Smart Contract Security Audit Report

Table Of Contents

1 Executive Summary

On 2025.03.25, the SlowMist security team received the benfenorg team's security audit application for bfc smart contracts, 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 party 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.
Suggestion	There are better practices for coding or architecture.

2 Audit Methodology

The security audit process of SlowMist security team for smart contract includes two steps:

Smart contract codes are scanned/tested for commonly known and more specific vulnerabilities using automated analysis tools.

Manual audit of the codes for security issues. The contracts are manually analyzed to look for any potential problems.

Following is the list of commonly known vulnerabilities that was considered during the audit of the smart contract:

- Reentrancy Vulnerability
- Replay Vulnerability
- Reordering Vulnerability
- Denial of Service Vulnerability
- Transaction Ordering Dependence Vulnerability
- Race Conditions Vulnerability
- Authority Control Vulnerability
- Integer Overflow and Underflow Vulnerability
- TimeStamp Dependence Vulnerability
- Unsafe External Call Audit
- Scoping and Declarations Audit

3 Project Overview

3.1 Project Introduction

Bfc is a next-generation smart contract platform with high throughput, low latency, and an asset-oriented programming model powered by the Move programming language.

3.2 Vulnerability Information

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

NO	Title	Category	Level	Status
N1	The issue of BUSD token not being processed correctly in the send_token function.	Design Logic Audit	Medium	Fixed
N2	There is a risk in having no expiration time for the signature information.	Design Logic Audit	Low	Acknowledged
N3	Missing event records	Others	Suggestion	Fixed
N4	Explanation of the deposit_external_coin_v2 event	Design Logic Audit	Information	Acknowledged
N5	Explanation of the pre_deposit_external_coin v2 event	Design Logic Audit	Information	Acknowledged
N6	The issue of lack of permission control in register_foreign_token	Others	Low	Acknowledged
N7	The limit of max_mint_busd_limit has not been used.	Others	Suggestion	Fixed
N8	Risk of unhandled messages in BCS.	Others	Suggestion	Fixed
N9	Suggest parameter checking for try_create_next_committee	Others	Suggestion	Fixed
N10	Risk of excessive authority	Authority Control Vulnerability Audit	Medium	Acknowledged
N11	Preemptive Initialization	Race Conditions Vulnerability	Suggestion	Acknowledged
N12	Missing event records	Others	Suggestion	Acknowledged
N13	Dao's management role has a centralization problem	Authority Control Vulnerability Audit	Medium	Acknowledged

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.

Audit version:

https://github.com/benfenorg/bfc

Audit Commit: b07f799795b79654bbcf6d92405e3c90cb933d38

Review Commit: 921407bd0080eb9fa706e2f5c8f468d25f7e460e

Scope:

packages/bridge/ *.move

packages/bfc-system/ *.move

packages/sui-framework/ *.move

4.2 Visibility Description

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

bfc_system::bfc_system		
Public Function Name	Params Check	Auth Objects
create_stake_manager_key	1/1	-
unstake_manager_key	3/3	-
change_round	2/2	-
request_gas_balance	2/2	-
load_bfc_system_state	1/1	-
load_bfc_system_state_mut	1/1	-
get_exchange_rate	1/1	-

bfc_system::bfc_system		
remove_propose	3/3	BFCDaoManageKey
remove_action	3/3	BFCDaoManageKey
destroy_terminated_proposal	4/4	BFCDaoManageKey
propose	7/7	-
create_bfcdao_action	5/5	-
judge_proposal_state	2/2	-
set_voting_period	3/3	BFCDaoManageKey
set_voting_quorum_rate	3/3	BFCDaoManageKey
set_min_action_delay	3/3	BFCDaoManageKey
withdraw_voting	3/3	-
create_voting_bfc	3/3	-
rebalance	3/3	-
rebalance_with_one_stablecoin	3/3	-
swap_bfc_to_stablecoin	7/7	-
swap_stablecoin_to_bfc	7/7	-
get_stablecoin_by_bfc	2/2	-
get_bfc_by_stablecoin	2/2	-
vault_info	1/1	-
vault_ticks	1/1	-
vault_positions	1/1	-
total_supply	1/1	-
get_bfc_exchange_rate	1/1	-

