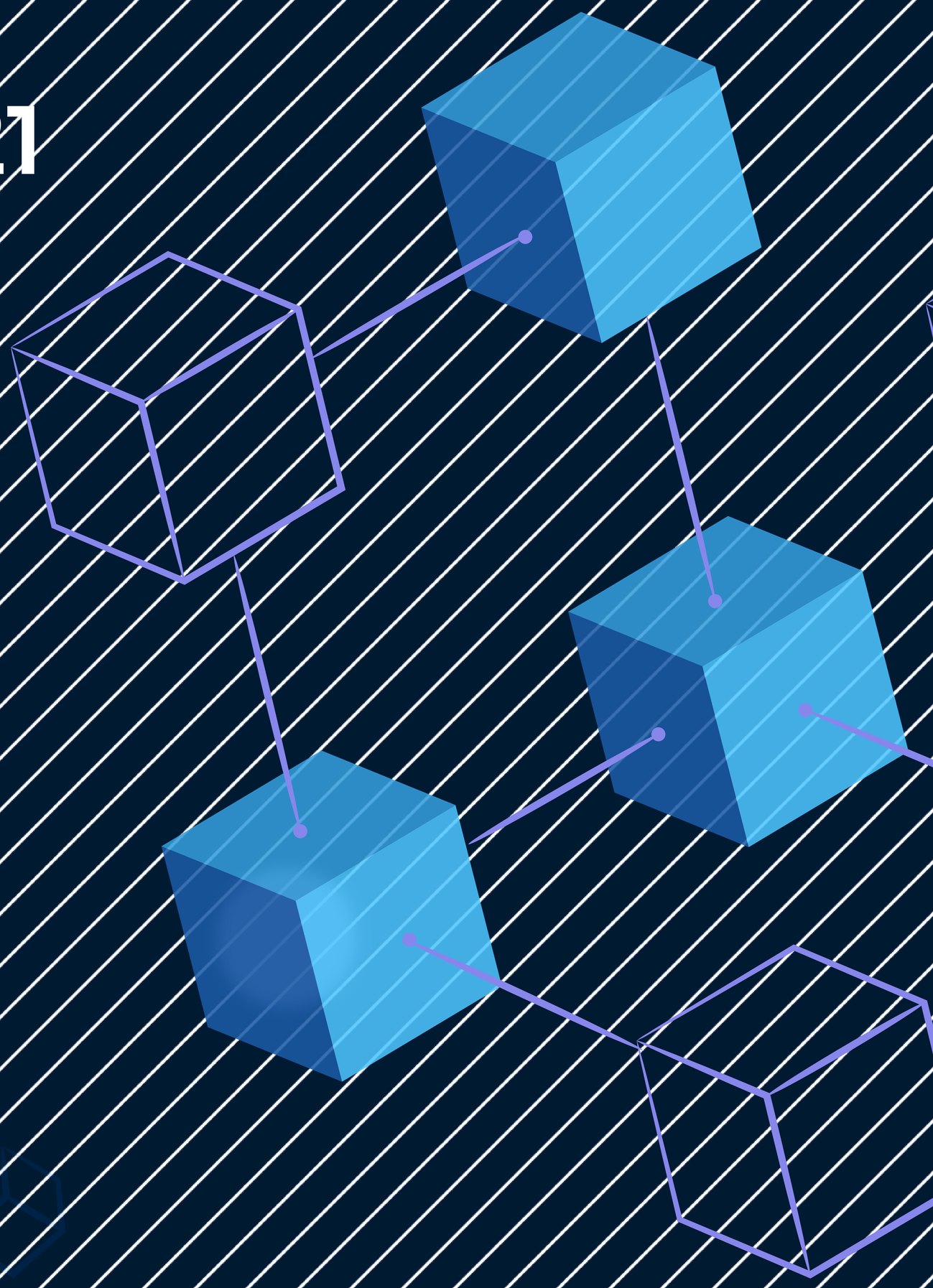
QuillAudits

Audit Report
December, 2021



For

Muslim Coins

Contents

Contents

Scope of the Audit

The scope of this audit was to analyze and document the MuslimCoins Token smart contract codebase for quality, security, and correctness.

