Smart Contract
Security Audit Report

Table Of Contents

1 Executive Summary

On 2022.05.30, the SlowMist security team received the team's security audit application for SubDao, 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

https://github.com/SubDAO-Network/app-contracts

commit: 7049b3b26ef4850f8874a7c6e838a3974571fa98

Fixed Version:

https://github.com/SubDAO-Network/app-contracts

commit: 1acd5ae4bf0a97c81a61bda467d9a6213e5e6bb4

3.2 Vulnerability Information

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

NO	Title	Category	Level	Status
N1	Pages calculation issue	Design Logic Audit	Low	Fixed
N2	Missing event records	Others	Suggestion	Fixed
N3	Owner update issue	Design Logic Audit	Critical	Fixed
N4	State Coverage Risk	Design Logic Audit	High	Confirmed
N5	TODO label issue	Others	Suggestion	Confirmed
N6	Length check issue	Design Logic Audit	High	Fixed
N7	Vote check issue	Design Logic Audit	Low	Confirmed
N8	Cancel voting issue	Design Logic Audit	Medium	Fixed

NO	Title	Category	Level	Status
N9	Risk of Governance Attacks	Design Logic Audit	Low	Confirmed
N10	Potential Fund Theft Risk	Design Logic Audit	Critical	Fixed
N11	Lack of access control	Authority Control Vulnerability	High	Fixed
N12	Period transfer issue	Design Logic Audit	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:

DaoFactory			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
calculatePages	Internal	-	-
utfStringLength	Internal	-	-
setConfig	External	Can Modify State	onlyOwner

DaoFactory			
setActionConfig	External	Can Modify State	onlyOwner
instanceByTemplate	External	Can Modify State	nonReentrant
listDaoInstanceByOwner	External	-	-
listDaoInstanceByAccount	External	-	-
listDaoInstanceByIds	External	-	-
addComponent	Public	Can Modify State	-
addComponents	Public	Can Modify State	-
updateTemplate	External	Can Modify State	-
transferOrgOwnerShip	External	Can Modify State	-
getComponentHash	Public	-	-
editDaoInfo	External	Can Modify State	-
editSocialUrl	External	Can Modify State	-
setDaoAlias	Public	Can Modify State	-
isAliasExists	Public	-	-

DaoFactoryConfig			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
editTemplateMap	External	Can Modify State	onlyOwner
editComponentMap	External	Can Modify State	onlyOwner

DaoFactoryConfig			
getTemplateInstance	External	Can Modify State	-
getComponentInstance	External	Can Modify State	-

DaoOrganization			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
setTemplate	External	Can Modify State	onlyOwner
setDaoConfig	External	Can Modify State	onlyOwner
setTemplateConfig	External	Can Modify State	onlyOwner
addComponent	External	Can Modify State	onlyOwner
updateTemplate	External	Can Modify State	onlyOwner
setActionConfig	External	Can Modify State	onlyOwner
transferOrgOwnerShip	External	Can Modify State	onlyOwner
templateOwner	External	-	-
action	External	Can Modify State	nonReentrant

TemplateManager			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
addNewTemplate	External	Can Modify State	onlyAdmin
listTemplates	External	-	-

TemplateManager			
queryTemplateByIndex	External	-	-

ActionConfig			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
addAction	External	Can Modify State	onlyOwner
addActions	External	Can Modify State	onlyOwner
removeAction	External	Can Modify State	onlyOwner
removeActions	External	Can Modify State	onlyOwner
isAction	External	-	-
listActions	External	-	-
actionLength	External	-	-

DaoTemplate			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
setActionConfig	External	Can Modify State	onlyOwner
addComponent	External	Can Modify State	onlyOwner
updateOwnership	External	Can Modify State	onlyOwner
action	External	Can Modify State	onlyOwner