bfc_system::bfc_system		
get_stablecoin_exchange_rate	1/1	-
bfc_required	1/1	-
next_epoch_bfc_required	1/1	-
bfc_required_with_one_stablecoin	1/1	-
treasury_balance	1/1	-
deposit_to_treasury	1/1	-
deposit_to_treasury_inner	2/2	-
deposit_to_treasury_pool	1/1	-
deposit_to_treasury_pool_no_entry	2/2	-
vault_set_pause	3/3	TreasuryPauseCap
set_voting_delay	3/3	BFCDaoManageKey
cast_vote	6/6	-
change_vote	6/6	-
queue_proposal_action	3/3	BFCDaoManageKey
revoke_vote	6/6	-
unvote_votes	4/4	-
vote_of	3/3	-
has_vote	2/2	-
get_operation_capability	2/2	-
get_operation_capability_by_key	3/3	-
add_operation_capability	5/5	BfcSystemAdminCap
remove_operation_capability	5/5	BfcSystemAdminCap

bfc_system::bfc_system		
set_operation_capability	5/5	BfcSystemAdminCap
set_single_operation_capability	5/5	BfcSystemAdminCap
set_oracle_address	3/3	-
get_oracle_address	2/2	-
mint_stable_entry	4/4	BfcSystemModifyCap
mint_stable	4/4	BfcSystemModifyCap
init_admin_capability	3/3	-
init_single_admin_capability	3/3	-
add_admin_capability	4/4	BfcSystemAdminCap
remove_admin_capability	4/4	BfcSystemAdminCap
judge_proposal_state_with_clock	2/2	-
add_external_stable_gas_coin	2/2	verify_admin_capability
delete_external_stable_gas_coin	3/3	verify_admin_capability

bfc_system::bfc_system_state_inner		
Public Function Name	Params Check	Auth Objects
next_epoch_bfc_required	1/1	-
treasury_balance	1/1	-
vault_info	1/1	-
vault_ticks	1/1	-
vault_positions	1/1	-
get_total_supply	1/1	-
vault_set_pause	3/3	TreasuryPauseCap

bfc_system::clmm_math		
Public Function Name	Params Check	Auth Objects
get_liquidity_from_a	3/4	-
get_liquidity_from_b	3/4	-
get_delta_a	3/4	-
get_delta_b	3/4	-
get_next_sqrt_price_a_up	4/4	-
get_next_sqrt_price_b_down	4/4	-
get_next_sqrt_price_from_input	4/4	-
get_next_sqrt_price_from_output	4/4	-
get_delta_up_from_input	4/4	-
get_delta_down_from_output	4/4	-
compute_swap_step	6/6	-
get_amount_by_liquidity	6/6	-
get_liquidity_by_amount	6/6	-

bfc_system::treasury		
Public Function Name	Params Check	Auth Objects
index	1/1	-
get_balance	1/1	-
get_vault_key	0/0	-
vault_info	1/1	-
fetch_ticks	1/1	-
fetch_positions	1/1	-

bfc_system::treasury		
mint	7/7	-
redeem	7/7	-
calculate_swap_result	3/3	-

bfc_system::position		
Public Function Name	Params Check	Auth Objects
get_vault_id	1/1	-
is_empty	1/1	-
get_liquidity	1/1	-
get_tick_range	1/1	-
is_position_exist	2/2	-
get_total_positions	1/1	-
fetch_positions	3/3	-
check_position_tick_range	3/3	-

bfc_system::tick		
Public Function Name	Params Check	Auth Objects
sqrt_price	1/1	-
liquidity_net	1/1	-
tick_index	1/1	-
tick_spacing	1/1	-
fetch_ticks	1/1	-

bfc_system::treasury_pool		
Public Function Name	Params Check	Auth Objects
get_balance	1/1	-

bfc_system::vault		
Public Function Name	Params Check	Auth Objects
calculated_swap_result_amount_out	1/1	-
calculated_swap_result_is_exceed	1/1	-
calculated_swap_result_amount_in	1/1	-
calculated_swap_result_after_sqrt_price	1/1	-
calculate_swap_result_step_results	1/1	-
get_position_amounts	3/3	-
get_position_liquidity	2/2	-
get_position_tick_range_and_price	2/2	-
fetch_ticks	1/1	-
fetch_positions	1/1	-
vault_info	1/1	-
vault_id	1/1	-
vault_current_sqrt_price	1/1	-
vault_current_tick_index	1/1	-
balances	1/1	-
get_liquidity	1/1	-
get_vault_state	1/1	-
get_last_rebalance_vault_state	1/1	-

