SECURITY AUDIT OF

LEISUREMETA DIAMOND

Public Report

Aug 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 Aug 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 Diamond. 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

The LeisureMeta Token (LM) is a ERC20 token used in the Leisure Metaverse ecosystem. The LM token is used as a form of payment for fees incurred from user activities within the Leisure Metaverse 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 Diamond. It was conducted on the source code provided by the LM LLC team.

The latest version of the following files was made available in the course of the review:

Address	Contract
https://etherscan.io/address/0x7C51d293616561b50EBbac65766ED909227C9d0f	DiamondCutFacet
https://etherscan.io/address/0x2526f15Ccb3153Afe55238444a89C7E6f3B8dd4b	OwnershipFacet
https://etherscan.io/address/0xf57792e22E6190AF2eecc0b746617075Ed62242A	TransferFacet
https://etherscan.io/address/0xCe4B36a4d963b5197203890EF435ad6962765d64	ControlFacet
https://etherscan.io/address/0xf3fd37012cED2f8FA759Aed15EED6C9DdFfa7ab7#code	DiamondLoupeFacet
https://etherscan.io/address/0x926043eaf378a40Ba9a477C9Cc90F43941893938	Diamond

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 1. 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

The LeisureMeta Diamond was written in `Solidity` language, with the required versions being `^0.8.19`. The source code was written based on OpenZeppelin's library.

2.2. Diamond proxy implementation

The LeisureMeta Diamond is a upgradeable smart contract that implements the Diamond proxy pattern. The Diamond is a pattern for building complex smart contracts that can be upgraded. The Diamond is based on the EIP-2535 standard.

There are many facets:

- `Diamond Cut Facet`: A diamond contains within it a mapping of function selectors to facet addresses. This facet that are add/replace/remove any number of functions and optionally execute a function with `delegatecall` are protected by contract owner checking.
- `Diamond Loupe Facet`: Provide information about and visualize diamonds.
- `Ownership Facet`: Manage ownership of the diamond, such as transfer ownership to another address, renounce ownership. All functions at this facet are protected by contract owner checking.
- `Control Facet`: Setup config by the owner, such as `setTokenStorage`, `storeToken`, `setGateway`, `setBoundary`, `setDeployedContract`, `addApprovers`, `removeApprovers`. All functions at this facet are protected by contract owner checking. ***Notes: owner can transfer tokens from this contract to `tokenStorage` address.***
- `Transfer Facet`: Transaction management, such as `addTransaction`, `confirmTransaction`, `revokeTransaction`, and `executeTransaction`. All functions at this facet are protected by gateway and approver checking.

Following security considerations in the EIP-2535 standard, we reviewed logic of contracts and following security issues:

SECURITY ISSUE	PASSED/FAILED
Ownership and Authentication	Passed. All operation functions are protected by operator user checking.
Arbitrary Execution with `diamondCut`	Passed. Because the `DiamondCut` function is protected by contract owner checking

SECURITY ISSUE	PASSED/FAILED
Do Not Self Destruct	Passed
Avoid Function Selector Clash	Passed
Transparency	Passed

Table 2. Security considerations in the EIP-2535 standard

2.3. Findings

During the audit process, the audit team found no vulnerability in the given version of LeisureMeta Diamond but we have a best practice recommendation for the LeisureMeta team.

#	Issue	Severity	Status
1	Missing emit event in some functions	INFORMATIVE	**FIXED**

2.3.1. Missing emit event in some functions - INFORMATIVE

Affected files:

- contracts/facets/ControlFacet.sol

Emit event allows people and software to monitor changes to a contract. If any bad acting function is added to a diamond then it can be seen.

Functions have not emitted events: `addApprovers`, `removeApprovers`, `setBoundary`, `setDeployedContract`.

RECOMMENDATION

Add emit event in these functions.

