MOVEBIT
Securing the Move Ecosystem

Starcoin Framework DAO Audit Report

https://twitter.com/movebit_

contact@movebit.xyz

Starcoin Framework
DAO Audit Report

1 Executive Summary

1.1 Project Information

Type	DAO
Auditor	MoveBit
Timeline	2022-09-12 to 2022-09-23
Language	Move
Methods	Architecture Review, Unit Testing, Formal Verification, Manual Review
Specification	SIP <https://github.com/starcoinorg/sips>
Source Code	V11 <https://github.com/starcoinorg/starcoin-framework/tree/v11>

1.2 Issue Statistics

Item	Count	Fixed	Pending
Total	3		3
Minor			
Medium	3		3
Major			
Critical			

1.3 Issue level

- **Minor** issues are typically suggestions relevant to best practices and readability. They do not post any immediate risks. Code owners should decide whether to fix these issues.
- **Medium** issues are not security vulnerabilities. They are not exploitable. These issues should be fixed unless there is a reason not to.
- **Major** issues are security vulnerabilities. They put a portion of users' sensitive information at risk, but are usually not directly exploitable. All major issues should be fixed.
- **Critical** issues are directly exploitable security vulnerabilities. They put users' sensitive information at risk. All critical issues should be fixed at highest priority.

1.4 Issue Status

- **Fixed:** The issue has been resolved.
- **Pending:** The issue has been acknowledged by the code owner, but has not yet been resolved. The code owner may take action to fix it in the future.

2 Summary of Findings

DAO plays a significant role in Starcoin-framework, as many other modules depend on DAO. We first took a review of the DAO architecture, then mainly focused on the code review and formal verification with the Move Prover. We have been in close contact with the Starcoin team for the past few weeks. As a result, we found a total of 3 issues.

We added formal specifications for most of the functions, except for native functions and some functions containing special elements that cannot be reasoned about (e.g., reflection, bitwise operators). All the verification code will be submitted as PR (pull requests) to the code repository, and got merged by the Starcoin team in later revisions.

Here is the list of general suggestions:

- Many files contain redundant code, which is meant to optimize the program, but is actually suboptimal and more computationally expensive than a simpler implementation, resulting in more gas usage;
- Coding style is inconsistent within the project, examples include line width limit and code indentation;
- Some method names do not conform to traditional English grammar.

3 MoveBit Audit BreakDown

MoveBit aims to assess repositories for security-related issues, code quality, and compliance with specifications and best practices. Possible issues we looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Integer overflow/underflow
- Number of rounding errors
- Denial of service / logical oversights
- Access control
- Centralization of power
- Business logic contradicting the specification
- Code clones, functionality duplication
- Gas usage
- Arbitrary token minting
- Unchecked CALL Return Values
- The flow of capability
- Witness Type

4 Methodology

The security team adopted the "**Testing and Automated Analysis**" , "**Code Review**" and "**Formal Verification**" strategy to perform a complete security test on the project party in a way that is closest to the real attack. The main entrance and scope of security testing are in the conventions in the "Audit Objective", and that can expand to the context beyond the scope based on the actual testing needs. The main types of this security audit include:

(1) Testing and Automated Analysis

Include: state consistency / failure rollback / unit test / value overflow / parameter verification

/ error unhandled / boundary check / coding specification.

(2) Code Review

See **Appendix 1** for code scope.

(3) Formal Verification

Perform formal verification for key functions with the move prover(MVP).

(4) Audit Process

- Carry out relevant security tests on the testnet or the mainnet;
- If there are any questions during the audit process, communicate with the project party in a timely manner, and the project party must actively cooperate (which may include sharing the

latest stable source code, relevant deployment scripts or methods, transaction signature scripts, exchange docking schemes, etc.);

- The necessary information on the audit process will be well recorded for the audit team and the project party communication in a timely manner.

5 Findings

5.1 Outdated documents for `DAO`

Severity: Medium
Status: Pending
Description: The DAO documentation on the website https://starcoin.org is outdated. Many commands listed there are no longer accepted by starcoin console. This is not reader-friendly and may easily frustrate novice users.

