terms of this disclaimer. If you do not agree to these terms, please stop reading now, and delete any duplicates of this report. Audits.finance Inc. is an official auditor utilizing the Solidity auditing industry standard. Audits.finance hereby excludes any liability and responsibility. Neither you nor any other person shall have any claim against Audits.finance for any economic loss or damages. Audits.finance Inc. does not guarantee the authenticity of a project, nor does it guarantee the project will not participate in one or any scamming including but not limited to, removing liquidity, selling off team supply, or exit scams. Audits.finance Inc. does not give investment advice in any way. Audits.finance Inc. supplies this presentation for information purposes only, and strongly suggests that none of this information be used as investment advice. Audits.finance in no way endorses or recommends any projects that it audits. Audits.finance is solely responsible for smart contract and project analysis of the projects that it is contracted to audit. Audits.finance may be contracted by teams, investors, or any other 3rd party in regard to a contract address or project. Audits.finance provides a full report for informational purposes only.

Smart Contract – Audit Overview

Project Summary

Project Name	Metabolic
Platform	Binance Smart Chain
Language	Solidity
Commits	0x1184c14af1ea6a7ce5df501ac91b8b0d9dcfad17

Audit Summary

Delivery Date	December 1, 2021
Method of Audit	Human and AI
Consultants Engaged	One
Timeline	November 30, 2021 – December 2, 2021

Vulnerability Summary

Vulnerability Level	Total	Resolved
Critical	0	✓
Major	0	✓
Medium	0	✓
Minor	3	X
Informational	1	X

Smart Contract - Contract Overview

All information is recorded as of 12/01/2021.

Contract Name	Metabolic.sol
Contract Ticker	MBTC
Contract Address	0x1184c14AF1Ea6A7CE5df501ac91B8B0D9dcFaD17
Contract Creator	0x2364fBb8321246a79C62D56cC2c6521412DB81Bb
Decimals	18
Total Supply	10,000,000,000
Token Holders	1
Token Transfers	2
Complier Version	v0.8.7+commit.e28d00a7
Source Code	Solidity
Optimization Enabled	Yes with 200 runs
Other Settings	default evmVersion, MIT license

Smart Contract - Vulnerabilities

Vulnerability Tested	Human Review	Ai Review	Line(s) Affected	Results
Function Default Visibility	✓	✓		✓
Integer Overflow and Underflow	✓	✓		✓
Outdated Compiler Version	✓	✓		✓
Floating Pragma	✓	✓	L12	✗
Unchecked Call Return Value	✓	✓		✓
Unprotected Ether Withdrawal	✓	✓		✓
Unprotected SELFDESTRUCT Instruction	✓	✓		✓
Unencrypted Private Data On-Chain	✓	✓		✓

Smart Contract - Vulnerabilities

Vulnerability Tested	Human Review	Ai Review	Line(s) Affected	Results
Reentrancy	✓	✓		✓
State Variable Default Visibility	✓	✓	L556	✗
Uninitialized Storage Pointer	✓	✓		✓
Assert Violation	✓	✓		✓
Use of Deprecated Solidity Functions	✓	✓		✓
Delegatecall to Untrusted Callee	✓	✓		✓
DoS with Failed Call	✓	✓		✓
Code With No Effects	✓	✓		✓

Smart Contract - Vulnerabilities

Vulnerability Tested	Human Review	Ai Review	Line(s) Affected	Results
Transaction Order Dependence	✓	✓		✓
Authorization through tx.origin	✓	✓	L688, L784	✗
Block values as a proxy for time	✓	✓		✓
Signature Malleability	✓	✓		✓
Incorrect Constructor Name	✓	✓		✓
Shadowing State Variables	✓	✓		✓
Weak Sources of Randomness from Chain Attributes	✓	✓		✓

Smart Contract - Vulnerabilities

Vulnerability Tested	Human Review	Ai Review	Line(s) Affected	Results
Missing Protection against Signature Replay Attacks	✓	✓		✓
Lack of Proper Signature Verification	✓	✓		✓
Requirement Violation	✓	✓		✓
Write to Arbitrary Storage Location	✓	✓		✓
Incorrect Inheritance Order	✓	✓		✓
Insufficient Gas Griefing	✓	✓		✓
Arbitrary Jump with Function Type Variable	✓	✓		✓