TemplateConfig			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
addComponent	External	Can Modify State	onlyOwner
removeComponent	External	Can Modify State	onlyOwner
isComponent	External	-	-
listComponents	External	-	-
componentLength	External	-	-
addAction	External	Can Modify State	onlyOwner
removeAction	External	Can Modify State	onlyOwner
isAction	External	-	-
listActions	External	-	-
actionLength	External	-	-

VentureTemplate			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
setActionConfig	External	Can Modify State	onlyOwner
addComponent	External	Can Modify State	onlyOwner
updateOwnership	External	Can Modify State	onlyOwner
action	External	Can Modify State	onlyOwner

AuthManagerCreator			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
create	External	Can Modify State	nonReentrant

GovTokenManagerCreator			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
create	External	Can Modify State	nonReentrant

GrantMethodManagerCreator			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
create	External	Can Modify State	nonReentrant

OrgManagerCreator			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
create	External	Can Modify State	nonReentrant

TemplateDaoCreator			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer

TemplateDaoCreator			
create	External	Can Modify State	nonReentrant

TemplateVentureCreator			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
create	External	Can Modify State	nonReentrant

VaultManagerCreator			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
create	External	Can Modify State	nonReentrant

VenturesManagerCreator			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
create	External	Can Modify State	nonReentrant

VenturesStockManagerCreator			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
create	External	Can Modify State	nonReentrant

VoteExecutionManagerCreator			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
create	External	Can Modify State	nonReentrant

VoteManagerCreator			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
create	External	Can Modify State	nonReentrant

HappyRedPacket			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
box	Internal	-	-
unbox	Internal	-	-
validRange	Internal	-	-
rewriteBox	Internal	-	-
transferToken	Internal	Can Modify State	-
random	Internal	-	-
wrap1	Internal	-	-
wrap2	Internal	-	-
_verify	Private	-	-

HappyRedPacket			
updateOwnership	External	Can Modify State	onlyOwner
createRedPacket	Public	Payable	-
claim	Public	Can Modify State	-
checkAvailability	External	-	-
checkClaimed	External	-	-
refund	Public	Can Modify State	-

AuthManager			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
updateOwnership	External	Can Modify State	onlyRole
addModerator	Public	Can Modify State	onlyRole
addMember	Public	Can Modify State	-
isMember	Public	-	-
removeModerator	Public	Can Modify State	onlyRole
removeMember	Public	Can Modify State	-
exit	Public	Can Modify State	-
getMemberCount	Public	-	-
getMember	Public	-	-
getModeratorCount	Public	-	-

AuthManager			
getModerator	Public	-	-

GovTokenManager			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
updateOwnership	External	Can Modify State	onlyOwner
mint	External	Can Modify State	onlyOwner

GrantMethodManager			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
updateOwnership	External	Can Modify State	onlyOwner
generateNewVoteId	External	Can Modify State	onlyOwner
registerOp	External	Can Modify State	onlyOwner
applyOp	External	Can Modify State	onlyOwner
setUserOpByOwner	External	Can Modify State	onlyOwner
checkMethodPermission	External	-	-
createSpendLimit	Public	Can Modify State	onlyOwner
spendTokenInLimit	External	Can Modify State	onlyOwner
getTokenLimit	External	-	-
removeUserOpByOwner	External	Can Modify State	onlyOwner

GrantMethodManager			
getKey	External	-	-

OrgManager			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
addDaoModerator	External	Can Modify State	onlyOwner
addDaoMember	External	Can Modify State	onlyOwner
removeDaoModerator	External	Can Modify State	onlyOwner
removeDaoMember	External	Can Modify State	onlyOwner
addApplyingMember	External	Can Modify State	onlyOwner
removeApplyingMember	External	Can Modify State	onlyOwner
getDaoMembersList	External	-	-
getDaoModeratorList	External	-	-
getDaoApplyingMemberList	External	-	-
checkRoleByAccount	External	-	-
getOrgCount	External	-	-
isOwnerOrMemberOrModerator	External	-	-
isOwner	External	-	-
setCanFreeAddMember	External	Can Modify State	onlyOwner
transferOwner	External	Can Modify State	onlyOwner

