Smart Contract
Security Audit Report

Table Of Contents

1 Executive Summary

On 2023.01.31, the SlowMist security team received the SuperCells team's security audit application for Cell Core, developed the audit plan according to the agreement of both parties and the characteristics of the project, and finally issued the security audit report.

The SlowMist security team adopts the strategy of "white box lead, black, grey box assists" to conduct a complete security test on the project in the way closest to the real attack.

The test method information:

Test method	Description
Black box testing	Conduct security tests from an attacker's perspective externally.
Grey box testing	Conduct security testing on code modules through the scripting tool, observing the internal running status, mining weaknesses.
White box testing	Based on the open source code, non-open source code, to detect whether there are vulnerabilities in programs such as nodes, SDK, etc.

The vulnerability severity level information:

Level	Description
Critical	Critical severity vulnerabilities will have a significant impact on the security of the DeFi project, and it is strongly recommended to fix the critical vulnerabilities.
High	High severity vulnerabilities will affect the normal operation of the DeFi project. It is strongly recommended to fix high-risk vulnerabilities.
Medium	Medium severity vulnerability will affect the operation of the DeFi project. It is recommended to fix medium-risk vulnerabilities.
Low	Low severity vulnerabilities may affect the operation of the DeFi project in certain scenarios. It is suggested that the project team should evaluate and consider whether these vulnerabilities need to be fixed.
Weakness	There are safety risks theoretically, but it is extremely difficult to reproduce in engineering.
Suggestion	There are better practices for coding or architecture.

2 Audit Methodology

The security audit process of SlowMist security team for smart contract includes two steps:

Smart contract codes are scanned/tested for commonly known and more specific vulnerabilities using automated analysis tools.

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 was considered during the audit of the smart contract:

Serial Number	Audit Class	Audit Subclass
1	Overflow Audit	-
2	Reentrancy Attack Audit	-
3	Replay Attack Audit	-
4	Flashloan Attack Audit	-
5	Race Conditions Audit	Reordering Attack Audit
6	Permission Vulnerability Audit	Access Control Audit
		Excessive Authority Audit

Serial Number	Audit Class	Audit Subclass
7	Security Design Audit	External Module Safe Use Audit
		Compiler Version Security Audit
		Hard-coded Address Security Audit
		Fallback Function Safe Use Audit
		Show Coding Security Audit
		Function Return Value Security Audit
		External Call Function Security Audit
		Block data Dependence Security Audit
		tx.origin Authentication Security Audit
8	Denial of Service Audit	-
9	Gas Optimization Audit	-
10	Design Logic Audit	-
11	Variable Coverage Vulnerability Audit	-
12	"False Top-up" Vulnerability Audit	-
13	Scoping and Declarations Audit	-
14	Malicious Event Log Audit	-
15	Arithmetic Accuracy Deviation Audit	-
16	Uninitialized Storage Pointer Audit	-

3 Project Overview

3.1 Project Introduction

Project: SuperCells Core

Module: Module: Token + Oracle + Proxy

3.2 Vulnerability Information

The following is the status of the vulnerabilities found in this audit:

NO	Title	Category	Level	Status
N1	Missing event record	Malicious Event Log Audit	Suggestion	Fixed
N2	Safe transfer issue	Others	Low	Fixed
N3	Risk of excessive authority	Authority Control Vulnerability Audit	Medium	Acknowledged

4 Code Overview

4.1 Contracts Description

Codebase:

https://github.com/SuperCellT/cell-core

commit: 7229ea55fee9ca96790d90a64fe397ea453ff518

The main network address of the contract is as follows:

The code was not deployed to the mainnet.

4.2 Visibility Description

The SlowMist Security team analyzed the visibility of major contracts during the audit, the result as follows:

Cell			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	ERC721A

Cell			
_hash	Internal	-	-
_verify	Internal	-	-
_recover	Internal	-	-
_baseURI	Internal	-	-
setURI	Public	Can Modify State	onlyOwner
addSigner	External	Can Modify State	onlyOwner
removeSigner	External	Can Modify State	onlyOwner
setProxyer	External	Can Modify State	onlyOwner
addStageRecord	External	Can Modify State	onlyOwner
updateStageRecordMaxCount	External	Can Modify State	onlyOwner
updateStageRecordCostPoint	External	Can Modify State	onlyOwner
updateStageRecordIncPoint	External	Can Modify State	onlyOwner
updateStageRecordValid	External	Can Modify State	onlyOwner
mint	External	Can Modify State	onlyProxy
mintWithSignature	External	Can Modify State	onlyProxy
mintWithPoints	External	Can Modify State	onlyProxy
_beforeTokenTransfers	Internal	Can Modify State	-
lock	External	Can Modify State	nonReentrant
unlock	External	Can Modify State	nonReentrant
use	External	Can Modify State	nonReentrant
IsActive	External	-	-
getDetailByTokenId	External	-	-

