Smart Contract
Security Audit Report

Table Of Contents

1 Executive Summary

On 2022.02.28, the SlowMist security team received the Starcrazy team's security audit application for Starcrazy, 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:

Project address:

https://github.com/GameFantasyDev/starcrazy-contracts

commit:

e9e11d234ac065726e108a73dfcd5efbad26f2c5

3.2 Vulnerability Information

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

NO	Title	Category	Level	Status
N1	Gas optimization	Others	Suggestion	Ignored
N2	Overflow reminder	Others	Suggestion	Ignored
N3	Hardcoded reminders	Others	Suggestion	Ignored
N4	Return value not checked	Others	Suggestion	Ignored
N5	Authority control issue	Authority Control Vulnerability	Low	Confirmed
N6	lack of authorization	Authority Control Vulnerability	Low	Confirmed
N7	Business logic issue	Design Logic Audit	Low	Confirmed
N8	Redundant code	Others	Suggestion	Ignored
N9	Parameter modification issue	Authority Control Vulnerability	Low	Confirmed
N10	Missing event record	Others	Suggestion	Ignored

4 Code Overview

4.1 Contracts Description

The main network address of the contract is as follows:

Aliana: io1asxdtswkr9p6r9du57ecrhrql865tf2qxue6hw

AlianaMint: io1mt5u8vzxzuwq7hut99t9tvduhykzgkfclwlsgj

AlianaSale: io14qqv4xz8jzk3hexhmp9qdtdghdpamvyzuqajrd

Auction: io1t7pkdvadx2mrnfukzvhsr0xhc2nsjuq9sren7p

DSAuthority: io12xh8ckeu5zt4ha9hv090r4yphpvhzfs42q9n65

FakeAliana: io1hsjyxj7j37gd9hre90eu6kfs28clc8tg7h9a0p

FlashSale: io1khx3s43tpszzpl9dvj69w87ea3mgyzlyzh4nax

GeneScienceSafe: io16t7hqruawrqllqvf7pqkekddvqezlc7483wqp2

GFS: io1t585efypl4e9ex7xks2upnjm8saawfk0v0s5sj

GFSBonus: io1gdr2vxxzu075e75zr6gjzm40jfaagmwae2pmpa

GFSMint: io195akzk9dus8c624zfzahnxyuypr60fw48r2fgy

GFT: io1zl0el07pek4sly8dmscccnm0etd8xr8j02t4y7

LPMint: io1mev3f2tuc5rxw5tzfuzn6uvlv7jddyurjake79

Timelock: io1eswwxp0n6zcf3mujv8npsncnvhcqulmmw3v2x6

WGFT: io1dxrdaqzw20d6rss7yce035uzs32etrnlmcpq7l

4.2 Visibility Description

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

AlianaBase			
Function Name	Visibility	Mutability	Modifiers
transferFrom	Public	Can Modify State	whenNotPaused
safeTransferFrom	Public	Can Modify State	whenNotPaused
safeTransferFrom	Public	Can Modify State	whenNotPaused
setGeneScienceAddress	Public	Can Modify State	onlyCEO
getGeneScienceAddress	External	-	onlyCLevelOrWhitelisted
isAliana	External	-	-
<Constructor>	Public	Can Modify State	ERC721Full
_createAliana	Internal	Can Modify State	-
getAliana	External	-	-

FlashSaleBase			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
updateMaxBiddingNum	Public	Can Modify State	onlyCEO
getStartingPrice	Public	-	-
setStartingPrice	Public	Can Modify State	onlyCEO
getCycleBlock	Public	-	-
setCycleBlock	Public	Can Modify State	onlyCEO

FlashSaleBase			
getStartBlockOffset	Public	-	-
setStartBlockOffset	Public	Can Modify State	onlyCEO
getMaxCycle	Public	-	-
setMaxCycle	Public	Can Modify State	onlyCEO
getDuration	Public	-	-
setDuration	Public	Can Modify State	onlyCEO
getLatestActivity	Public	-	-
isNowInActivity	Public	-	-
getCurrentCycleNum	Public	-	-
getMinAddPrice	Public	-	-
setMinAddPrice	Public	Can Modify State	onlyCEO
minAddPrice	Public	-	-
_updateBidding	Internal	Can Modify State	-
_isBidding	Internal	-	-
_isBiddingNoCheck	Internal	-	-
_biddingNum	Internal	-	-
_biddingIdList	Internal	-	-
_getGene	Internal	-	-
_getBiddingGene	Internal	-	-
_getBiddingGeneNoCheck	Internal	-	-

FlashSaleBase			
_tokensOfOwnerAuctionOn	Internal	-	-
_hasAuction	Internal	-	-
_hasAuctionByInfo	Internal	-	-
_bidFrom	Internal	Can Modify State	whenNotPaused
_takeBid	Internal	Can Modify State	-
_takeBids	Internal	Can Modify State	-
_takeBids256	Internal	Can Modify State	-
_addAuction	Internal	Can Modify State	whenNotPaused
_isOnAuction	Internal	-	-

FlashSale			
Function Name	Visibility	Mutability	Modifiers
isFlashSale	Public	-	-
<Constructor>	Public	Can Modify State	FlashSaleBase
getAuction	External	-	-
getCurrentPrice	External	-	-
updateBidding	External	Can Modify State	-
biddingNum	External	-	-
biddingIdList	External	-	-
getBiddingGene	External	-	-

FlashSale			
tokensOfOwnerAuction	External	-	-
tokensOfOwnerAuctionOn	External	-	-
bid	External	Can Modify State	-
takeBid	External	Can Modify State	-
takeBids	External	Can Modify State	-
setTokenGene	External	Can Modify State	onlyCEO
unsetTokenGene	External	Can Modify State	onlyCEO
takeMyBids	External	Can Modify State	-
takeBidsOf	External	Can Modify State	-
receiveApproval	Public	Can Modify State	-

GFAccessControl			
Function Name	Visibility	Mutability	Modifiers
addAddressToWhitelist	External	Can Modify State	onlyCEO
_addAddressToWhitelist	Private	Can Modify State	onlyCEO
addAddressesToWhitelist	External	Can Modify State	onlyCEO
removeAddressFromWhitelist	External	Can Modify State	onlyCEO
_removeAddressFromWhitelist	Private	Can Modify State	onlyCEO
removeAddressesFromWhitelist	External	Can Modify State	onlyCEO
<Constructor>	Public	Can Modify State	-

GFAccessControl			
setCandidateCEO	External	Can Modify State	onlyCEO
acceptCEO	External	Can Modify State	-
setCFO	External	Can Modify State	onlyCEO
setCOO	External	Can Modify State	onlyCEO
pause	External	Can Modify State	onlyCEO whenNotPaused
unpause	Public	Can Modify State	onlyCEO whenPaused
setNewAddress	External	Can Modify State	onlyCEO whenPaused
claimTokens	External	Can Modify State	onlyCEO
withdrawTokens	External	Can Modify State	onlyCEO

IAliana			
Function Name	Visibility	Mutability	Modifiers
createOfficialAliana	Public	Can Modify State	-
createOfficialAliana	Public	Can Modify State	-
burn	External	Can Modify State	-
isAliana	Public	Can Modify State	-
geneLpLabor	Public	-	-
geneLpLabors	Public	-	-
tokensOfOwner	External	-	-
getAliana	External	-	-

AuctionOwner			
Function Name	Visibility	Mutability	Modifiers
_tokensOfOwnerAuction	Internal	-	-
_addTokenToOwnerEnumerationAuction	Internal	Can Modify State	-
_removeTokenFromOwnerEnumerationAuction	Internal	Can Modify State	-

IGeneScience			
Function Name	Visibility	Mutability	Modifiers
isGeneScience	Public	-	-
mixGenes	Public	Can Modify State	-
mixGenesBySeed	Public	-	-
geneLpLabor	Public	-	-
isValid	Public	-	-
totalQuality	Public	-	-
getAuctionGene	Public	-	-

ISaleClockAuction			
Function Name	Visibility	Mutability	Modifiers
isAuction	Public	-	-
createAuction	External	Can Modify State	-
claimTokens	Public	Can Modify State	-

AlianaOwnership			
Function Name	Visibility	Mutability	Modifiers
totalAlianaSupply	Public	-	-
tokensOfOwner	External	-	-
approveAndCall	Public	Can Modify State	-
setApprovalForAllAndCall	Public	Can Modify State	-

IApproveAndCallFallBack			
Function Name	Visibility	Mutability	Modifiers
receiveApproval	Public	Can Modify State	-

IAlianaMint			
Function Name	Visibility	Mutability	Modifiers
depositedTokens	Public	-	-