OrgManager			
updateOwnership	External	Can Modify State	onlyOwner

VaultManager			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
addVaultToken	External	Can Modify State	onlyOwner
removeVaultToken	External	Can Modify State	onlyOwner
withdrawApplying	External	Can Modify State	onlyOwner
withdrawByVote	External	Can Modify State	onlyOwner
getWithdrawApplyDetail	External	-	-
getTokenList	External	-	-
getBalanceOf	External	-	-
deposit	External	Payable	-
withdraw	Public	Can Modify State	onlyOwner
_withdraw	Private	Can Modify State	-
updateOwnership	External	Can Modify State	onlyOwner

VenturesManager			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
_initBaseInfo	Internal	Can Modify State	onlyOwner

VenturesManager			
_openGPRaiseMoney	Internal	Can Modify State	onlyOwner mustSetToken
transferVenturePeriod	Public	Can Modify State	onlyOwner
setGPBonusPercent	Public	Can Modify State	onlyOwner
addLP	Public	Can Modify State	onlyOwner
getLPCount	Public	-	-
setWhiteList	Public	Can Modify State	onlyOwner
updateOwnership	External	Can Modify State	onlyOwner
getVenturePeriod	External	-	-
openLPRaiseMoney	External	Can Modify State	onlyOwner mustSetToken
setTotalGPBonusPercent	External	Can Modify State	onlyOwner
getInvestLength	External	-	-
getTokenType	External	-	-
gpRaiseMoney	External	Can Modify State	mustLessHardTop isGpRaise
lpRaiseMoney	External	Can Modify State	mustLessHardTop isLpRaise
raiseMoneyFail	External	Can Modify State	onlyOwner
getUserStock	External	-	-
getRaisedAmount	External	-	-
getManageFeeAmount	External	-	-
initManageFee	External	Can Modify State	onlyOwner
getClaimedAmount	External	-	-

VenturesManager			
getClaimedGPBonus	External	-	-
updateClaimedAmount	External	Can Modify State	onlyOwner
updateSettlementSnap	External	Can Modify State	onlyOwner
increaseSettlementSnap	External	Can Modify State	onlyOwner
getSettlementSnap	External	-	-
getVentureMaster	External	-	-
getTotalGPBonusPercent	External	-	-
getGPBonusPercent	External	-	-
getRaiseMinSoftCap	External	-	-
getRaiseMaxHardCap	External	-	-
isWhiteListEnable	External	-	-
inWhiteList	External	-	-
validateProof	Public	-	-

VenturesStockManager			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
updateOwnership	External	Can Modify State	onlyOwner
mint	External	Can Modify State	onlyOwner
burn	External	Can Modify State	onlyOwner

VenturesStockManager			
transferStock	External	Can Modify State	-

VoteExecutionManager			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
updateOwnership	External	Can Modify State	onlyOwner
generateNewSeq	External	Can Modify State	onlyOwner
registerImpeach	External	Can Modify State	onlyOwner
getImpeachGp	External	-	-
getAcceptGp	External	-	-
registerTransferVenturePeriod	External	Can Modify State	onlyOwner
registerAccept	External	Can Modify State	onlyOwner

VoteManager			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
_newVote	Internal	Can Modify State	-
_vote	Internal	Can Modify State	-
_executeVote	Internal	Can Modify State	-
_unsafeExecuteVote	Internal	Can Modify State	-
_canExecute	Internal	-	-

