Smart Contract Security Audit Report

[2021]

Table Of Contents

1 Executive Summary

On 2021.10.03, the SlowMist security team received the Vee Finance team's security audit application for Vee Finance Phase 2, 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:

- Reentrancy Vulnerability
- Replay Vulnerability
- Reordering Vulnerability
- Short Address Vulnerability
- Denial of Service Vulnerability
- Transaction Ordering Dependence Vulnerability
- Race Conditions Vulnerability
- Authority Control Vulnerability
- Integer Overflow and Underflow Vulnerability
- TimeStamp Dependence Vulnerability
- Uninitialized Storage Pointers Vulnerability
- Arithmetic Accuracy Deviation Vulnerability
- tx.origin Authentication Vulnerability

- "False top-up" Vulnerability
- Variable Coverage Vulnerability
- Gas Optimization Audit
- Malicious Event Log Audit
- Redundant Fallback Function Audit
- Unsafe External Call Audit
- Explicit Visibility of Functions State Variables Aduit
- Design Logic Audit
- Scoping and Declarations Audit

3 Project Overview

3.1 Project Introduction

Audie version:

https://github.com/VeeFinance/veefinance-protocol *(Does not contain the periphery directory)*

commit: 2c727d4b7b7ba9253592197f0663e7d050a1e885

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	Confirmed

4 Code Overview

4.1 Contracts Description

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:

BaseJumpRateModelV2			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
updateJumpRateModel	External	Can Modify State	-
utilizationRate	Public	-	-
getBorrowRateInternal	Internal	-	-
getSupplyRate	External	-	-
updateJumpRateModelInternal	Internal	Can Modify State	-

CErc20			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	-
mint	External	Can Modify State	-
redeem	External	Can Modify State	-
redeemUnderlying	External	Can Modify State	-

CErc20			
borrow	External	Can Modify State	-
repayBorrow	External	Can Modify State	-
repayBorrowBehalf	External	Can Modify State	-
liquidateBorrow	External	Can Modify State	-
sweepToken	External	Can Modify State	-
_addReserves	External	Can Modify State	-
getCashPrior	Internal	-	-
doTransferIn	Internal	Can Modify State	-
doTransferOut	Internal	Can Modify State	-
mintBehalf	External	Can Modify State	-
requireNoError	Internal	-	-

CEther			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
mint	External	Payable	-
redeem	External	Can Modify State	-
redeemUnderlying	External	Can Modify State	-
borrow	External	Can Modify State	-
repayBorrow	External	Payable	-

CEther			
repayBorrowBehalf	External	Payable	-
liquidateBorrow	External	Payable	-
<Fallback>	External	Payable	-
getCashPrior	Internal	-	-
doTransferIn	Internal	Can Modify State	-
doTransferOut	Internal	Can Modify State	-
requireNoError	Internal	-	-
mintBehalf	External	Payable	-

ChainlinkAnchoredView			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
price	External	-	-
priceInternal	Internal	-	-
getUnderlyingPrice	External	-	-
getTokenConfigBySymbol	Public	-	-
getTokenConfigBySymbolHash	Public	-	-
getTokenConfigByCToken	Public	-	-

Comptroller			
Function Name	Visibility	Mutability	Modifiers

Comptroller			
<Constructor>	Public	Can Modify State	-
getAssetsIn	External	-	-
checkMembership	External	-	-
enterMarkets	Public	Can Modify State	-
addToMarketInternal	Internal	Can Modify State	-
exitMarket	External	Can Modify State	-
mintAllowed	External	Can Modify State	-
mintVerify	External	Can Modify State	-
redeemAllowed	External	Can Modify State	-
redeemAllowedInternal	Internal	-	-
redeemVerify	External	Can Modify State	-
borrowAllowed	External	Can Modify State	-
borrowVerify	External	Can Modify State	-
repayBorrowAllowed	External	Can Modify State	-
repayBorrowVerify	External	Can Modify State	-
liquidateBorrowAllowed	External	Can Modify State	-
liquidateBorrowVerify	External	Can Modify State	-
seizeAllowed	External	Can Modify State	-
seizeVerify	External	Can Modify State	-
transferAllowed	External	Can Modify State	-