IAlianaSale			
Function Name	Visibility	Mutability	Modifiers
getAlianaSaleInfo	External	-	-
allTokensSale	External	-	-
listTokensSale	External	-	-
tokensOfOwnerSale	External	-	-
totalSale	Public	-	-

IAuction			
Function Name	Visibility	Mutability	Modifiers
isAuction	Public	Can Modify State	-
getAuction	External	-	-
biddingIdList	External	-	-
tokensOfOwnerAuctionOn	External	-	-
takeBid	External	Can Modify State	-

IFlashSale			
Function Name	Visibility	Mutability	Modifiers
isFlashSale	Public	Can Modify State	-
getAuction	External	-	-
biddingIdList	External	-	-
tokensOfOwnerAuctionOn	External	-	-
takeBid	External	Can Modify State	-

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

NFTokenEnumerable			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
totalSupply	External	-	-
tokenByIndex	External	-	-
tokenOfOwnerByIndex	External	-	-
_mint	Internal	Can Modify State	-
_burn	Internal	Can Modify State	-
_removeNFToken	Internal	Can Modify State	-
_addNFToken	Internal	Can Modify State	-
_getOwnerNFTCount	Internal	-	-

NFToken			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
safeTransferFrom	External	Can Modify State	-
safeTransferFrom	External	Can Modify State	-
transferFrom	External	Can Modify State	canTransfer validNFToken
approve	External	Can Modify State	canOperate validNFToken
setApprovalForAll	External	Can Modify State	-
balanceOf	External	-	-

NFToken			
ownerOf	External	-	-
getApproved	External	-	validNFToken
isApprovedForAll	External	-	-
_transfer	Internal	Can Modify State	-
_mint	Internal	Can Modify State	-
_burn	Internal	Can Modify State	validNFToken
_removeNFToken	Internal	Can Modify State	-
_addNFToken	Internal	Can Modify State	-
_getOwnerNFTCount	Internal	-	-
_safeTransferFrom	Private	Can Modify State	canTransfer validNFToken
_clearApproval	Private	Can Modify State	-

SupportsInterface			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
supportsInterface	External	-	-

NFTokenMetadata			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
name	External	-	-

NFTokenMetadata			
symbol	External	-	-
tokenURI	External	-	validNFToken
_burn	Internal	Can Modify State	-
_setTokenUri	Internal	Can Modify State	validNFToken

Pausable			
Function Name	Visibility	Mutability	Modifiers
pause	Public	Can Modify State	onlyOwner whenNotPaused
unpause	Public	Can Modify State	onlyOwner whenPaused

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

DSAuthEvents			
Function Name	Visibility	Mutability	Modifiers

DSAuth			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
setOwner	Public	Can Modify State	auth

DSAuth			
setAuthority	Public	Can Modify State	auth
isAuthorized	Internal	-	-

Whitelist			
Function Name	Visibility	Mutability	Modifiers
addAddressToWhitelist	Public	Can Modify State	onlyOwner
addAddressesToWhitelist	Public	Can Modify State	onlyOwner
removeAddressFromWhitelist	Public	Can Modify State	onlyOwner
removeAddressesFromWhitelist	Public	Can Modify State	onlyOwner

BasicToken			
Function Name	Visibility	Mutability	Modifiers
totalSupply	Public	-	-
transfer	Public	Can Modify State	-
balanceOf	Public	-	-

IERC20Basic			
Function Name	Visibility	Mutability	Modifiers
totalSupply	Public	-	-
balanceOf	Public	-	-
transfer	Public	Can Modify State	-

DSToken			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
name	Public	-	-
symbol	Public	-	-
decimals	Public	-	-
totalSupply	Public	-	-
balanceOf	Public	-	-
allowance	Public	-	-
approve	External	Can Modify State	-
approve	Public	Can Modify State	stoppable
transfer	External	Can Modify State	-
transferFrom	Public	Can Modify State	stoppable
push	External	Can Modify State	-
pull	External	Can Modify State	-
move	External	Can Modify State	-
mint	External	Can Modify State	-
burn	External	Can Modify State	-
mint	Public	Can Modify State	auth stoppable
burn	Public	Can Modify State	auth stoppable
destroy	Public	Can Modify State	auth stoppable

DSToken			
dsStop	Public	Can Modify State	auth
start	Public	Can Modify State	auth

IERC20			
Function Name	Visibility	Mutability	Modifiers
name	External	-	-
symbol	External	-	-
allowance	Public	-	-
transferFrom	Public	Can Modify State	-
approve	Public	Can Modify State	-

IMintableToken			
Function Name	Visibility	Mutability	Modifiers
mint	External	Can Modify State	-
burn	External	Can Modify State	-

ITokenController			
Function Name	Visibility	Mutability	Modifiers
proxyPayment	Public	Payable	-
onTransfer	Public	Can Modify State	-
onApprove	Public	Can Modify State	-

ShadowToken			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
mint	Public	Can Modify State	onlyMinter whenNotPaused
burn	Public	Can Modify State	-

StandardToken			
Function Name	Visibility	Mutability	Modifiers
transferFrom	Public	Can Modify State	-
approve	Public	Can Modify State	-
allowance	Public	-	-
increaseApproval	Public	Can Modify State	-
decreaseApproval	Public	Can Modify State	-

Aliana			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
mix	External	Can Modify State	whenNotPaused
burn	External	Can Modify State	-
geneLpLabor	Public	-	-
geneLpLabors	External	-	-
createOfficialAliana	External	Can Modify State	onlyWhitelisted

Aliana			
createOfficialAliana	External	Can Modify State	onlyWhitelisted
createPromoAliana	External	Can Modify State	onlyCLevel

AlianaMinting			
Function Name	Visibility	Mutability	Modifiers
insert	Internal	Can Modify State	-
remove	Internal	Can Modify State	-
contains	Internal	-	-
iterate_start	Internal	-	-
iterate_valid	Internal	-	-
iterate_next	Internal	-	-
iterate_get	Internal	-	-
<Constructor>	Public	Can Modify State	-
setMaxMintingNumPerAddress	Public	Can Modify State	onlyCEO
setRewardPerBlock	Public	Can Modify State	onlyCEO
pendingReward	External	-	-
rewardPending	Internal	-	-
updateBlockReward	Public	Can Modify State	-
_depositFrom	Internal	Can Modify State	whenNotPaused
deposit	Public	Can Modify State	whenNotPaused

AlianaMinting			
deposits	Public	Can Modify State	whenNotPaused
withdrawPending	Public	Can Modify State	-
withdraw	Public	Can Modify State	-
_withdrawFrom	Internal	Can Modify State	-
withdraws	Public	Can Modify State	-
withdrawsByCEO	Public	Can Modify State	onlyCEO
emergencyWithdraw	Public	Can Modify State	-
emergencyWithdrawByCEO	Public	Can Modify State	onlyCEO
_emergencyWithdrawFrom	Internal	Can Modify State	-
safeGFTTransfer	Internal	Can Modify State	-
depositedTokens	Public	-	-
depositedBalanceOf	Public	-	-
receiveApproval	Public	Can Modify State	-

AlianaSale			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
setOwnerCutSale	External	Can Modify State	onlyCEO
withdrawOwnerCutSale	External	Can Modify State	onlyCLevel
getAlianaSaleInfo	External	-	-

AlianaSale			
createAlianaSale	External	Can Modify State	whenNotPaused
_createAlianaSaleFrom	Internal	Can Modify State	whenNotPaused
adminCancelAlianaSales	Public	Can Modify State	onlyCLevel
cancelAlianaSale	External	Can Modify State	-
_cancelAlianaSaleFrom	Internal	Can Modify State	-
buySaleAliana	External	Can Modify State	whenNotPaused
_buySaleAlianaFrom	Internal	Can Modify State	whenNotPaused
_computeCutSale	Internal	-	-
_burnSale	Internal	Can Modify State	-
_isOnSale	Internal	-	-
totalSale	Public	-	-
allTokensSale	External	-	-
listTokensSale	External	-	-
tokensOfOwnerSale	External	-	-
_tokensOfOwnerSale	Internal	-	-
_addTokenToOwnerEnumerationSale	Private	Can Modify State	-
_addTokenToAllTokensEnumerationSale	Private	Can Modify State	-
_removeTokenFromOwnerEnumerationSale	Private	Can Modify State	-
_removeTokenFromAllTokensEnumerationSale	Private	Can Modify State	-
receiveApproval	Public	Can Modify State	-

