Smart Contract Security Audit Report

[2021]

Table Of Contents

1 Executive Summary

On 2021.11.01, the SlowMist security team received the Alpaca Finance team's security audit application for AlpacaStablecoin, 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 Audit
- Design Logic Audit
- Scoping and Declarations Audit

3 Project Overview

3.1 Project Introduction

Audit Version:

https://github.com/alpaca-finance/alpaca-stablecoin

commit: 013108fd9faeca01a4293b7611563d8b7a86319f

Fixed Version:

https://github.com/alpaca-finance/alpaca-stablecoin/tree

commit:f107b48045d7dd2e6797f70e3625de0d1c01e50e

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	Ignored
N2	Miss event records	Others	Suggestion	Fixed
N3	Coding Standards issues	Others	Suggestion	Ignored
N4	Address management recommendations	Others	Suggestion	Ignored
N5	Miss range of values	Others	Suggestion	Ignored
N6	DOS issues	Others	Suggestion	Ignored
N7	Miss permission checks	Authority Control Vulnerability	Medium	Fixed
N8	Blank function	Others	Suggestion	Ignored
N9	Prices can be manipulated	Others	Medium	Ignored
N10	Possible risk of false top-up	"False top-up" Vulnerability	Medium	Fixed
N11	Miss resetSafeApprove	Design Logic Audit	Low	Fixed
N12	Miss a zero-check	Others	Suggestion	Fixed
N13	Excessive authority issue	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:

AlpacaToken			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
cap	Public	-	-
unlockedSupply	External	-	-
totalLock	Public	-	-
manualMint	Public	Can Modify State	onlyOwner
mint	Public	Can Modify State	onlyOwner
burn	External	Can Modify State	onlyOwner
transfer	Public	Can Modify State	-
transferFrom	Public	Can Modify State	-
totalBalanceOf	External	-	-
lockOf	External	-	-
lastUnlockBlock	External	-	-
lock	External	Can Modify State	onlyOwner
canUnlockAmount	Public	-	-

AlpacaToken			
unlock	External	Can Modify State	-
transferAll	External	Can Modify State	-
delegates	External	-	-
delegate	External	Can Modify State	-
delegateBySig	External	Can Modify State	-
getCurrentVotes	External	-	-
getPriorVotes	External	-	-
_delegate	Internal	Can Modify State	-
_moveDelegates	Internal	Can Modify State	-
_writeCheckpoint	Internal	Can Modify State	-
safe32	Internal	-	-
getChainId	Internal	-	-

Ownable			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Internal	Can Modify State	-
owner	Public	-	-
renounceOwnership	Public	Can Modify State	onlyOwner
transferOwnership	Public	Can Modify State	onlyOwner

Context			
Function Name	Visibility	Mutability	Modifiers
_msgSender	Internal	-	-
_msgData	Internal	-	-

ERC20			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
name	Public	-	-
symbol	Public	-	-
decimals	Public	-	-
totalSupply	Public	-	-
balanceOf	Public	-	-
transfer	Public	Can Modify State	-
allowance	Public	-	-
approve	Public	Can Modify State	-
transferFrom	Public	Can Modify State	-
increaseAllowance	Public	Can Modify State	-
decreaseAllowance	Public	Can Modify State	-
_transfer	Internal	Can Modify State	-
_mint	Internal	Can Modify State	-

ERC20			
_burn	Internal	Can Modify State	-
_approve	Internal	Can Modify State	-
_setupDecimals	Internal	Can Modify State	-
_beforeTokenTransfer	Internal	Can Modify State	-

FairLaunch			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
setDev	Public	Can Modify State	-
setAlpacaPerBlock	Public	Can Modify State	onlyOwner
setBonus	Public	Can Modify State	onlyOwner
addPool	Public	Can Modify State	onlyOwner
setPool	Public	Can Modify State	onlyOwner
isDuplicatedPool	Public	-	-
poolLength	External	-	-
manualMint	Public	Can Modify State	onlyOwner
getMultiplier	Public	-	-
pendingAlpaca	External	-	-
massUpdatePools	Public	Can Modify State	-
updatePool	Public	Can Modify State	-

FairLaunch			
deposit	Public	Can Modify State	-
withdraw	Public	Can Modify State	-
withdrawAll	Public	Can Modify State	-
_withdraw	Internal	Can Modify State	-
harvest	Public	Can Modify State	-
_harvest	Internal	Can Modify State	-
emergencyWithdraw	Public	Can Modify State	-
safeAlpacaTransfer	Internal	Can Modify State	-

ReentrancyGuard			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Internal	Can Modify State	-

Shield			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
setAlpacaPerBlock	External	Can Modify State	onlyOwner
setBonus	External	Can Modify State	onlyOwner
mintWarchest	External	Can Modify State	onlyOwner
addPool	External	Can Modify State	onlyOwner
setPool	External	Can Modify State	onlyOwner

FlashLoanReceiverBase			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
rad	Internal	-	-
add	Internal	-	-
mul	Internal	-	-
approvePayback	Internal	Can Modify State	-
payBackBookKeeper	Internal	Can Modify State	-

FlashMintModule			
Function Name	Visibility	Mutability	Modifiers
initialize	External	Can Modify State	initializer
_add	Internal	-	-
_mul	Internal	-	-
setMax	External	Can Modify State	onlyOwner
setFeeRate	External	Can Modify State	onlyOwner
maxFlashLoan	External	-	-
flashFee	External	-	-
flashLoan	External	Can Modify State	lock
bookKeeperFlashLoan	External	Can Modify State	lock

FlashMintModule			
convert	External	Can Modify State	lock
accrue	External	Can Modify State	lock

PausableUpgradeable			
Function Name	Visibility	Mutability	Modifiers
__Pausable_init	Internal	Can Modify State	initializer
__Pausable_init_unchained	Internal	Can Modify State	initializer
paused	Public	-	-
_pause	Internal	Can Modify State	whenNotPaused
_unpause	Internal	Can Modify State	whenPaused

