Smart Contract Security Audit

REHASHCRYPTO

MOONTRAVELS TOKEN

REHASHCRYPTO received the application for a smart contract security audit of MOONTRAVELS on February 26, 2022. The following are the details and results of this smart contract security audit:

Token Name: Moon Travels

Contract address: 0xaebb04A119CD6035f2aD573e9C716BC0158a0513

Address Link:
https://bscscan.com/token/0xaebb04A119CD6035f2aD573e9C716BC0158a0513

Uniswap V2 pair: 0xe163682fc068b2ae17833aba97986df232ee6c72

Contract Deployer address: 0xAc7700C6BB59186DE8C455e4e04C320Da5e90417

Contract's current owner address: 0xAc7700C6BB59186DE8C455e4e04C320Da5e90417

The audit items and results:

(Other unknown security vulnerabilities are not included in the audit responsibility scope)

Audit Result: Passed Ownership: Not renounced (Can be Renounced)
(The contract contains ownership functionality and ownership is not renounced which allows the creator or current owner to modify contract behavior)
(Contract owner can modify fees and can modify max tx)
KYC Verification: Not verified

Audit Team: REHASHCRYPTO https://rehashcrypto.com/

Introduction

This Audit Report mainly focuses on the overall security of MOONTRAVELS Smart Contract. With this report, we have tried to ensure the reliability and correctness of their smart contract by complete and rigorous assessment of their system's architecture and the smart contract codebase.

Auditing Approach and Methodologies applied

The REHASHCRYPTO team has performed rigorous testing of the project starting with analyzing the code design patterns in which we reviewed the smart contract architecture to ensure it is structured and safe use of third-party smart contracts and libraries.

Our team then performed a formal line by line inspection of the Smart Contract to find any potential issue like race conditions, transaction-ordering dependence, timestamp dependence, and denial of service attacks.

In the Unit testing Phase, we coded/conducted custom unit tests written for each function in the contract to verify that each function works as expected.

In Automated Testing, we tested the Smart Contract with our in-house developed tools to identify vulnerabilities and security flaws.

The code was tested in collaboration of our multiple team members and this included -

- Testing the functionality of the Smart Contract to determine proper logic has been followed throughout the whole process.
- Analyzing the complexity of the code in depth and detailed, manual review of the code, line- by-line.
- Deploying the code on testnet using multiple clients to run live tests.
- Analyzing failure preparations to check how the Smart Contract performs in case of any bugs and vulnerabilities.
- Checking whether all the libraries used in the code are on the latest version.
- Analyzing the security of the on-chain data.

Audit Details

Project Name: MOONTRAVELS

Website: https://moon-travels.com/

Platform: Binance Smart Chain
Type of Token: BEP20

Languages: Solidity (Smart contract)
Platforms and Tools: Remix IDE, Truffle, Ganache, Solhint, Mythril, Contract Library

Audit Goals

The focus of the audit was to verify that the Smart Contract System is secure, resilient and working according to the specifications. The audit activities can be grouped in the following three categories:

Security

Identifying security related issues within each contract and the system of contract.

Sound Architecture

Evaluation of the architecture of this system through the lens of established smart contract best practices and general software best practices.

Code Correctness and Quality

A full review of the contract source code. The primary areas of focus include:

- Accuracy
- Readability
- Sections of code with high complexity
- Quantity and quality of test coverage

Issue Categories

Every issue in this report was assigned a severity level from the following:

High level severity issues

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

Medium level severity issues

Issues on this level could potentially bring problems and should eventually be fixed.

Low level severity issues

Issues on this level are minor details and warnings that can remain unfixed but would be better fixed at some point in the future.

Manual Audit:

For this section the code was tested/read line by line by our developers. We also used Remix IDE's
JavaScript VM and Kovan networks to test the contract functionality.