ClockAuctionBase			
Function Name	Visibility	Mutability	Modifiers
setGeneScienceAddress	Public	Can Modify State	onlyCEO
<Constructor>	Public	Can Modify State	-
updateMaxBiddingNum	Public	Can Modify State	onlyCEO
getStartingPrice	Public	-	-
setStartingPrice	Public	Can Modify State	onlyCEO
getDuration	Public	-	-
setDuration	Public	Can Modify State	onlyCEO
getMinAddPrice	Public	-	-
setMinAddPrice	Public	Can Modify State	onlyCEO
minAddPrice	Public	-	-
_updateBidding	Internal	Can Modify State	-
_isBidding	Internal	-	-
_isBiddingNoCheck	Internal	-	-
_biddingNum	Internal	-	-
_biddingIdList	Internal	-	-
_getGene	Internal	-	-
_getBiddingGene	Internal	-	-
_getBiddingGeneNoCheck	Internal	-	-

ClockAuctionBase			
_tokensOfOwnerAuctionOn	Internal	-	-
_hasAuction	Internal	-	-
_hasAuctionByInfo	Internal	-	-
_bidFrom	Internal	Can Modify State	whenNotPaused
_takeBid	Internal	Can Modify State	-
_takeBids	Internal	Can Modify State	-
_takeBids256	Internal	Can Modify State	-
_addAuction	Internal	Can Modify State	whenNotPaused
_isOnAuction	Internal	-	-

Auction			
Function Name	Visibility	Mutability	Modifiers
isAuction	Public	-	-
<Constructor>	Public	Can Modify State	ClockAuctionBase
getAuction	External	-	-
getCurrentPrice	External	-	-
updateBidding	External	Can Modify State	-
biddingNum	External	-	-
biddingIdList	External	-	-
getBiddingGene	External	-	-

Auction			
tokensOfOwnerAuction	External	-	-
tokensOfOwnerAuctionOn	External	-	-
bid	External	Can Modify State	-
takeBid	External	Can Modify State	-
takeBids	External	Can Modify State	-
takeMyBids	External	Can Modify State	-
takeBidsOf	External	Can Modify State	-
receiveApproval	Public	Can Modify State	-

DSAuthority			
Function Name	Visibility	Mutability	Modifiers
canCall	External	-	-

FakeAliana			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
_mixFrom	Internal	Can Modify State	whenNotPaused
mix	Public	Can Modify State	-
haveFake	Public	-	-
receiveApproval	Public	Can Modify State	-

GeneMath			
Function Name	Visibility	Mutability	Modifiers
_newSeedWithBlock	Internal	-	-
_random	Internal	-	-
_random3	Internal	-	-
sliceUint8	Internal	-	-
isCollections	Public	-	-
getTargetQualityList	Public	-	-
_randQuality	Internal	-	-
_randStyle	Internal	-	-
mixGenesBySeed	Public	-	-
_mixGenesBySeed	Internal	-	-
geneLpLaborDetail	Public	-	-
geneLpLabor	Public	-	-
totalQuality	Public	-	-
isValid	Public	-	-
getAuctionGene	Public	-	-

GeneScienceTask			
Function Name	Visibility	Mutability	Modifiers
mixTaskOf	Public	-	-

GeneScienceTask			
mixTask	Public	-	-
cleanAddrExpiredTasks	Public	Can Modify State	-
addrInProgressTasks	Public	-	-
_addTaskToOwnerEnumeration	Internal	Can Modify State	-
_removeTaskToOwnerEnumeration	Internal	Can Modify State	-

GeneScienceSafe			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
setMathGene	Public	Can Modify State	onlyCEO
isGeneScience	Public	-	-
mixGenes	Public	-	-
trySeedAddrMixTask	External	Can Modify State	-
trySeedAddrMixTask	External	Can Modify State	-
trySeedAddrMixTask	External	Can Modify State	-
_trySeedAddrMixTask	Internal	Can Modify State	-
_trySeedAddrMixTask	Internal	Can Modify State	-
newMixTask	External	Can Modify State	-
newMixTask	External	Can Modify State	-
_newMixTask	Internal	Can Modify State	whenNotPaused

GeneScienceSafe			
doneMixTask	External	Can Modify State	-
doneMixTask	External	Can Modify State	-
doneMixTask	External	Can Modify State	-
_doneMixTask	Internal	Can Modify State	-
_doneMixTask	Internal	Can Modify State	-

GFS			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
transferOwnership	Public	Can Modify State	auth
acceptOwnership	Public	Can Modify State	-
disableTransfers	Public	Can Modify State	auth
issue	Public	Can Modify State	auth stoppable
cap	Public	-	-
changeCap	Public	Can Modify State	auth
changeController	Public	Can Modify State	auth
transferFrom	Public	Can Modify State	transfersAllowed
transferFrom	Public	Can Modify State	transfersAllowed
transfer	Public	Can Modify State	-
approve	Public	Can Modify State	-

GFS			
mint	Public	Can Modify State	auth stoppable
burn	Public	Can Modify State	auth stoppable
approveAndCall	Public	Can Modify State	-
isContract	Internal	-	-
<Fallback>	External	Payable	-
claimTokens	Public	Can Modify State	auth
withdrawTokens	Public	Can Modify State	auth

GFSBonus			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
setRewardPerBlock	Public	Can Modify State	onlyCEO
pendingReward	External	-	-
getReward	Public	-	-
addReward	Public	Can Modify State	onlyCLevelOrWhitelisted
updateBlockReward	Public	Can Modify State	-
deposit	Public	Can Modify State	-
_depositFrom	Internal	Can Modify State	whenNotPaused
withdraw	Public	Can Modify State	whenNotPaused
emergencyWithdraw	Public	Can Modify State	-

GFSBonus			
sendWithdraw	Internal	Can Modify State	-
receiveApproval	Public	Can Modify State	-

GFSMint			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
setRewardPerBlock	Public	Can Modify State	onlyCEO
pendingReward	External	-	-
addReward	Public	Can Modify State	onlyWhitelisted
updateBlockReward	Public	Can Modify State	-
deposit	Public	Can Modify State	-
_depositFrom	Internal	Can Modify State	whenNotPaused
withdraw	Public	Can Modify State	whenNotPaused
emergencyWithdraw	Public	Can Modify State	-
safeMintTransfer	Internal	Can Modify State	-
receiveApproval	Public	Can Modify State	-

GFT			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
transferOwnership	Public	Can Modify State	auth

GFT			
acceptOwnership	Public	Can Modify State	-
disableTransfers	Public	Can Modify State	auth
issue	Public	Can Modify State	auth stoppable
cap	Public	-	-
changeController	Public	Can Modify State	auth
transferFrom	Public	Can Modify State	transfersAllowed
transferFrom	Public	Can Modify State	transfersAllowed
transfer	Public	Can Modify State	-
approve	Public	Can Modify State	-
mint	Public	Can Modify State	auth stoppable
burn	Public	Can Modify State	auth stoppable
approveAndCall	Public	Can Modify State	-
isContract	Internal	-	-
<Fallback>	External	Payable	-
claimTokens	Public	Can Modify State	auth
withdrawTokens	Public	Can Modify State	auth

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

LPMint			
setRewardPerBlock	Public	Can Modify State	onlyCEO
pendingReward	External	-	-
addReward	Public	Can Modify State	onlyWhitelisted
updateBlockReward	Public	Can Modify State	-
deposit	Public	Can Modify State	-
_depositFrom	Internal	Can Modify State	whenNotPaused
withdraw	Public	Can Modify State	whenNotPaused
emergencyWithdraw	Public	Can Modify State	-
safeMintTransfer	Internal	Can Modify State	-
receiveApproval	Public	Can Modify State	-

Timelock			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
<Fallback>	External	Payable	-
setDelay	Public	Can Modify State	-
acceptAdmin	Public	Can Modify State	-
setPendingAdmin	Public	Can Modify State	-
queueTransaction	Public	Can Modify State	-
cancelTransaction	Public	Can Modify State	-

Timelock			
executeTransaction	Public	Payable	-
getBlockTimestamp	Internal	-	-

WGFT			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	-
transferOwnership	Public	Can Modify State	auth
acceptOwnership	Public	Can Modify State	-
disableTransfers	Public	Can Modify State	auth
issue	Public	Can Modify State	auth stoppable
cap	Public	-	-
changeCap	Public	Can Modify State	auth
changeController	Public	Can Modify State	auth
transferFrom	Public	Can Modify State	transfersAllowed
transferFrom	Public	Can Modify State	transfersAllowed
transfer	Public	Can Modify State	-
approve	Public	Can Modify State	-
mint	Public	Can Modify State	auth stoppable
burn	Public	Can Modify State	auth stoppable
approveAndCall	Public	Can Modify State	-

WGFT			
isContract	Internal	-	-
<Fallback>	External	Payable	-
claimTokens	Public	Can Modify State	auth
withdrawTokens	Public	Can Modify State	auth
receiveApproval	Public	Can Modify State	-
_swapBurn	Private	Can Modify State	stoppable
_swapMint	Private	Can Modify State	stoppable
swapTo	External	Can Modify State	stoppable
_swapTo	Internal	Can Modify State	stoppable
swapFrom	External	Can Modify State	stoppable
_swapFrom	Internal	Can Modify State	stoppable

4.3 Vulnerability Summary

[N1] [Suggestion] Gas optimization

Category: Others

Content

Using assert in the contract will consume all the Gas.

