QuillAudits

AUDIT REPORT

February, 2025

For

PoSciDonDAO

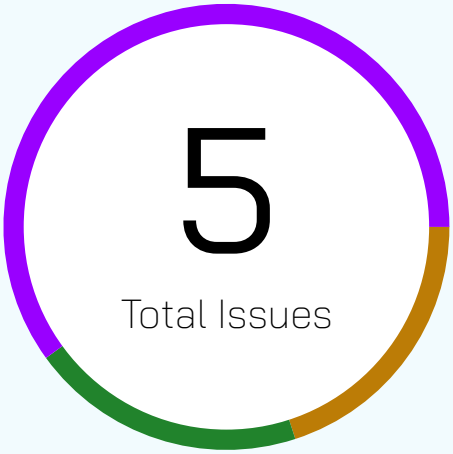
Table of Content

Executive Summary

Project name	PoSciDonDao
Overview	This contract is a token management contract (SciManager) for handling governance functionalities. It implements a system where users can lock SCI tokens to gain voting rights, with built-in timelock mechanisms for both voting and proposal actions. The contract maintains snapshots of user voting rights at different blocks for governance purposes and includes emergency functions and role-based access control
Update code Received	2025-02-19
Second Review	20th February 2025
Method	Manual Review, Functional Testing, Automated Testing, etc. All the raised flags were manually reviewed and re-tested to identify any false positives.
Blockchain	Base
Audit Scope	The scope of this Audit was to analyze the PoSciDonDao Smart Contracts for quality, security, and correctness. Git Repo link : https://github.com/PoSciDonDAO/poscidondao_contracts/blob/main/contracts/sciManager/SciManager.sol Branch: Main
Contracts In Scope	SciManager.sol
Fixed In	Branch: main 210e2c08e7349e1d8fca6bd38b80621fc50a9424

Project URL	https://www.poscidondao.com/
Commit hash	7271b184cc09ce3971be832203fcda72dbcf787d
Language	Solidity
Method	Manual Analysis, Functional Testing, Automated Testing
Review_1	11th February 2025 - 17th February 2025

Number of Issues per Severity



High	0 (0.00%)
Medium	1 (20.00%)
Low	1 (20.00%)
Informational	3 (60.00%)

Severity

Issues	High	Medium	Low	Informational
Open	0	0	0	0
Resolved	0	**1**	**1**	**3**
Acknowledged	0	0	0	0
Partially Resolved	0	0	0	0

Checked Vulnerabilities

- ☑ Access Management
- ☑ Arbitrary Write to Storage
- ☑ Centralization of Control
- ☑ Ether Theft
- ☑ Improper or Missing Events
- ☑ Logical Issues and Flaws
- ☑ Arithmetic Computations Correctness
- ☑ Race Conditions/Front Running
- ☑ SWC Registry
- ☑ Re-entrancy
- ☑ Timestamp Dependence
- ☑ Exception Disorder
- ☑ Gasless Send
- ☑ Use of tx.origin
- ☑ Malicious Libraries
- ☑ Compiler Version Not Fixed
- ☑ Address Hardcoded
- ☑ Divide Before Multiply
- ☑ Integer Overflow/Underflow
- ☑ ERC's Conformance
- ☑ Dangerous Strict Equalities
- ☑ Tautology or Contradiction
- ☑ Return Values of Low-Level Calls
- ☑ Missing Zero Address Validation
- ☑ Private Modifier
- ☑ Revert/Require Functions
- ☑ Multiple Sends
- ☑ Using Suicide
- ☑ Upgradeable Safety
- ☑ Using Throw
- ☑ Using Inline Assembly
- ☑ Style Guide Violation

- [x] Unsafe Type Inference
- [x] Implicit Visibility Level

Techniques and Methods

Throughout the audit of smart contracts, care was taken to ensure:

- The overall quality of code.
- Use of best practices.
- Code documentation and comments, match logic and expected behavior.
- Token distribution and calculations are as per the intended behavior mentioned in the whitepaper.
- Implementation of ERC standards.
- Efficient use of gas.
- Code is safe from re-entrancy and other vulnerabilities.

The following techniques, methods, and tools were used to review all the smart contracts.

Structural Analysis

In this step, we have analyzed the design patterns and structure of smart contracts. A thorough check was done to ensure the smart contract is structured in a way that will not result in future problems.

Static Analysis

A static Analysis of Smart Contracts was done to identify contract vulnerabilities. In this step, a series of automated tools are used to test the security of smart contracts.

Code Review / Manual Analysis

Manual Analysis or review of code was done to identify new vulnerabilities or verify the vulnerabilities found during the static analysis. Contracts were completely manually analyzed, their logic was checked and compared with the one described in the whitepaper. Besides, the results of the automated analysis were manually verified.

Gas Consumption

In this step, we have checked the behavior of smart contracts in production. Checks were done to know how much gas gets consumed and the possibilities of optimization of code to reduce gas consumption.

Tools And Platforms Used For Audit

Remix IDE, Foundry, Solhint, Mythril, Slither, Solidity statistical analysis.

