SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: Milky Way

Date: August 21st, 2021

This document may contain confidential information about IT systems and the intellectual property of the Customer and information about potential vulnerabilities and mBNBods of their exploitation.

The report containing confidential information can be used internally by the Customer, or it can be disclosed publicly after all vulnerabilities fixed - upon a decision of the Customer.

Document

Name	Smart Contract Code Review and Security Analysis Report for Milky Way (18 pages)
Approved by	Andrew Matiukhin | CTO Hacken OU
Type	Token
Platform	Binance / Solidity
MBNBods	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
Address	0X481DE76D5AB31E28A33B0EA1C1063ADCB5B1769A
Timeline	20TH Aug 2021 – 21ST Aug 2021
Changelog	21ST Aug 2021- Initial Audit

Table of contents

Introduction

Hacken OÜ (Consultant) was contracted by Milky Way (Customer) to conduct a Smart Contract Code Review and Security Analysis. This report presents the findings of the security assessment of the Customer's smart contract and its code review conducted between August 20th, 2021 – August 21st, 2021.

Scope

The scope of the project is smart contract in the mainnet:
Address: 0x6c56d9D857998E1a098e98569f257D9D85999A2C

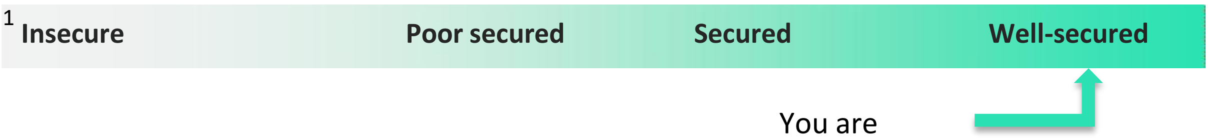
We have scanned this smart contract for commonly known and more specific vulnerabilities. Here are some of the commonly known vulnerabilities that are considered:

Category	Check Item
Code review	▪ Reentrancy ▪ Ownership Takeover ▪ Timestamp Dependence ▪ Gas Limit and Loops ▪ DoS with (Unexpected) Throw ▪ DoS with Block Gas Limit ▪ Transaction-Ordering Dependence ▪ Style guide violation ▪ Costly Loop ▪ BEP20 API violation ▪ Unchecked external call ▪ Unchecked math ▪ Unsafe type inference ▪ Implicit visibility level ▪ Deployment Consistency ▪ Repository Consistency ▪ Data Consistency

Functional review	■ Business Logics Review ■ Functionality Checks ■ Access Control & Authorization ■ Escrow manipulation ■ Token Supply manipulation ■ Assets integrity ■ User Balances manipulation ■ Data Consistency manipulation ■ Kill-Switch Mechanism ■ Operation Trails & Event Generation

Executive Summary

According to the assessment, the Customer's smart contract has not critical vulnerabilities and can be considered secure.



Our team performed an analysis of code functionality, manual audit, and automated checks with Mythril and Slither. All issues found during automated analysis were manually reviewed, and important vulnerabilities are presented in the Audit overview section. A general overview is presented in AS-IS section, and all found issues can be found in the Audit overview section.

Security engineers found no severity issues during the audit.

[1] Look for details and justification in conclusion section

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to assets loss or data manipulations.
High	High-level vulnerabilities are difficult to exploit; however, they also have a significant impact on smart contract execution, e.g., public access to crucial functions
Medium	Medium-level vulnerabilities are essential to fix; however, they can't lead to assets loss or data manipulations.
Low	Low-level vulnerabilities are mostly related to outdated, unused, etc. code snippets that can't have a significant impact on execution
Lowest / Code Style / Best Practice	Lowest-level vulnerabilities, code style violations, and info statements can't affect smart contract execution and can be ignored.

AS-IS overview

Description

Milky Way is an BEP20 token contract based on the *OpenZeppelin* source code. This contract cannot accept BNB because the receive function is always reverted. The *constructor* function mints 26,000 Milky Way tokens. After the contract is deployed, new tokens cannot be minted.

Imports

Milky Way contract has 3 imports:

- *Ownable* — from *OpenZeppelin*
- *IBEP20* — from *OpenZeppelin*. Comments have been stripped and the interface has been renamed *ERC*.
- *SafeMath* — from *OpenZeppelin*. Unused functions have been removed.

Inheritance

Milky Way contract inherits *BEP20* and *Ownable*.

Usings

Milky Way contract use *SafeMath* for *uint256*.