Checked Vulnerabilities

We have scanned the smart contract for commonly known and more specific vulnerabilities. Here are some of the commonly known vulnerabilities that we considered:

- Re-entrancy
- Timestamp Dependence
- Gas Limit and Loops
- DoS with Block Gas Limit
- Transaction-Ordering Dependence
- Use of tx.origin
- Exception disorder
- Gasless send
- Balance equality
- Byte array
- Transfer forwards all gas
- BEP20 API violation
- Malicious libraries
- Compiler version not fixed
- Redundant fallback function
- Send instead of transfer
- Style guide violation
- Unchecked external call
- Unchecked math
- Unsafe type inference
- Implicit visibility level

Techniques and Methods

Throughout the audit of smart contract, care was taken to ensure:

- The overall quality of code.
- Use of best practices.
- Code documentation and comments match logic and expected behaviour.
- Token distribution and calculations are as per the intended behaviour mentioned in the whitepaper.
- Implementation of BEP-20 token standards.
- Efficient use of gas.
- Code is safe from re-entrancy and other vulnerabilities.

The following techniques, methods and tools were used to review all the smart contracts.

Structural Analysis
In this step, we have analysed the design patterns and structure of smart contracts. A thorough check was done to ensure the smart contract is structured in a way that will not result in future problems.

Static Analysis
Static analysis of smart contracts was done to identify contract vulnerabilities. In this step, a series of automated tools are used to test the security of smart contracts.

Code Review / Manual Analysis
Manual analysis or review of code was done to identify new vulnerabilities or verify the vulnerabilities found during the static analysis. Contracts were completely manually analysed, their logic was checked and compared with the one described in the whitepaper. Besides, the results of the automated analysis were manually verified.

Gas Consumption
In this step, we have checked the behaviour of smart contracts in production. Checks were done to know how much gas gets consumed and the possibilities of optimization of code to reduce gas consumption.

Tools and Platforms used for Audit
Remix IDE, Truffle, Truffle Team, Solhint, Mythril, Slither, Solidity statistic analysis.

Issue Categories

Every issue in this report has been assigned to a severity level. There are four levels of severity, and each of them has been explained below.

Risk-level	Description
High	A high severity issue or vulnerability means that your smart contract can be exploited. Issues on this level are critical to the smart contract's performance or functionality, and we recommend these issues be fixed before moving to a live environment.
Medium	The issues marked as medium severity usually arise because of errors and deficiencies in the smart contract code. Issues on this level could potentially bring problems, and they should still be fixed.
Low	Low-level severity issues can cause minor impact and or are just warnings that can remain unfixed for now. It would be better to fix these issues at some point in the future.
Informational	These are severity issues that indicate an improvement request, a general question, a cosmetic or documentation error, or a request for information. There is low-to-no impact.

Number of issues per severity

Type	High	Medium	Low	Informational
Open	0	0	0	0
Acknowledged	0	0	1	12
Closed	0	0	0	0

Introduction

During the period of **November 24, 2021 to December 1, 2021** - QuillAudits Team performed a security audit for **MuslimCoins** smart contracts.

The code for the audit was taken from following the official link:
Codebase: 0x8B93585978B81E4FC0aE063fe526dBfBE9B8D42D

Ver No	Date	Contract address	Network
1	24 November	0x8B93585978B81E4FC0aE 063fe526dBfBE9B8D42D	Binance

Issues Found

A. Contract – MuslimCoins

High severity issues

No issues were found.

Medium severity issues

No issues were found.

