Smart contract security audit report

Cradle smart contract security audit report

Audit Team： Noneage security team

Audit date： March 15, 2021

Cradle Smart Contract Security Audit Report

1. Overview

On March 12, 2021, the security team of Zero Hour Technology received the security audit request of the **Cradle project**. The team completed the security audit of the **Cradle smart contract** on March 15, 2021. The audit process is in progress. The security audit experts of Zero Hour Technology communicate with the relevant interface people of the Cradle project, maintain information symmetry, conduct security audits under controllable operational risks, and try to avoid risks to project generation and operations during the testing process.

Through communicat and feedback with Cradle project party, it is confirmed that the loopholes and risks found in the audit process have been repaired or within the acceptable range. The result of this Cradle smart contract security audit: **passed.**

Audit Report MD5：CA5B8D4DA90F9FF7EB3CE45632D23068

2. Background

2.1 Project Description

> Project name: Cradle
>
> Project official: https://cradle.finance/
>
> Contract type: DeFi Token contract
>
> Code language: Solidity
>
> Contract documents: Cradle.sol

2.2 Audit Range

Cradle officially provides contract documents and MD5:

Cradle.sol e39958c72fb31f2de43e89f8c8374e3a

2.3 Security Audit List

The security experts of Noneage Technology conduct security audits on the security audit list within the agreement, The scope of this smart contract security audit does not include new attack methods that may appear in the future, does not include the code after contract upgrades or tampering, and is not included in the subsequent cross-country, does not include cross-chain

deployment, does not include project front-end code security and project platform server security.

This smart contract security audit list includes the following:

- Integer overflow
- Reentry attack
- Floating point numbers and numerical precision
- Default visibility
- Tx.origin authentication
- Wrong constructor
- Return value not verified
- Insecure random numbers
- Timestamp dependency
- Transaction order is dependent
- Delegatecall
- Call
- Denial of service
- Logic design flaws
- Fake recharge vulnerability
- Short address attack
- Uninitialized storage pointer
- Additional token issuance
- Frozen account bypass
- Access control
- Gas usage

3. Contract Structure Analysis

3.1 Directory Structure

```
└─Cradle
    Cradle.sol
```

3.2 Cradle contract

Contract

Context

- _msgSender()
- _msgData()

Ownable

- owner()
- renounceOwnership()
- transferOwnership(address newOwner)

Operator

- operator()
- isOperator()

- transferOperator(address newOperator_)
- *transferOperator(address newOperator*)

Cradle

- setStableToken(address money, bool result)
- setFeeAccount(address _feeAccount)
- setFeeRate(uint256 _feeRate)
- projectInfo(address projectId)
- create(address stock,address money,uint256 stockAmount,uint256 moneyAmount,uint256[] memory times)
- buy(address projectId, uint256 targetStockAmount)
- userBuyInfo(address projectId, address user)
- userRelease(address projectId)
- release(address projectId) public updateProject(projectId)
- transferOut(bool isStock,address projectId,address to,uint256 amount)
- transferIn(bool isStock,address projectId,address from,uint256 amount)
- _safeTransfer(address token,address to,uint256 value)
- safeTransferFrom(address token,address from,uint256 value)
- _safeTransferFrom(address token,address from,address to,uint256 value)

Interface

IERC20

- totalSupply()
- balanceOf(address account)
- transfer(address recipient, uint256 amount)
- allowance(address owner, address spender)
- approve(address spender, uint256 amount)
- transferFrom(address sender, address recipient, uint256 amount)