ContextUpgradeable			
Function Name	Visibility	Mutability	Modifiers
__Context_init	Internal	Can Modify State	initializer
__Context_init_unchained	Internal	Can Modify State	initializer
_msgSender	Internal	-	-
_msgData	Internal	-	-

Initializable			
Function Name	Visibility	Mutability	Modifiers
_isConstructor	Private	-	-

AccessControlUpgradeable			
Function Name	Visibility	Mutability	Modifiers
__AccessControl_init	Internal	Can Modify State	initializer
__AccessControl_init_unchained	Internal	Can Modify State	initializer
hasRole	Public	-	-
getRoleMemberCount	Public	-	-
getRoleMember	Public	-	-
getRoleAdmin	Public	-	-
grantRole	Public	Can Modify State	-
revokeRole	Public	Can Modify State	-
renounceRole	Public	Can Modify State	-
_setupRole	Internal	Can Modify State	-
_setRoleAdmin	Internal	Can Modify State	-
_grantRole	Private	Can Modify State	-
_revokeRole	Private	Can Modify State	-

GetPositions			
Function Name	Visibility	Mutability	Modifiers
initialize	External	Can Modify State	initializer
getAllPositionsAsc	External	-	-
getPositionsAsc	External	-	-

GetPositions			
_getPositionsAsc	Internal	-	-
getAllPositionsDesc	External	-	-
getPositionsDesc	External	-	-
_getPositionsDesc	Internal	-	-
getPositionWithSafetyBuffer	External	-	-
calculateSafetyBuffer	Internal	-	-

PositionManager			
Function Name	Visibility	Mutability	Modifiers
initialize	External	Can Modify State	initializer
_safeAdd	Internal	-	-
_safeSub	Internal	-	-
_safeToInt	Internal	-	-
allowManagePosition	Public	Can Modify State	whenNotPaused onlyOwnerAllowed
allowMigratePosition	Public	Can Modify State	whenNotPaused
open	Public	Can Modify State	whenNotPaused
give	Public	Can Modify State	whenNotPaused onlyOwnerAllowed
adjustPosition	Public	Can Modify State	whenNotPaused onlyOwnerAllowed
moveCollateral	Public	Can Modify State	whenNotPaused onlyOwnerAllowed

PositionManager			
moveCollateral	Public	Can Modify State	whenNotPaused onlyOwnerAllowed
moveStablecoin	Public	Can Modify State	whenNotPaused onlyOwnerAllowed
exportPosition	Public	Can Modify State	whenNotPaused onlyOwnerAllowed onlyMigrationAllowed
importPosition	Public	Can Modify State	whenNotPaused onlyMigrationAllowed onlyOwnerAllowed
movePosition	Public	Can Modify State	whenNotPaused onlyOwnerAllowed onlyOwnerAllowed
pause	External	Can Modify State	-
unpause	External	Can Modify State	-
redeemLockedCollateral	Public	Can Modify State	whenNotPaused onlyOwnerAllowed

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

AlpacaOraclePriceFeed			
Function Name	Visibility	Mutability	Modifiers
initialize	External	Can Modify State	initializer
setPriceLife	External	Can Modify State	onlyOwner
pause	External	Can Modify State	onlyOwner
unpause	External	Can Modify State	onlyOwner

AlpacaOraclePriceFeed			
readPrice	External	-	-
peekPrice	External	-	-
_isPriceFresh	Internal	-	-
_isPriceOk	Internal	-	-

IbTokenPriceFeed			
Function Name	Visibility	Mutability	Modifiers
initialize	External	Can Modify State	initializer
pause	External	Can Modify State	onlyOwnerOrGov
unpause	External	Can Modify State	onlyOwnerOrGov
readPrice	External	-	-
peekPrice	External	-	-

SimplePriceFeed			
Function Name	Visibility	Mutability	Modifiers
initialize	External	Can Modify State	initializer
setPrice	External	Can Modify State	onlyOwner
setPriceLife	External	Can Modify State	onlyOwner
pause	External	Can Modify State	onlyOwner
unpause	External	Can Modify State	onlyOwner
readPrice	External	-	-

SimplePriceFeed			
peekPrice	External	-	-
_isPriceFresh	Internal	-	-
_isPriceOk	Internal	-	-

StrictAlpacaOraclePriceFeed			
Function Name	Visibility	Mutability	Modifiers
initialize	External	Can Modify State	initializer
setPriceLife	External	Can Modify State	onlyOwner
setMaxPriceDiff	External	Can Modify State	onlyOwner
pause	External	Can Modify State	onlyOwnerOrGov
unpause	External	Can Modify State	onlyOwnerOrGov
readPrice	External	-	-
peekPrice	External	-	-
_isPriceOk	Internal	-	-
_isPriceFresh	Internal	-	-
_isPriceStable	Internal	-	-

DexPriceOracle			
Function Name	Visibility	Mutability	Modifiers
initialize	External	Can Modify State	initializer
getPrice	External	-	-

AlpacaStablecoinProxyActions			
Function Name	Visibility	Mutability	Modifiers
_safeSub	Internal	-	-
_safeToInt	Internal	-	-
_safeMul	Internal	-	-
_toRad	Internal	-	-
convertTo18	Internal	Can Modify State	-
_getDrawDebtShare	Internal	Can Modify State	-
_getWipeDebtShare	Internal	-	-
_getWipeAllStablecoinAmount	Internal	-	-
stablecoinAdapterDeposit	Public	Can Modify State	-
transfer	Public	Can Modify State	-
bnbAdapterDeposit	Public	Payable	-
tokenAdapterDeposit	Public	Can Modify State	-
whitelist	External	Can Modify State	-
blacklist	External	Can Modify State	-
open	Public	Can Modify State	-
transferOwnership	Public	Can Modify State	-
transferOwnershipToProxy	External	Can Modify State	-
allowManagePosition	External	Can Modify State	-