Contract functions details

+ Context
  - [Int] _msgSender
  - [Int] _msgData

+ [Int] IERC20
  - [Ext] totalSupply
  - [Ext] balanceOf
  - [Ext] transfer #
  - [Ext] allowance
  - [Ext] approve #
  - [Ext] transferFrom #

+ [Lib] SafeMath
  - [Int] add
  - [Int] sub
  - [Int] sub
  - [Int] mul
  - [Int] div
  - [Int] div
  - [Int] mod
  - [Int] mod

+ [Lib] Address
  - [Int] isContract
  - [Int] sendValue #
  - [Int] functionCall #
  - [Int] functionCall #
  - [Int] functionCallWithValue #
  - [Int] functionCallWithValue #
  - [Prv] _functionCallWithValue #

+ Ownable (Context)
  - [Pub] <Constructor> #
  - [Pub] owner
  - [Pub] renounceOwnership #
    - modifiers: onlyOwner
  - [Pub] transferOwnership #
    - modifiers: onlyOwner
  - [Pub] getUnlockTime
  - [Pub] getTime

- [Pub] lock #
  - modifiers: onlyOwner
- [Pub] unlock #

+ [Int] IUniswapV2Factory
- [Ext] feeTo
- [Ext] feeToSetter
- [Ext] getPair
- [Ext] allPairs
- [Ext] allPairsLength
- [Ext] createPair #

- [Ext] setFeeTo #
- [Ext] setFeeToSetter #

+ [Int] IUniswapV2Pair
- [Ext] name
- [Ext] symbol
- [Ext] decimals
- [Ext] totalSupply
- [Ext] balanceOf
- [Ext] allowance
- [Ext] approve #
- [Ext] transfer #
- [Ext] transferFrom #
- [Ext] DOMAIN_SEPARATOR
- [Ext] PERMIT_TYPEHASH
- [Ext] nonces
- [Ext] permit #
- [Ext] MINIMUM_LIQUIDITY
- [Ext] factory
- [Ext] token0
- [Ext] token1
- [Ext] getReserves
- [Ext] price0CumulativeLast
- [Ext] price1CumulativeLast
- [Ext] kLast
- [Ext] burn #
- [Ext] swap #
- [Ext] skim #
- [Ext] sync #
- [Ext] initialize #

+ [Int] IUniswapV2Router01
- [Ext] factory
- [Ext] WETH
- [Ext] addLiquidity #
- [Ext] addLiquidityETH ($)
- [Ext] removeLiquidity #
- [Ext] removeLiquidityETH #
- [Ext] removeLiquidityWithPermit #

- [Ext] removeLiquidityETHWithPermit #
- [Ext] swapExactTokensForTokens #
- [Ext] swapTokensForExactTokens #
- [Ext] swapExactETHForTokens ($)
- [Ext] swapTokensForExactETH #
- [Ext] swapExactTokensForETH #
- [Ext] swapETHForExactTokens ($)
- [Ext] quote
- [Ext] getAmountOut
- [Ext] getAmountIn
- [Ext] getAmountsOut
- [Ext] getAmountsIn

+ [Int] IUniswapV2Router02 (IUniswapV2Router01)
- [Ext] removeLiquidityETHSupportingFeeOnTransferTokens #
- [Ext] removeLiquidityETHWithPermitSupportingFeeOnTransferTokens #
- [Ext] swapExactTokensForTokensSupportingFeeOnTransferTokens #
- [Ext] swapExactETHForTokensSupportingFeeOnTransferTokens ($)
- [Ext] swapExactTokensForETHSupportingFeeOnTransferTokens #

+ MoonTravels (Context, IERC20, Ownable)
- [Pub] <Constructor> #
- [Pub] name
- [Pub] symbol
- [Pub] decimals
- [Pub] totalSupply
- [Pub] balanceOf
- [Pub] transfer #
- [Pub] allowance
- [Pub] approve #
- [Pub] transferFrom #
- [Pub] increaseAllowance #
- [Pub] decreaseAllowance #
- [Pub] isExcludedFromReward
- [Pub] totalFees
- [Pub] minimumTokensBeforeSwapAmount
- [Pub] buyBackUpperLimitAmount
- [Pub] deliver #
- [Pub] reflectionFromToken
- [Pub] tokenFromReflection
- [Pub] excludeFromReward #
- modifiers: onlyOwner
- [Ext] includeInReward #
- modifiers: onlyOwner
- [Prv] _approve #
- [Prv] _transfer #
- [Prv] swapTokens #
- modifiers: lockTheSwap
- [Prv] buyBackTokens #
- modifiers: lockTheSwap
- [Prv] swapTokensForEth #

