Smart Contract Security Audit Report

Table Of Contents

1 Executive Summary

On 2024.05.08, the security team received the souland team's security audit application, 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 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.

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 the 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
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

Token smart contract for souland.

3.2 Vulnerability Information

The following is the status of the vulnerabilities found in this audit,during the audit work we found 1 vulnerability.

No	Title	Category	Level	Status
1	Using private rather than public for constants, saves gas	Gas Optimization	Suggestion	

4 Code Overview

4.1 Contracts Description

Codebase:
https://github.com/deFang/souland_contract/commit/c66e26240f9a67a7a45518564132a520519e67f2
File: souland_contract-main.zip
sha256: 6c36bda46f0ad6c36fdb775fcedb397f93df6477b9dd15357311f73328dffb97

4.2 Vulnerability Summary

N1 [Suggestion] Using private rather than public for constants, saves gas

Category: Gas Optimization

Content:
Using private instead of public for constants in Solidity can significantly reduce gas consumption during deployment.

souland_contract-main/contracts/Sltoken.sol#9-10:

```
uint256 constant public TOTALSUPPLY = 1_000_000_000;
uint256 constant ONE = 1e18;
```

Solution:

```
uint256 constant private  TOTALSUPPLY = 1_000_000_000;
uint256 constant private ONE = 1e18;
```

5 Statement

The security team 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, the security team 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 the security team by the information provider till the date of the insurance report (referred to as "provided information"). The security team 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 security team shall not be liable for any loss or adverse effect resulting therefrom. The security team only conducts the agreed security audit on the security situation of the project and issues this report. The security team is not responsible for the background and other conditions of the project.