Location:

```
/// web https://starcoin.org/zh/developers/node/command/modify_dao_config/

// wrong command in tutorial
account execute-function -s 0x84b4a430c50322a66007469a645a6a06 --function 0x1::ModifyDaoConfigProposal::propose -t 0x1::STC::STC --arg 60000 600000 4u8 1000 0
// correct
account execute-function -s 0x84b4a430c50322a66007469a645a6a06 --function 0x1::ModifyDaoConfigProposal::propose -t 0x1::STC::STC --arg 60000 --arg 600000 --arg 4u8 --arg 1000 --arg 0


// wrong command in tutorial
account execute-function -s 0x84b4a430c50322a66007469a645a6a06 --function 0x1::Dao::destroy_terminated_proposal -t 0x1::STC::STC -t 0x1::ModifyDaoConfigProposal::DaoConfigUpdate --arg 0x84b4a430c50322a66007469a645a6a06 0
// correct
account execute-function -s 0x84b4a430c50322a66007469a645a6a06 --function 0x1::Dao::destroy_terminated_proposal -t 0x1::STC::STC -t 0x1::ModifyDaoConfigProposal::DaoConfigUpdate --arg 0x84b4a430c50322a66007469a645a6a06 --arg 0



/// web https://starcoin.org/zh/developer/cli/modify_onchain_config/

// wrong command in tutorial
account execute-function -s 0xc95f6a6f3845f09395a422b2f9959a97 --function 0x1::OnChainConfigScripts::propose_update_txn_publish_option --arg true true 0
// correct
account execute-function -s 0xc95f6a6f3845f09395a422b2f9959a97 --function 0x1::OnChainConfigScripts::propose_update_txn_publish_option --arg true --arg true --arg 0

// wrong command in tutorial
account execute-function -s 0xc95f6a6f3845f09395a422b2f9959a97 --function  0x1::DaoVoteScripts::unstake_vote -t 0x1::STC::STC -t 0x1::OnChainConfigDao::OnChainConfigUpdate<0x1::TransactionPublishOption::TransactionPublishOption> --arg 0xc95f6a6f3845f09395a422b2f9959a97 0
// correct
account execute-function -s 0xc95f6a6f3845f09395a422b2f9959a97 --function  0x1::DaoVoteScripts::unstake_vote -t 0x1::STC::STC -t 0x1::OnChainConfigDao::OnChainConfigUpdate<0x1::TransactionPublishOption::TransactionPublishOption> --arg 0xc95f6a6f3845f09395a422b2f9959a97 --arg 0
```

Recommendation: Update the documents.

5.2 Proposals of the same type with different values exist at the same time

Severity: Medium
Status: Pending
Description: We found that more than one `ModifyDaoConfigProposal` can exist at the same time. If they have different values, and all pass, what is the right behavior? Suppose there are two different `ModifyDaoConfigProposal` , A and B. Some users support A, and some users support B. After the final update of `ModifyDaoConfigProposa,` if the users supporting A see the result as B, it's confusing. The same problem may exist for `OnChainConfigScript` and `UpgradeModuleDaoProposal` .

Code Location:

```
// setup two ModifyDaoConfigProposal at the same time
account unlock 0x4245dd5c48ebf53f8728748d6276636b
account execute-function -s 0x4245dd5c48ebf53f8728748d6276636b --function 0x1::ModifyDaoConfigProposal::propose -t 0x1::STC::STC --arg 60000 --arg 600000 --arg 4u8 --arg 1000 --arg 0

account unlock -p "" 0xc4e7ce4f9f5983b802cb819f900c567f
account execute-function -s 0xc4e7ce4f9f5983b802cb819f900c567f --function 0x1::ModifyDaoConfigProposal::propose -t 0x1::STC::STC --arg 60000 --arg 700000 --arg 4u8 --arg 1000 --arg 0
```

Recommendation: Supporting only one proposal for the same type at the same time.