Low severity issues

1. **Missing zero address validation**

   Description
   Missing zero address check for to address in the following methods:
   - _transferBothExcluded()
   - _approve()
   - _transfer()
   - _tokenTransfer()

   Remediation
   Add a ‘require’ to check to address != address(0)

   Status: Acknowledged

Informational issues

2. **Missing comments and description**

   Description
   Comments and Description of the methods and the variables are missing, it's hard to read and understand the purpose of the variables and the methods in context of the whole picture

   Remediation
   Consider adding NatSpec format comments for the comments and state variables

   Status: Acknowledged

3. Same execution for two different cases can be merged

```
if (_isExcluded[sender] && !_isExcluded[recipient]) {
        _transferFromExcluded(sender, recipient, amount);
    } else if (!_isExcluded[sender] && _isExcluded[recipient]) {
        _transferToExcluded(sender, recipient, amount);
    } else if (!_isExcluded[sender] && !_isExcluded[recipient]) {
        _transferStandard(sender, recipient, amount);
    } else if (_isExcluded[sender] && _isExcluded[recipient]) {
        _transferBothExcluded(sender, recipient, amount);
    } else {
        _transferStandard(sender, recipient, amount);
    }
```

Remediation

```
if (_isExcluded[sender] && !_isExcluded[recipient]) {
        _transferFromExcluded(sender, recipient, amount);
    } else if (!_isExcluded[sender] && _isExcluded[recipient]) {
        _transferToExcluded(sender, recipient, amount);
    } else if (_isExcluded[sender] && _isExcluded[recipient]) {
        _transferBothExcluded(sender, recipient, amount);
    } else {
        _transferStandard(sender, recipient, amount);
    }
```

Status: Acknowledged

4. Less meaningful variable

Description
MAX variable names do not provide a clear picture of their purpose

Remediation
MAX should be renamed to **MAX_INT_256**

Status: Acknowledged

5. Variable defined but never used

Description
‘contractTokenBalance’ defined but never used

Remediation
Remove unused variables

Status: Acknowledged

6. Public methods only being used externally

Description
'public' functions that are not inherited by the contract should be declared 'external' to save gas.

Remediation
Make these methods external -
- name()
- symbol()
- decimals()
- totalSupply()
- balanceOf()
- transfer()
- allowance()
- approve()
- transferFrom()
- increaseAllowance()
- decreaseAllowance()
- isExcludedFromReward()
- totalFees()
- deliver()
- reflectionFromToken()
- tokenFromReflection()
- excludeFromReward()
- excludeFromFee()
- includeInFee()
- isExcludedFromFee()

Status: Acknowledged

7. Constant calculations in the contract

Precalculated initialization will save 2847 units of gas in deployment

```
uint256 private _tTotal = 500**1 * 10**6 * 10**18;
uint256 private _rTotal = (MAX - (MAX % _tTotal));
```

Recommendation
Replace the initialization as -

```
uint256 private _tTotal =
500000000000000000000000000;
uint256 private _rTotal =
115792089237316195423570985008687907853269984665640
500000000000000000000000000;
```

10**2 should be a precalculated constant and then used in **calculateTaxFee**.

Status: Acknowledged

8. Use a constructor to set addresses

Description
The address is set in the contract should be provided dynamically via constructor on deployment for readability and correctness

Recommendation
Pass the address of “walletZakat” as a parameter of the constructor and set the value

Status: Acknowledged

9. Getters should be at the bottom of the contract

Status: Acknowledged

10. **Length calculation within the loop**

Description
Reading from the state and fetching its length is a costly operation and doing such within a loop should be avoided

Recommendation
_excluded.length should be calculated and stored in a variable before using in the for loop inside `**_getCurrentSupply**` and `**includeInReward**` method.

Status: Acknowledged

11. **Inconsistent error messages**

Description
The require checks in the contract have error messages containing the contract name in some places and the same has been skipped in most cases.

Recommendation
The error message should follow the same pattern and we recommend using the contract name along with the error message at all places
Eg.

```
require(_isExcluded[account], "MuslimCoins: Account is
already excluded");
```

Status: Acknowledged

12. **Naming Conventions**

Description
The contract follows a consistent naming convention of private variables with leading “_” and public variables without it.
But we have missed complying with the condition for certain variable names - "**calculateTaxFee**", "**removeAllFee**" and “**restoreAllFee**” which are private.