Library

Math

SafeMath

SafeERC20

Address

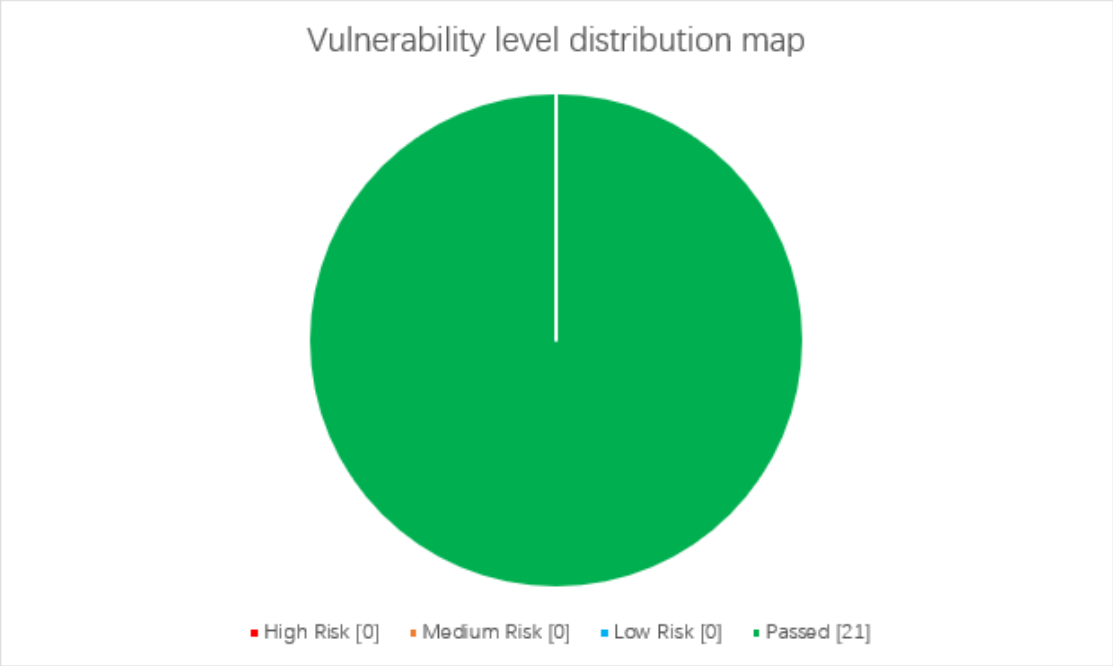
- isContract(address account)
- functionCall(address target, bytes memory data)
- functionCall(address target,bytes memory data,string memory errorMessage)
- functionCallWithValue(address target,bytes memory data,uint256 value)
- functionCallWithValue(address target,bytes memory data,uint256 value,string memory errorMessage)
- functionStaticCall(address target, bytes memory data)
- functionStaticCall(address target,bytes memory data,string memory errorMessage)
- functionDelegateCall(address target, bytes memory data)
- functionDelegateCall(address target,bytes memory data,string memory errorMessage)
- _verifyCallResult(bool success,bytes memory returndata,string memory errorMessage)

4. Audit Details

4.1 Vulnerabilities Distribution

Vulnerabilities in this security audit are distributed by risk level, as follows:

Vulnerability level distribution			
High risk	Medium risk	Low risk	Passed
0	0	0	21



This smart contract security audit has 0 high-risk vulnerabilities, 0 medium-risk vulnerabilities, 0 low-risk vulnerabilities, and 21 passed, with a high security level.

4.2 Vulnerabilities Details

A security audit was conducted on the smart contract within the agreement, and no security vulnerabilities that could be directly exploited and generated security problems were found, and the security audit was passed.

4.3 Other Risks

Other risks refer to the code that contract security auditors consider to be risky. Under certain circumstances, it may affect the stability of the project, but it cannot constitute a security issue that directly harms.

4.3.1 Security of Admin Token Transfer

- **Issue causes**

The existence of the smart contract administrator sets core variables. When the administrator key is lost or controlled by malicious people, it will affect the normal operation of the project.

- **Question detail**

Through the audit contract, it is found that some settings in the Cradle contract are controlled by the administrator. If the administrator key is lost or is controlled by a malicious person, the stability of the project will be affected. The specific code is as follows:

```
function setStableToken(address money, bool result) public onlyOwner {
    stableTokens[money] = result;
}

function setFeeAccount(address _feeAccount) public onlyOwner {
    feeAccount = _feeAccount;
}

function setFeeRate(uint256 _feeRate) public onlyOwner {
    feeRate = _feeRate;
}
```

- **Safety advice**

It is necessary to store the private key of the deployer's address safely and effectively to avoid loss or acquisition by malicious persons.

5. Security Audit Tool

Tool name	Tool Features
Oyente	Can be used to detect common bugs in smart contracts
securify	Common types of smart contracts that can be verified
MAIAN	Multiple smart contract vulnerabilities can be found and classified
Noneage Internal Toolkit	Noneage(hawkeye system) self-developed toolkit + https://audit.noneage.com

6. Vulnerability assessment criteCradle

Vulnerability level	Vulnerability description
High risk	Vulnerabilities that can directly lead to the loss of contracts or users' digital assets, such as integer overflow vulnerabilities, false recharge vulnerabilities, re-entry vulnerabilities, illegal token issuance, etc. Vulnerabilities that can directly cause the ownership change of the token contract or verification bypass, such as: permission verification bypass, call code injection, vaCradleble coverage, unverified return value, etc. Vulnerabilities that can directly cause the token to work normally, such as denial of service vulnerabilities, insecure random numbers, etc.
Medium risk	Vulnerabilities that require certain conditions to trigger, such as vulnerabilities triggered by the token owner's high authority, and transaction sequence dependent vulnerabilities. Vulnerabilities that cannot directly cause asset loss, such as function default visibility errors, logic design flaws, etc.
Low risk	Vulnerabilities that are difficult to trigger, or vulnerabilities that cannot lead to asset loss, such as vulnerabilities that need to be triggered at a cost higher than the benefit of the attack, cannot lead to incorrect coding of security vulnerabilities.

零时科技
NONEAGE

Telephone：86-17391945345 18511993344
Email ：support@noneage.com
Site ：www.noneage.com
Weibo ：weibo.com/noneage