AlpacaStablecoinProxyActions			
allowMigratePosition	External	Can Modify State	-
moveCollateral	Public	Can Modify State	-
moveStablecoin	Public	Can Modify State	-
adjustPosition	Public	Can Modify State	-
exportPosition	External	Can Modify State	-
importPosition	External	Can Modify State	-
movePosition	External	Can Modify State	-
bnbToIbBNB	Public	Payable	-
ibBNBToBNB	Public	Payable	-
tokenToIbToken	Public	Can Modify State	-
ibTokenToToken	Public	Can Modify State	-
lockBNB	Public	Payable	-
safeLockBNB	External	Payable	-
lockToken	Public	Can Modify State	-
safeLockToken	External	Can Modify State	-
unlockBNB	Public	Can Modify State	-
unlockToken	Public	Can Modify State	-
withdrawBNB	External	Can Modify State	-
withdrawToken	External	Can Modify State	-
draw	External	Can Modify State	-

AlpacaStablecoinProxyActions			
wipe	Public	Can Modify State	-
safeWipe	External	Can Modify State	-
wipeAll	Public	Can Modify State	-
safeWipeAll	External	Can Modify State	-
lockBNBAndDraw	Public	Payable	-
openLockBNBAndDraw	External	Payable	-
lockTokenAndDraw	Public	Can Modify State	-
openLockTokenAndDraw	Public	Can Modify State	-
convertAndLockToken	Public	Can Modify State	-
convertLockTokenAndDraw	Public	Can Modify State	-
convertBNBAndLockToken	Public	Payable	-
convertBNBLockTokenAndDraw	Public	Payable	-
convertBNBOpenLockTokenAndDraw	External	Payable	-
convertOpenLockTokenAndDraw	External	Can Modify State	-
wipeAndUnlockBNB	External	Can Modify State	-
wipeAllAndUnlockBNB	External	Can Modify State	-
wipeAndUnlockToken	Public	Can Modify State	-
wipeUnlockIbBNBAndCovertToBNB	External	Can Modify State	-
wipeUnlockTokenAndConvert	External	Can Modify State	-
wipeAllAndUnlockToken	Public	Can Modify State	-

AlpacaStablecoinProxyActions			
wipeAllUnlockIbBNBAndConvertToBNB	External	Can Modify State	-
wipeAllUnlockTokenAndConvert	External	Can Modify State	-
harvest	External	Can Modify State	-
redeemLockedCollateral	External	Can Modify State	-

AlpacaAuthEvents			
Function Name	Visibility	Mutability	Modifiers

AlpacaAuth			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
setOwner	Public	Can Modify State	auth
setAuthority	Public	Can Modify State	auth
isAuthorized	Internal	-	-

AlpacaNote			
Function Name	Visibility	Mutability	Modifiers

ProxyWallet			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
<Receive Ether>	External	Payable	-

ProxyWallet			
execute	Public	Payable	-
execute	Public	Payable	auth note
setCache	Public	Can Modify State	auth note

ProxyWalletCache			
Function Name	Visibility	Mutability	Modifiers
read	Public	-	-
write	Public	Can Modify State	-

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

ProxyWalletRegistry			
Function Name	Visibility	Mutability	Modifiers
initialize	External	Can Modify State	initializer
build	Public	Can Modify State	-
build	Public	Can Modify State	-
setOwner	Public	Can Modify State	-

OwnableUpgradeable			
Function Name	Visibility	Mutability	Modifiers
__Ownable_init	Internal	Can Modify State	initializer
__Ownable_init_unchained	Internal	Can Modify State	initializer
owner	Public	-	-
renounceOwnership	Public	Can Modify State	onlyOwner
transferOwnership	Public	Can Modify State	onlyOwner

IbTokenAdapter			
Function Name	Visibility	Mutability	Modifiers
initialize	External	Can Modify State	initializer
add	Internal	-	-
sub	Internal	-	-
mul	Internal	-	-
div	Internal	-	-
divup	Internal	-	-
wmul	Internal	-	-
wdiv	Internal	-	-
wdivup	Internal	-	-
rmul	Internal	-	-
rmulup	Internal	-	-

IbTokenAdapter			
rdiv	Internal	-	-
setTreasuryFeeBps	External	Can Modify State	onlyOwner
setTreasuryAccount	External	Can Modify State	onlyOwner
netAssetValuation	Public	-	-
netAssetPerShare	Public	-	-
_harvest	Internal	Can Modify State	-
harvest	Internal	Can Modify State	-
pendingRewards	External	-	-
_pendingRewards	Internal	-	-
deposit	External	Payable	nonReentrant whenNotPaused
_deposit	Private	Can Modify State	-
withdraw	External	Can Modify State	nonReentrant whenNotPaused
_withdraw	Private	Can Modify State	-
emergencyWithdraw	External	Can Modify State	nonReentrant whenNotPaused
_emergencyWithdraw	Private	Can Modify State	-
moveStake	External	Can Modify State	nonReentrant whenNotPaused
_moveStake	Private	Can Modify State	-
onAdjustPosition	External	Can Modify State	nonReentrant whenNotPaused
onMoveCollateral	External	Can Modify State	nonReentrant whenNotPaused
cage	External	Can Modify State	nonReentrant

IbTokenAdapter			
uncage	External	Can Modify State	-
pause	External	Can Modify State	-
unpause	External	Can Modify State	-

ReentrancyGuardUpgradeable			
Function Name	Visibility	Mutability	Modifiers
__ReentrancyGuard_init	Internal	Can Modify State	initializer
__ReentrancyGuard_init_unchained	Internal	Can Modify State	initializer

AuthTokenAdapter			
Function Name	Visibility	Mutability	Modifiers
initialize	External	Can Modify State	initializer
cage	External	Can Modify State	-
uncage	External	Can Modify State	-
mul	Internal	-	-
deposit	External	Can Modify State	nonReentrant whenNotPaused
withdraw	External	Can Modify State	nonReentrant whenNotPaused
pause	External	Can Modify State	-
unpause	External	Can Modify State	-

StablecoinAdapter			
Function Name	Visibility	Mutability	Modifiers
initialize	External	Can Modify State	initializer
cage	External	Can Modify State	-
uncage	External	Can Modify State	-
mul	Internal	-	-
deposit	External	Payable	nonReentrant whenNotPaused
withdraw	External	Can Modify State	nonReentrant whenNotPaused
pause	External	Can Modify State	-
unpause	External	Can Modify State	-