bfc_system::vault		
bfc_required	2/2	-
min_liquidity_rate	0/0	-
max_liquidity_rate	0/0	-
base_liquidity_rate	0/0	-

sui_system::stable_pool		
Public Function Name	Params Check	Auth Objects
stable_balance	1/1	-
rewards_pool	1/1	-
pool_id	1/1	-
staked_sui_amount	1/1	-
stake_activation_epoch	1/1	-
is_preactive	1/1	-
is_inactive	1/1	-
split	3/3	-
split_staked_sui (entry)	3/3	-
join_staked_sui (entry)	2/2	-
is_equal_staking_metadata	2/2	-
pool_token_exchange_rate_at_epoch	2/2	-
pending_stake_amount	1/1	-
pending_stake_withdraw_amount	1/1	-
sui_amount	1/1	-
pool_token_amount	1/1	-

sui_system::stake_subsidy		
Public Function Name	Params Check	Auth Objects
current_epoch_subsidy_amount	1/1	-

sui_system::staking_pool		
Public Function Name	Params Check	Auth Objects
sui_balance	1/1	-
pool_id	1/1	-
staked_sui_amount	1/1	-
stake_activation_epoch	1/1	-
is_preactive	1/1	-
is_inactive	1/1	-
split	3/3	-
split_staked_sui (entry)	3/3	-
join_staked_sui (entry)	2/2	-
is_equal_staking_metadata	2/2	-
pool_token_exchange_rate_at_epoch	2/2	-
pending_stake_amount	1/1	-
pending_stake_withdraw_amount	1/1	-
sui_amount	1/1	-
pool_token_amount	1/1	-

sui_system::storage_fund		
Public Function Name	Params Check	Auth Objects

sui_system::storage_fund		
total_object_storage_rebates	0/1	-
total_balance	0/1	-

sui_system::sui_system		
Public Function Name	Params Check	Auth Objects
request_add_validator_candidate	16/16	-
request_remove_validator_candidate	2/2	-
request_add_validator	2/2	-
request_remove_validator	2/2	-
request_set_gas_price	3/3	UnverifiedValidatorOperationCap
set_candidate_validator_gas_price	3/3	UnverifiedValidatorOperationCap
request_set_commission_rate	3/3	-
set_candidate_validator_commission_rate	3/3	-
request_add_stake	3/3	-
request_add_stable_stake	3/3	-
request_add_stable_stake_non_entry	3/3	-
request_add_stake_non_entry	3/3	-
request_add_stake_mul_coin	4/4	-
request_withdraw_stake	3/3	-
request_withdraw_stable_stake	3/3	-
request_withdraw_stake_non_entry	3/3	-
request_withdraw_stable_stake_non_entry	3/3	-
report_validator	3/3	UnverifiedValidatorOperationCap

sui_system::sui_system		
undo_report_validator	3/3	UnverifiedValidatorOperationCap
rotate_operation_cap	2/2	-
update_validator_name	3/3	-
update_validator_description	3/3	-
update_validator_image_url	3/3	-
update_validator_project_url	3/3	-
update_validator_next_epoch_network_address	3/3	-
update_candidate_validator_network_address	3/3	-
update_validator_next_epoch_p2p_address	3/3	-
update_candidate_validator_p2p_address	3/3	-
update_validator_next_epoch_primary_address	3/3	-
update_candidate_validator_primary_address	3/3	-
update_validator_next_epoch_worker_address	3/3	-
update_candidate_validator_worker_address	3/3	-
update_validator_next_epoch_protocol_pubkey	4/4	-
update_candidate_validator_protocol_pubkey	4/4	-
update_validator_next_epoch_worker_pubkey	3/3	-
update_candidate_validator_worker_pubkey	3/3	-
update_validator_next_epoch_network_pubkey	3/3	-
update_candidate_validator_network_pubkey	3/3	-
pool_exchange_rates	2/2	-
pool_exchange_stable_rates	2/2	-