- [Prv] swapETHForTokens #
- [Prv] addLiquidity #
- [Prv] _tokenTransfer #
- [Prv] _transferStandard #
- [Prv] _transferToExcluded #
- [Prv] _transferFromExcluded #
- [Prv] _transferBothExcluded #
- [Prv] _reflectFee #
- [Prv] _getValues
- [Prv] _getTValues
- [Prv] _getRValues
- [Prv] _getRate
- [Prv] _getCurrentSupply
- [Prv] _takeLiquidity #
- [Prv] calculateTaxFee
- [Prv] calculateLiquidityFee
- [Prv] removeAllFee #
- [Prv] restoreAllFee #
- [Pub] isExcludedFromFee
- [Pub] excludeFromFee #
  - modifiers: onlyOwner
- [Pub] includeInFee #
  - modifiers: onlyOwner
- [Ext] setTaxFee #
- modifiers: onlyOwner
- [Ext] setBuybackFee #
  - modifiers: onlyOwner
- [Ext] setMaxTxAmount #
- modifiers: onlyOwner
- [Ext] setMarketingFee #
- modifiers: onlyOwner
- [Ext] setNumTokensSellToAddToLiquidity #
- modifiers: onlyOwner
- [Ext] setBuybackUpperLimit #
- modifiers: onlyOwner
- [Ext] setMarketingAddress #
- modifiers: onlyOwner
- [Pub] setSwapAndLiquifyEnabled #
- modifiers: onlyOwner
- [Pub] setBuyBackEnabled #
- modifiers: onlyOwner
- [Ext] presale #
- modifiers: onlyOwner
- [Prv] transferToAddressETH #
- [Ext] <Fallback> ($)

($) = payable function
= non-constant function

Automated Audit

Remix Compiler Warnings

It throws warnings by Solidity's compiler. If it encounters any errors the contract cannot be compiled and deployed. No issues found.

Number of issues per severity

Critical	High	Medium	Low	Note
0	0	0	1	0

Issues Checking Status

№	Issue description.	Checking
1	Compiler warnings.	Passed
2	Race conditions and Reentrancy. Cross-function race	Passed
3	Possible delays in data delivery.	Passed
4	Oracle calls.	Passed
5	Front running.	Passed
6	Timestamp dependence.	Passed
7	Integer Overflow and Underflow.	Passed
8	DoS with Revert.	Passed
9	DoS with block gas limit.	Low issues
10	Methods execution permissions.	Passed
11	Economy model.	Passed
12	The impact of the exchange rate on the logic.	Passed
13	Private user data leaks.	Passed
14	Malicious Event log.	Passed
15	Scoping and Declarations.	Passed
16	Uninitialized storage pointers.	Passed
17	Arithmetic accuracy.	Passed

18	Design Logic.	Passed
19	Cross-function race conditions.	Passed
20	Safe Zeppelin module.	Passed
21	Fallback function security.	Passed

Critical Severity Issues

No critical severity issues found.

High Severity Issues

No high severity issues found.

Medium Severity Issues

No medium severity issues found.

Low Severity Issues

1. Out of gas

Issue:

- The function `includeInReward()` uses the loop to find and remove addresses from the `_excluded` list. Function will be aborted with `OUT_OF_GAS` exception if there will be a long excluded addresses list.
- The function `_getCurrentSupply` also uses the loop for evaluating total supply. It also could be aborted with `OUT_OF_GAS` exception ifthere will be a long excluded addresses list.

Recommendation:

Check that the excluded array length is not too big.

Notes:

- addLiquidity function is not used.

Owner privileges (In the period when the ownership is notrenounced)

- Owner can change the tax, buyback and marketing fee.
- Owner can change the maximum transaction amount.
- Owner can exclude from the fee.
- Owner can change minimum number of tokens to add to liquidity.
- Owner can change buyBackUpperLimit.
- Owner can change marketing address.
- Owner can enable and disable buyBack.
- Owner can enable before and after presale modes.
- Owner can lock and unlock. By the way, using these functions theowner could retake privileges even after the ownership was renounced.

Disclaimer

By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and ReHashCrypto and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (ReHashCrypto) owe no duty of care towards you or any other person, nor does ReHashCrypto make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and ReHashCrypto hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, ReHashCrypto hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against ReHashCrypto, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report.

The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

Summary

Smart contracts do not contain any high severity issues.

Note:
Please check the disclaimer above and note, the audit makes no statements or warranties on business model, investment attractiveness or code sustainability. The report is provided for the only contract mentioned in the report.