Recommendation
Add “_” in private method names

Status: Acknowledged

13. Presence of not implemented payable method

Description
The method `receive()` is not implemented but is present in the contract.

Recommendation
Remove the method from the contract or call the `revert()` method inside this `receive()` method.
Eg.

```
receive() external payable {
    revert("NOT_IMPLEMENTED");
}
```

Binance Testnet Test Contract

MuslimCoins: 0x23Ab9286aF3713783d994d675b1421824396Aeef

Functional Tests

- Deliver
- Exclude from fee
- Exclude from reward
- Include in fee
- Include in reward
- Set zakat and fee

Automated Tests

Slither

```
▶slither .
'npx hardhat compile --force' running
Compiling 1 file with 0.6.12
Compilation finished successfully
 ·-----------------|-------------·
 |  Contract Name  ·  Size (Kb)  |
 ··················|··············
 |  Address        ·       0.08  |
 ··················|··············
 |  MuslimCoins    ·       8.81  |
 ··················|··············
 |  SafeMath       ·       0.08  |
 ·-----------------|-------------·
Creating Typechain artifacts in directory types for target ethers-v5
Successfully generated Typechain artifacts!

contracts/MuslimCoins.sol:901:9: Warning: Unused local variable.
        uint256 contractTokenBalance = balanceOf(address(this));
        ^--------------------------^



Contract locking ether found:
        Contract MuslimCoins (contracts/MuslimCoins.sol#533-1019) has payable functions:
         - MuslimCoins.receive() (contracts/MuslimCoins.sol#780)
        But does not have a function to withdraw the ether
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#contracts-that-lock-ether

MuslimCoins.allowance(address,address).owner (contracts/MuslimCoins.sol#617) shadows:
        - Ownable.owner() (contracts/MuslimCoins.sol#469-471) (function)
MuslimCoins._approve(address,address,uint256).owner (contracts/MuslimCoins.sol#877) shadows:
        - Ownable.owner() (contracts/MuslimCoins.sol#469-471) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing

MuslimCoins.setZakatAndFee(address,uint256,uint256) (contracts/MuslimCoins.sol#598-606) should emit an event for:
        - _taxZakat = taxZakat (contracts/MuslimCoins.sol#604)
        - taxFee = _taxFee (contracts/MuslimCoins.sol#605)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic

MuslimCoins.setZakatAndFee(address,uint256,uint256).account (contracts/MuslimCoins.sol#599) lacks a zero-check on :
                - walletZakat = account (contracts/MuslimCoins.sol#603)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

Ownable.unlock() (contracts/MuslimCoins.sol#522-530) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(now > _lockTime,Contract is locked until 7 days) (contracts/MuslimCoins.sol#527)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp

Address.isContract(address) (contracts/MuslimCoins.sol#278-289) uses assembly
        - INLINE ASM (contracts/MuslimCoins.sol#285-287)
Address._functionCallWithValue(address,bytes,uint256,string) (contracts/MuslimCoins.sol#404-432) uses assembly
        - INLINE ASM (contracts/MuslimCoins.sol#424-427)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Address._functionCallWithValue(address,bytes,uint256,string) (contracts/MuslimCoins.sol#404-432) is never used and should be removed
```

```
Address._functionCallWithValue(address,bytes,uint256,string) (contracts/MuslimCoins.sol#404-432) is never used and should be removed
Address.functionCall(address,bytes) (contracts/MuslimCoins.sol#339-344) is never used and should be removed
Address.functionCall(address,bytes,string) (contracts/MuslimCoins.sol#352-358) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/MuslimCoins.sol#371-383) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (contracts/MuslimCoins.sol#391-402) is never used and should be removed
Address.isContract(address) (contracts/MuslimCoins.sol#278-289) is never used and should be removed