sui_system::sui_system		
active_validator_addresses	2/2	-

sui_system::validator_set		
Public Function Name	Params Check	Auth Objects
derive_reference_gas_price	1/1	-
total_stake	1/1	-
validator_total_stake_amount	2/2	-
validator_total_stake_amount_with_stable	3/3	-
validator_stake_amount	2/2	-
validator_stable_stake_amount	2/2	-
validator_staking_pool_id	2/2	-
validator_stable_pool_id	2/2	-
staking_pool_mappings	1/1	-
stalbe_staking_pool_mappings	1/1	-
sum_voting_power_by_addresses	3/3	-
active_validators	2/2	-
is_validator_candidate	2/2	-
is_inactive_validator	2/2	-

sui_system::voting_power		
Public Function Name	Params Check	Auth Objects
total_voting_power	0/0	-
quorum_threshold	0/0	-

bridge::bridge		
Public Function Name	Params Check	Auth Objects
committee_registration	4/4	validators
update_node_url	2/2	committee
register_foreign_token	3/3	-
send_token	5/5	-
send_busd	6/6	-
send_back_token	7/7	refund_admin
approve_token_transfer	3/3	-
get_max_mint_busd_amount	1/1	-
set_max_mint_busd_amount	5/5	BfcSystemModifyCap
claim_token	6/6	token_owner + BfcSystemModifyCap
claim_and_transfer_token	4/4	-
claim_and_transfer_busd	6/6	BfcSystemModifyCap
execute_system_message	3/3	-
get_available_claim_amount	2/2	-
pre_deposit_external_coin	7/7	external_coin_admin, coin_witness
deposit_external_coin	7/7	external_coin_admin, coin_witness
deposit_external_coin_v2	7/7	external_coin_admin
approval_and_claimed_external_coin	4/4	committee
withdraw_external_coin	5/5	-

bridge::message		
Public Function Name	Params Check	Auth Objects

bridge::message		
extract_token_bridge_payload	1/1	-
extract_add_witness_poyload	1/1	-
extract_remove_witness_poyload	1/1	-
extract_add_external_target_address_poyload	1/1	-
extract_remove_external_target_address_poyload	1/1	-
extract_emergency_op_payload	1/1	-
extract_blocklist_payload	1/1	-
extract_refund_admin_payload	1/1	-
extract_update_bridge_limit	1/1	-
extract_add_external_coin_admin	1/1	-
extract_remove_external_coin_admin	1/1	-
extract_update_asset_price	1/1	-
extract_add_tokens_on_sui	1/1	-
create_bitcoin_message	6/6	-
create_token_bridge_message	9/9	-
create_emergency_op_message	3/3	-
create_blocklist_message	4/4	-
create_refund_admin_message	4/4	-
create_update_bridge_limit_message	4/4	-
create_update_asset_price_message	4/4	-
create_add_external_coin_admin_message	4/4	-
create_add_external_coin_witness_message	4/4	-

bridge::message		
create_remove_external_coin_witness_message	4/4	-
create_add_external_coin_target_message	4/4	-
create_remove_external_coin_target_message	4/4	-
create_remove_external_coin_admin_message	4/4	-
create_add_tokens_on_sui_message	6/6	-
required_voting_power	1/1	-

bridge::chain_ids		
Public Function Name	Params Check	Auth Objects
sui_mainnet	0/0	-
sui_testnet	0/0	-
sui_custom	0/0	-
eth_mainnet	0/0	-
eth_sepolia	0/0	-
eth_custom	0/0	-
btc_mainnet	0/0	-
btc_testnet	0/0	-
route_source	1/1	-
route_destination	1/1	-
assert_valid_chain_id	1/1	-
valid_routes	0/0	-
is_valid_route	2/2	-
get_route	2/2	-

bridge::committee		
Public Function Name	Params Check	Auth Objects
verify_signatures	3/3	-

bridge::limiter		
Public Function Name	Params Check	Auth Objects
get_route_limit	1/1	-
get_mint_busd_max_limit	1/1	-
get_available_claim_amount	3/3	-
check_and_record_sending_transfer	4/4	-
update_route_limit	3/3	-