Code location:starcrazy-contracts/contract/aliana/GFAccessControl.sol #L308-L315

```
function withdrawTokens(
    IERC20 token_,
    address to_,
    uint256 amount_
) external onlyCEO {
```

```
        assert(token_.transfer(to_, amount_));
        emit WithdrawTokens(address(token_), address(msg.sender), to_, amount_);
    }
```

Code location:starcrazy-contracts/contract/math/SafeMath.sol #L20-L79

```
    function mul(uint256 a, uint256 b) internal pure returns (uint256 c) {
        if (a == 0) {
            return 0;
        }
        c = a * b;
        assert(c / a == b);
        return c;
    }

    /**
     * @dev Integer division of two numbers, truncating the quotient.
     */
    /*@CTK SafeMath_div
    @tag spec
    @pre b != 0
    @post __reverted == __has_assertion_failure
    @post __has_overflow == true -> __has_assertion_failure == true
    @post __reverted == false -> __return == a / b
    @post msg == msg__post
   */
    /* CertiK Smart Labelling, for more details visit: https://certik.org */
    function div(uint256 a, uint256 b) internal pure returns (uint256) {
        // assert(b > 0); // Solidity automatically throws when dividing by 0
        // uint256 c = a / b;
        // assert(a == b * c + a % b); // There is no case in which this doesn't hold
        return a / b;
    }

    /**
     * @dev Subtracts two numbers, throws on overflow (i.e. if subtrahend is greater
than minuend).
     */
    /*@CTK SafeMath_sub
    @tag spec
    @post __reverted == __has_assertion_failure
    @post __has_overflow == true -> __has_assertion_failure == true
```

```
 @post __reverted == false -> __return == a - b
 @post msg == msg__post
*/
 /* CertiK Smart Labelling, for more details visit: https://certik.org */
 function sub(uint256 a, uint256 b) internal pure returns (uint256) {
     assert(b <= a);
     return a - b;
 }

 /**
  * @dev Adds two numbers, throws on overflow.
  */
 /*@CTK SafeMath_add
 @tag spec
 @post __reverted == __has_assertion_failure
 @post __has_assertion_failure == __has_overflow
 @post __reverted == false -> c == a + b
 @post msg == msg__post
*/
 /* CertiK Smart Labelling, for more details visit: https://certik.org */
 function add(uint256 a, uint256 b) internal pure returns (uint256 c) {
     c = a + b;
     assert(c >= a);
     return c;
 }
```

Code location:starcrazy-contracts/contract/GFS.sol #L22-L25

```
 modifier transfersAllowed() {
     assert(_transfersEnabled);
     _;
 }
```

Code location:starcrazy-contracts/contract/GFS.sol #L282-L288

```
 function withdrawTokens(
     IERC20 token_,
     address to_,
     uint256 amount_
 ) public auth {
```

```
        assert(token_.transfer(to_, amount_));
    }
```

Code location:starcrazy-contracts/contract/GFT.sol #L22-L25

```
    modifier transfersAllowed() {
        assert(_transfersEnabled);
        _;
    }
```

Code location:starcrazy-contracts/contract/GFT.sol #L273-L279

```
    function withdrawTokens(
        IERC20 token_,
        address to_,
        uint256 amount_
    ) public auth {
        assert(token_.transfer(to_, amount_));
    }
```

Code location:starcrazy-contracts/contract/WGFT.sol #L29-L32

```
    modifier transfersAllowed() {
        assert(_transfersEnabled);
        _;
    }
```

Code location:starcrazy-contracts/contract/WGFT.sol #L290-L296

```
    function withdrawTokens(
        IERC20 token_,
        address to_,
        uint256 amount_
    ) public auth {
        assert(token_.transfer(to_, amount_));
    }
```

Solution

It is recommended to replace all asserts in contracts with require.

Status

Ignored

[N2] [Suggestion] Overflow reminder

Category: Others

Content

There are many places in the contract that do not use safe-math for operations, which may cause overflow.

Solution

It is recommended to apply safe-math to all operations in the contract to avoid overflow.

Status

Ignored

[N3] [Suggestion] Hardcoded reminders

Category: Others

Content

The address in the contract is hardcoded and cannot be modified.

Code location:starcrazy-contracts/contract/gfc/utils/address-utils.sol #L15-L33

```
    function isContract(address _addr)
        internal
        view
        returns (bool addressCheck)
    {
        // This method relies in extcodesize, which returns 0 for contracts in
        // construction, since the code is only stored at the end of the
        // constructor execution.

        // According to EIP-1052, 0x0 is the value returned for not-yet created
accounts
```

```
        // and 0xc5d2460186f7233c927e7db2dcc703c0e500b653ca82273b7bfad8045d85a470 is
returned
        // for accounts without code, i.e. `keccak256('')`
        bytes32 codehash;
        bytes32 accountHash =
0xc5d2460186f7233c927e7db2dcc703c0e500b653ca82273b7bfad8045d85a470;
        assembly {
            codehash := extcodehash(_addr)
        } // solhint-disable-line
        addressCheck = (codehash != 0x0 && codehash != accountHash);
    }
```

Solution

It is not recommended to use hard coding, which is convenient for later modification.

Status

Ignored

[N4] [Suggestion] Return value not checked

Category: Others

Content

The _depositFrom function calls the insert function to insert data. There is a return value in the insert function but the return value is not checked when it is called.

Code location:starcrazy-contracts/contract/AlianaMinting.sol #L31-L46

```
    function insert(
        itmap storage self,
        uint256 key,
        uint256 value
    ) internal returns (bool replaced) {
        uint256 keyIndex = self.data[key].keyIndex;
        self.data[key].value = value;
        if (keyIndex > 0) return true;
        else {
            keyIndex = self.keys.length++;
            self.data[key].keyIndex = keyIndex + 1;
            self.keys[keyIndex].key = key;
```

```
            self.size++;
            return false;
        }
    }
```

Code location:starcrazy-contracts/contract/AlianaMinting.sol #L222-L261

```
    function _depositFrom(address _from, uint256 _tokenId)
        internal
        whenNotPaused
    {
        require(
            aliana.ownerOf(_tokenId) == _from,
            "AlianaMinting: must be the owner"
        );
        UserInfo storage user = userInfo[_from];
        if (maxMintingNumPerAddress > 0) {
            require(
                user.amountToken.size < maxMintingNumPerAddress,
                "AlianaMinting: too much mining at the same time"
            );
        }
        (, , , , uint256 _amount) = aliana.getAliana(_tokenId);
        require(_amount > 0, "AlianaMinting: gene _amount must > 0");

        updateBlockReward();
        uint256 takePending;
        if (user.amount > 0) {
            uint256 pending = user.amount.mul(accGFTPerShare).div(1e12).sub(
                user.rewardDebt
            );
            if (pending > 0) {
                safeGFTTransfer(_from, pending);
            }
            takePending = pending;
        }

        aliana.transferFrom(address(_from), address(this), _tokenId);
        insert(user.amountToken, _tokenId, _amount);
        user.balance = user.balance.add(1);

        user.amount = user.amount.add(_amount);
```

```
        labor = labor.add(_amount);

        user.rewardDebt = user.amount.mul(accGFTPerShare).div(1e12);
        emit Deposit(_from, _tokenId, _amount, takePending);
    }
```

Solution

If the business logic needs to check the return value of the insert function, please add require to check the return value, if not, delete the return value in the insert function.

Status

Ignored

[N5] [Low] Authority control issue

Category: Authority Control Vulnerability

Content

The CEO has the right to modify the address of the genScience contract. If the modified genScience contract is an unaudited contract, there may be security risks.

Code location:starcrazy-contracts/contract/aliana/AlianaBase.sol #L42-L47

```
    function setGeneScienceAddress(IGeneScience _address) public onlyCEO {
        require(_address.isGeneScience(), "Aliana: not gene");

        // Set the new contract address
        geneScience = _address;
    }
```

In the GFAccessControl contract, the CEO has the right to add and delete roles in the whitelist, and has the right to modify the roles of CFO and COO to any address.

