SMART CONTRACT AUDIT

ZOKYO.

April 28th, 2022 | v. 1.0

PASS

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

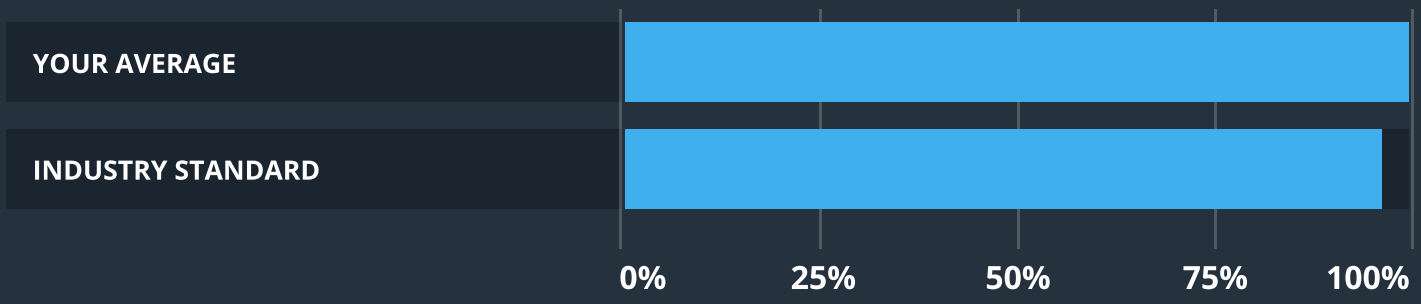
TECHNICAL SUMMARY

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

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

Contract Status

Testable Code



The testable code is 98%, 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 Symbiosis 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 Smart contract's source code was taken from the Symbiosis repository.

Repository:
https://github.com/symbiosis-finance/terra-audit
Initial commit
154852354f49f893e7dd69e9134d3ab76edc14f1
Last commit
9d47b1e0cf0adb1f596cb0ed8401ca28311a5ce2

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

- metarouter_gateway
- metarouter

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;
- Distributes tokens in a manner that matches calculations;
- 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 Symbiosis 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.
2	Cross-comparison with other, similar smart contracts by industry leaders.
3	Testing contract logic against common and uncommon attack vectors.
4	Thorough, manual review of the codebase, line-by-line.

EXECUTIVE SUMMARY

There were no critical issues found during the audit. All the mentioned findings may have an effect only in case of specific conditions performed by the contract owner or are connected to the code quality, performance and best practices following.

Contracts are well written and structured. The findings during the audit have no impact on contract performance or security, so it is fully production-ready.

Symbiosis team has successfully fixed all issues discovered during the audit. Contracts logic was also verified within the testset written by Zokyo Security 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

INFORMATIONAL | RESOLVED

Clones an owned value that is going to be dropped without further use
metarouter_gateway/src/execute.rs:30:35
Line 30. It is not always possible for the compiler to eliminate useless allocations and deallocations generated by redundant clone()s.

Recommendation:
remove to_string()

INFORMATIONAL | RESOLVED

Duplicate code
metarouter/src/execute.rs
Line. 34-56. Duplicate code makes your codebase unnecessary large and adds extra technical debt. 34-44 and 46-56 parts of the code differ only in the variables

Recommendation:
It is better to use a closures instead of duplicating the code

INFORMATIONAL | RESOLVED

Duplicate code
metarouter/src/execute.rs
Line. 58-72. Duplicate code makes your codebase unnecessary large and adds extra technical debt. 58-64 and 66-72 parts of the code differ only in the variables

Recommendation:
It is better to use a closures instead of duplicating the code

INFORMATIONAL | RERESOLVED

Unnecessary wrap
metarouter/src/query.rs
Line. 5. There is no need to wrap type into StdResult since function always return Ok response

Recommendation:
Remove StdResult returning type and leave a raw type

INFORMATIONAL | RESOLVED

Needlessly taken reference of left operand
metarouter/src/execute.rs
Line. 22,26. Checks for arguments to == which have their address taken to satisfy a bound and suggests to dereference the other argument instead

Recommendation:
Use the left value directly: `number_router`

	metarouter_gateway	metarouter
Re-entrancy	Pass	Pass
Access Management Hierarchy	Pass	Pass
Arithmetic Over/Under Flows	Pass	Pass
Unexpected Ether	Pass	Pass
Delegatecall	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 (backdoors in the underlying ERC-20)	Pass	Pass

CODE COVERAGE AND TEST RESULTS FOR ALL FILES

Tests written by Zokyo Secured team

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

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

List of methods

contracts/metarouter_gateway/

src/contract.rs
- ✓ instantiate
- ✓ execute
- ✓ query
- ✓ migrate

src/error.rs

src/execute.rs
- ✓ claim_tokens

src/lib.rs

src/query.rs
- ✓ metarouter

src/state.rs
- ✓ is_metarouter

contracts/metarouter/

src/contract.rs
- ✓ instantiate
- ✓ reply
- ✓ execute
- ✓ query
- ✓ migrate

src/error.rs

src/execute.rs
- ✓ meta_route_v2

src/lib.rs

src/query.rs
- ✓ gateway

src/state.rs

✓ is_transmitter

List of native tests

contracts/metarouter/

contracts/metarouter_gateway/

src/contract.rs

✓ test_instantiate

List of tests by Zokyo team

contracts/metarouter/

src/contract.rs

✓ test_instantiate
✓ test_reply
✓ test_reply_unimplemented
✓ test_meta_route_v2_no_cw20route_no_calldata
✓ test_meta_route_v2_wrong_first_router
✓ test_meta_route_v2_wrong_second_router
✓ test_meta_route_v2_cw20route0
✓ test_meta_route_v2_cw20route1
✓ test_meta_route_v2_calldata1
✓ test_meta_route_v2_calldata2
✓ test_meta_route_v2_cw20route0_cw20route1
✓ test_meta_route_v2_cw20route0_calldata1
✓ test_meta_route_v2_cw20route0_calldata2
✓ test_meta_route_v2_cw20route1_calldata1
✓ test_meta_route_v2_cw20route1_calldata2
✓ test_meta_route_v2_calldata1_calldata2
✓ test_meta_route_v2_cw20route1_calldata1_calldata2
✓ test_meta_route_v2_cw20route0_cw20route0_calldata1
✓ test_meta_route_v2_cw20route0_cw20route1_calldata2
✓ test_meta_route_v2_cw20route0_calldata1_calldata2
✓ test_meta_route_v2_cw20route0_cw20route1_calldata1_calldata2
✓ test_query
✓ test_migrate

contracts/metarouter_gateway/

src/contract.rs

✓ test_claim_native
✓ test_claim_cw20
✓ test_claim_unauthorized
✓ test_query
✓ test_migrate

FILE	% BRANCH	% FUNCS	% LINES	Uncovered Lines
metarouter/src	100	83.12	100	1333 / 1333
metarouter_gateway/src	100	67.44	100	222 / 222
All files	**100**	**77.5**	**100**	

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

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 Symbiosis team put in place a bug bounty program to encourage further analysis of the smart contract by third parties.

ZOKYO.