bridge::message_types		
Public Function Name	Params Check	Auth Objects
token	0/0	-
committee_blocklist	0/0	-
emergency_op	0/0	-
update_bridge_limit	0/0	-
update_asset_price	0/0	-
add_tokens_on_sui	0/0	-
add_external_coin_admin	0/0	-
remove_external_coin_admin	0/0	-
refund_admin_operate	0/0	-
add_bitcoin_witness	0/0	-

bridge::message_types		
remove_bitcoin_witness	0/0	-

bridge::treasury		
Public Function Name	Params Check	Auth Objects
token_id	1/1	-
decimal_multiplier	1/1	-
notional_value	1/1	-
verify_bitcoin_signatures	6/6	-

4.3 Vulnerability Summary

[N1] [Medium] The issue of BUSD token not being processed correctly in the send_token function.

Category: Design Logic Audit

Content

In the `send_token` function, when `token_id` is 5, it will trigger `assert!(token_id != 5, EUseSendBusd);`

This will cause the logic of `if (token_id == 5 && bfc_system_state.is_some())` to never be executed.

- sui-framework/packages/bridge/sources/bridge.move

```
public fun send_token<T>(
        bridge: &mut Bridge,
        bfc_system_state: &mut Option<BfcSystemState>,
        target_chain: u8,
        target_address: vector<u8>,
        token: Coin<T>,
        ctx: &mut TxContext
    ) {
        ...
        assert!(token_id != 5, EUseSendBusd);

        ...


        if (token_id == 5 && bfc_system_state.is_some()) {
```

```
            bfc_system_state.borrow_mut().burn_stable(
                token,
                ctx
            );
        } else {
            inner.treasury.burn(token);
        };

        ...
    }
```

Solution

Need to confirm whether the send_token function can handle the case where token_id == 5. If it is not necessary to handle, then delete the corresponding judgment.

Status

Fixed

[N2] [Low] There is a risk in having no expiration time for the signature information.

Category: Design Logic Audit

Content

The signed message does not contain a valid timestamp, which may result in previously signed but failed messages being collected and resubmitted for approval after the signer's weight is increased.

```
public fun serialize_message(message: BridgeMessage): vector<u8> {
    let BridgeMessage {
        message_type,
        message_version,
        seq_num,
        source_chain,
        payload
    } = message;

    let mut message = vector[
        message_type,
        message_version,
    ];

    message.append(reverse_bytes(bcs::to_bytes(&seq_num)));
    message.push_back(source_chain);
    message.append(payload);
    message
```

```
}
```

Solution

Need to add the signing time in the signature information, and check if the signature time has expired in the contract.

Status

Acknowledged; Project party: Will ensure that once the signature is generated, it can be executed successfully.

[N3] [Suggestion] Missing event records

Category: Others

Content

The modification of the following key parameters did not have corresponding event records.

- packages/bridge/sources/treasury.move

```
add_external_coin_admin
remove_external_coin_admin
add_external_coin_witness
execute_remove_external_coin_witness
add_external_coin_target
remove_external_coin_target
```

Solution

Record events corresponding to parameter changes.

Status

Fixed

[N4] [Information] Explanation of the deposit_external_coin_v2 event

Category: Design Logic Audit

Content

In deposit external coin v2, after executing the logic, the key was not stored in the external bridge records, which can lead to duplicate message submissions.This part meets the design requirements. The backend for message processing will filter out the duplicate parts.