5.3 Proposals of different types exist at the same time, but an account can only vote for one

Severity: Medium
Status: Pending
Description: We found that proposals of `TransactionPublishOption::TransactionPublishOption` and `ConsensusConfig::ConsensusConfig` can exist at the same time. These proposals are of different types. We also found that one account is unable to vote for more than one kind of proposal at the same time. The below commands will issue two different types of proposals, and an account 0x848982bd8790c4d54e9b2aa4f0783700 will succeed for the first one, and fail for the second with error code ERR_VOTED_OTHERS_ALREADY. This seems to be unreasonable.

```
// setup two kinds of proposals at the same time
account execute-function -s 0xa46e618fa8f11ab045cf76ec786e692d --function 0x1::OnChainConfigScripts::propose_update_txn_publish_option --arg true --arg true --arg 0

account execute-function -s 0x4245dd5c48ebf53f8728748d6276636b --function 0x1::OnChainConfigScripts::propose_update_consensus_config --arg 500 --arg 10000 --arg 10000000000u128 --arg 10 --arg 240 --arg 24 --arg 5000 --arg 60000 --arg 2 --arg 50000000 --arg 3u8 --arg 0


// voting
// the first succeeds
account execute-function -s 0x848982bd8790c4d54e9b2aa4f0783700 --function 0x1::DaoVoteScripts::cast_vote -t 0x1::STC::STC -t 0x1::OnChainConfigDao::OnChainConfigUpdate<0x1::TransactionPublishOption::TransactionPublishOption> --arg 0xa46e618fa8f11ab045cf76ec786e692d --arg 4 --arg true --arg 666666u128

// the second fails
account execute-function -s 0x848982bd8790c4d54e9b2aa4f0783700 --function 0x1::DaoVoteScripts::cast_vote -t 0x1::STC::STC -t 0x1::OnChainConfigDao::OnChainConfigUpdate<0x1::ConsensusConfig::ConsensusConfig> --arg 0xa46e618fa8f11ab045cf76ec786e692d --arg 5 --arg true --arg 7777777u128
```

Code Location: sources/Dao.move, line 300

```
public fun cast_vote<TokenT: copy + drop + store, ActionT: copy + drop + store>(
    signer: &signer,
    proposer_address: address,
    proposal_id: u64,
    stake: Token::Token<TokenT>,
    agree: bool,
) acquires Proposal, DaoGlobalInfo, Vote {
    {
        let state = proposal_state<TokenT, ActionT>(proposer_address, proposal_id);
        // only when proposal is active, use can cast vote.
        assert!(state == ACTIVE, Errors::invalid_state(ERR_PROPOSAL_STATE_INVALID));
    };
    let proposal = borrow_global_mut<Proposal<TokenT, ActionT>>(proposer_address);
    assert!(proposal.id == proposal_id, Errors::invalid_argument(ERR_PROPOSAL_ID_MISMATCH));
    let sender = Signer::address_of(signer);
    let total_voted = if (exists<Vote<TokenT>>(sender)) {
        let my_vote = borrow_global_mut<Vote<TokenT>>(sender);
        assert!(my_vote.id == proposal_id, Errors::invalid_argument(ERR_VOTED_OTHERS_ALREADY));
        assert!(my_vote.agree == agree, Errors::invalid_state(ERR_VOTE_STATE_MISMATCH));

        do_cast_vote(proposal, my_vote, stake);
        Token::value(&my_vote.stake)
    } else {
        let my_vote = Vote<TokenT> {
            proposer: proposer_address,
            id: proposal_id,
            stake: Token::zero(),
            agree,
        };
        do_cast_vote(proposal, &mut my_vote, stake);
        let total_voted = Token::value(&my_vote.stake);
        move_to(signer, my_vote);
        total_voted
    };

    // emit event
    let gov_info = borrow_global_mut<DaoGlobalInfo<TokenT>>(Token::token_address<TokenT>());
    Event::emit_event(
        &mut gov_info.vote_changed_event,
        VoteChangedEvent {
            proposal_id,
            proposer: proposer_address,
            voter: sender,
            agree,
            vote: total_voted,
        },
    );
}
```

Recommendation: Supporting vote for different kinds of proposals at the same time.

6 Prover Formal Verification

The formal verification report of all files and modules is as follows:

ConsensusConfig

General Description
The module provides configuration of `consensus` parameters.
It mainly provides the module initialization function `initialize()` , and the function to obtain part of the configuration, `new_consensus_config()` is to create a new consensus configuration, mainly used for `DAO` .

Formally Verified Properties

- Adding `aborts_if` based on `assert` makes the program never abort.
- Verified that `genesis_address` has the necessary resources to proceed.
- In `initialize()` , it is verfied that the acoount has to be the genesis account.

