SMART CONTRACT AUDIT

ZOKYO.

November 26th, 2021 | v. 1.0

PASS

Zokyo's Security Team has concluded that this smart contract passes security qualifications to be listed on digital asset exchanges.

TECHNICAL SUMMARY

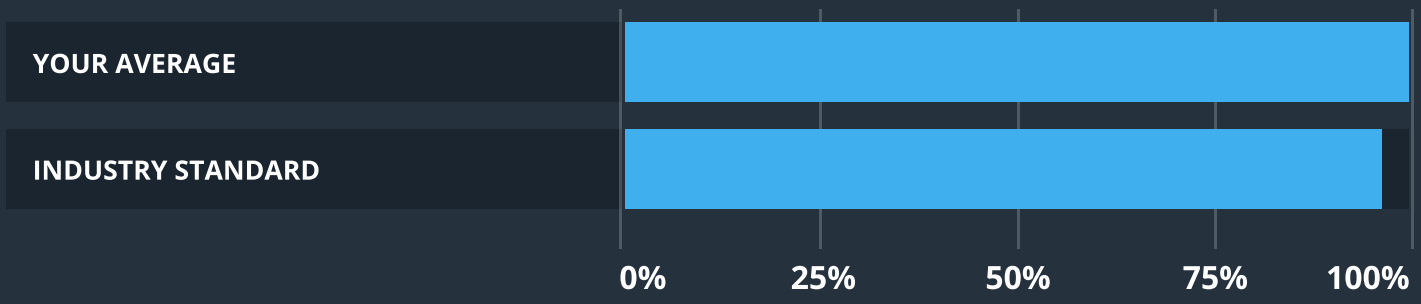
This document outlines the overall security of the SyncDAO smart contracts, evaluated by Zokyo's Blockchain Security team.

The scope of this audit was to analyze and document the SyncDAO smart contract codebase for quality, security, and correctness.

Contract Status

There were no critical and high issues found during the audit.

Testable Code



The testable code is 99,36%, which is above the industry standard of 95%.

It should be noted that this audit is not an endorsement of the reliability or effectiveness of the contract, rather limited to an assessment of the logic and implementation. In order to ensure a security of the contract we at Zokyo recommend that the SyncDAO team put in place a bug bounty program to encourage further and active analysis of the smart contract.

TABLE OF CONTENTS

AUDITING STRATEGY AND TECHNIQUES APPLIED

The SyncDAO smart contract's source code was taken from the repositories provided by the SyncDAO team: https://github.com/syncdao/governanceToken

Initial commits (audited): 50a80a7133179584416a2acd0efab3443d39a559

Last commits (post-audit): bf8985bdbc36b4918b837ac5a92a1352faf45216

Within the scope of this audit Zokyo auditors have reviewed the following contract(s):

- GovernanceDistribution.sol
- GovernanceToken.sol

Throughout the review process, care was taken to ensure that the contract:

- Implements and adheres to existing standards appropriately and effectively;
- Documentation and code comments match logic and behavior;
- Follows best practices in efficient use of resources, without unnecessary waste;
- Uses methods safe from reentrance attacks;
- Is not affected by the latest vulnerabilities;
- Whether the code meets best practices in code readability, etc.

Zokyo's Security Team has followed best practices and industry-standard techniques to verify the implementation of SyncDAO smart contracts. To do so, the code is reviewed line-by-line by our smart contract developers, documenting any issues as they are discovered. Part of this work includes writing a unit test suite. In summary, our strategies consist largely of manual collaboration between multiple team members at each stage of the review:

1	Due diligence in assessing the overall code quality of the codebase.	3	Testing contract logic against common and uncommon attack vectors.
2	Cross-comparison with other, similar smart contracts by industry leaders.	4	Thorough, manual review of the codebase, line-by-line.

EXECUTIVE SUMMARY

There were several medium and low issues discovered during audit that relate to the following unsafe ERC20 transfers, extra checks and overall code quality. The code itself is well written, though lacks gas optimisation in several places. Though, there were no critical issues found during the audit.
After recommendations by Zokyo auditors, all issues that influence security and efficiency were fixed by SyncDAO team.

STRUCTURE AND ORGANIZATION OF DOCUMENT

For ease of navigation, sections are arranged from most critical to least critical. Issues are tagged “Resolved” or “Unresolved” depending on whether they have been fixed or addressed. Furthermore, the severity of each issue is written as assessed by the risk of exploitation or other unexpected or otherwise unsafe behavior:

Critical

The issue affects the contract in such a way that funds may be lost, allocated incorrectly, or otherwise result in a significant loss.

High

The issue affects the ability of the contract to compile or operate in a significant way.

Medium

The issue affects the ability of the contract to operate in a way that doesn’t significantly hinder its behavior.

Low

The issue has minimal impact on the contract’s ability to operate.

Informational

The issue has no impact on the contract’s ability to operate.

COMPLETE ANALYSIS

Common

LOW | RESOLVED

Missing license type.

Contracts should contain license type.

Recommendation:
Specify the license type using a comment in the first line of the code of the contracts. (https://docs.soliditylang.org/en/latest/layout-of-source-files.html?highlight=license#spdx-license-identifier).