Comptroller			
transferVerify	External	Can Modify State	-
getAccountLiquidity	Public	-	-
getAccountLiquidityInternal	Internal	-	-
getHypotheticalAccountLiquidity	Public	-	-
getHypotheticalAccountLiquidityInternal	Internal	-	-
liquidateCalculateSeizeTokens	External	-	-
_setVeeHub	Public	Can Modify State	-
_setPriceOracle	Public	Can Modify State	-
_setCloseFactor	External	Can Modify State	-
_setCollateralFactor	External	Can Modify State	-
_setLiquidationIncentive	External	Can Modify State	-
_supportMarket	External	Can Modify State	-
_setTokenAddress	External	Can Modify State	-
_addMarketInternal	Internal	Can Modify State	-
_setMarketBorrowCaps	External	Can Modify State	-
_setBorrowCapGuardian	External	Can Modify State	-
_setPauseGuardian	Public	Can Modify State	-
_setMintPaused	Public	Can Modify State	-
_setBorrowPaused	Public	Can Modify State	-
_setTransferPaused	Public	Can Modify State	-

Comptroller			
_setSeizePaused	Public	Can Modify State	-
_become	Public	Can Modify State	-
adminOrInitializing	Internal	-	-
setVeeSpeedInternal	Internal	Can Modify State	-
updateVeeSupplyIndex	Internal	Can Modify State	-
updateVeeBorrowIndex	Internal	Can Modify State	-
distributeSupplierVee	Internal	Can Modify State	-
distributeBorrowerVee	Internal	Can Modify State	-
updateContributorRewards	Public	Can Modify State	-
claimVee	Public	Can Modify State	-
claimVee	Public	Can Modify State	-
claimVee	Public	Can Modify State	-
grantVeeInternal	Internal	Can Modify State	-
_grantVee	Public	Can Modify State	-
_setVeeSpeed	Public	Can Modify State	-
_setContributorVeeSpeed	Public	Can Modify State	-
getAllMarkets	Public	-	-
getBlockNumber	Public	-	-
getVeeAddress	Public	-	-
mintBehalf	Public	Can Modify State	-

Comptroller			
mintBehalf	Public	Payable	-

CToken			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	-
transferTokens	Internal	Can Modify State	-
transfer	External	Can Modify State	nonReentrant
transferFrom	External	Can Modify State	nonReentrant
approve	External	Can Modify State	-
allowance	External	-	-
balanceOf	External	-	-
balanceOfUnderlying	External	Can Modify State	-
getAccountSnapshot	External	-	-
getBlockNumber	Internal	-	-
borrowRatePerBlock	External	-	-
supplyRatePerBlock	External	-	-
totalBorrowsCurrent	External	Can Modify State	nonReentrant
borrowBalanceCurrent	External	Can Modify State	nonReentrant
borrowBalanceStored	Public	-	-
borrowBalanceStoredInternal	Internal	-	-

CToken			
exchangeRateCurrent	Public	Can Modify State	nonReentrant
exchangeRateStored	Public	-	-
exchangeRateStoredInternal	Internal	-	-
getCash	External	-	-
accrueInterest	Public	Can Modify State	-
mintInternal	Internal	Can Modify State	nonReentrant
mintFresh	Internal	Can Modify State	-
redeemInternal	Internal	Can Modify State	nonReentrant
redeemUnderlyingInternal	Internal	Can Modify State	nonReentrant
redeemFresh	Internal	Can Modify State	-
borrowInternal	Internal	Can Modify State	nonReentrant
borrowFresh	Internal	Can Modify State	-
repayBorrowInternal	Internal	Can Modify State	nonReentrant
repayBorrowBehalfInternal	Internal	Can Modify State	nonReentrant
repayBorrowFresh	Internal	Can Modify State	-
liquidateBorrowInternal	Internal	Can Modify State	nonReentrant
liquidateBorrowFresh	Internal	Can Modify State	-
seize	External	Can Modify State	nonReentrant
seizeInternal	Internal	Can Modify State	-
_setPendingAdmin	External	Can Modify State	-