Code location:starcrazy-contracts/contract/aliana/GFAccessControl.sol #L26-L117

```
    function addAddressToWhitelist(address addr)
        external
```

```
        onlyCEO
        returns (bool success)
    {
        return _addAddressToWhitelist(addr);
    }

    /**
     * @dev add an address to the whitelist
     * @param addr address
     * @return true if the address was added to the whitelist, false if the address
was already in the whitelist
     */
    function _addAddressToWhitelist(address addr)
        private
        onlyCEO
        returns (bool success)
    {
        if (!whitelist[addr]) {
            whitelist[addr] = true;
            emit WhitelistedAddressAdded(addr);
            success = true;
        }
    }

    /**
     * @dev add addresses to the whitelist
     * @param addrs addresses
     * @return true if at least one address was added to the whitelist,
     * false if all addresses were already in the whitelist
     */
    function addAddressesToWhitelist(address[] calldata addrs)
        external
        onlyCEO
        returns (bool success)
    {
        for (uint256 i = 0; i < addrs.length; i++) {
            if (_addAddressToWhitelist(addrs[i])) {
                success = true;
            }
        }
    }

    /**
     * @dev remove an address from the whitelist
```

```
 * @param addr address
 * @return true if the address was removed from the whitelist,
 * false if the address wasn't in the whitelist in the first place
 */
function removeAddressFromWhitelist(address addr)
    external
    onlyCEO
    returns (bool success)
{
    return _removeAddressFromWhitelist(addr);
}

/**
 * @dev remove an address from the whitelist
 * @param addr address
 * @return true if the address was removed from the whitelist,
 * false if the address wasn't in the whitelist in the first place
 */
function _removeAddressFromWhitelist(address addr)
    private
    onlyCEO
    returns (bool success)
{
    if (whitelist[addr]) {
        whitelist[addr] = false;
        emit WhitelistedAddressRemoved(addr);
        success = true;
    }
}

/**
 * @dev remove addresses from the whitelist
 * @param addrs addresses
 * @return true if at least one address was removed from the whitelist,
 * false if all addresses weren't in the whitelist in the first place
 */
function removeAddressesFromWhitelist(address[] calldata addrs)
    external
    onlyCEO
    returns (bool success)
{
    for (uint256 i = 0; i < addrs.length; i++) {
        if (_removeAddressFromWhitelist(addrs[i])) {
            success = true;
```

```
            }
        }
    }
```

Code location:starcrazy-contracts/contract/aliana/GFAccessControl.sol #L224-L238

```
    function setCFO(address _newCFO) external onlyCEO {
        require(_newCFO != address(0), "addr can't be 0");

        cfoAddress = _newCFO;
        emit SetCFO(cfoAddress);
    }

    /// @dev Assigns a new address to act as the COO. Only available to the current
CEO.
    /// @param _newCOO The address of the new COO
    function setCOO(address _newCOO) external onlyCEO {
        require(_newCOO != address(0), "addr can't be 0");

        cooAddress = _newCOO;
        emit SetCOO(cooAddress);
    }
```

The CEO has the authority to suspend the contract. When the contract is suspended, users will not be able to withdraw coins through normal channels, and can only use emergency withdrawals to withdraw coins. Emergency withdrawals do not count for user rewards.

Code location:starcrazy-contracts/contract/aliana/GFAccessControl.sol #L256-L259

```
    function pause() external onlyCEO whenNotPaused {
        paused = true;
        emit Pause(msg.sender);
    }
```

The Minter role has the right to mint coins for any address, and the number of coins minted is unlimited.

Code location:starcrazy-contracts/contract/token/ShadowToken.sol #L39-L50

```
function mint(address _to, uint256 _amount)
    public
    onlyMinter
    whenNotPaused
    returns (bool)
{
    totalSupply_ = totalSupply_.add(_amount);
    balances[_to] = balances[_to].add(_amount);
    emit Minted(_to, _amount);
    emit Transfer(address(0), _to, _amount);
    return true;
}
```

The auth role can mint coins for any address and has no upper limit. The auth role can also burn tokens from any address.

Code location:starcrazy-contracts/contract/GFS.sol #L194-L206

```
function mint(address guy_, uint256 wad_) public auth stoppable {
    require(totalSupply().add(wad_) <= _cap, "GFS-insufficient-cap");

    super.mint(guy_, wad_);

    emit Transfer(address(0), guy_, wad_);
}

function burn(address guy_, uint256 wad_) public auth stoppable {
    super.burn(guy_, wad_);

    emit Transfer(guy_, address(0), wad_);
}
```

Code location:starcrazy-contracts/contract/WGFT.sol #L202-L214

```
function mint(address guy_, uint256 wad_) public auth stoppable {
    require(totalSupply().add(wad_) <= _cap, "WGFT-insufficient-cap");

    super.mint(guy_, wad_);

    emit Transfer(address(0), guy_, wad_);
```

```
    }

    function burn(address guy_, uint256 wad_) public auth stoppable {
        super.burn(guy_, wad_);

        emit Transfer(guy_, address(0), wad_);
    }
```

In the Aliana contract, the roles in the whitelist and the roles of CEO, COO, and CFO can create NFTs with specified attributes for any address.

Code location:starcrazy-contracts/contract/Aliana.sol #L106-L140

```
    function createOfficialAliana(uint256 _genes, address _owner)
        external
        onlyWhitelisted
        returns (uint256)
    {
        return _createAliana(0, 0, _genes, _owner);
    }

    /// @dev we can create Official alianas, up to a limit. Only callable by Official
contract
    /// @param _genes the encoded genes of the kitten to be created, any value is
accepted
    /// @param _owner the future owner of the created alianas. Default to contract
COO
    function createOfficialAliana(
        uint256 _matronId,
        uint256 _sireId,
        uint256 _genes,
        address _owner
    ) external onlyWhitelisted returns (uint256) {
        return _createAliana(_matronId, _sireId, _genes, _owner);
    }

    /// @dev we can create promo alianas, up to a limit. Only callable by COO
    /// @param _genes the encoded genes of the kitten to be created, any value is
accepted
    /// @param _owner the future owner of the created alianas. Default to contract
COO
    function createPromoAliana(uint256 _genes, address _owner)
```

```
        external
        onlyCLevel
        returns (uint256)
    {
        address alianaOwner = _owner;
        if (alianaOwner == address(0)) {
            alianaOwner = msg.sender;
        }

        return _createAliana(0, 0, _genes, alianaOwner);
    }
```

In the FlashSale contract, the CEO has the right to arbitrarily modify the genes of the NFT.

Code location:starcrazy-contracts/contract/FlashSale.sol #L554-L562

```
    function setTokenGene(
        uint256[] calldata _tokenIds,
        uint256[] calldata _gene
    ) external onlyCEO {
        for (uint256 i = 0; i < _tokenIds.length; i++) {
            tokenIdToGene[_tokenIds[i]].gene = _gene[i];
            tokenIdToGene[_tokenIds[i]].used = true;
        }
    }
```

Solution

It is recommended to transfer the permissions of roles with excessive permissions to governance contracts or timelock contracts. At least multisig should be used.

Status

Confirmed; The project team will transfer the permissions to the timeLock contract management, but this can only play a buffering role, and cannot completely solve the problem of excessive permissions.

[N6] [Low] lack of authorization

Category: Authority Control Vulnerability

Content

In the AlianaMinting contract, the CEO has the right to call the withdrawsByCEO function (this function will calculate the reward) and emergencyWithdrawByCEO function (this function will not calculate the reward) to withdraw the tokens for the user without the user's authorization, and the withdrawn tokens will be in the same way returned to the user.

Code location:starcrazy-contracts/contract/AlianaMinting.sol #L330-L337

```
function withdrawsByCEO(address _from, uint256[] memory _tokenIds)
    public
    onlyCEO
{
    for (uint256 i = 0; i < _tokenIds.length; i++) {
        _withdrawFrom(_from, _tokenIds[i]);
    }
}
```

Code location:starcrazy-contracts/contract/AlianaMinting.sol #L345-L347

```
function emergencyWithdrawByCEO(address _from) public onlyCEO {
    _emergencyWithdrawFrom(_from);
}
```

In the AlianaSale function, the CEO, COO, and CFO have the right to call the adminCancelAlianaSales function to cancel the user's pending order.

Code location:starcrazy-contracts/contract/AlianaSale.sol #L134-L151

```
function adminCancelAlianaSales(uint256[] memory _tokenIds)
    public
    onlyCLevel
{
    for (uint256 i = 0; i < _tokenIds.length; i++) {
        uint256 _tokenId = _tokenIds[i];
        AlianaSaleInfo storage info = _allSaleAlianaInfo[_tokenId];
        if (_isOnSale(info)) {
            aliana.transferFrom(
                address(this),
```

```
                address(info.seller),
                _tokenId
            );
            _burnSale(info.seller, _tokenId);
            emit CancelAlianaSale(info.seller, _tokenId);
        }
    }
}
```

Solution

It is recommended to add a switch that turns on the function call when the user authorizes the CEO to manage, and turns off if the user does not authorize it.