VoteManager			
_cancelVote	Internal	Can Modify State	-
_unsafeCancelVote	Internal	Can Modify State	-
_canCancel	Internal	-	-
_canVote	Internal	-	-
_isVoteArchive	Internal	-	-
_isVoteOpen	Internal	-	-
_isVotePending	Internal	-	-
_isCancelled	Internal	-	-
_isValuePct	Internal	-	-
updateOwnership	External	Can Modify State	onlyOwner
newVote	External	Can Modify State	onlyOwner
vote	External	Can Modify State	onlyOwner voteExists
setMinRequire	External	Can Modify State	onlyOwner
setSupportRequiredPct	External	Can Modify State	onlyOwner
canExecute	External	-	voteExists
executeVote	External	Can Modify State	onlyOwner voteExists
cancelVote	External	Can Modify State	onlyOwner voteExists
setIsOpenVote	External	Can Modify State	onlyOwner
hasVote	External	-	voteExists
getVoterState	External	-	voteExists

VoteManager			
getVote	External	-	voteExists
getVotesLength	Public	-	-
isHistoryVote	Public	-	voteExists
isActiveVote	Public	-	voteExists
isPendingVote	Public	-	voteExists
isCancelledVote	Public	-	voteExists

4.3 Vulnerability Summary

[N1] [Low] Pages calculation issue

Category: Design Logic Audit

Content

In the DaoFactory contract, calculatePages is used to calculate the start index and end index of a page. The size, start index and end index are checked in the function `size <= 0 || start >= total || start < end`, but in fact, size should not be less than 0, and start should not be greater than total.

Code location: contracts/DAOCenter.sol

```
function calculatePages(
  uint256 page,
  uint256 size,
  uint256 total
)
  internal
  pure
  returns (
    uint256,
    uint256,
    uint256
```

```
    )
  {
    uint256 start = page * size;
    uint256 end = start + size;
    if (end > total) {
      end = total;
    }
    require(size <= 0 || start >= total || start < end, 'wrong params');
    uint256 pages = total / size;
    if (total % size > 0) {
      pages += 1;
    }
    return (start, end, pages);
  }
```

Solution

It is recommended to check that size is greater than or equal to 0, and start is less than or equal to total.

Status

Fixed

[N2] [Suggestion] Missing event records

Category: Others

Content

In the DaoTemplate contract, the user can modify the actionConfig parameter through the setActionConfig function, but no event recording is performed.

The same is true for the setActionConfig function in the VentureTemplate contract.

The same is true for the setCanFreeAddMember, transferOwner and updateOwnership functions in the OrgManager contract.

Code location:

contracts/templates/DaoTemplate.sol

```
  function setActionConfig(address newActionConfig) external override onlyOwner {
    actionConfig = newActionConfig;
  }
```

contracts/templates/VentureTemplate.sol

```
  function setActionConfig(address newActionConfig) external override onlyOwner {
    actionConfig = newActionConfig;
  }
```

Solution

It is recommended to record events when sensitive parameters are modified for self-inspection or community review.

Status

Fixed

[N3] [Critical] Owner update issue

Category: Design Logic Audit

Content

In the DaoTemplate contract, the owner can update the owner of all components through the updateOwnership function. But it calls the updateOwnership interface of the templateConfig contract by mistake.

The same is true for the updateOwnership function in the VentureTemplate contract.

Code location: contracts/templates/DaoTemplate.sol

```
  function updateOwnership(address newOwner) external override onlyOwner {
    uint256 length = ITemplateConfig(templateConfig).componentLength();
    address[] memory components = ITemplateConfig(templateConfig).listComponents(0,
length);
    for (uint256 i = 0; i < components.length; i++) {
      IComponent(templateConfig).updateOwnership(newOwner);
    }
  }
```

Solution

The updateOwnership interface of the component contract should be called to update the owner.

Status

Fixed

[N4] [High] State Coverage Risk

Category: Design Logic Audit

Content

In the GrantMethodManager contract, DaoTemplate can operate the applyOp and setUserOpByOwner functions through the action contract. Since the parameters it receives are all passed in from the outside, if the incoming data is repeated, the encoded key will be repeated, which will cause the existing data to be overwritten.