CToken			
_acceptAdmin	External	Can Modify State	-
_setComptroller	Public	Can Modify State	-
_setReserveFactor	External	Can Modify State	nonReentrant
_setReserveFactorFresh	Internal	Can Modify State	-
_addReservesInternal	Internal	Can Modify State	nonReentrant
_addReservesFresh	Internal	Can Modify State	-
_reduceReserves	External	Can Modify State	nonReentrant
_reduceReservesFresh	Internal	Can Modify State	-
_setInterestRateModel	Public	Can Modify State	-
_setInterestRateModelFresh	Internal	Can Modify State	-
getCashPrior	Internal	-	-
doTransferIn	Internal	Can Modify State	-
doTransferOut	Internal	Can Modify State	-
mintBehalfInternal	Internal	Can Modify State	nonReentrant

EmergencyStop			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
setGuardian	External	Can Modify State	-
systemStop	External	Can Modify State	-

JumpRateModel			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
utilizationRate	Public	-	-
getBorrowRate	Public	-	-
getSupplyRate	Public	-	-

JumpRateModelV2			
Function Name	Visibility	Mutability	Modifiers
getBorrowRate	External	-	-
getSupplyRate	External	-	-
<Constructor>	Public	Can Modify State	BaseJumpRateModelV2

Maximillion			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
repayBehalf	Public	Payable	-
repayBehalfExplicit	Public	Payable	-

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

Timelock			
<Receive Ether>	External	Payable	-
<Fallback>	External	Payable	-
setDelay	Public	Can Modify State	-
setSignerWhitelist	Public	Can Modify State	-
queueTransaction	Public	Can Modify State	-
cancelTransaction	Public	Can Modify State	-
executeTransaction	Public	Payable	-
getBlockTimestamp	Internal	-	-
execute	Internal	Can Modify State	-
executeImmediate	Public	Payable	-

Unitroller			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
_setPendingImplementation	Public	Can Modify State	-
_acceptImplementation	Public	Can Modify State	-
_setPendingAdmin	Public	Can Modify State	-
_acceptAdmin	Public	Can Modify State	-
<Fallback>	External	Payable	-

WhitePaperInterestRateModel

WhitePaperInterestRateModel			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
utilizationRate	Public	-	-
getBorrowRate	Public	-	-
getSupplyRate	Public	-	-

4.3 Vulnerability Summary

[N1] [Medium] Risk of excessive authority

Category: Authority Control Vulnerability

Content

In the Comptroller and cToken contracts, the admin role can modify key sensitive parameters such as the oracle, the rate model, and the market, which will lead to the risk of excessive authority of the admin role.

Solution

It is recommended that the authority division processing be carried out in the early stage of the project:

1. The authority related to user funds should be transferred to the timelock contract or community governance. The timelock contract admin can use multi-signature to avoid the risk of private key leakage.
2. In the early stage of the project, some parameters may be frequently modified. This part of the permissions can be controlled separately.
3. Consider retaining the authority to temporarily suspend the project in order to respond to an emergency in the early stage of the project, which can quickly suspend the project and stop the loss in time.
4. After the project has passed the early stage of smooth operation, the authority can be transferred to community governance.

Status

Confirmed; After communicating with the project and feedback, the project stated that it will transfer the ownership to the timelock contract, and in order to deal with emergency situations, it will retain the right to perform any operation immediately without time delay within half a year of the transfer of the authority. Half a year after the transfer of ownership, the immediate execution interface of the timelock contract will no longer be available.

5 Audit Result

Audit Number	Audit Team	Audit Date	Audit Result
0X002110040001	SlowMist Security Team	2021.10.03 - 2021.10.04	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 vulnerability. And 1 medium risk vulnerabilities were confirmed and being fixed; All other findings were fixed. The code was not deployed to the mainnet. At present, the ownership of some contracts has not been transferred to timelock, so the project still has the risk of excessive authority.

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