Fields

Milky Way contract has 6 fields:

- *string _name* — a name;
- *string _symbol* — a symbol;
- *uint256 _totalSupply* — the total supply;
- *uint256 _decimal* — a decimal;
- *mapping(address => uint256) _balances* — a mapping of balances;
- *mapping(address => mapping (address => uint256)) _allowances* — a mapping of allowances;

Functions

Milky Way contract has 13 functions:

- ***constructor***

  Description

  Initializes the contract. Mints 26,000 Milky Way tokens.

  Visibility

  public

  Input parameters

  None

  Constraints

  None

  Events emit

  None

  Output

  None

- ***name***

  Description

  Used to get the name.

  Visibility

  public view

  Input parameters

  None

  Constraints

  None

Events emit

None

Output

Returns the name.

- ***symbol***

Description

Used to get the symbol.

Visibility

public view

Input parameters

None

Constraints

None

Events emit

None

Output

Returns the symbol.

- ***decimals***

Description

Used to get decimals.

Visibility

public view

Input parameters

None

Constraints

None

Events emit

None

Output

Returns decimals.

- ***totalSupply***

Description

Used to get the total supply.

Visibility

external view

Input parameters

None

Constraints

None

Events emit

None

Output

Returns the total supply.

- ***balanceOf***

Description

Used to get the balance of the address.

Visibility

external view

Input parameters

- *address _tokenOwner* — an address;

Constraints

None

Events emit

None

Output

Returns the balance.

- ***transfer***

Description

Used to transfer tokens.

Visibility

external

Input parameters

- *address _to* — an address of recipient;
- *uint256 _tokens* — an amount of tokens;

Constraints

None

Events emit

None

Output

None

- ***_transfer***

Description

Used to transfer tokens.

Visibility

internal

Input parameters

- *address _sender* — an address of sender;
- *address _recipient* — an address of recipient;
- *uint256 _amount* — an amount of tokens;

Constraints

- The sender should not be a zero address.
- The recipient should not be a zero address.

Events emit

- *emit Transfer(_sender, _recipient, _amount);*

Output

None

- ***allowance***

Description

Used to get allowance.

Visibility

external view

Input parameters

- *address _tokenOwner* — an address of owner;
- *address _spender* — an address of spender;

Constraints

None

Events emit

None

Output

Returns allowance.

- ***approve***

Description

Used to approve transfer.

Visibility

external

Input parameters

- *address _spender* — an address of spender;
- *uint256 _tokens* — an amount of tokens;

Constraints

None

Events emit

None

Output

None

- ***_approve***

Description

Used to approve transfer.

Visibility

internal

Input parameters

- *address _owner* — an address of owner;
- *address _spender* — an address of spender;
- *uint256 _value* — an amount of tokens;

Constraints

- The owner should not be a zero address.
- The spender should not be a zero address.

Events emit

- *emit Approval(_owner, _spender, _value);*

Output

None

- ***transferFrom***

Description

Used to transfer tokens.

Visibility

external

Input parameters

  - *address _from* — an address of sender;
  - *address _to* — an address of recipient;
  - *uint256 _tokens* — an amount of tokens;

Constraints

None

Events emit

None

Output

None

- ***receive***

Description

Reverts BNB payments.

Visibility

external payable

Input parameters

None

Constraints

None

Events emit

None

Output

None

Audit overview

🟥🟥🟥🟥 Critical

No critical issues were found.

🟨🟨🟨 High

No high issues were found.

🟩🟩 Medium

No medium issues were found.

🟦 Low

No low severity issues were found.

🟦 Lowest / Code style / Best Practice

No lowest severity issues were found.

Conclusion

Smart contracts within the scope were manually reviewed and analyzed with static analysis tools. For the contract, high-level description of functionality was presented in As-is overview section of the report.

Security engineers found no severity issues during the audit.

Disclaimers

Hacken Disclaimer

The smart contracts given for audit have been analyzed in accordance with the best industry practices at the date of this report, in relation to cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report (Source Code); the Source Code compilation, deployment, and functionality (performing the intended functions).

The audit makes no statements or warranties on the security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bugfree status, or any other statements of the contract. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only - we recommend proceeding with several independent audits and a public bug bounty program to ensure the security of smart contracts.

Technical Disclaimer

Smart contracts are deployed and executed on blockchain platform. The platform, its programming language, and other software related to the smart contract can have its own vulnerabilities that can lead to hacks. Thus, the audit can't guarantee explicit security of the audited smart contracts.