Code location: contracts/components/GrantMethodManager.sol

```
  function applyOp(uint256 voteId) external override onlyOwner {
    Operation storage op_ = registerVotes[voteId];
    require(op_.exist, 'no such vote');
    bytes32 key = keccak256(abi.encodePacked(op_.gp, op_.callee, op_.method));
    Operation storage applyOp = ops[key];
    applyOp.gp = op_.gp;
    applyOp.callee = op_.callee;
    applyOp.method = op_.method;
    applyOp.exist = true;
    uint64 deadline = op_.expire + getTimestamp64();
    applyOp.expire = deadline;

    if (keccak256(abi.encodePacked(applyOp.method)) ==
keccak256(abi.encodePacked('GP_Buy'))) {
      (address token, address user, uint256 limit) = abi.decode(op_.extra, (address,
address, uint256));
      createSpendLimit(token, user, limit);
    }
    emit ApplyOperation(op_.gp, op_.callee, op_.method, deadline);
  }

  function setUserOpByOwner(
```

```
    address account,
    address callee,
    string calldata method,
    uint64 deadline
) external override onlyOwner {
    bytes32 key = keccak256(abi.encodePacked(account, callee, method));
    Operation storage applyOp = ops[key];
    applyOp.gp = account;
    applyOp.callee = callee;
    applyOp.method = method;
    applyOp.exist = true;
    applyOp.expire = deadline + getTimestamp64();
    emit ApplyOperation(account, callee, method, deadline);
}
```

Solution

It is recommended to check whether the key exists and exists.

Status

Confirmed; After communicating with the project team, the project team stated that this is the expected design.

[N5] [Suggestion] TODO label issue

Category: Others

Content

There is still a TODO label in the spendTokenInLimit function of the GrantMethodManager contract. Is there still a function not perfect?

The same is true for the _unsafeCancelVote function in the VoteManager contract.

Code location:

contracts/components/GrantMethodManager.sol

```
// TODO validate overflow
function spendTokenInLimit(
    address token,
    address user,
```

```
    uint256 spend
  ) external override onlyOwner {
    SpendLimit storage sl = spendLimitMapping[token][user];
    require(sl.spend + spend <= sl.limit, 'spend more');
    sl.spend += spend;
  }
```

contracts/components/VoteManager.sol

```
  function _unsafeCancelVote(uint256 _voteId) internal {
    Vote storage vote_ = votes[_voteId];

    vote_.st = VoteST.Cancelled;

    // bytes memory input = new bytes(0); // TODO: Consider input for voting scripts
    // runScript(vote_.executionScript, input, new address[](0));

    emit CancelVote(_voteId);
  }
```

Solution

If the label is not expected, it is recommended to remove it.

Status

Confirmed

[N6] [High] Length check issue

Category: Design Logic Audit

Content

The initialize function exists in the VenturesStockManager , VenturesManager and GovTokenManager contracts to initialize the contract according to the incoming parameters. It checks the byte length of the incoming parameter, but because some parameters are variable-length data, forcing an `equals` check will lead to unsuccessful initialization.

Code location:

contracts/components/VenturesStockManager.sol

```
  function initialize(bytes memory param) public override initializer {
    __ReentrancyGuard_init();









    __Ownable_init();

    require(param.length == 3, 'VenturesManager: init params length error!');
    (bytes memory param1, bytes memory param2, bytes memory param3) =
abi.decode(param, (bytes, bytes, bytes));
    ...
  }
```

contracts/components/VenturesManager.sol

```
  function initialize(bytes memory param) public override initializer {
    __ReentrancyGuard_init();
    __Ownable_init();

    require(param.length == 2, 'VenturesStockManager: init param length error!');
    (string memory tokenName, string memory tokenSymbol) = abi.decode(param, (string,
string));
    __ERC20_init(tokenName, tokenSymbol);
  }
```

contracts/components/GovTokenManager.sol

```
  function initialize(bytes memory param) public override initializer {
    __ReentrancyGuard_init();
    __Ownable_init();

    require(param.length == 4, 'GovTokenManager: init param length error!');
    (string memory name, string memory symbol, uint256 supply, address owner) =
abi.decode(
      param,
      (string, string, uint256, address)
```

```
    );
    ...
}
```

Solution

It is recommended to check that the byte length is equal to a certain threshold.