Smart Contract - Vulnerabilities

Vulnerability Tested	Human Review	Ai Review	Line(s) Affected	Results
DoS With Block Gas Limit	✓	✓		✓
Typographical Error	✓	✓		✓
Right-To-Left-Override control character	✓	✓		✓
Presence of unused variables	✓	✓		✓
Unexpected Ether balance	✓	✓		✓
Hash Collisions With Multiple Variable Length Arguments	✓	✓		✓
Message call with hardcoded gas amount	✓	✓		✓

Smart Contract - Code Analysis

Floating Pragma
Severity: Informational
Metabolic.sol
Line: 12

The current pragma Solidity directive is ""^0.8.3"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

```
10 // Dev Teemo
11 //No Comments made below to prevent skids
12 pragma solidity ^0.8.3;
13 abstract contract Context {
14 function _msgSender() internal view virtual returns (address payable) {
```

State variable visibility is not set
Severity: Minor
Metabolic.sol
Line: 556

It is best practice to set the visibility of state variables explicitly. The default visibility for "inSwapAndLiquify" is internal. Other possible visibility settings are public and private.

```
554 IUniswapV2Router02 public uniswapV2Router;
555 address public uniswapV2Pair;
556 bool inSwapAndLiquify;
557 bool public swapAndLiquifyEnabled = false;
558 bool public tradingActive = false;
```

Smart Contract - Code Analysis

Potential use of "block.number" as source of randonmness
Severity: Minor
Metabolic.sol
Line: 688

The environment variable "block.number" looks like it might be used as a source of randomness. Note that the values of variables like coinbase, gaslimit, block number and timestamp are predictable and can be manipulated by a malicious miner. Also keep in mind that attackers know hashes of earlier blocks. Don't use any of those environment variables as sources of randomness and be aware that use of these variables introduces a certain level of trust into miners.

```
686 tradingActive = true;
687 swapAndLiquifyEnabled = true;
688 tradingActiveBlock = block.number;
689 earlyBuyPenaltyEnd = block.timestamp + 24 hours;
690 }
```

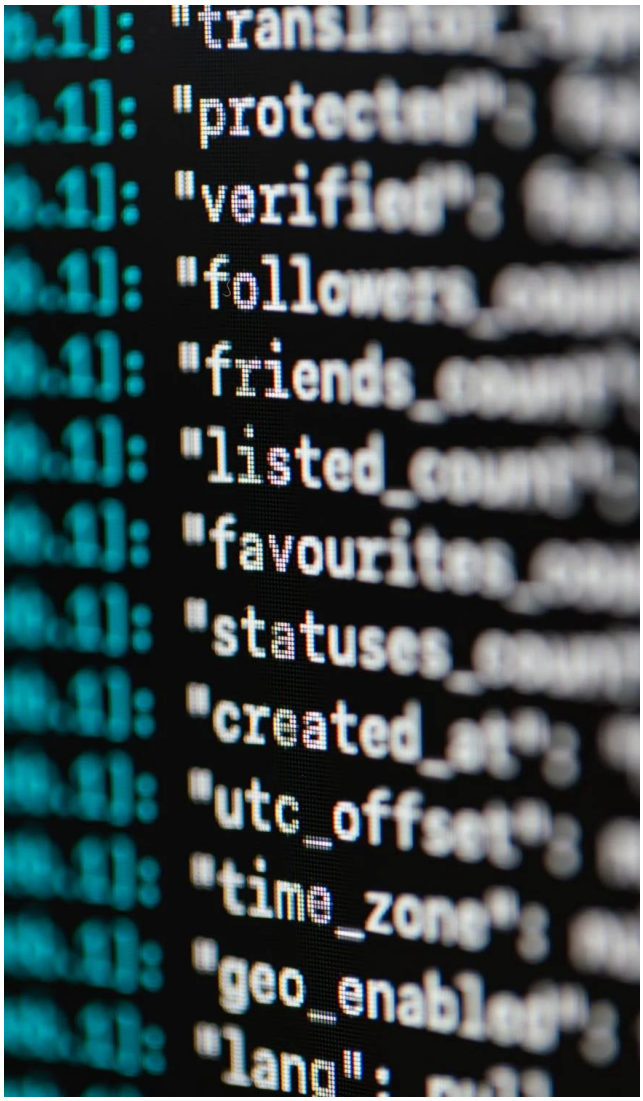
Potential use of "block.number" as source of randonmness
Severity: Minor
Metabolic.sol
Line: 784

The environment variable "block.number" looks like it might be used as a source of randomness. Note that the values of variables like coinbase, gaslimit, block number and timestamp are predictable and can be manipulated by a malicious miner. Also keep in mind that attackers know hashes of earlier blocks. Don't use any of those environment variables as sources of randomness and be aware that use of these variables introduces a certain level of trust into miners.

```
782 require(_isExcludedFromFee[from] || _isExcludedFromFee[to], "Trading is not active yet.");
783 }
784 if(from != owner() && to != uniswapV2Pair && block.number == tradingActiveBlock){
785 boughtEarly[to] = true;
786 }
```

Smart Contract –
Mint function

This contract cannot mint new MTCB tokens. We were unable to locate a mint function that is used to mint new Metabolic tokens.



