verichains

SECURITY AUDIT OF

LEISUREMETA TOKEN

Public Report

Jun 14, 2023

Verichains Lab

info@verichains.io

https://www.verichains.io

Driving Technology > Forward

ABBREVIATIONS

Name	Description
Ethereum	An open source platform based on blockchain technology to create and distribute smart contracts and decentralized applications.
Ether (ETH)	A cryptocurrency whose blockchain is generated by the Ethereum platform. Ether is used for payment of transactions and computing services in the Ethereum network.
Smart contract	A computer protocol intended to digitally facilitate, verify or enforce the negotiation or performance of a contract.
Solidity	A contract-oriented, high-level language for implementing smart contracts for the Ethereum platform.
Solc	A compiler for Solidity.
ERC20	ERC20 (BEP20 in Binance Smart Chain or *x*RP20 in other chains) tokens are blockchain-based assets that have value and can be sent and received. The primary difference with the primary coin is that instead of running on their own blockchain, ERC20 tokens are issued on a network that supports smart contracts such as Ethereum or Binance Smart Chain.

EXECUTIVE SUMMARY

This Security Audit Report was prepared by Verichains Lab on Jun 14, 2023. We would like to thank the LeisureMeta for trusting Verichains Lab in auditing smart contracts. Delivering high-quality audits is always our top priority.

This audit focused on identifying security flaws in code and the design of the LeisureMeta Token. The scope of the audit is limited to the source code files provided to Verichains. Verichains Lab completed the assessment using manual, static, and dynamic analysis techniques.

During the audit process, the audit team had identified no vulnerable issues in the smart contracts code.

TABLE OF CONTENTS

1. MANAGEMENT SUMMARY

1.1. About LeisureMeta Token

The LeisureMeta Token (LM) is a ERC20 token used in the LeisureMetaverse ecosystem. The LM token is used as a form of payment for fees incurred from user activities within the LeisureMetaverse and all fees are burned, helping to prevent inflation and maintain a healthy ecosystem. The token also serves as a bridge between the virtual and real economies. Users are rewarded with LM tokens for participating in activities and collecting NFTs, with the rewards being based on the total user activity score.

1.2. Audit scope

This audit focused on identifying security flaws in code and the design of the LeisureMeta Token.

The audited contract is the LeisureMeta Token that deployed on Ethereum Mainnet at address `0xc064f4f215b6a1e4e7f39bd8530c4de0fc43ee9d`. The details of the deployed smart contract are listed in Table 1.

FIELD	VALUE
Contract Name	LeisureMeta
Contract Address	0xc064f4f215b6a1e4e7f39bd8530c4de0fc43ee9d
Compiler Version	v0.8.20+commit.a1b79de6
Optimization Enabled	No with 200 runs
Explorer	*https://etherscan.io/token/0xc064f4f215b6a1e4e7f39bd8530c4de0fc43ee9d*

Table 1. The deployed smart contract details

1.3. Audit methodology

Our security audit process for smart contract includes two steps:

- Smart contract codes are scanned/tested for commonly known and more specific vulnerabilities using public and RK87, our in-house smart contract security analysis tool.
- Manual audit of the codes for security issues. The contracts are manually analyzed to look for any potential problems.

Following is the list of commonly known vulnerabilities that were considered during the audit of the smart contract:

- Integer Overflow and Underflow
- Timestamp Dependence
- Race Conditions
- Transaction-Ordering Dependence
- DoS with (Unexpected) revert
- DoS with Block Gas Limit
- Gas Usage, Gas Limit and Loops
- Redundant fallback function
- Unsafe type Inference
- Reentrancy
- Explicit visibility of functions state variables (external, internal, private and public)
- Logic Flaws

For vulnerabilities, we categorize the findings into categories as listed in table below, depending on their severity level:

SEVERITY LEVEL	DESCRIPTION
CRITICAL	A vulnerability that can disrupt the contract functioning; creates a critical risk to the contract; required to be fixed immediately.
HIGH	A vulnerability that could affect the desired outcome of executing the contract with high impact; needs to be fixed with high priority.
MEDIUM	A vulnerability that could affect the desired outcome of executing the contract with medium impact in a specific scenario; needs to be fixed.
LOW	An issue that does not have a significant impact, can be considered as less important.

Table 2. Severity levels

1.4. Disclaimer

Please note that security auditing cannot uncover all existing vulnerabilities, and even an audit in which no vulnerabilities are found is not a guarantee for a 100% secure smart contract. However, auditing allows discovering vulnerabilities that were unobserved, overlooked during development and areas where additional security measures are necessary.

2. AUDIT RESULT

2.1. Overview

Table 2 lists some properties of the audited LeisureMeta Token (as of the report writing time).

PROPERTY	VALUE
Name	LeisureMeta
Symbol	LM
Decimals	18
Total Supply	5,000,000,000 x10^{18} Note: the number of decimals is 18, so the total representation token will be 5,000,000,000 or 5 billion.

Table 3. The LeisureMeta Token properties

Note: The maximum supply of LeisureMeta Token is 5,000,000,000 tokens, which are minted only during deployment. Once deployed, the total supply cannot be increased.

2.2. LeisureMeta.sol

The `LeisureMeta` contract was written in `Solidity` language, with the required version to be `^0.8.13`.

This contract extends `ERC20Burnable`, `Ownable` and `Pausable` contracts. With `Ownable`, by default, the token Owner is the contract deployer but he can transfer ownership to another address at any time. `ERC20Burnable` allows token holders to destroy both their own tokens and those that they have an allowance for.

The token Owner can pause/unpause contract using `Pausable` contract, user can only transfer unlocked tokens and only when contract is not paused.

There are 4 types of locks, and for each lock type, the token owner will transfer tokens to the user's wallet address and lock them for multiple periods, with each period being 30 days (1 month):

- daoLock: there are 60 stages, and the tokens will be unlocked after 1 month from the `_dDay`
- saleLock: there are 7 stages, and the tokens will be unlocked after 4 months from the `_dDay`

- generalLock: there are 20 stages, and the tokens will be unlocked after 6 months from the `_dDay`
- customLock: the token owner can freely choose the number of stages and decide when to start them.

The value of the variable `_dDay` is set by the token owner.

The token owner has the ability to revoke any locked tokens from any address locked by generalLock and customLock using the `revoke()` function.

2.3. Findings

During the audit process, the audit team found no vulnerability in the given version of LeisureMeta Token.

2.4. Additional notes and recommendations

2.4.1. Should use constants instead of initializing variables multiple times INFORMATIVE

The variable `aDay` is repeated multiple times in the lock functions, so it is advisable to use a constant for this variable. Using constants is a good practice for declaring values that do not change in your source code. This helps make the code clearer, more readable, and reduces memory load during execution.

UPDATES

- *Jun 14, 2023*: This issue has been acknowledged by the LeisureMeta team.

2.4.2. Gas saving INFORMATIVE

To retrieve items from `LockedItem` (line 111, 120), you can use the memory keyword instead of storage.

UPDATES

- *Jun 14, 2023*: This issue has been acknowledged by the LeisureMeta team.

3. VERSION HISTORY

Version	Date	Status/Change	Created by
1.0	*Jun 13, 2023*	Public Report	Verichains Lab
1.1	*Jun 14, 2023*	Public Report	Verichains Lab

Table 4. Report versions history