TokenAdapter			
Function Name	Visibility	Mutability	Modifiers
initialize	External	Can Modify State	initializer
cage	External	Can Modify State	-
uncage	External	Can Modify State	-
deposit	External	Payable	nonReentrant whenNotPaused
withdraw	External	Can Modify State	nonReentrant whenNotPaused
onAdjustPosition	External	Can Modify State	nonReentrant
onMoveCollateral	External	Can Modify State	nonReentrant
pause	External	Can Modify State	-

TokenAdapter			
unpause	External	Can Modify State	-

AccessControlConfig			
Function Name	Visibility	Mutability	Modifiers
initialize	External	Can Modify State	initializer

CollateralPoolConfig			
Function Name	Visibility	Mutability	Modifiers
collateralPools	External	-	-
initialize	External	Can Modify State	initializer
initCollateralPool	External	Can Modify State	onlyOwner
setPriceWithSafetyMargin	External	Can Modify State	-
setDebtCeiling	External	Can Modify State	onlyOwner
setDebtFloor	External	Can Modify State	onlyOwner
setPriceFeed	External	Can Modify State	onlyOwner
setLiquidationRatio	External	Can Modify State	onlyOwner
setStabilityFeeRate	External	Can Modify State	onlyOwner
setAdapter	External	Can Modify State	onlyOwner
setCloseFactorBps	External	Can Modify State	onlyOwner
setLiquidatorIncentiveBps	External	Can Modify State	onlyOwner
setTreasuryFeesBps	External	Can Modify State	onlyOwner

CollateralPoolConfig			
setTotalDebtShare	External	Can Modify State	-
setDebtAccumulatedRate	External	Can Modify State	-
setStrategy	External	Can Modify State	onlyOwner
updateLastAccumulationTime	External	Can Modify State	-
getTotalDebtShare	External	-	-
getDebtAccumulatedRate	External	-	-
getPriceWithSafetyMargin	External	-	-
getDebtCeiling	External	-	-
getDebtFloor	External	-	-
getPriceFeed	External	-	-
getLiquidationRatio	External	-	-
getStabilityFeeRate	External	-	-
getLastAccumulationTime	External	-	-
getAdapter	External	-	-
getCloseFactorBps	External	-	-
getLiquidatorIncentiveBps	External	-	-
getTreasuryFeesBps	External	-	-
getStrategy	External	-	-

FixedSpreadLiquidationStrategy			
Function Name	Visibility	Mutability	Modifiers
initialize	External	Can Modify State	initializer
mul	Internal	-	-
rdiv	Internal	-	-
setFlashLendingEnabled	External	Can Modify State	-
getFeedPrice	Internal	-	-
_calculateLiquidationInfo	Internal	-	-
execute	External	Can Modify State	nonReentrant whenNotPaused
pause	External	Can Modify State	-
unpause	External	Can Modify State	-

AlpacaStablecoin			
Function Name	Visibility	Mutability	Modifiers
add	Internal	-	-
sub	Internal	-	-
initialize	External	Can Modify State	initializer
transfer	External	Can Modify State	-
transferFrom	Public	Can Modify State	-
mint	External	Can Modify State	-
burn	External	Can Modify State	-

AlpacaStablecoin			
approve	External	Can Modify State	-
push	External	Can Modify State	-
pull	External	Can Modify State	-
move	External	Can Modify State	-
permit	External	Can Modify State	-

BookKeeper			
Function Name	Visibility	Mutability	Modifiers
pause	External	Can Modify State	-
unpause	External	Can Modify State	-
whitelist	External	Can Modify State	whenNotPaused
blacklist	External	Can Modify State	whenNotPaused
wish	Internal	-	-
initialize	External	Can Modify State	initializer
add	Internal	-	-
sub	Internal	-	-
mul	Internal	-	-
add	Internal	-	-
sub	Internal	-	-
mul	Internal	-	-

BookKeeper			
setTotalDebtCeiling	External	Can Modify State	-
setAccessControlConfig	External	Can Modify State	-
setCollateralPoolConfig	External	Can Modify State	-
cage	External	Can Modify State	-
uncage	External	Can Modify State	-
addCollateral	External	Can Modify State	nonReentrant whenNotPaused
moveCollateral	External	Can Modify State	nonReentrant whenNotPaused
moveStablecoin	External	Can Modify State	nonReentrant whenNotPaused
either	Internal	-	-
both	Internal	-	-
adjustPosition	External	Can Modify State	nonReentrant whenNotPaused
movePosition	External	Can Modify State	nonReentrant whenNotPaused
confiscatePosition	External	Can Modify State	nonReentrant whenNotPaused
settleSystemBadDebt	External	Can Modify State	nonReentrant whenNotPaused
mintUnbackedStablecoin	External	Can Modify State	nonReentrant whenNotPaused
accrueStabilityFee	External	Can Modify State	nonReentrant whenNotPaused

LiquidationEngine			
Function Name	Visibility	Mutability	Modifiers
initialize	External	Can Modify State	initializer

LiquidationEngine			
liquidate	External	Can Modify State	nonReentrant whenNotPaused
cage	External	Can Modify State	-
uncage	External	Can Modify State	-
pause	External	Can Modify State	-
unpause	External	Can Modify State	-

PriceOracle			
Function Name	Visibility	Mutability	Modifiers
initialize	External	Can Modify State	initializer
mul	Internal	-	-
rdiv	Internal	-	-
setStableCoinReferencePrice	External	Can Modify State	-
setPrice	External	Can Modify State	whenNotPaused
cage	External	Can Modify State	-
uncage	External	Can Modify State	-
pause	External	Can Modify State	-
unpause	External	Can Modify State	-

ShowStopper			
Function Name	Visibility	Mutability	Modifiers
initialize	External	Can Modify State	initializer