UPDATES

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

APPENDIX

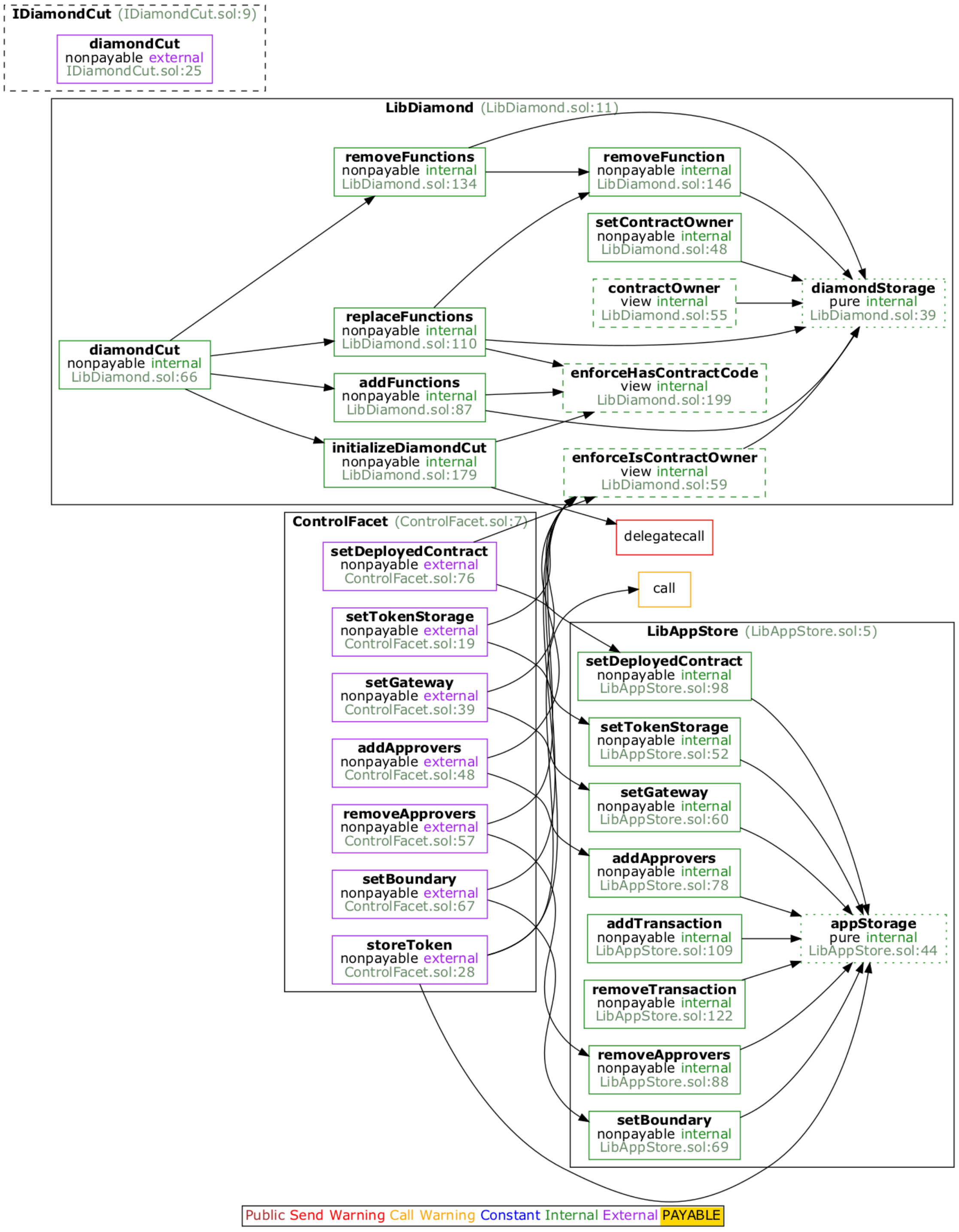
IDiamondCut (IDiamondCut.sol:9)
diamondCut nonpayable external IDiamondCut.sol:25
LibDiamond (LibDiamond.sol:11)
removeFunctions nonpayable internal LibDiamond.sol:134
removeFunction nonpayable internal LibDiamond.sol:146
setContractOwner nonpayable internal LibDiamond.sol:48
contractOwner view internal LibDiamond.sol:55
diamondStorage pure internal LibDiamond.sol:39
replaceFunctions nonpayable internal LibDiamond.sol:110
diamondCut nonpayable internal LibDiamond.sol:66
enforceHasContractCode view internal LibDiamond.sol:199
addFunctions nonpayable internal LibDiamond.sol:87
initializeDiamondCut nonpayable internal LibDiamond.sol:179
enforceIsContractOwner view internal LibDiamond.sol:59
delegatecall
call
ControlFacet (ControlFacet.sol:7)
setDeployedContract nonpayable external ControlFacet.sol:76
setTokenStorage nonpayable external ControlFacet.sol:19
setGateway nonpayable external ControlFacet.sol:39
addApprovers nonpayable external ControlFacet.sol:48
removeApprovers nonpayable external ControlFacet.sol:57
setBoundary nonpayable external ControlFacet.sol:67
storeToken nonpayable external ControlFacet.sol:28
LibAppStore (LibAppStore.sol:5)
setDeployedContract nonpayable internal LibAppStore.sol:98
setTokenStorage nonpayable internal LibAppStore.sol:52
setGateway nonpayable internal LibAppStore.sol:60
addApprovers nonpayable internal LibAppStore.sol:78
addTransaction nonpayable internal LibAppStore.sol:109
appStorage pure internal LibAppStore.sol:44
removeTransaction nonpayable internal LibAppStore.sol:122
removeApprovers nonpayable internal LibAppStore.sol:88
setBoundary nonpayable internal LibAppStore.sol:69
Public Send Warning Call Warning Constant Internal External PAYABLE