Address.sendValue(address,uint256) (contracts/MuslimCoins.sol#307-319) is never used and should be removed
Context._msgData() (contracts/MuslimCoins.sol#251-254) is never used and should be removed
SafeMath.mod(uint256,uint256) (contracts/MuslimCoins.sol#138-140) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (contracts/MuslimCoins.sol#154-161) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

MuslimCoins._rTotal (contracts/MuslimCoins.sol#548) is set pre-construction with a non-constant function or state variable:
        - (MAX - (MAX % _tTotal))
MuslimCoins._previousTaxFee (contracts/MuslimCoins.sol#557) is set pre-construction with a non-constant function or state variable:
        - taxFee
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#function-initializing-state

Low level call in Address.sendValue(address,uint256) (contracts/MuslimCoins.sol#307-319):
        - (success) = recipient.call{value: amount}() (contracts/MuslimCoins.sol#314)
Low level call in Address._functionCallWithValue(address,bytes,uint256,string) (contracts/MuslimCoins.sol#404-432):
        - (success,returndata) = target.call{value: weiValue}(data) (contracts/MuslimCoins.sol#413-415)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter MuslimCoins.setZakatAndFee(address,uint256,uint256)._taxFee (contracts/MuslimCoins.sol#601) is not in mixedCase
Parameter MuslimCoins.calculateTaxFee(uint256)._amount (contracts/MuslimCoins.sol#856) is not in mixedCase
Variable MuslimCoins._taxZakat (contracts/MuslimCoins.sol#555) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Redundant expression "this (contracts/MuslimCoins.sol#252)" inContext (contracts/MuslimCoins.sol#246-255)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

Variable MuslimCoins.reflectionFromToken(uint256,bool).rTransferAmount (contracts/MuslimCoins.sol#711) is too similar to MuslimCoins._transferToExcluded(address,address,uint256).tTransferAmount (contracts/M
uslimCoins.sol#991)
Variable MuslimCoins._transferToExcluded(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#989) is too similar to MuslimCoins._transferToExcluded(address,address,uint256).tTransferAmount (
contracts/MuslimCoins.sol#991)
```

```
Variable MuslimCoins._getValues(uint256).rTransferAmount (contracts/MuslimCoins.sol#799) is too similar to MuslimCoins._transferBothExcluded(address,address,uint256).tTransferAmount (contracts/MuslimCoins.s
ol#761)
Variable MuslimCoins._getValues(uint256).rTransferAmount (contracts/MuslimCoins.sol#799) is too similar to MuslimCoins._transferFromExcluded(address,address,uint256).tTransferAmount (contracts/MuslimCoins.s
ol#1010)
Variable MuslimCoins._transferStandard(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#960) is too similar to MuslimCoins._transferStandard(address,address,uint256).tTransferAmount (cont
racts/MuslimCoins.sol#962)
Variable MuslimCoins._transferBothExcluded(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#759) is too similar to MuslimCoins._transferStandard(address,address,uint256).tTransferAmount (
contracts/MuslimCoins.sol#962)
Variable MuslimCoins.reflectionFromToken(uint256,bool).rTransferAmount (contracts/MuslimCoins.sol#711) is too similar to MuslimCoins._transferStandard(address,address,uint256).tTransferAmount (contracts/Mus
limCoins.sol#962)
Variable MuslimCoins._transferToExcluded(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#989) is too similar to MuslimCoins._transferStandard(address,address,uint256).tTransferAmount (co
ntracts/MuslimCoins.sol#962)
Variable MuslimCoins._getValues(uint256).rTransferAmount (contracts/MuslimCoins.sol#799) is too similar to MuslimCoins._getTValues(uint256).tTransferAmount (contracts/MuslimCoins.sol#813)
Variable MuslimCoins.reflectionFromToken(uint256,bool).rTransferAmount (contracts/MuslimCoins.sol#711) is too similar to MuslimCoins._getValues(uint256).tTransferAmount (contracts/MuslimCoins.sol#798)