Status

Confirmed; After communication, we learned that the reason why the project team set up these functions is to consider that the project team will bear the gas fee and call these functions to retrieve the user's assets in the event of an emergency, so as to help the user stop losses in time. In the future, the authority will be handed over to the timelock contract for management.

[N7] [Low] Business logic issue

Category: Design Logic Audit

Content

The receiveApproval function exists in multiple contracts. The function is public and can be called by anyone. Anyone can use these functions to manipulate other users' assets, as long as the user has an authorization limit to the contract. However, this authorization limit is not set by the user. The authorization limit for each transaction of the user is calculated off-chain by the project team. The code here is not within the scope of this audit, so we will not be able to guarantee the security here.

If there is an error in the calculation of the authorization limit.Anyone can call the receiveApproval function to deposit any user's tokens without authorization.

Code location:starcrazy-contracts/contract/AlianaMinting.sol #L418-L442

```
    function receiveApproval(
        address _sender,
        uint256 _value,
        address _tokenContract,
        bytes memory _extraData
    ) public {
        require(_value >= 0, "AlianaMinting: approval negative");
        uint256 action;
        assembly {
            action := mload(add(_extraData, 0x20))
        }
        require(action == 1, "AlianaMinting: unknow action");
        if (action == 1) {
            // deposit
            require(
                _tokenContract == address(aliana),
                "AlianaMinting: approval and want mint use aliana, but used token
isn't Aliana"
            );
            uint256 tokenId;
            assembly {
                tokenId := mload(add(_extraData, 0x40))
            }
            _depositFrom(_sender, tokenId);
        }
    }
```

Code location:starcrazy-contracts/contract/GFSBonus.sol #L243-L274

```
    function receiveApproval(
        address _sender,
        uint256 _value,
        address _tokenContract,
        bytes memory _extraData
    ) public {
        require(_value > 0, "approval zero");
        uint256 action;
        assembly {
            action := mload(add(_extraData, 0x20))
        }
        emit ReceiveApproval(
```

```
            _sender,
            _value,
            _tokenContract,
            _extraData,
            action
        );
        require(action == 3, "unknow action");
        if (action == 3) {
            // deposit
            require(
                _tokenContract == address(gfsToken),
                "approval and want deposit, but used token isn't GFT"
            );
            uint256 amount;
            assembly {
                amount := mload(add(_extraData, 0x40))
            }
            _depositFrom(_sender, amount);
        }
    }
```

Code location:starcrazy-contracts/contract/GFSMint.sol #L205-L236

```
    function receiveApproval(
        address _sender,
        uint256 _value,
        address _tokenContract,
        bytes memory _extraData
    ) public {
        require(_value > 0, "GFSMint: approval zero");
        uint256 action;
        assembly {
            action := mload(add(_extraData, 0x20))
        }
        emit ReceiveApproval(
            _sender,
            _value,
            _tokenContract,
            _extraData,
            action
        );
        require(action == 3, "GFSMint: unknow action");
```

```
    if (action == 3) {
        // buy
        require(
            _tokenContract == address(lpToken),
            "GFSMint: approval and want deposit, but used token isn't GFT"
        );
        uint256 amount;
        assembly {
            amount := mload(add(_extraData, 0x40))
        }
        _depositFrom(_sender, amount);
    }
}
```

Code location:starcrazy-contracts/contract/LPMint.sol #L205-L236

```
function receiveApproval(
    address _sender,
    uint256 _value,
    address _tokenContract,
    bytes memory _extraData
) public {
    require(_value > 0, "approval zero");
    uint256 action;
    assembly {
        action := mload(add(_extraData, 0x20))
    }
    emit ReceiveApproval(
        _sender,
        _value,
        _tokenContract,
        _extraData,
        action
    );
    require(action == 3, "unknow action");
    if (action == 3) {
        // buy
        require(
            _tokenContract == address(lpToken),
            "approval and want deposit, but used token isn't GFT"
        );
        uint256 amount;
```

```
            assembly {
                amount := mload(add(_extraData, 0x40))
            }
            _depositFrom(_sender, amount);
        }
    }
```

In the AlianaSale contract,If the user is attacked by phishing or miscalculated off-chain, resulting in incorrect authorization, anyone can sell any user's NFT and buy NFT on behalf of any user without authorization (the tokens used for the purchase are the user's). If this function is exploited by an attacker, the attacker can sell the target user's rare attribute NFT at any price. Attackers can also create malicious orders, place a low-attribute NFT at a high price, and then use the target user's tokens to buy this NFT, eventually resulting in the loss of the user's assets.

Code location:starcrazy-contracts/contract/AlianaSale.sol #L386-L423

```
    function receiveApproval(
        address _sender,
        uint256 _value,
        address _tokenContract,
        bytes memory _extraData
    ) public {
        require(_value >= 0, "AlianaSale: approval negative");
        uint256 action;
        assembly {
            action := mload(add(_extraData, 0x20))
        }
        require(action == 1 || action == 2, "AlianaSale: unknow action");
        if (action == 1) {
            // buy
            require(
                _tokenContract == address(gaeToken),
                "AlianaSale: approval and want buy a aliana, but used token isn't
GFT"
            );
            uint256 tokenId;
            assembly {
                tokenId := mload(add(_extraData, 0x40))
            }
            _buySaleAlianaFrom(_sender, tokenId);
        } else if (action == 2) {
```

```
            // sale
            require(
                _tokenContract == address(aliana),
                "AlianaSale: approval and want sale a aliana, but used token isn't
Aliana"
            );
            uint256 tokenId;
            uint256 price;
            assembly {
                tokenId := mload(add(_extraData, 0x40))
                price := mload(add(_extraData, 0x60))
            }
            _createAlianaSaleFrom(_sender, tokenId, price);
        }
    }
```

In the Auction contract, If a user suffers a phishing attack resulting in incorrect authorization, anyone can call the receiveApproval function without authorization to bid for any user at any price higher than the minimum bid, which can be exploited by an attacker to force the victim to bid on their own lot.

Code location:starcrazy-contracts/contract/Auction.sol #L504-530

```
    function receiveApproval(
        address _sender,
        uint256 _value,
        address _tokenContract,
        bytes memory _extraData
    ) public {
        require(_value > 0, "Auction: approval zero");
        uint256 action;
        assembly {
            action := mload(add(_extraData, 0x20))
        }
        require(action == 2, "Auction: unknow action");
        if (action == 2) {
            // buy
            require(
                _tokenContract == address(gaeToken),
                "Auction: approval and want buy a aliana, but used token isn't GFT"
            );
            uint256 tokenId;
```

```
            uint256 price;
            assembly {
                tokenId := mload(add(_extraData, 0x40))
                price := mload(add(_extraData, 0x60))
            }
            _bidFrom(_sender, tokenId, price);
        }
    }
```

In the FlashSale contract, Incorrect authorization will result in anyone can call the receiveApproval function without authorization to bid for any user at any price higher than the minimum bid, which can be exploited by an attacker to force the victim to bid on their own lot.

Code location:starcrazy-contracts/contract/FlashSale.sol #L580-L606

```
    function receiveApproval(
        address _sender,
        uint256 _value,
        address _tokenContract,
        bytes memory _extraData
    ) public {
        require(_value > 0, "FlashSale: approval zero");
        uint256 action;
        assembly {
            action := mload(add(_extraData, 0x20))
        }
        require(action == 2, "FlashSale: unknow action");
        if (action == 2) {
            // buy
            require(
                _tokenContract == address(gaeToken),
                "FlashSale: approval and want buy a aliana, but used token isn't GFT"
            );
            uint256 tokenId;
            uint256 price;
            assembly {
                tokenId := mload(add(_extraData, 0x40))
                price := mload(add(_extraData, 0x60))
            }
            _bidFrom(_sender, tokenId, price);
```

```
        }
    }
```

Solution

It is recommended to check the user's authorization limit after each transaction to ensure that the authorization limit is not too high. In addition, it is recommended to add an anti-phishing reminder on the user page to prevent users from being attacked by phishing and causing unnecessary losses.

Status

Confirmed; The project team promises to strictly control the calculation of the authorization limit under the chain, and there will be no error in the calculation of the authorization limit.

[N8] [Suggestion] Redundant code

Category: Others

Content

The require check for the action parameter in the following function is redundant.

The location of redundant code will be marked with (//Slowmist// Redundant code here).