ShowStopper			
add	Internal	-	-
sub	Internal	-	-
mul	Internal	-	-
min	Internal	-	-
rmul	Internal	-	-
wdiv	Internal	-	-
setBookKeeper	External	Can Modify State	-
setLiquidationEngine	External	Can Modify State	-
setSystemDebtEngine	External	Can Modify State	-
setPriceOracle	External	Can Modify State	-
setCageCoolDown	External	Can Modify State	-
cage	External	Can Modify State	-
cage	External	Can Modify State	-
accumulateBadDebt	External	Can Modify State	-
redeemLockedCollateral	External	Can Modify State	-
finalizeDebt	External	Can Modify State	-
finalizeCashPrice	External	Can Modify State	-
accumulateStablecoin	External	Can Modify State	-
redeemStablecoin	External	Can Modify State	-

StabilityFeeCollector			
Function Name	Visibility	Mutability	Modifiers
initialize	External	Can Modify State	initializer
rpow	Internal	-	-
add	Internal	-	-
diff	Internal	-	-
rmul	Internal	-	-
setGlobalStabilityFeeRate	External	Can Modify State	-
setSystemDebtEngine	External	Can Modify State	-
collect	External	Can Modify State	whenNotPaused nonReentrant
_collect	Internal	Can Modify State	-
pause	External	Can Modify State	-
unpause	External	Can Modify State	-

StableSwapModule			
Function Name	Visibility	Mutability	Modifiers
initialize	External	Can Modify State	initializer
add	Internal	-	-
sub	Internal	-	-
mul	Internal	-	-
setFeeIn	External	Can Modify State	-

StableSwapModule			
setFeeOut	External	Can Modify State	-
whitelist	External	Can Modify State	-
blacklist	External	Can Modify State	-
swapTokenToStablecoin	External	Can Modify State	nonReentrant whenNotPaused
swapStablecoinToToken	External	Can Modify State	nonReentrant whenNotPaused
pause	External	Can Modify State	-
unpause	External	Can Modify State	-

SystemDebtEngine			
Function Name	Visibility	Mutability	Modifiers
initialize	External	Can Modify State	initializer
add	Internal	-	-
sub	Internal	-	-
min	Internal	-	-
withdrawCollateralSurplus	External	Can Modify State	-
withdrawStablecoinSurplus	External	Can Modify State	-
setSurplusBuffer	External	Can Modify State	whenNotPaused
settleSystemBadDebt	External	Can Modify State	whenNotPaused nonReentrant
cage	External	Can Modify State	-
uncage	External	Can Modify State	-

SystemDebtEngine			
pause	External	Can Modify State	-
unpause	External	Can Modify State	-

4.3 Vulnerability Summary

[N1] [Medium] Risk of excessive authority

Category: Authority Control Vulnerability

Content

The Owner role can use the burn function to destroy any user's tokens.

Code location:ALPACA-STABLECOIN/contracts/6.12/apis/alpaca/FairLaunch.sol #L133-L154

```
function addPool(
   uint256 _allocPoint,
   address _stakeToken,
   bool _withUpdate
 ) public override onlyOwner {
   if (_withUpdate) {
     massUpdatePools();
   }
   require(_stakeToken != address(0), "add: not stakeToken addr");
   require(!isDuplicatedPool(_stakeToken), "add: stakeToken dup");
   uint256 lastRewardBlock = block.number > startBlock ? block.number : startBlock;
   totalAllocPoint = totalAllocPoint.add(_allocPoint);
   poolInfo.push(
     PoolInfo({
       stakeToken: _stakeToken,
       allocPoint: _allocPoint,
       lastRewardBlock: lastRewardBlock,
       accAlpacaPerShare: 0,
       accAlpacaPerShareTilBonusEnd: 0
     })
   );
 }
```

Solution

It is recommended to transfer the ownership of the owner to community governance.

Status

Ignored; After communication, the project team said that the contract was not within the audit scope.

[N2] [Suggestion] Miss event records

Category: Others

Content

The function for modifying contract variables has no event record, which is not conducive to the review of community users.

Code location:ALPACA-STABLECOIN/contracts/6.12/proxy-wallet/ProxyWallet.sol #L71-L75

```
function setCache(address _cacheAddr) public auth note returns (bool) {
    require(_cacheAddr != address(0), "proxy-wallet-cache-address-required");
    cache = ProxyWalletCache(_cacheAddr); // overwrite cache
    return true;
  }
```

Code location:ALPACA-STABLECOIN/contracts/6.12/proxy-wallet/ProxyWalletRegistry.sol #L47-53

```
  function setOwner(address _newOwner) public {
    require(proxies[_newOwner] == ProxyWallet(0));
    ProxyWallet _proxy = proxies[msg.sender];
    require(_proxy.owner() == _newOwner);
    proxies[_newOwner] = _proxy;
    proxies[msg.sender] = ProxyWallet(0);
  }
```

Code location:ALPACA-STABLECOIN/contracts/6.12/stablecoin-core/BookKeeper.sol #L463-L468

```
  function settleSystemBadDebt(uint256 _value) external override nonReentrant
whenNotPaused {
```

```
    systemBadDebt[msg.sender] = sub(systemBadDebt[msg.sender], _value);
    stablecoin[msg.sender] = sub(stablecoin[msg.sender], _value);
    totalUnbackedStablecoin = sub(totalUnbackedStablecoin, _value);
    totalStablecoinIssued = sub(totalStablecoinIssued, _value);
  }
```

Code location:ALPACA-STABLECOIN/contracts/6.12/stablecoin-core/SystemDebtEngine.sol #L102-L106

```
  function settleSystemBadDebt(uint256 _value) external override whenNotPaused
nonReentrant {
    require(_value <= bookKeeper.stablecoin(address(this)),
"SystemDebtEngine/insufficient-surplus");
    require(_value <= bookKeeper.systemBadDebt(address(this)),
"SystemDebtEngine/insufficient-debt");
    bookKeeper.settleSystemBadDebt(_value);
  }
```

Code location:ALPACA-STABLECOIN/contracts/6.12/stablecoin-

core/adapters/FarmableTokenAdapter/IbTokenAdapter.sol #L187-L197

```
  function setTreasuryFeeBps(uint256 _treasuryFeeBps) external onlyOwner {
    require(live == 1, "IbTokenAdapter/not-live");
    require(_treasuryFeeBps <= 5000, "IbTokenAdapter/bad treasury fee bps");
    treasuryFeeBps = _treasuryFeeBps;
  }

  function setTreasuryAccount(address _treasuryAccount) external onlyOwner {
    require(live == 1, "IbTokenAdapter/not-live");
    require(_treasuryAccount != address(0), "IbTokenAdapter/bad treasury account");
    treasuryAccount = _treasuryAccount;
  }
```

Solution

It is recommended to add event records when modifying contract variables.