- packages/bridge/sources/message.move

```
    public fun deposit_external_coin_v2<T>(
        bridge: &mut Bridge,
        source_chain: u8,
        source_address: vector<u8>,
        target_address: vector<u8>,
        amount: u64,
        tx_hash: ascii::String,
        _signatures: vector<u8>,
        ctx: &mut TxContext
    ) {
        let sender = ctx.sender();
        let coin_type = type_name::into_string(type_name::get<T>());

        let inner = load_inner_mut(bridge);
        assert!(!inner.paused, EBridgeUnavailable);
        assert!(chain_ids::is_valid_route(source_chain, inner.chain_id),
EInvalidBridgeRoute);
        if (!inner.treasury.is_external_coin_admin(coin_type,
sender.to_ascii_string())) {
            abort EUnknownExternalCoinOrSender
        };

        let key = ExternalBridgeMessageKey{
            source_chain,
            source_address,
            target_address,
            amount,
            tx_hash,
        };
        if (!inner.multi_signature_passed(key, coin_type)) {
            abort EUnpassedMultiSignature
        };

        // check records
        let key = ExternalBridgeMessageKey{
            source_chain,
            source_address,
            target_address,
            amount,
            tx_hash,
        };
        if (inner.external_bridge_records.contains(key)) {
            abort EDuplicatedMessage
        };
        let seq_num = inner.get_current_seq_num_and_increment(message_types::token());
        let token_id = inner.treasury.token_id<T>();

        emit(
```

```
            ExternalDepositStartEvent {
                seq_num,
                tx_hash,
                token_id,
                source_chain,
                target_chain: inner.chain_id,
                source_address,
                target_address,
                amount,
            },
        )
    }
```

Solution

Status

Acknowledged

[N5] [Information] Explanation of the pre_deposit_external_coin v2 event

Category: Design Logic Audit

Content

In deposit external coin v2, after executing the logic, the key was not stored in the external bridge records, which can lead to duplicate message submissions.

This part meets the design expectations, and the backend will filter out duplicate events that can be triggered multiple times.

- packages/bridge/sources/bridge.move

```
    public fun deposit_external_coin_v2<T>(
        bridge: &mut Bridge,
        source_chain: u8,
        source_address: vector<u8>,
        target_address: vector<u8>,
        amount: u64,
        tx_hash: ascii::String,
        _signatures: vector<u8>,
        ctx: &mut TxContext
    ) {
        let sender = ctx.sender();
        let coin_type = type_name::into_string(type_name::get<T>());
```

```
        let inner = load_inner_mut(bridge);
        assert!(!inner.paused, EBridgeUnavailable);
        assert!(chain_ids::is_valid_route(source_chain, inner.chain_id),
EInvalidBridgeRoute);
        if (!inner.treasury.is_external_coin_admin(coin_type,
sender.to_ascii_string())) {
            abort EUnknownExternalCoinOrSender
        };

        let key = ExternalBridgeMessageKey{
            source_chain,
            source_address,
            target_address,
            amount,
            tx_hash,
        };
        if (!inner.multi_signature_passed(key, coin_type)) {
            abort EUnpassedMultiSignature
        };

        // check records
        let key = ExternalBridgeMessageKey{
            source_chain,
            source_address,
            target_address,
            amount,
            tx_hash,
        };
        if (inner.external_bridge_records.contains(key)) {
            abort EDuplicatedMessage
        };
        let seq_num = inner.get_current_seq_num_and_increment(message_types::token());
        let token_id = inner.treasury.token_id<T>();

        emit(
            ExternalDepositStartEvent {
                seq_num,
                tx_hash,
                token_id,
                source_chain,
                target_chain: inner.chain_id,
                source_address,
                target_address,
                amount,
            },
        )
    }
```

Solution

Status

Acknowledged

[N6] [Low] The issue of lack of permission control in register_foreign_token

Category: Others

Content

Anyone can create a token to add to treasuries. This may result in some malicious tokens being present in the treasuries.

- packages/bridge/sources/bridge.move

```
public fun register_foreign_token<T>(
    bridge: &mut Bridge,
    tc: TreasuryCap<T>,
    uc: UpgradeCap,
    metadata: &CoinMetadata<T>,
) {
    load_inner_mut(bridge)
        .treasury
        .register_foreign_token<T>(tc, uc, metadata)
}
```

Solution

Need to add permission control to ensure that the tokens on the bridge are within expectations.

Status

Acknowledged

[N7] [Suggestion] The limit of max_mint_busd_limit has not been used.

Category: Others

Content

In the limiter contract, the max_mint_busd_limit is not used. However, in the bridge contract, there is a separate transfer for busd, but there is no separate restriction on busd.

- packages/bridge/sources/limiter.move

Solution

Suggest adding a separate restriction logic for BUSD. If it is confirmed that it is not needed, it can be deleted.

Status

Fixed

[N8] [Suggestion] Risk of unhandled messages in BCS.