Status

Fixed

[N7] [Low] Vote check issue

Category: Design Logic Audit

Content

In the VoteManager contract, the _canExecute function is used to check whether the proposal can be executed, but it does not check whether the number of yes votes is greater than the number of negative votes.

Code location: contracts/components/VoteManager.sol

```
function _canExecute(uint256 _voteId) internal view returns (uint256) {
    ...
}
```

Solution

It is recommended to check that the number of affirmative votes is greater than the number of negative votes before it can be executed.

Status

Confirmed; After communicating with the project team, the project team stated that this is the expected design.

[N8] [Medium] Cancel voting issue

Category: Design Logic Audit

Content

In the VoteManager contract, the _canCancel function user checks whether the current vote can be cancelled. If the voting period for a proposal has passed, but the execution conditions are still not met, the proposal cannot be executed or cancelled.

Code location: contracts/components/VoteManager.sol

```
function _canCancel(uint256 _voteId) internal view returns (uint256) {
  ...
}
```

Solution

It is recommended to check whether the proposal has passed the voting period and still cannot be executed.

Status

Fixed

[N9] [Low] Risk of Governance Attacks

Category: Design Logic Audit

Content

DAO members can create new proposals through ActionVoteNew , ActionGrantMethodRegister , ActionVoteTransferPeriodRegister and other contracts. However, the proposal does not contain the data that needs to be executed. After the proposal is passed, the community members will pass in the specific execution data for execution. If malicious data is passed in, there is a risk that the protocol will be maliciously broken during proposal execution.

Solution

It is recommended that the data to be executed be parsed and written into the proposal when the proposal is created, so that the community can review it, and the data in the approved proposal should be used for execution.

Status

Confirmed; After communicating with the project team, the project team stated that although the proposal is not filled

with data to be executed, the executed data will be written into the register queue and displayed to community members at the front end, and the data in the register queue will be used during execution.

[N10] [Critical] Potential Fund Theft Risk

Category: Design Logic Audit

Content

As mentioned in N12, when a DAO member creates an Operation through ActionGrantMethodRegister, the user's specific execution data is not recorded in the newVote operation. Although registerOp records the `extra` data passed in by the user, it is not used `op_.extra` in actual execution. Therefore, the user can pass in valid execution data when performing the registerOp operation. And malicious data is passed in during the ActionVaultUniswapV2Router02Swap operation. This will result in funds managed by the DAO being approved for malicious router contracts, or swapping through extremely illiquid pools, allowing malicious users to easily arbitrage. This would create huge risks for DAOs.

The same is true in the ActionVaultUniswapV2SwapToken contract.

Code location: contracts/actions/ActionVaultUniswapV2Router02Swap.sol

```
  function action(address sender, bytes memory extra) external override nonReentrant
{
    // Decoding parameters
    (address router, address tokenIn, address tokenOut, uint256 amountIn, uint256
amountOutMin) = abi.decode(
      extra,
      (address, address, address, uint256, uint256)
    );

    ...

    require(IGrantMethodManager(grantAddr).checkMethodPermission(sender, vaultAddr,
'GP_Buy'), 'gp not allowed');
    IGrantMethodManager(grantAddr).spendTokenInLimit(tokenIn, sender, amountIn);

    require(IVaultManager(vaultAddr).withdraw(tokenIn, address(this), amountIn),
```

```
'withdraw tokenIn fail');

    uint256 amountOut = swapByRouter(router, tokenIn, tokenOut, amountIn,
amountOutMin);

    // transfer back
    require(IERC20(tokenOut).approve(vaultAddr, amountOut), 'approve vault fail');
    require(IVaultManager(vaultAddr).addVaultToken(tokenOut), 'add swap token fail');
    require(IVaultManager(vaultAddr).deposit(tokenOut, address(this), amountOut),
'deposit swap token fail');

   ...
  }
```

Solution

It is recommended that the data to be executed be parsed and written into the proposal when the proposal is created, so that the community can review it, and the data in the approved proposal should be used for execution. In the long run, we recommend checking the legitimacy of approved targets or externally invoked targets.