Status

Fixed; The project team added the corresponding event record.

[N3] [Suggestion] Coding Standards issues

Category: Others

Content

In the event of an emergency, users can withdraw their staked assets through the emergencyWithdraw function. However, during the withdrawal process, it first transfers the staked funds to the user through the safeTransfer function, and then sets user.amount and user.rewardDebt to 0, which does not comply with the Checks-Effects-Interactions principle.

Code location:ALPACA-STABLECOIN/contracts/6.12/apis/alpaca/FairLaunch.sol #L339-L346

```
function emergencyWithdraw(uint256 _pid) public {
  PoolInfo storage pool = poolInfo[_pid];
  UserInfo storage user = userInfo[_pid][msg.sender];
  IERC20(pool.stakeToken).safeTransfer(address(msg.sender), user.amount);
  emit EmergencyWithdraw(msg.sender, _pid, user.amount);
  user.amount = 0;
  user.rewardDebt = 0;
}
```

Solution

It is recommended to follow the Checks-Effects-Interactions principle and change the status first before performing the transfer operation.

Status

Ignored; After communication, the project team said that the contract was not within the audit scope.

[N4] [Suggestion] Address management recommendations

Category: Others

Content

Set the developer address, it is recommended to use multi-signature contract, so as to prevent the private key from

being stolen and causing the loss of the team's interests.

Code location:ALPACA-STABLECOIN/contracts/6.12/apis/alpaca/FairLaunch.sol #L109-L112

```
function setDev(address _devaddr) public {
  require(msg.sender == devaddr, "dev: wut?");
  devaddr = _devaddr;
}
```

Solution

It is recommended to use multi-signature contract.

Status

Ignored; After communication, the project team said that the contract was not within the audit scope.

[N5] [Suggestion] Miss range of values

Category: Others

Content

There is no limit to the value range when setting the alpacaPerBlock parameter.

Code location:ALPACA-STABLECOIN/contracts/6.12/apis/alpaca/FairLaunch.sol #L114-L116

```
function setAlpacaPerBlock(uint256 _alpacaPerBlock) public onlyOwner {
  alpacaPerBlock = _alpacaPerBlock;
}
```

Solution

It is recommended to add a range of values.

Status

Ignored; After communication, the project team said that the contract was not within the audit scope.

[N6] [Suggestion] DOS issues

Category: Others

Content

There is no maximum length limit in the for loop, so when the number of loops is too large, it will cause out of gas and then revert, causing the gas consumed by the previous operation to be wasted.

Code location:ALPACA-STABLECOIN/contracts/6.12/apis/alpaca/FairLaunch.sol #L178-L184

```
function isDuplicatedPool(address _stakeToken) public view returns (bool) {
  uint256 length = poolInfo.length;
  for (uint256 _pid = 0; _pid < length; _pid++) {
    if (poolInfo[_pid].stakeToken == _stakeToken) return true;
  }
  return false;
}
```

Code location:ALPACA-STABLECOIN/contracts/6.12/apis/alpaca/FairLaunch.sol #L221-L226

```
function massUpdatePools() public {
  uint256 length = poolInfo.length;
  for (uint256 pid = 0; pid < length; ++pid) {
    updatePool(pid);
  }
}
```

Solution

It is recommended to have a maximum number of times limit, or in the chain There must be a limit to avoid the waste of gas after DoS.

Status

Ignored; After communication, the project team said that the contract was not within the audit scope.

[N7] [Medium] Miss permission checks

Category: Authority Control Vulnerability

Content

These two functions did not perform permission checks.

Code location:ALPACA-STABLECOIN/contracts/6.12/stablecoin-core/adapters/FarmableTokenAdapter/IbTokenAdapter.sol #L369-L418

```
  function moveStake(
    address _source,
    address _destination,
    uint256 _share,
    bytes calldata _data
  ) external override nonReentrant whenNotPaused {
    _moveStake(_source, _destination, _share, _data);
  }

  /// @dev Move wad amount of staked balance from source to destination.
  /// Can only be moved if underlaying assets make sense.
  /// @param _source The address to be moved staked balance from
  /// @param _destination The address to be moved staked balance to
  /// @param _share The amount of staked balance to be moved
  function _moveStake(
    address _source,
    address _destination,
    uint256 _share,
    bytes calldata /* data */
  ) private {
    // 1. Update collateral tokens for source and destination
    uint256 _stakedAmount = stake[_source];
    stake[_source] = sub(_stakedAmount, _share);
    stake[_destination] = add(stake[_destination], _share);
    // 2. Update source's rewardDebt due to collateral tokens have
    // moved from source to destination. Hence, rewardDebt should be updated.
    // rewardDebtDiff is how many rewards has been paid for that share.
    uint256 _rewardDebt = rewardDebts[_source];
    uint256 _rewardDebtDiff = mul(_rewardDebt, _share) / _stakedAmount;
    // 3. Update rewardDebts for both source and destination
    // Safe since rewardDebtDiff <= rewardDebts[source]
    rewardDebts[_source] = _rewardDebt - _rewardDebtDiff;
    rewardDebts[_destination] = add(rewardDebts[_destination], _rewardDebtDiff);
    // 4. Sanity check.
    // - stake[source] must more than or equal to collateral + lockedCollateral that
source has
    // to prevent a case where someone try to steal stake from source
```

```
    // - stake[destination] must less than or eqal to collateral + lockedCollateral
that destination has
    // to prevent destination from claim stake > actual collateral that he has
    (uint256 _lockedCollateral, ) = bookKeeper.positions(collateralPoolId, _source);
    require(
      stake[_source] >= add(bookKeeper.collateralToken(collateralPoolId, _source),
_lockedCollateral),
      "IbTokenAdapter/stake[source] < collateralTokens + lockedCollateral"
    );
    (_lockedCollateral, ) = bookKeeper.positions(collateralPoolId, _destination);
    require(
      stake[_destination] <= add(bookKeeper.collateralToken(collateralPoolId,
_destination), _lockedCollateral),
      "IbTokenAdapter/stake[destination] > collateralTokens + lockedCollateral"
    );
    emit LogMoveStake(_source, _destination, _share);
  }
```

Solution

It is recommended to add permission checks.