Category: Others

Content

The following function does not execute bcs.into remainder bytes().is empty() check. There may be unprocessed data.

- packages/bridge/sources/message.move

```
extract_add_witness_poyload
extract_remove_witness_poyload
extract_emergency_op_payload
extract_remove_external_target_address_poyload
extract_remove_external_coin_admin
```

Solution

Check the execution of `assert!(bcs.into_remainder_bytes().is_empty(), ETrailingBytes);`.

Status

Fixed

[N9] [Suggestion] Suggest parameter checking for try_create_next_committee

Category: Others

Content

min_stake_participation_percentage is an external parameter that may have certain risks. It needs to be restricted in the contract.

- packages/bridge/sources/committee.move

```
    public(package) fun try_create_next_committee(
        self: &mut BridgeCommittee,
        active_validator_voting_power: VecMap<address, u64>,
```

```
        min_stake_participation_percentage: u64,
        ctx: &TxContext
    ) {

        let mut i = 0;
        let mut new_members = vec_map::empty();
        let mut stake_participation_percentage = 0;

        while (i < self.member_registrations.size()) {
            let (_, registration) = self.member_registrations.get_entry_by_idx(i);


            let voting_power =
active_validator_voting_power.try_get(&registration.sui_address);
            if (voting_power.is_some()) {
                let voting_power = voting_power.destroy_some();
                stake_participation_percentage = stake_participation_percentage +
voting_power;

                let member = CommitteeMember {
                    sui_address: registration.sui_address,
                    bridge_pubkey_bytes: registration.bridge_pubkey_bytes,
                    voting_power: (voting_power as u64),
                    http_rest_url: registration.http_rest_url,
                    blocklisted: false,
                };


                new_members.insert(registration.bridge_pubkey_bytes, member)
            };

            i = i + 1;
        };

        if (stake_participation_percentage >= min_stake_participation_percentage) {
            self.member_registrations = vec_map::empty();
            self.members = new_members;
            self.last_committee_update_epoch = ctx.epoch();

            emit(CommitteeUpdateEvent {
                members: new_members,
                stake_participation_percentage
            })
        }
    }
```

Solution

The value of min_stake_participation_percentage should be defined inside the contract rather than passed in externally.

Status

Fixed

[N10] [Medium] Risk of excessive authority

Category: Authority Control Vulnerability Audit

Content

The `refund_admin` executing the send back token does not require additional signature coordination. If the refund admin's private key is leaked, it may lead to serious consequences.

- packages/bridge/sources/bridge.move

```
validators can committee_registration
refund_admin can send_back_token
BfcSystemModifyCap can set_max_mint_busd_amount
BfcSystemModifyCap can claim_and_transfer_busd
committee can approval_and_claimed_external_coin<T>
external_coin_admin + coin_witness can pre_deposit_external_coin
external_coin_admin + coin_witness can deposit_external_coin
external_coin_admin + coin_witness can deposit_external_coin_v2
```

UnverifiedValidatorOperationCap can set some key parameters to control the Validator. If the private key of the UnverifiedValidatorOperationCap object is compromised, it will have an impact on the system.

- crates/sui-framework/packages/sui-system/sources/sui_system.move

```
UnverifiedValidatorOperationCap can request_set_gas_price
UnverifiedValidatorOperationCap can set_candidate_validator_gas_price
UnverifiedValidatorOperationCap can report_validator
UnverifiedValidatorOperationCap can undo_report_validator
```

`TreasuryPauseCap` can set Vault pause, `BFCDaoManageKey` can set proposal-related configurations, `BfcSystemModifyCap` can mint tokens, `BfcSystemAdminCap` can create and delete `BfcSystemModifyCap`. If the private keys controlling these 4 objects are leaked, it will cause serious impact on the system.

- crates/sui-framework/packages/bfc-system/sources/bfc_system.move

```
TreasuryPauseCap can vault_set_pause
BFCDaoManageKey can queue_proposal_action
BFCDaoManageKey can  set_voting_delay
BFCDaoManageKey can  remove_propose
BFCDaoManageKey can  remove_action
BFCDaoManageKey can  destroy_terminated_proposal
BFCDaoManageKey can  set_voting_period
BFCDaoManageKey can  set_voting_quorum_rate
BFCDaoManageKey can  set_min_action_delay

BfcSystemModifyCap can mint_stable_entry
BfcSystemModifyCap can mint_stable
BfcSystemAdminCap can add_operation_capability
BfcSystemAdminCap can remove_operation_capability
BfcSystemAdminCap can set_operation_capability
BfcSystemAdminCap can set_single_operation_capability
```

Solution

It is recommended to use multi-signature addresses for privileged role management.