Status

Fixed; After communicating with the project team, the project team stated that this is an expected design, and users must be authorized by the community proposal to operate before the swap operation. When community users vote, they should understand that once the authorization is successful, the authorized users will have the right to freely operate the funds within the authorized amount, and the risks arising from this will also be borne by the community.

[N11] [High] Lack of access control

Category: Authority Control Vulnerability

Content

In the VenturesManager contract, the gpRaiseMoney and lpRaiseMoney functions are not restricted to be called by the owner.

Code location: contracts/components/VenturesManager.sol

```
function gpRaiseMoney(address user, uint256 _amount)
  external
  override
  mustLessHardTop(_amount)
  isGpRaise(_amount, user)
{
  ...
}

function lpRaiseMoney(address user, uint256 _amount)
  external
  override
  mustLessHardTop(_amount)
  isLpRaise(_amount, user)
{
  ...
}
```

Solution

It is recommended to restrict calls only by the owner.

Status

Fixed

[N12] [Medium] Period transfer issue

Category: Design Logic Audit

Content

In the ActionVoteTransferPeriodApply contract, when performing a period transfer, the period will be obtained through the transferPeriodStates function of the VoteExecutionManager contract, and then the period will be transferred through the transferVenturePeriod function of the VenturesManager contract. But in the current action, period is directly transferred to SettlementPeriod.

Code location: contracts/actions/ActionVoteTransferPeriodApply.sol

```
  function action(address sender, bytes memory extra) external override nonReentrant
{
    address gpcom = ITemplateConfig(templateConfig).labelComponent('OrgManager');
    bool isInOrg = IOrgManager(gpcom).isOwnerOrMemberOrModerator(sender);
    require(isInOrg, 'you must be join the org first');
    // Decoding parameters
    uint256 voteId = abi.decode(extra, (uint256));
    // Get the components you need to use
    address voteCom = ITemplateConfig(templateConfig).labelComponent('VoteManager');

    IVote(voteCom).executeVote(voteId);

    address voteExecutionCom =
ITemplateConfig(templateConfig).labelComponent('VoteExecutionManager');

    IVoteExecutionManager(voteExecutionCom).transferPeriodStates(voteId);

    address ventureCom =
ITemplateConfig(templateConfig).labelComponent('VenturesManager');
    address vaultCom =
ITemplateConfig(templateConfig).labelComponent('VaultManager');


IMiniVenturesManager(ventureCom).transferVenturePeriod(IVenturesManager.VenturePeriod
.SettlementPeriod);
    // close vote
    if (IVote(voteCom).isOpenVote()) {
      IVote(voteCom).setIsOpenVote(false);
    }

    address token = IVenturesManager(ventureCom).getTokenType();

IVenturesManager(ventureCom).updateSettlementSnap(IVaultManager(vaultCom).getBalanceO
f(token));
  }
```

Solution

The set period should be the period obtained from the VoteExecutionManager contract according to the voteId.

Status

Confirmed; After communicating with the project team, the project team stated that this is the expected design.

5 Audit Result

Audit Number	Audit Team	Audit Date	Audit Result
0X002206170002	SlowMist Security Team	2022.05.30 - 2022.06.17	Passed

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 2 critical risks, 3 high risks, 2 medium risks, 3 low risks, and 2 suggestions. And 1 high risk, 1 medium risk, 2 low risks, and 1 suggestion were 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