Smart Contract
Security Audit Report

Table Of Contents

1 Executive Summary

On 2022.06.29, the SlowMist security team received the team's security audit application for KelePoolStaking, 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.

Level	Description
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

Audit Version:

Proxy: 0xACBA4cFE7F30E64dA787c6Dc7Dc34f623570e758

implementation: 0xeee5dd6c3d57207d08ce090b62d34d67620ed44a

Fixed Version:

Proxy: 0xACBA4cFE7F30E64dA787c6Dc7Dc34f623570e758

implementation: 0x3B27417D971D6aec8a8406143c507095F729BFF0

3.2 Vulnerability Information

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

NO	Title	Category	Level	Status
N1	Risk of excessive authority	Authority Control Vulnerability	Medium	Fixed
N2	Risk of excessive authority	Authority Control Vulnerability	Medium	Fixed
N3	Risk of excessive authority	Authority Control Vulnerability	Medium	Fixed

4 Code Overview

4.1 Contracts Description

The main network address of the contract is as follows:

Proxy: 0xACBA4cFE7F30E64dA787c6Dc7Dc34f623570e758

implementation: 0x3B27417D971D6aec8a8406143c507095F729BFF0

4.2 Visibility Description

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

KelePoolStaking			
Function Name	Visibility	Mutability	Modifiers
_authorizeUpgrade	Internal	Can Modify State	onlyOwner
initialize	Public	Can Modify State	initializer
deposit	External	Payable	-
createValidator	External	Payable	-
withdraw	Public	Can Modify State	onlyFoundation
takeOutFee	Public	Can Modify State	onlyFoundation
changeOwner	Public	Can Modify State	onlyOwner
changeOperator	Public	Can Modify State	onlyOwner
changeFoundation	Public	Can Modify State	onlyOwner
changeFee	Public	Can Modify State	onlyOperator
changeStatus	Public	Can Modify State	onlyOperator
changeMinimum	Public	Can Modify State	onlyOperator
getSystemInfo	Public	-	-

ERC1967Proxy			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Payable	-

ERC1967Proxy			
_implementation	Internal	-	-

4.3 Vulnerability Summary

[N1] [Medium] Risk of excessive authority

Category: Authority Control Vulnerability

Content

The Owner of the kelePoolStaking contract has the right to change the `_system.owner`, `_system.operator` and `_system.foundation`. The operators and foundation have sensitive permissions such as withdrawing users' funds, changing and withdrawing the fee. If they can be arbitrarily set by the owner, there is a risk to the users' funds.

Code location: contracts/KelePoolStaking.sol #L1085-1104

```
    // change ownership
    function changeOwner(address owner) public onlyOwner {
        require(owner != address(0), "Error address");
        _system.owner = owner;
        emit OnOwnerChanged(msg.sender, owner);
    }

    // change operator
    function changeOperator(address operator) public onlyOwner {
        require(operator != address(0), "Error address");
        _system.operator = operator;
        emit OnOperatorChanged(msg.sender, operator);
    }

    // change foundation
    function changeFoundation(address foundation) public onlyOwner {
        require(foundation != address(0), "Error address");
        _system.foundation = foundation;
        emit OnFoundationChanged(msg.sender, foundation);
    }
```

Solution

It is recommended to transfer the permissions of the Owner role to timelock contract management or use multi-sign.

Status

Fixed; The transaction hash to fix the problem is

0x5f7edda8586ffe46496881efea073ad2a6a7785e732015edc19bd9cb1258ced0.

[N2] [Medium] Risk of excessive authority

Category: Authority Control Vulnerability

Content

The operator role has the right to arbitrarily change the fee to be charged, the system status and the minimum deposit amount of the system. And there is no upper limit on the fee and the minimum deposit amount. If they are set too large, the user's deposit needs to pay a very high price.

Code location: contracts/KelePoolStaking.sol #L1106-1122

```
// change fee
function changeFee(uint256 fee) public onlyOperator {
    _system.fee = fee;
    emit OnFeeChanged(msg.sender, fee);
}

// change status
function changeStatus() public onlyOperator {
    _system.status = !_system.status;
    emit OnStatusChanged(msg.sender, _system.status);
}

// change minimum
function changeMinimum(uint256 amount) public onlyOperator {
    _system.minimum = amount;
    emit OnMinimumChanged(msg.sender, amount);
}
```

Solution

It is recommended to transfer the permissions of the operator role to timelock contract management or use multi-sign and set caps on the fee and the minimum deposit amount.

Status

Fixed

[N3] [Medium] Risk of excessive authority

Category: Authority Control Vulnerability

Content

The foundation role has the right to transfer users' funds in the contract to any address and the maximum transfer amount at one time is the total amount of funds in the contract (`_system.balance`).

Code location: contracts/KelePoolStaking.sol #L1057-1071

```
// withdraw
function withdraw(address receiver, uint256 amount) public onlyFoundation {
    require(receiver != address(0), "Invalid receiver");
    require(
        amount > 0 && address(this).balance >= amount,
        "Invalid amount"
    );
    if (_system.balance >= amount) {
        _system.balance -= amount;
    } else {
        _system.balance = 0;
    }
    payable(receiver).transfer(amount);
    emit OnWithdraw(msg.sender, receiver, amount);
}
```

Solution

It is recommended to transfer the permissions of the foundation role to the timelock contract management or use multi-sign.

Status

Fixed; The transaction hash to fix the problem is

0x99a042ac5542bfcd2b9962197df09fb0da4178ed3fe2250739055e1a62d4de10

5 Audit Result

Audit Number	Audit Team	Audit Date	Audit Result
0X002207010002	SlowMist Security Team	2022.06.29 - 2022.07.01	Passed

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 3 medium risk. All findings were fixed.

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