Code location:starcrazy-contracts/contract/AlianaMinting.sol #L418-L442

```
    function receiveApproval(
        address _sender,
        uint256 _value,
        address _tokenContract,
        bytes memory _extraData
    ) public {
        require(_value >= 0, "AlianaMinting: approval negative");
        uint256 action;
        assembly {
            action := mload(add(_extraData, 0x20))
        }
//Slowmist// Redundant code here.
        require(action == 1, "AlianaMinting: unknow action");
        if (action == 1) {
            // deposit
            require(
```

```
                _tokenContract == address(aliana),
                "AlianaMinting: approval and want mint use aliana, but used token
isn't Aliana"
            );
            uint256 tokenId;
            assembly {
                tokenId := mload(add(_extraData, 0x40))
            }
            _depositFrom(_sender, tokenId);
        }
    }
```

Code location:starcrazy-contracts/contract/AlianaSale.sol #L386-L423

```
    function receiveApproval(
        address _sender,
        uint256 _value,
        address _tokenContract,
        bytes memory _extraData
    ) public {
        require(_value >= 0, "AlianaSale: approval negative");
        uint256 action;
        assembly {
            action := mload(add(_extraData, 0x20))
        }
//Slowmist// Redundant code here.
        require(action == 1 || action == 2, "AlianaSale: unknow action");
        if (action == 1) {
            // buy
            require(
                _tokenContract == address(gaeToken),
                "AlianaSale: approval and want buy a aliana, but used token isn't
GFT"
            );
            uint256 tokenId;
            assembly {
                tokenId := mload(add(_extraData, 0x40))
            }
            _buySaleAlianaFrom(_sender, tokenId);
        } else if (action == 2) {
            // sale
            require(
```

```
                _tokenContract == address(aliana),
                "AlianaSale: approval and want sale a aliana, but used token isn't
Aliana"
            );
            uint256 tokenId;
            uint256 price;
            assembly {
                tokenId := mload(add(_extraData, 0x40))
                price := mload(add(_extraData, 0x60))
            }
            _createAlianaSaleFrom(_sender, tokenId, price);
        }
    }
```

Code location:starcrazy-contracts/contract/Auction.sol #L504-L530

```
    function receiveApproval(
        address _sender,
        uint256 _value,
        address _tokenContract,
        bytes memory _extraData
    ) public {
        require(_value > 0, "Auction: approval zero");
        uint256 action;
        assembly {
            action := mload(add(_extraData, 0x20))
        }
//Slowmist// Redundant code here.
        require(action == 2, "Auction: unknow action");
        if (action == 2) {
            // buy
            require(
                _tokenContract == address(gaeToken),
                "Auction: approval and want buy a aliana, but used token isn't GFT"
            );
            uint256 tokenId;
            uint256 price;
            assembly {
                tokenId := mload(add(_extraData, 0x40))
                price := mload(add(_extraData, 0x60))
            }
            _bidFrom(_sender, tokenId, price);
```

```
        }
    }
```

Code location:starcrazy-contracts/contract/FakeAliana.sol #L48-L72

```
    function receiveApproval(
        address _sender,
        uint256 _value,
        address _tokenContract,
        bytes memory _extraData
    ) public {
        require(_value >= 0, "FakeAliana: approval negative");
        uint256 action;
        assembly {
            action := mload(add(_extraData, 0x20))
        }
//Slowmist// Redundant code here.
        require(action == 1, "FakeAliana: unknow action");
        if (action == 1) {
            // mix
            require(
                _tokenContract == address(aliana),
                "FakeAliana: approval and want mint use aliana, but used token isn't
Aliana"
            );
            uint256 tokenId;
            assembly {
                tokenId := mload(add(_extraData, 0x40))
            }
            _mixFrom(_sender, tokenId);
        }
    }
```

Code location:starcrazy-contracts/contract/FlashSale.sol #L580-L606

```
    function receiveApproval(
        address _sender,
        uint256 _value,
        address _tokenContract,
        bytes memory _extraData
    ) public {
```

```
        require(_value > 0, "FlashSale: approval zero");
        uint256 action;
        assembly {
            action := mload(add(_extraData, 0x20))
        }
//Slowmist// Redundant code here.
        require(action == 2, "FlashSale: unknow action");
        if (action == 2) {
            // buy
            require(
                _tokenContract == address(gaeToken),
                "FlashSale: approval and want buy a aliana, but used token isn't GFT"
            );
            uint256 tokenId;
            uint256 price;
            assembly {
                tokenId := mload(add(_extraData, 0x40))
                price := mload(add(_extraData, 0x60))
            }
            _bidFrom(_sender, tokenId, price);
        }
    }
```

Code location:starcrazy-contracts/contract/GFSBonus.sol #L243-L274

```
    function receiveApproval(
        address _sender,
        uint256 _value,
        address _tokenContract,
        bytes memory _extraData
    ) public {
        require(_value > 0, "approval zero");
        uint256 action;
        assembly {
            action := mload(add(_extraData, 0x20))
        }
        emit ReceiveApproval(
            _sender,
            _value,
            _tokenContract,
            _extraData,
            action
```

```
        );
//Slowmist// Redundant code here.
        require(action == 3, "unknow action");
        if (action == 3) {
            // deposit
            require(
                _tokenContract == address(gfsToken),
                "approval and want deposit, but used token isn't GFT"
            );
            uint256 amount;
            assembly {
                amount := mload(add(_extraData, 0x40))
            }
            _depositFrom(_sender, amount);
        }
    }
```

Code location:starcrazy-contracts/contract/GFSMint.sol #L205-L236

```
    function receiveApproval(
        address _sender,
        uint256 _value,
        address _tokenContract,
        bytes memory _extraData
    ) public {
        require(_value > 0, "GFSMint: approval zero");
        uint256 action;
        assembly {
            action := mload(add(_extraData, 0x20))
        }
        emit ReceiveApproval(
            _sender,
            _value,
            _tokenContract,
            _extraData,
            action
        );
//Slowmist// Redundant code here.
        require(action == 3, "GFSMint: unknow action");
        if (action == 3) {
            // buy
            require(
```

```
                _tokenContract == address(lpToken),
                "GFSMint: approval and want deposit, but used token isn't GFT"
            );
            uint256 amount;
            assembly {
                amount := mload(add(_extraData, 0x40))
            }
            _depositFrom(_sender, amount);
        }
    }
```

Code location:starcrazy-contracts/contract/LPMint.sol #L205-L236

```
    function receiveApproval(
        address _sender,
        uint256 _value,
        address _tokenContract,
        bytes memory _extraData
    ) public {
        require(_value > 0, "approval zero");
        uint256 action;
        assembly {
            action := mload(add(_extraData, 0x20))
        }
        emit ReceiveApproval(
            _sender,
            _value,
            _tokenContract,
            _extraData,
            action
        );
//Slowmist// Redundant code here.
        require(action == 3, "unknow action");
        if (action == 3) {
            // buy
            require(
                _tokenContract == address(lpToken),
                "approval and want deposit, but used token isn't GFT"
            );
            uint256 amount;
            assembly {
                amount := mload(add(_extraData, 0x40))
```

```
            }
            _depositFrom(_sender, amount);
        }
    }
```

Code location:starcrazy-contracts/contract/WGFT.sol #L316-L347

```
    function receiveApproval(
        address _sender,
        uint256 _value,
        address _tokenContract,
        bytes memory _extraData
    ) public {
        require(_value > 0, "approval zero");
        uint256 action;
        assembly {
            action := mload(add(_extraData, 0x20))
        }
        emit ReceiveApproval(
            _sender,
            _value,
            _tokenContract,
            _extraData,
            action
        );
//Slowmist// Redundant code here.
        require(action == 5, "unknow action");
        if (action == 5) {
            // swapFrom
            require(
                _tokenContract == address(wrappedToken),
                "approval and want deposit, but used token isn't GFT"
            );
            uint256 amount;
            assembly {
                amount := mload(add(_extraData, 0x40))
            }
            _swapFrom(_sender, amount);
        }
    }
```

In the setCycleBlock of the FlashSale contract, the value of the incoming _cycleBlock will be checked. Since the c_duration parameter is passed in when the contract is constructed, the default value is 50, so it is only necessary to judge _cycleBlock > c_duration, no need to judge _cycleBlock > 0 .

Code location:starcrazy-contracts/contract/FlashSale.sol #L92-L96

```
function setCycleBlock(uint256 _cycleBlock) public onlyCEO {
    require(_cycleBlock > 0, "require _cycleBlock > 0");
    require(_cycleBlock > c_duration, "require _cycleBlock > c_duration");
    c_cycleBlock = _cycleBlock;
}
```

In the _swapBurn function of the WGFT contract, this function can only be called by the _swapTo function. The _swapTo function is called by the swapTo function. When calling the _swapTo function, the default from is msg.sender, so the value of guy is always msg .sender , so there is no guy != msg.sender here, which means that the if judgment here is meaningless.