GovernanceToken.sol

INFORMATIONAL | RESOLVED

Repeatable code.

Lines 35, 40. ‘Require’ statement repeats.

Recommendation:
It is recommended to replace the requirement with hasRole function with onlyRole modifier from AccessControl.sol.
(https://docs.openzeppelin.com/contracts/4.x/api/access#AccessControl-onlyRole-bytes32-).

GovernanceDistribution.sol

MEDIUM | RESOLVED

Use safeERC20 library.

Lines 312, 327. SafeERC20 already performs all the checks, which are performed in the contract.

Recommendation:
Use SafeERC20 library instead. This is necessary in order to be able to withdraw tokens with non-standard ERC20 implementation(like USDT). SafeERC20 resolves this problem.

INFORMATIONAL | RESOLVED

Repeatable code.

Lines 162, 252, 323. 'Require' statement repeats.

Recommendation:
It is recommended to replace the requirement with hasRole function with onlyRole modifier from AccessControl.sol.
(https://docs.openzeppelin.com/contracts/4.x/api/access#AccessControl-onlyRole-bytes32-).

INFORMATIONAL | RESOLVED

Unnecessary check.

Line 294. The vesting value does not change in the code after the allocation is set, so that there is no case so that there is no case when the value is zero and the allocation exists.

Recommendation:
Remove the 'if' statement, since it can never succeed

INFORMATIONAL | RESOLVED

Unnecessary check.

Line 165. An unnecessary requirement, since this check is already included in enum type, that is, when passed to the function, AllocationType will be checked for a range of its values.

Recommendation:
Remove this requirement.

	GovernanceDistribution	GovernanceToken
Re-entrancy	Pass	Pass
Access Management Hierarchy	Pass	Pass
Arithmetic Over/Under Flows	Pass	Pass
Delegatecall Unexpected Ether	Pass	Pass
Default Public Visibility	Pass	Pass
Hidden Malicious Code	Pass	Pass
Entropy Illusion (Lack of Randomness)	Pass	Pass
External Contract Referencing	Pass	Pass
Short Address/ Parameter Attack	Pass	Pass
Unchecked CALL Return Values	Pass	Pass
Race Conditions / Front Running	Pass	Pass
General Denial Of Service (DOS)	Pass	Pass
Uninitialized Storage Pointers	Pass	Pass
Floating Points and Precision	Pass	Pass
Tx.Origin Authentication	Pass	Pass
Signatures Replay	Pass	Pass
Pool Asset Security	Pass	Pass

CODE COVERAGE AND TEST RESULTS FOR ALL FILES

Tests written by Zokyo Secured team

As part of our work assisting SyncDAO in verifying the correctness of their contract code, our team was responsible for writing integration tests using Truffle testing framework.

Tests were based on the functionality of the code, as well as review of the SyncDAO contract requirements for details about issuance amounts and how the system handles these.

GovernanceDistribution
- ✓ Administrator can refund other tokens (810ms)

Setting of an allocation
- ✓ Distributor can set an allocation (272ms)
- ✓ Distributor can set an allocation to add tokens (241ms)
- ✓ Distributor can set allocations with different types (1345ms)
- ✓ Distributor can renew a canceled allocation (235ms)
- ✓ Distributor can not set an allocation if a recipient zero address (49ms)
- ✓ Distributor can not set an allocation if a zero allocated amount (62ms)
- ✓ Distributor can not change an allocation type (158ms)
- ✓ Distributor can not change an allocation release status (153ms)

Cancellation of an allocation
- ✓ Distributor can cancel an allocation (190ms)
- ✓ Distributor can cancel allocations with different types (1714ms)
- ✓ Distributor can not cancel if a nonexistent allocation (67ms)
- ✓ Distributor can not cancel if an allocation with claimed tokens (238ms)

Claiming
- ✓ Everybody can claim (256ms)
- ✓ Everybody can claim if an instant release allocation (187ms)
- ✓ Everybody can claim if a monthly unlock style (196ms)
- ✓ Nobody can claim if a nonexistent allocation (66ms)
- ✓ Nobody can claim if an allocation have already been transferred (239ms)
- ✓ Nobody can claim if tokens for the period have already been transferred (298ms)

Only a distributor has the listed abilities
- ✓ Setting of an allocation (92ms)
- ✓ Cancellation of an allocation (74ms)

GovernanceToken
- ✓ Pause functionality (555ms)

22 passing (10s)

FILE	% STMTS	% BRANCH	% FUNCS	% LINES	UNCOVERED LINES
contracts/	99.36	93.18	100.00	99.27	
GovernanceDistribution.sol	99.33	92.86	100.00	99.23	295
GovernanceToken.sol	100.00	100.00	100.00	100.00	
All files	**99.36**	**93.18**	**100.00**	**99.27**	

We are grateful to have been given the opportunity to work with the SyncDAO.

The statements made in this document should not be interpreted as investment or legal advice, nor should its authors be held accountable for decisions made based on them.

Zokyo's Security Team recommends that the SyncDAO put in place a bug bounty program to encourage further analysis of the smart contract by third parties.

ZOKYO.