Status

Fixed; The project team has added permission checks.

[N8] [Suggestion] Blank function

Category: Others

Content

These two functions do not have a function body.

Code location:ALPACA-STABLECOIN/contracts/6.12/stablecoin-core/adapters/TokenAdapter.sol #L133-L146

```
  function onAdjustPosition(
    address src,
    address dst,
    int256 collateralValue,
    int256 debtShare,
    bytes calldata data
  ) external override nonReentrant {}
```

```
  function onMoveCollateral(
    address src,
    address dst,
    uint256 wad,
    bytes calldata data
  ) external override nonReentrant {}
```

Solution

If there is no business need, it is recommended to delete it.

Status

Ignored; The projet team said:These functions are implemented due to the usage of IGenericTokenAdapter. They are intended to perform extra actions for that TokenAdapter. IbTokenAdapter has extra logic on this whilst TokenAdapter does not. Therefore, we intentionally left it blank.

[N9] [Medium] Prices can be manipulated

Category: Others

Content

The getprice function obtains the price on the chain in real time through the getreserves function, which will lead to malicious manipulation of the price by the attacker.

Attackers can change the number of tokens in the pool through swap or Flash loan, so as to manipulate the price and make the price reach the desired amount.

Code location:ALPACA-STABLECOIN/contracts/6.12/price-oracles/DexPriceOracle.sol #L38-L44

```
  function getPrice(address token0, address token1) external view override returns
(uint256, uint256) {
    if (token0 == token1) return (1e18, uint64(now));

    (uint256 r0, uint256 r1) = PancakeLibraryV2.getReserves(dexFactory, token0,
token1);
    uint256 price = r0.mul(1e18).div(r1);
```

```
    return (price, uint64(now));
}
```

Solution

It is recommended to use Chainlink to obtain the price.

Status

Ignored; The projet team said:We have decided to refrain from using prices from DEX at all. We will be using Chainlink and Band Protocol off-chain oracle as our go to. Chainlink will be our primary price source which is the price to be used. Band Protocol will be used as secondary price source to check against Chainlink for price anomaly.

[N10] [Medium] Possible risk of false top-up

Category: "False top-up" Vulnerability

Content

The code uses the transferFrom function called externally (see the slowmist note for specific location: //SlowMist//), but the return value of the function is not checked. If the externally called contract is a fake recharge contract, it will result in the borrowed money Not returned.

Code location:ALPACA-STABLECOIN/contracts/6.12/flash-mint/FlashMintModule.sol #L114-L142

```
function flashLoan(
    IERC3156FlashBorrower _receiver,
    address _token,
    uint256 _amount,
    bytes calldata _data
) external override lock returns (bool) {
    require(_token == address(stablecoin), "FlashMintModule/token-unsupported");
    require(_amount <= max, "FlashMintModule/ceiling-exceeded");

    uint256 _amt = _mul(_amount, RAY);
    uint256 _fee = _mul(_amount, feeRate) / WAD;
    uint256 _total = _add(_amount, _fee);

    bookKeeper.mintUnbackedStablecoin(address(this), address(this), _amt);
    stablecoinAdapter.withdraw(address(_receiver), _amount, abi.encode(0));
```

```
    emit LogFlashLoan(address(_receiver), _token, _amount, _fee);

    require(
      _receiver.onFlashLoan(msg.sender, _token, _amount, _fee, _data) ==
CALLBACK_SUCCESS,
      "FlashMintModule/callback-failed"
    );
//SlowMist//Issue location here.
    stablecoin.transferFrom(address(_receiver), address(this), _total); // The fee is
also enforced here
    stablecoinAdapter.deposit(address(this), _total, abi.encode(0));
    bookKeeper.settleSystemBadDebt(_amt);

    return true;
  }
```

Solution

It is recommended to check the return value of transferFrom or use safeTransferFrom.

Status

Fixed; The project team has used the safeTransferFrom.

[N11] [Low] Miss resetSafeApprove

Category: Design Logic Audit

Content

The fairlaunch role will be granted an authorization quota when it is initialized. According to common sense, fairlaunch will consume this quota every time a transfer is made. When this quota is exhausted, the contract will be unavailable.And there isn't any function in the code that can consume this authorized quota.

Code location:ALPACA-STABLECOIN/contracts/6.12/stablecoin-core/adapters/FarmableTokenAdapter/IbTokenAdapter.sol #L92-L140

```
  function initialize(
    address _bookKeeper,
    bytes32 _collateralPoolId,
    address _collateralToken,
```

```
    address _rewardToken,
    address _fairlaunch,
    uint256 _pid,
    address _shield,
    address _timelock,
    uint256 _treasuryFeeBps,
    address _treasuryAccount,
    address _positionManager
  ) external initializer {
    // 1. Initialized all dependencies
    PausableUpgradeable.__Pausable_init();
    ReentrancyGuardUpgradeable.__ReentrancyGuard_init();

    // 2. Sanity checks
    (address _stakeToken, , , , ) = IAlpacaFairLaunch(_fairlaunch).poolInfo(_pid);
    require(_stakeToken == _collateralToken, "IbTokenAdapter/collateralToken-not-
match");
    require(IAlpacaFairLaunch(_fairlaunch).alpaca() == _rewardToken,
"IbTokenAdapter/reward-token-not-match");
    require(IAlpacaFairLaunch(_fairlaunch).owner() == _shield,
"IbTokenAdapter/shield-not-match");
    require(IShield(_shield).owner() == _timelock, "IbTokenAdapter/timelock-not-
match");

    fairlaunch = IAlpacaFairLaunch(_fairlaunch);
    shield = IShield(_shield);
    timelock = ITimeLock(_timelock);
    pid = _pid;

    live = 1;

    bookKeeper = IBookKeeper(_bookKeeper);
    collateralPoolId = _collateralPoolId;
    collateralToken = _collateralToken;
    decimals = IToken(collateralToken).decimals();
    require(decimals <= 18, "IbTokenAdapter/decimals > 18");

    to18ConversionFactor = 10**(18 - decimals);
    toTokenConversionFactor = 10**decimals;
    rewardToken = IToken(_rewardToken);

    require(_treasuryAccount != address(0), "IbTokenAdapter/bad treasury account");
    treasuryFeeBps = _treasuryFeeBps;
    treasuryAccount = _treasuryAccount;
```

```
    positionManager = IManager(_positionManager);

    address(collateralToken).safeApprove(address(fairlaunch), uint256(-1));
  }
```