Variable MuslimCoins._transferToExcluded(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#989) is too similar to MuslimCoins._getValues(uint256).tTransferAmount (contracts/MuslimCoins.sol
#798)
Variable MuslimCoins._transferStandard(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#960) is too similar to MuslimCoins._transferFromExcluded(address,address,uint256).tTransferAmount (
contracts/MuslimCoins.sol#1010)
```

```
Variable MuslimCoins._transferStandard(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#960) is too similar to MuslimCoins._transferFromExcluded(address,address,uint256).tTransferAmount (
contracts/MuslimCoins.sol#1010)
Variable MuslimCoins._transferBothExcluded(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#759) is too similar to MuslimCoins._transferBothExcluded(address,address,uint256).tTransferAmou
nt (contracts/MuslimCoins.sol#761)
Variable MuslimCoins._transferBothExcluded(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#759) is too similar to MuslimCoins._transferFromExcluded(address,address,uint256).tTransferAmou
nt (contracts/MuslimCoins.sol#1010)
Variable MuslimCoins.reflectionFromToken(uint256,bool).rTransferAmount (contracts/MuslimCoins.sol#711) is too similar to MuslimCoins._transferBothExcluded(address,address,uint256).tTransferAmount (contracts
/MuslimCoins.sol#761)
Variable MuslimCoins._transferToExcluded(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#989) is too similar to MuslimCoins._transferBothExcluded(address,address,uint256).tTransferAmount
 (contracts/MuslimCoins.sol#761)
Variable MuslimCoins._transferToExcluded(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#989) is too similar to MuslimCoins._transferFromExcluded(address,address,uint256).tTransferAmount
 (contracts/MuslimCoins.sol#1010)
Variable MuslimCoins.reflectionFromToken(uint256,bool).rTransferAmount (contracts/MuslimCoins.sol#711) is too similar to MuslimCoins._transferFromExcluded(address,address,uint256).tTransferAmount (contracts
/MuslimCoins.sol#1010)
Variable MuslimCoins._transferFromExcluded(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#1008) is too similar to MuslimCoins._transferFromExcluded(address,address,uint256).tTransferAmo
unt (contracts/MuslimCoins.sol#1010)
Variable MuslimCoins._transferStandard(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#960) is too similar to MuslimCoins._getTValues(uint256).tTransferAmount (contracts/MuslimCoins.sol#
813)
Variable MuslimCoins._getRValues(uint256,uint256,uint256).rTransferAmount (contracts/MuslimCoins.sol#832) is too similar to MuslimCoins._getTValues(uint256).tTransferAmount (contracts/MuslimCoins.sol#813)
Variable MuslimCoins._transferBothExcluded(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#759) is too similar to MuslimCoins._getTValues(uint256).tTransferAmount (contracts/MuslimCoins.
sol#813)
Variable MuslimCoins._transferFromExcluded(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#1008) is too similar to MuslimCoins._getTValues(uint256).tTransferAmount (contracts/MuslimCoins
.sol#813)
Variable MuslimCoins._getValues(uint256).rTransferAmount (contracts/MuslimCoins.sol#799) is too similar to MuslimCoins._transferStandard(address,address,uint256).tTransferAmount (contracts/MuslimCoins.sol#9
62)
Variable MuslimCoins.reflectionFromToken(uint256,bool).rTransferAmount (contracts/MuslimCoins.sol#711) is too similar to MuslimCoins._getTValues(uint256).tTransferAmount (contracts/MuslimCoins.sol#813)
Variable MuslimCoins._transferToExcluded(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#989) is too similar to MuslimCoins._getTValues(uint256).tTransferAmount (contracts/MuslimCoins.so
l#813)
Variable MuslimCoins._transferBothExcluded(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#759) is too similar to MuslimCoins._getValues(uint256).tTransferAmount (contracts/MuslimCoins.s