ConsensusStrategy

General Description
The module provides the information of current consensus strategy.

Formally Verified Properties

- Check in `initialize()` that the blockchain must be in the genesis state and the signer must be the genesis address.
- Verified that the genesis address must have `Config<ConsensusStrategy>` and `ModifyConfigCapabilityHolder<ConsensusStrategy>` after initialization.
- Verified that `get()` returns only when `Config<ConsensusStrategy>` exists.

Dao

General Description
Dao mainly includes the following parts:DaoGlobalInfo,DaoConfig,Proposal and Vote.
Formally Verified Properties

- Verify the parameter aborts of `plugin()` . Verify `DaoGlobalInfo` , `Config<DaoConfig<TokenT>>` , `ModifyConfigCapabilityHolder<DaoConfig<TokenT>>` does not exist under `sender` when `plugin()` is called.
- Verify the parameter aborts of `propose()` .Verify the impact of the existence of resources

under the sender on the execution of the `propose()` .Integer overflow check for `propose()` ignored.

- Integer overflow check for `cast_vote()` are ignored.

DaoVoteScripts

Verified. similar to dao.

OnChainConfigDao

General Description
OnChainConfigDao is a DAO proposal for modify onchain configuration
Formally Verified Properties

- Verified that when `plugin ()` WrappedConfigModifyCapability is added to address, WrappedConfigModifyCapability does not exist under address
- When verifying the execution of the proposal, the proposer address has WrappedConfigModifyCapability
- Other parts verified.similar to dao.

OnChainConfigScripts

Verified. Similar to dao.

TransactionPublishOption

General Description
TransactionPublishOption provides an option to limit whether user can use script or publish custom modules on chain
Formally Verified Properties
Verified. This deprecated module is similar to config.

UpgradeModuleDaoProposal

General Description
A proposal module for upgrading the contract code under the token.

Formally Verified Properties

- The signer identity is the token issuer to take the next step.
- The signer does not own the UpgradeModuleCapability resource, and will have the UpgradeModuleCapability after the function is executed.

Appendix 1 - Files in Scope

This audit covered the following files:

Files	SHA-1 Hash
sources/ConsensusConfig.move	5adbe8752d299a90ffee76696dcd20b88e5eab44
sources/ConsensusStrategy.move	a96ed80411d66e82164f2c557ae6aae9d1d43eee
sources/Dao.move	237d0db3ea26a03ab8cbb95517138446859ab8d4
sources/DaoVoteScripts.move	8f7c1dd90c67fc6b96e8408ca8b722ab8bfa9293
sources/ModifyDaoConfigProposal.move	41f53c6be50f482381936400186940dc5b725447
sources/ModuleUpgradeScripts.move	dc7bfd291de644ad8db1c903e0ef11d3730cfdab
sources/OnChainConfigDao.move	89b11a439526bbe8a927426dec8515fc34c69271
sources/OnChainConfigScripts.move	357e8b7b8621be54b1ff2defec11ba2e7ce25b8c
sources/TransactionPublishOption.move	18f8abac5a1d425200ea5f931a19c2cab97c7daf
sources/TreasuryWithdrawDaoProposal.move	ac668aee48ac1f46cd74d38064c27e0eac964b2b
sources/UpgradeModuleDaoProposal.move	cf1912ee8bd0191e457d3b0856ddacccff3e9d78

Appendix 2 - Disclaimer

This report is based on the scope of materials and documents provided, with a limited review at the time provided. Results may not be complete and do not include all vulnerabilities. The review and this report are provided on an as-is, where-is, and as-available basis. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your own risk. A report does not imply an endorsement of any particular project or team, nor does it guarantee its security. These reports should not be relied upon in any way by any third party, including for the purpose of making any decision to buy or sell products, services, or any other assets. TO THE FULLEST EXTENT PERMITTED BY LAW, WE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, IN CONNECTION WITH THIS REPORT, ITS CONTENT, RELATED SERVICES AND PRODUCTS, AND YOUR USE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NOT INFRINGEMENT.

MOVEBIT

Securing the Move Ecosystem

https://twitter.com/movebit_

contact@movebit.xyz