Image 1. LeisureMeta's ControlFacet call graph

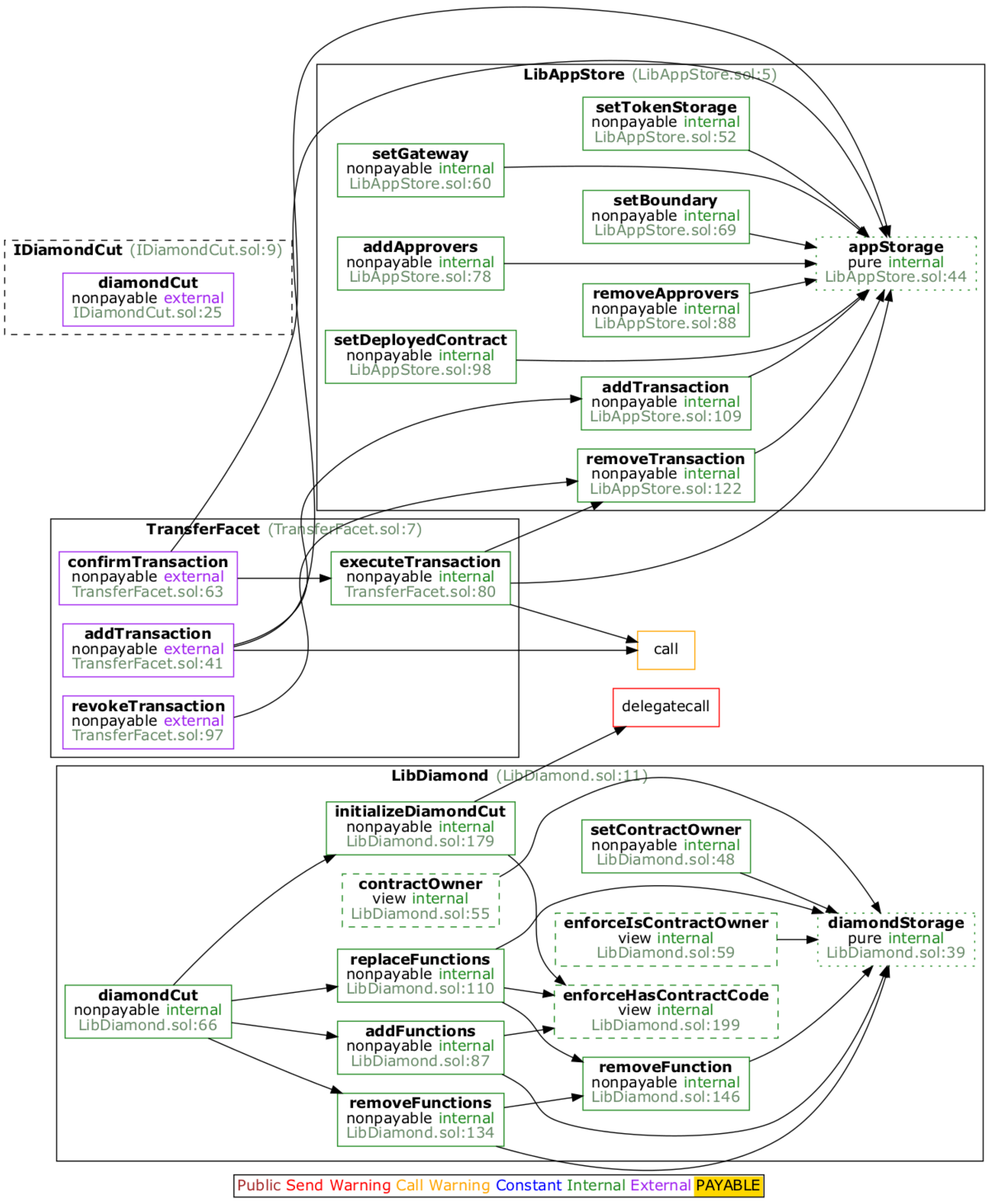
LibAppStore (LibAppStore.sol:5)
setTokenStorage nonpayable internal LibAppStore.sol:52
setGateway nonpayable internal LibAppStore.sol:60
setBoundary nonpayable internal LibAppStore.sol:69
addApprovers nonpayable internal LibAppStore.sol:78
appStorage pure internal LibAppStore.sol:44
removeApprovers nonpayable internal LibAppStore.sol:88
setDeployedContract nonpayable internal LibAppStore.sol:98
addTransaction nonpayable internal LibAppStore.sol:109
removeTransaction nonpayable internal LibAppStore.sol:122
IDiamondCut (IDiamondCut.sol:9)
diamondCut nonpayable external IDiamondCut.sol:25
TransferFacet (TransferFacet.sol:7)
confirmTransaction nonpayable external TransferFacet.sol:63
executeTransaction nonpayable internal TransferFacet.sol:80
addTransaction nonpayable external TransferFacet.sol:41
revokeTransaction nonpayable external TransferFacet.sol:97
call
delegatecall
LibDiamond (LibDiamond.sol:11)
initializeDiamondCut nonpayable internal LibDiamond.sol:179
setContractOwner nonpayable internal LibDiamond.sol:48
contractOwner view internal LibDiamond.sol:55
enforceIsContractOwner view internal LibDiamond.sol:59
diamondStorage pure internal LibDiamond.sol:39
replaceFunctions nonpayable internal LibDiamond.sol:110
diamondCut nonpayable internal LibDiamond.sol:66
enforceHasContractCode view internal LibDiamond.sol:199
addFunctions nonpayable internal LibDiamond.sol:87
removeFunction nonpayable internal LibDiamond.sol:146
removeFunctions nonpayable internal LibDiamond.sol:134
Public Send Warning Call Warning Constant Internal External PAYABLE

Image 2. LeisureMeta's TransferFacet call graph

3. VERSION HISTORY

Version	Date	Status/Change	Created by
1.0	*Jul 28, 2023*	Public Report	Verichains Lab
1.1	*Aug 14, 2023*	Public Report	Verichains Lab

Table 3. Report versions history