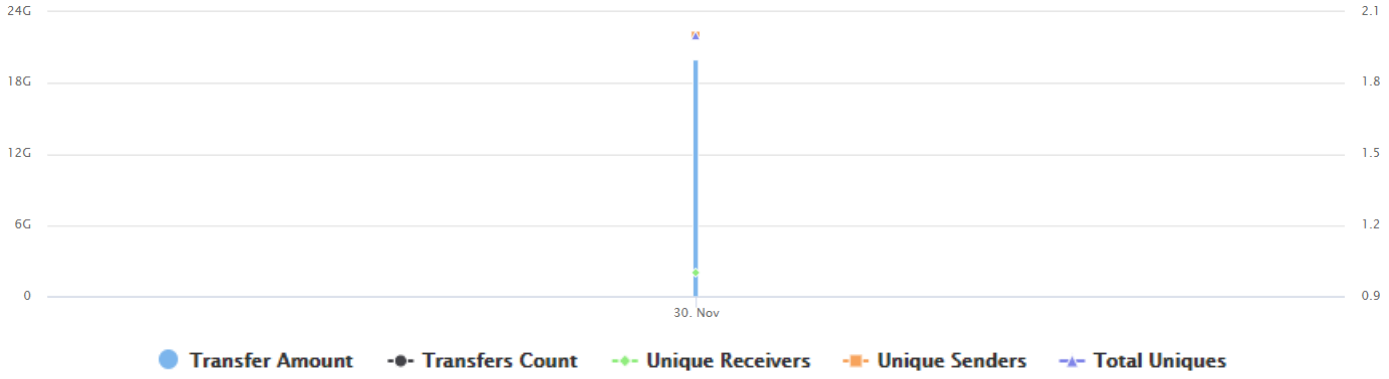
Smart Contract – Contract Ownership

Contract ownership has not been renounced at the time of the audit. The owner's address is shown as: **0x2364fbb8321246a79c62d56cc2c6521412db81bb**

Token Holders & Contract Analytics

Top 100 Token Holders

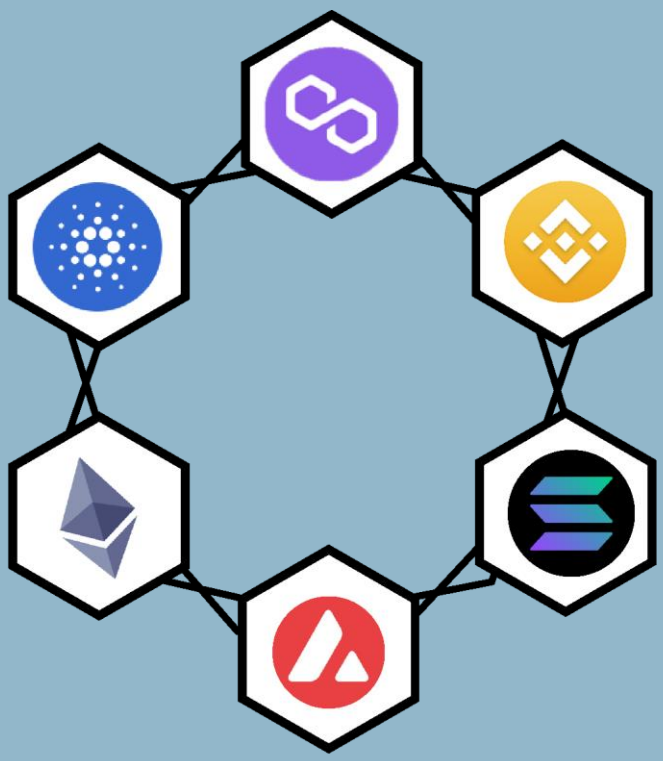
Token Contract Analytics



Team Overview

KYC NOT CERTIFIED

Audits.finance has not completed a KYC for the project. Audits.finance has not verified the identity of any team member(s) with government issued ID and photo evidence to match. This project is anonymous.



audits.finance

Contact information:
Website: audits.finance
Telegram: auditsfinancegroup
Twitter: auditsfinance
Email: hello@audits.finance