ol#798)
Variable MuslimCoins._transferFromExcluded(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#1008) is too similar to MuslimCoins._getValues(uint256).tTransferAmount (contracts/MuslimCoins.
sol#798)
Variable MuslimCoins._getRValues(uint256,uint256,uint256).rTransferAmount (contracts/MuslimCoins.sol#832) is too similar to MuslimCoins._getValues(uint256).tTransferAmount (contracts/MuslimCoins.sol#798)
Variable MuslimCoins._getValues(uint256).rTransferAmount (contracts/MuslimCoins.sol#799) is too similar to MuslimCoins._transferToExcluded(address,address,uint256).tTransferAmount (contracts/MuslimCoins.sol
#991)
Variable MuslimCoins._getValues(uint256).rTransferAmount (contracts/MuslimCoins.sol#799) is too similar to MuslimCoins._getValues(uint256).tTransferAmount (contracts/MuslimCoins.sol#798)
Variable MuslimCoins._getRValues(uint256,uint256,uint256).rTransferAmount (contracts/MuslimCoins.sol#832) is too similar to MuslimCoins._transferFromExcluded(address,address,uint256).tTransferAmount (contra
cts/MuslimCoins.sol#1010)
Variable MuslimCoins._transferStandard(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#960) is too similar to MuslimCoins._transferToExcluded(address,address,uint256).tTransferAmount (co
ntracts/MuslimCoins.sol#991)
Variable MuslimCoins._transferStandard(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#960) is too similar to MuslimCoins._transferBothExcluded(address,address,uint256).tTransferAmount (
contracts/MuslimCoins.sol#761)
Variable MuslimCoins._transferBothExcluded(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#759) is too similar to MuslimCoins._transferToExcluded(address,address,uint256).tTransferAmount
 (contracts/MuslimCoins.sol#991)
Variable MuslimCoins._transferFromExcluded(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#1008) is too similar to MuslimCoins._transferToExcluded(address,address,uint256).tTransferAmoun
t (contracts/MuslimCoins.sol#991)
Variable MuslimCoins._transferStandard(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#960) is too similar to MuslimCoins._getValues(uint256).tTransferAmount (contracts/MuslimCoins.sol#7
98)
Variable MuslimCoins._transferFromExcluded(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#1008) is too similar to MuslimCoins._transferStandard(address,address,uint256).tTransferAmount
(contracts/MuslimCoins.sol#962)
Variable MuslimCoins._transferFromExcluded(address,address,uint256).rTransferAmount (contracts/MuslimCoins.sol#1008) is too similar to MuslimCoins._transferBothExcluded(address,address,uint256).tTransferAmo
unt (contracts/MuslimCoins.sol#761)
Variable MuslimCoins._getRValues(uint256,uint256,uint256).rTransferAmount (contracts/MuslimCoins.sol#832) is too similar to MuslimCoins._transferToExcluded(address,address,uint256).tTransferAmount (contract
s/MuslimCoins.sol#991)
Variable MuslimCoins._getRValues(uint256,uint256,uint256).rTransferAmount (contracts/MuslimCoins.sol#832) is too similar to MuslimCoins._transferStandard(address,address,uint256).tTransferAmount (contracts/
Variable MuslimCoins._getRValues(uint256,uint256,uint256).rTransferAmount (contracts/MuslimCoins.sol#832) is too similar to MuslimCoins._transferStandard(address,address,uint256).tTransferAmount (contracts/
MuslimCoins.sol#962)
Variable MuslimCoins._getRValues(uint256,uint256,uint256).rTransferAmount (contracts/MuslimCoins.sol#832) is too similar to MuslimCoins._transferBothExcluded(address,address,uint256).tTransferAmount (contra
cts/MuslimCoins.sol#761)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar

MuslimCoins._decimals (contracts/MuslimCoins.sol#554) should be constant
MuslimCoins._tTotal (contracts/MuslimCoins.sol#547) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (contracts/MuslimCoins.sol#491-494)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (contracts/MuslimCoins.sol#500-507)
geUnlockTime() should be declared external:
        - Ownable.geUnlockTime() (contracts/MuslimCoins.sol#509-511)
lock(uint256) should be declared external:
        - Ownable.lock(uint256) (contracts/MuslimCoins.sol#514-519)
unlock() should be declared external:
        - Ownable.unlock() (contracts/MuslimCoins.sol#522-530)
name() should be declared external:
        - MuslimCoins.name() (contracts/MuslimCoins.sol#577-579)
symbol() should be declared external:
        - MuslimCoins.symbol() (contracts/MuslimCoins.sol#581-583)
decimals() should be declared external:
        - MuslimCoins.decimals() (contracts/MuslimCoins.sol#585-587)
totalSupply() should be declared external:
        - MuslimCoins.totalSupply() (contracts/MuslimCoins.sol#589-591)
transfer(address,uint256) should be declared external:
        - MuslimCoins.transfer(address,uint256) (contracts/MuslimCoins.sol#608-615)
allowance(address,address) should be declared external:
        - MuslimCoins.allowance(address,address) (contracts/MuslimCoins.sol#617-624)
approve(address,uint256) should be declared external:
        - MuslimCoins.approve(address,uint256) (contracts/MuslimCoins.sol#626-633)
transferFrom(address,address,uint256) should be declared external:
        - MuslimCoins.transferFrom(address,address,uint256) (contracts/MuslimCoins.sol#635-650)
increaseAllowance(address,uint256) should be declared external:
        - MuslimCoins.increaseAllowance(address,uint256) (contracts/MuslimCoins.sol#652-663)
decreaseAllowance(address,uint256) should be declared external:
        - MuslimCoins.decreaseAllowance(address,uint256) (contracts/MuslimCoins.sol#665-679)
isExcludedFromReward(address) should be declared external:
        - MuslimCoins.isExcludedFromReward(address) (contracts/MuslimCoins.sol#681-683)
totalFees() should be declared external:
        - MuslimCoins.totalFees() (contracts/MuslimCoins.sol#685-687)
deliver(uint256) should be declared external:
        - MuslimCoins.deliver(uint256) (contracts/MuslimCoins.sol#689-699)
reflectionFromToken(uint256,bool) should be declared external:
        - MuslimCoins.reflectionFromToken(uint256,bool) (contracts/MuslimCoins.sol#701-714)
excludeFromReward(address) should be declared external:
        - MuslimCoins.excludeFromReward(address) (contracts/MuslimCoins.sol#729-737)
excludeFromFee(address) should be declared external:
        - MuslimCoins.excludeFromFee(address) (contracts/MuslimCoins.sol#772-774)
includeInFee(address) should be declared external:
        - MuslimCoins.includeInFee(address) (contracts/MuslimCoins.sol#776-778)
isExcludedFromFee(address) should be declared external:
```

Results

No major issues were found. Some false positive errors were reported by the tool. All the other issues have been categorized above according to their level of severity.

Closing Summary

Overall, smart contracts are very well written, documented, and adhere to guidelines. Several issues of Low severity and information issues have been reported,which have been acknowledged by the MuslimCoins Team.
We recommend working on suggestions that are reported in order to improve the code quality of smart contracts.

Disclaimer

Quillhash audit is not a security warranty, investment advice, or endorsement of the **MuslimCoins** platform. This audit does not provide a security or correctness guarantee of the audited smart contracts. The statements made in this document should not be interpreted as investment or legal advice, nor should its authors be held accountable for decisions made based on them. Securing smart contracts is a multistep process. One audit cannot be considered enough. We recommend that the **MuslimCoins** Team put in place a bug bounty program to encourage further analysis of the smart contract by other third parties.

Audit Report
December, 2021

For

Muslim Coins

Canada, India, Singapore, United Kingdom

audits.quillhash.com

audits@quillhash.com

QuillAudits