There is also an authorization quota here. If the externally called contract does not remake the quota, the quota will be exhausted and the contract cannot be used normally.

Code location:ALPACA-STABLECOIN/contracts/6.12/flash-mint/FlashMintModule.sol #L61-L72

```
  function initialize(address _stablecoinAdapter, address _systemDebtEngine) external
initializer {
    // 1. Initialized all dependencies
    PausableUpgradeable.__Pausable_init();

    bookKeeper = IBookKeeper(IStablecoinAdapter(_stablecoinAdapter).bookKeeper());
    stablecoinAdapter = IStablecoinAdapter(_stablecoinAdapter);
    stablecoin = IStablecoin(IStablecoinAdapter(_stablecoinAdapter).stablecoin());
    systemDebtEngine = _systemDebtEngine;

    bookKeeper.whitelist(_stablecoinAdapter);
    stablecoin.approve(_stablecoinAdapter, type(uint256).max);
  }
```

Code location:ALPACA-STABLECOIN/contracts/6.12/stablecoin-core/StableSwapModule.sol #L51-L68

```
  function initialize(
    address _authTokenAdapter,
    address _stablecoinAdapter,
    address _systemDebtEngine
  ) external initializer {
    PausableUpgradeable.__Pausable_init();
    ReentrancyGuardUpgradeable.__ReentrancyGuard_init();

    IAuthTokenAdapter __authTokenAdapter = authTokenAdapter =
IAuthTokenAdapter(_authTokenAdapter);
    IStablecoinAdapter __stablecoinAdapter = stablecoinAdapter =
IStablecoinAdapter(_stablecoinAdapter);
    IBookKeeper _bookKeeper = bookKeeper =
```

```
IBookKeeper(address(__authTokenAdapter.bookKeeper()));
    IStablecoin _stablecoin = stablecoin =
IStablecoin(address(__stablecoinAdapter.stablecoin()));
    collateralPoolId = __authTokenAdapter.collateralPoolId();
    systemDebtEngine = _systemDebtEngine;
    to18ConversionFactor = 10**(18 - __authTokenAdapter.decimals());
    _stablecoin.approve(_stablecoinAdapter, uint256(-1));
    _bookKeeper.whitelist(_stablecoinAdapter);
  }
```

Solution

Should add a function similar to resetApprove to reset the allowance, so that when the allowance is insufficient, calling this function will reset the allowance to the maximum value again.

Status

Fixed; The project team has added the refreshApproval function to reset the allowance.

[N12] [Suggestion] Miss a zero-check

Category: Others

Content

The _systemDebtEngine address is passed in during initialization, and it is used directly without verifying whether the address is 0.

Code location:ALPACA-STABLECOIN/contracts/6.12/flash-mint/FlashMintModule.sol #L61-L72

```
  function initialize(address _stablecoinAdapter, address _systemDebtEngine) external
initializer {
    // 1. Initialized all dependencies
    PausableUpgradeable.__Pausable_init();

    bookKeeper = IBookKeeper(IStablecoinAdapter(_stablecoinAdapter).bookKeeper());
    stablecoinAdapter = IStablecoinAdapter(_stablecoinAdapter);
    stablecoin = IStablecoin(IStablecoinAdapter(_stablecoinAdapter).stablecoin());
    systemDebtEngine = _systemDebtEngine;

    bookKeeper.whitelist(_stablecoinAdapter);
```

```
    stablecoin.approve(_stablecoinAdapter, type(uint256).max);
  }
```

Solution

It is recommended to add the zero-check.

Status

Fixed; The project team has added the zero-check.

[N13] [Medium] Excessive authority issue

Category: Authority Control Vulnerability

Content

The owner can change the source contract of the price feed. If the owner's private key is stolen, the attacker can manipulate the price by modifying the malicious price feed contract.

Code location:ALPACA-STABLECOIN/contracts/6.12/stablecoin-core/config/CollateralPoolConfig.sol #L134-L137

```
function setPriceFeed(bytes32 _poolId, address _priceFeed) external onlyOwner {
    _collateralPools[_poolId].priceFeed = _priceFeed;
    emit LogSetPriceFeed(msg.sender, _poolId, _priceFeed);
  }
```

Solution

It is recommended to set the Owner as a governance contract and add a timelock mechanism, or set the Owner authority to a multi-signature contract to avoid a single private key management.

Status

Confirmed; The project team said that the Owner authority will be transferred to the TimeLock contract.

5 Audit Result

Audit Number	Audit Team	Audit Date	Audit Result
0X002111260001	SlowMist Security Team	2021.11.01 - 2021.11.26	Medium Risk

Summary conclusion: The SlowMist security team uses a manual and SlowMist team's analysis tool to audit the project, during the audit work we found 5 medium risk, 1 low risk, 7 suggestion vulnerabilities. And 2 medium risk, 5 suggestion vulnerabilities were ignored; 1 medium risk(N13) has been confirmed; All other findings were fixed. The code was not deployed to the mainnet.

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