Types of Severity

Every issue in this report has been assigned to a severity level. There are four levels of severity, and each of them has been explained below

High Severity Issues

A high severity issue or vulnerability means that your smart contract can be exploited. Issues on this level are critical to the smart contract's performance or functionality, and we recommend these issues be fixed before moving to a live environment.

Medium Severity Issues

The issues marked as medium severity usually arise because of errors and deficiencies in the smart contract code. Issues on this level could potentially bring problems, and they should still be fixed.

Low Severity Issues

Low-level severity issues can cause minor impact and are just warnings that can remain unfixed for now. It would be better to fix these issues at some point in the future.

Informational Issues

These are four severity issues that indicate an improvement request, a general question, a cosmetic or documentation error, or a request for information. There is low-to-no impact.

Types of Issues

Open

Security vulnerabilities identified that must be resolved and are currently unresolved.

Resolved

Security vulnerabilities identified that must be resolved and are currently unresolved.

Acknowledged

Vulnerabilities which have been acknowledged but are yet to be resolved.

Partially Resolved

Considerable efforts have been invested to reduce the risk/ impact of the security issue, but are not completely resolved.

Medium Severity Issues

Ownership Renouncement Risk in AccessControl Implementation

Resolved

Path

SciManager.sol

Function

NA

Description

The contract inherits from OpenZeppelin's AccessControl and uses DEFAULT_ADMIN_ROLE. If the admin renounces their role using the inherited renounceRole() function, it would set the owner address to zero, potentially leaving the contract without administrative control.

Recommendation

Override the renounceRole() function from AccessControl to prevent the admin from renouncing the DEFAULT_ADMIN_ROLE

```
function renounceRole(bytes32 role, address account)
        public
        virtual
        override
    {
        require(role != DEFAULT_ADMIN_ROLE, "AccessControl: cannot renounce Admin
role");
        super.renounceRole(role, account);
    }
```

Low Severity Issues

Admin Role Changes Should Be Two Step

Resolved

Path

SciManager.sol

Function

setAdmin()

Description

The setAdmin() function transfers administrative privileges in a single step. If the admin address is incorrectly set, it could result in a loss of administrative control over the contract.

Recommendation

Implement a two-step transfer pattern where the new admin must accept the role before the transfer is complete.

Informational Severity Issues

Incorrect Comment for voteLockEnd Parameter

Resolved

Path

SciManager.sol

Function

proposed(),voted()

Description

In the voted() and proposed() function, the comments for the voteLockEnd and proposeLockEnd parameter incorrectly states it's a block number when it's actually a timestamp.

```
/**
    * @dev is called by gov contracts upon proposing
    * @param user the user's address holding SCI tokens
    * @param proposeLockEnd the block number where the vote lock ends
    */
   function proposed(
       address user,
       uint256 proposeLockEnd
```

Recommendation

Update the comments to correctly reflect that voteLockEnd represents a timestamp

Missing Same-Address Check in setGovOps()

Resolved

Path

SciManager.sol

Function

setGovOps()

Description

The setGovOps() function lacks a check for setting the same address that is currently set, while setGovRes() includes this check. This inconsistency could lead to unnecessary state updates

Recommendation

Add the same address check as setGovRes()

Missing Total Supply Validation in Lock Function

Resolved

Path

SciManager.sol

Function

lock()

Description

The lock() function does not validate that the newly locked amount plus already locked tokens (_totLocked) doesn't exceed the TOTAL_SUPPLY_SCI. This could allow locking more tokens than should exist

Recommendation

Add validation to ensure total locked amount doesn't exceed total supply

Automated Tests

No major issues were found. Some false positive errors were reported by the tools. All the other issues have been categorized above according to their level of severity.

Closing Summary

In this report, we have considered the security of PoSciDonDao. We performed our audit according to the procedure described above.In the End,PoSciDon Dao team Resolved all Issues.

Disclaimer

At QuillAudits, we have spent years helping projects strengthen their smart contract security. However, security is not a one-time event—threats evolve, and so do attack vectors. Our audit provides a security assessment based on the best industry practices at the time of review, identifying known vulnerabilities in the submitted smart contract source code, including its compilation, deployment, and intended functionality.

This report does not serve as a security guarantee, investment advice, or an endorsement of any platform. It reflects our findings based on the provided code at the time of analysis and may no longer be relevant after any modifications. The presence of an audit does not imply that the contract is free of vulnerabilities or fully secure.

While we have conducted a thorough review, security is an ongoing process. We strongly recommend multiple independent audits, continuous monitoring, and a public bug bounty program to enhance resilience against emerging threats.

Stay proactive. Stay secure.

About QuillAudits

QuillAudits is a secure smart contracts audit platform designed by QuillHash Technologies. We are a team of dedicated blockchain security experts and smart contract auditors determined to ensure that Smart Contract-based Web3 projects can avail the latest and best security solutions to operate in a trustworthy and risk-free ecosystem

7+ Years of Expertise	**1M+** Lines of Code Audited
$30B+ Secured in Digital Assets	**1400+** Projects Secured

Follow Our Journey

AUDIT REPORT

February, 2025

For

PoSciDonDAO

QuillAudits

Canada, India, Singapore, UAE, UK

www.quillaudits.com audits@quillaudits.com