Status

Acknowledged

[N11] [Suggestion] Preemptive Initialization

Category: Race Conditions Vulnerability

Content

The function to initialize the administrator does not have permission restrictions, so anyone can call it, leading to a race condition issue.

- packages/bfc-system/sources/bfc_system.move

```
public entry fun init_admin_capability(wrapper: &mut BfcSystemState, addresses:
vector<address>, ctx: &mut TxContext) {
    let (inner_state, _ctx) = load_system_state_mut(wrapper, ctx);
    bfc_system_state_inner::init_bfc_system_admins(inner_state, _ctx, addresses);
}

public fun init_single_admin_capability(wrapper: &mut BfcSystemState, address:
address, ctx: &mut TxContext) {
```

```
    let (inner_state, _ctx) = load_system_state_mut(wrapper, ctx);
    bfc_system_state_inner::add_bfc_system_admin_cap(inner_state, _ctx,
vector[address]);
}
```

Solution

Suggest adding call restrictions.

Status

Acknowledged

[N12] [Suggestion] Missing event records

Category: Others

Content

- crates/sui-framework/packages/bfc-system/sources/bfc_dao.move

```
modify_dao_config
set_voting_delay
set_voting_period
set_voting_quorum_rate
set_min_action_delay
create_stake_manager_key
unstake_manager_key
modify_proposal_obj
set_current_status_into_dao
```

- crates/sui-framework/packages/sui-system/sources/sui_system.move
-

```
update_validator_name
update_validator_description
update_validator_image_url
update_validator_project_url
update_validator_next_epoch_network_address
update_candidate_validator_network_address
update_validator_next_epoch_p2p_address
update_candidate_validator_p2p_address
update_validator_next_epoch_primary_address
update_candidate_validator_primary_address
update_validator_next_epoch_worker_address
update_candidate_validator_worker_address
update_validator_next_epoch_protocol_pubkey
```

```
update_candidate_validator_protocol_pubkey
update_validator_next_epoch_worker_pubkey
update_candidate_validator_worker_pubkey
update_validator_next_epoch_network_pubkey
update_candidate_validator_network_pubkey
```

- packages/bfc-system/sources/bfc_system.move

```
init_admin_capability
init_single_admin_capability
add_admin_capability
remove_admin_capability
set_operation_capability
set_single_operation_capability
set_oracle_address
add_operation_capability
remove_operation_capability
add_external_stable_gas_coin
delete_external_stable_gas_coin
```

Solution

Record events corresponding to parameter changes.

Status

Acknowledged

[N13] [Medium] Dao's management role has a centralization problem

Category: Authority Control Vulnerability Audit

Content

`BFCDaoManageKey` poses a high centralization risk for roles. As long as there are enough BFC payments, anyone can obtain `BFCDaoManageKey`. `BFCDaoManageKey` can call functions such as set_voting_delay, queue_proposal_action, `set_voting_quorum_rate`, `set_min_action_delay`, and `set_voting_period to` modify the configuration of the Dao. In addition, the modifications to these configurations do not require a fixed effective time and take effect immediately. Even if there are multiple `BFCDaoManageKeys`, modification can proceed with the agreement of just one `BFCDaoManageKey`, which brings significant centralization risks.

- crates/sui-framework/packages/bfc-system/sources/bfc_dao.move

Solution

It is recommended to have a delay when modifying parameters. Parameters cannot be set by a single administrator.

Status

Acknowledged

5 Audit Result

Audit Number	Audit Team	Audit Date	Audit Result
0X002504020001	SlowMist Security Team	2025.03.25 - 2025.04.02	Medium Risk

Summary conclusion: The SlowMist security team use a manual and SlowMist team's analysis tool to audit the project, during the audit work we found 3 medium risk, 2 low risk, 6 suggestion vulnerabilities.

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