Code location:starcrazy-contracts/contract/WGFT.sol #L349-L358

```
function _swapBurn(address guy, uint256 wad) private stoppable {
    if (guy != msg.sender && _allowances[guy][msg.sender] != uint256(-1)) {
        require(
            _allowances[guy][msg.sender] >= wad,
            "ds-token-insufficient-approval"
        );
        _allowances[guy][msg.sender] = _allowances[guy][msg.sender].sub(
            wad
        );
    }
```

Code location:starcrazy-contracts/contract/WGFT.sol #L376-L383

```
function swapTo(uint256 _amount) external stoppable {
    _swapTo(msg.sender, _amount);
}

function _swapTo(address _from, uint256 _amount) internal stoppable {
```

```
        _swapBurn(_from, _amount);
        wrappedToken.transferFrom(address(this), msg.sender, _amount);
    }
```

Solution

It is recommended to remove redundant code.

Status

Ignored

[N9] [Low] Parameter modification issue

Category: Authority Control Vulnerability

Content

The NFT in the project can be auctioned, and the CEO has the right to modify some sensitive parameters in the auction information, which will affect the results of the auction.

Code location:starcrazy-contracts/contract/Auction.sol #L103-L105

```
    function setMinAddPrice(uint256 _value) public onlyCEO {
        c_minAddPrice = _value;
    }
```

Code location:starcrazy-contracts/contract/FlashSale.sol #L92-L96

```
    function setCycleBlock(uint256 _cycleBlock) public onlyCEO {
        require(_cycleBlock > 0, "require _cycleBlock > 0");
        require(_cycleBlock > c_duration, "require _cycleBlock > c_duration");
        c_cycleBlock = _cycleBlock;
    }
```

Code location:starcrazy-contracts/contract/FlashSale.sol #L102-L108

```
    function setStartBlockOffset(uint256 _startBlockOffset) public onlyCEO {
        require(
            _startBlockOffset <= block.number,
```

```
            "require _startBlockOffset <= block.number"
        );
        c_startBlockOffset = _startBlockOffset;
    }
```

Code location:starcrazy-contracts/contract/FlashSale.sol #L114-L116

```
    function setMaxCycle(uint256 _maxCycle) public onlyCEO {
        c_maxCycle = _maxCycle;
    }
```

Code location:starcrazy-contracts/contract/FlashSale.sol #L122-L126

```
    function setDuration(uint256 _value) public onlyCEO {
        require(_value > 0, "require _value > 0");
        require(c_cycleBlock > _value, "require c_cycleBlock > _value");
        c_duration = _value;
    }
```

Code location:starcrazy-contracts/contract/FlashSale.sol #L156-L158

```
    function setMinAddPrice(uint256 _value) public onlyCEO {
        c_minAddPrice = _value;
    }
```

Code location:starcrazy-contracts/contract/FlashSale.sol #L554-L562

```
    function setTokenGene(
        uint256[] calldata _tokenIds,
        uint256[] calldata _gene
    ) external onlyCEO {
        for (uint256 i = 0; i < _tokenIds.length; i++) {
            tokenIdToGene[_tokenIds[i]].gene = _gene[i];
            tokenIdToGene[_tokenIds[i]].used = true;
        }
    }
```

Solution

It is recommended to add a switch, you can modify the parameters when the auction has not started, and it is forbidden to modify the auction information during the auction.

Status

Confirmed

[N10] [Suggestion] Missing event record

Category: Others

Content

Events are not logged when sensitive parameters are modified in several places in the contract.

Code location:starcrazy-contracts/contract/aliana/AlianaBase.sol #L42-L47

```
    function setGeneScienceAddress(IGeneScience _address) public onlyCEO {
        require(_address.isGeneScience(), "Aliana: not gene");

        // Set the new contract address
        geneScience = _address;
    }
```

Code location:starcrazy-contracts/contract/AlianaMinting.sol #L166-L176

```
    function setMaxMintingNumPerAddress(uint256 _maxMintingNumPerAddress)
        public
        onlyCEO
    {
        maxMintingNumPerAddress = _maxMintingNumPerAddress;
    }

    function setRewardPerBlock(uint256 _rewardPerBlock) public onlyCEO {
        updateBlockReward();
        rewardPerBlock = _rewardPerBlock;
    }
```

Code location:starcrazy-contracts/contract/AlianaSale.sol #L63-L66

```
    function setOwnerCutSale(uint256 num) external onlyCEO {
        require(num <= 10000, "AlianaSale: num not valid");









        ownerCutSale = num;
    }
```

Code location:starcrazy-contracts/contract/Auction.sol #L45-L51

```
    function setGeneScienceAddress(IGeneScience _address) public onlyCEO {
        // NOTE: verify that a contract is what we expect -
https://github.com/Lunyr/crowdsale-
contracts/blob/cfadd15986c30521d8ba7d5b6f57b4fefcc7ac38/contracts/LunyrToken.sol#L117
        require(_address.isGeneScience(), "Aliana: not gene");

        // Set the new contract address
        geneScience = _address;
    }
```

Code location:starcrazy-contracts/contract/Auction.sol #L87-L89

```
    function setStartingPrice(uint256 _value) public onlyCEO {
        c_startingPrice = _value;
    }
```

Code location:starcrazy-contracts/contract/Auction.sol #L95-L97

```
    function setDuration(uint256 _value) public onlyCEO {
        c_duration = _value;
    }
```

Code location:starcrazy-contracts/contract/Auction.sol #L103-L105

```
function setMinAddPrice(uint256 _value) public onlyCEO {
    c_minAddPrice = _value;
}
```

Code location:starcrazy-contracts/contract/FlashSale.sol #L84-L86

```
function setStartingPrice(uint256 _value) public onlyCEO {
    c_startingPrice = _value;
}
```

Code location:starcrazy-contracts/contract/FlashSale.sol #L92-L96

```
function setCycleBlock(uint256 _cycleBlock) public onlyCEO {
    require(_cycleBlock > 0, "require _cycleBlock > 0");
    require(_cycleBlock > c_duration, "require _cycleBlock > c_duration");
    c_cycleBlock = _cycleBlock;
}
```

Code location:starcrazy-contracts/contract/FlashSale.sol #L102-L108

```
function setStartBlockOffset(uint256 _startBlockOffset) public onlyCEO {
    require(
        _startBlockOffset <= block.number,
        "require _startBlockOffset <= block.number"
    );
    c_startBlockOffset = _startBlockOffset;
}
```

Code location:starcrazy-contracts/contract/FlashSale.sol #L114-L116

```
function setMaxCycle(uint256 _maxCycle) public onlyCEO {
    c_maxCycle = _maxCycle;
}
```

Code location:starcrazy-contracts/contract/FlashSale.sol #L122-L126

```
function setDuration(uint256 _value) public onlyCEO {
    require(_value > 0, "require _value > 0");
    require(c_cycleBlock > _value, "require c_cycleBlock > _value");
    c_duration = _value;
}
```

Code location:starcrazy-contracts/contract/FlashSale.sol #L156-L158

```
function setMinAddPrice(uint256 _value) public onlyCEO {
    c_minAddPrice = _value;
}
```

Code location:starcrazy-contracts/contract/FlashSale.sol #L554-L562

```
function setTokenGene(
    uint256[] calldata _tokenIds,
    uint256[] calldata _gene
) external onlyCEO {
    for (uint256 i = 0; i < _tokenIds.length; i++) {
        tokenIdToGene[_tokenIds[i]].gene = _gene[i];
        tokenIdToGene[_tokenIds[i]].used = true;
    }
}
```

Code location:starcrazy-contracts/contract/GeneScienceSafe.sol #L501-L506

```
function setMathGene(IGeneScience _address) public onlyCEO {
    require(_address.isGeneScience(), "GeneScience: not gene");

    // Set the new contract address
    mathGene = _address;
}
```

Code location:starcrazy-contracts/contract/GFSBonus.sol #L76-L79

```
function setRewardPerBlock(uint256 _rewardPerBlock) public onlyCEO {
    updateBlockReward();
```

```
        rewardPerBlock = _rewardPerBlock;
    }
```

Code location:starcrazy-contracts/contract/GFSMint.sol #L67-L70

```
    function setRewardPerBlock(uint256 _rewardPerBlock) public onlyCEO {
        updateBlockReward();
        rewardPerBlock = _rewardPerBlock;
    }
```

Code location:starcrazy-contracts/contract/LPMint.sol #L67-L70

```
    function setRewardPerBlock(uint256 _rewardPerBlock) public onlyCEO {
        updateBlockReward();
        rewardPerBlock = _rewardPerBlock;
    }
```

Solution

It is recommended to add event records when modifying contract sensitive parameters for subsequent self-inspection or community review.

Status

Ignored

5 Audit Result

Audit Number	Audit Team	Audit Date	Audit Result
0X002203150002	SlowMist Security Team	2022.02.28 - 2022.03.15	Low 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 4 low risks and 6 suggestion vulnerabilities. And 4 low-risk vulnerabilities

were confirmed; 6 suggestion vulnerabilities were ignored; The code was 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