Nucleus			
Function Name	Visibility	Mutability	Modifiers
<Receive Ether>	External	Payable	-
<Constructor>	Public	Can Modify State	-
setOracleAddress	External	Can Modify State	onlyOwner
claim	External	Payable	nonReentrant
release	External	Can Modify State	onlyOwner
releaseBNB	External	Can Modify State	onlyOwner
cal	Public	-	-

Oracle			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
addOracleAddress	External	Can Modify State	onlyOwner
removeOracleAddress	External	Can Modify State	onlyOwner
setTokenNameAddress	External	Can Modify State	onlyOwner
setRates	External	Can Modify State	onlyOracle
setRatePeriod	External	Can Modify State	onlyOwner
setFixedPrice	External	Can Modify State	onlyOwner
revokeFixedPrice	External	Can Modify State	onlyOwner
getRatesDetail	External	-	-

Proxy			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-

Proxy			
setOracleAddress	External	Can Modify State	onlyOwner
setCellAddress	External	Can Modify State	onlyOwner
setMintPrice	External	Can Modify State	onlyOwner
mint	External	Can Modify State	nonReentrant
mintWithSignature	External	Can Modify State	nonReentrant
mintWithPoints	External	Can Modify State	nonReentrant
release	External	Can Modify State	nonReentrant onlyOwner

SCT			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	ERC20 ERC20Permit
mint	Public	Can Modify State	onlyOwner

4.3 Vulnerability Summary

[N1] [Suggestion] Missing event record

Category: Malicious Event Log Audit

Content

The following functions do not log events.

- contracts/Cell.sol

updateStageRecordMaxCount

updateStageRecordCostPoint

updateStageRecordIncPoint

updateStageRecordValid

addSigner

removeSigner

setProxyer

- contracts/Nucleus.sol

setOracleAddress

- contracts/Oracle.sol

addOracleAddress

removeOracleAddress

setTokenNameAddress

setRates

setRatePeriod

setFixedPrice

revokeFixedPrice

- contracts/Proxy.sol

setOracleAddress

setCellAddress

setMintPrice

Solution

It is recommended to record the modification of sensitive parameters for subsequent community review or self-examination.

Status

Fixed

[N2] [Low] Safe transfer issue

Category: Others

Content

- contracts/Nucleus.sol

Use transferFrom in the `claim` function to transfer the token. If the token in the operation does not meet the `eip20` standard, the transaction may fail.

- contracts/Proxy.sol

Use transferFrom in the `mint` function to transfer the token. If the token in the operation does not meet the `eip20` standard, the transaction may fail.

Solution

It is recommended to use OpenZeppelin's SafeERC20 library for token transfers.

Status

Fixed

[N3] [Medium] Risk of excessive authority

Category: Authority Control Vulnerability Audit

Content

- contracts/Cell.sol

The owner has too much authority, and if the owner's private key is leaked, the attacker can control the casting of NFT.

- contracts/Oracle.sol

The owner has too much authority. If the owner's private key is leaked, the attacker can manipulate the price by setting `setRates` and `setFixedPrice`.

- contracts/Proxy.sol

The owner's authority is too large. If the owner's private key is leaked, the attacker can withdraw the revenue in the contract. You can also set a malicious Oracle contract through `setOracleAddress` to control the price.

- contracts/Nucleus.sol

The owner has too much authority. If the owner's private key is leaked, the attacker can set a malicious Oracle contract through `setOracleAddress` to control the price.

Solution

It is recommended to hand over the Owner role to the community or timeLock contract governance, at least multi-signature should be used.

Status

Acknowledged; In the future, multi-signature contracts will be used to manage the owner.Users need to verify whether the owner is a multi-sign contract.

5 Audit Result

Audit Number	Audit Team	Audit Date	Audit Result
0X002302060001	SlowMist Security Team	2023.01.31 - 2023.02.06	Medium Risk

Summary conclusion: The SlowMist security team use a manual and SlowMist team's analysis tool to audit the project, during the audit work we found 1 medium risk, 1 low risk, 1 suggestion vulnerabilities.

6 Statement

SlowMist issues this report with reference to the facts that have occurred or existed before the issuance of this report, and only assumes corresponding responsibility based on these.

For the facts that occurred or existed after the issuance, SlowMist is not able to judge the security status of this project, and is not responsible for them. The security audit analysis and other contents of this report are based on the documents and materials provided to SlowMist by the information provider till the date of the insurance report (referred to as "provided information"). SlowMist assumes: The information provided is not missing, tampered with, deleted or concealed. If the information provided is missing, tampered with, deleted, concealed, or inconsistent with the actual situation, the SlowMist shall not be liable for any loss or adverse effect resulting therefrom. SlowMist only conducts the agreed security audit on the security situation of the project and issues this report. SlowMist is not responsible for the background and other conditions of the project.

Official Website

www.slowmist.com

E-mail

team@slowmist.com

Twitter

@SlowMist_Team

Github

https://github.com/slowmist