ZOKYO.

ZERO Exchange

Smart Contract Audit

Report date	January 15th, 2021
Report version	1.0

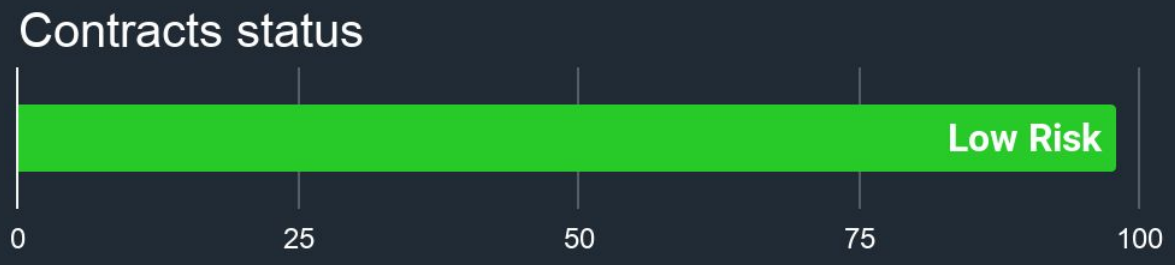
PASS

Zokyo's Blockchain 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 ZERO Exchange smart contracts, evaluated by Zokyo's Blockchain Security team.

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



There were no critical issues found during the audit. (See Complete Analysis)

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 secure contract that's able to withstand the Ethereum network's fast-paced and rapidly changing environment, we at Zokyo recommend that the ZERO Exchange 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

Summary

Structure and Organization of Document

Complete Analysis

- Informational, Unresolved: Overpowered role

Auditing Strategy and Techniques Applied

The Smart contract's source code was taken from:

1) ZERO Exchange: the contracts repo master branch (commit - _____)
2) ZERO Bridge: the _____ repository _____ branch (commit - _____)
3) ZERO Token: the _____ repository _____ branch (commit - _____)

Requirements: ZERO Exchange contracts are the fork of the Uniswap contracts and inherit the logic from Uniswap exchange. ZERO Bridge contracts are the fork of the ChainBridge system and inherit all the logic. ZERO Token is the standard ERC-20 token which implements ERC-3009.

ZERO Exchange consists of:

1. ZEROERC20.sol - Liquidity Pool token.
2. ZEROFactory.sol - Factory for deployment of Liquidity Pools.
3. ZEROPair - Liquidity Pool.

ZERO Bridge consists of:

1. Bridge.sol - contract with bridging logic.
2. ERC20Handler.sol - service contract with logic for managing the ERC20 transfers to/from bridge.
3. ERC721Handler.sol - service contract with logic for managing the ERC721 transfers to/from bridge.

ZERO Token is:

1. ZERO.sol - pure ERC-20 logic with ERC-3009 implementation for enabling metatransactions.

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

- Implements and adheres to existing Token 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 gas, 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 xBTC 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 using the Truffle testing framework. 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.

Summary

There were no critical issues found during the **manual audit**. All the mentioned findings may have effect only in case of specific conditions performed by the contract creator or contract owner. Also, there are no issues related to compliance with requirements. It is strongly recommended to fix mentioned low severity findings and setup multisig contract for managing admin-only functionality (set FeeTo, FeeToSetter in Factory contract). Although there are no issues with medium and higher severity level.

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:

- **Informational** - The issue has no impact on the contract’s ability to operate.
- **Low** - The issue has minimal impact on the contract’s ability to operate.
- **Medium** - The issue affects the ability of the contract to operate in a way that doesn’t significantly hinder its behavior.
- **High** - The issue affects the ability of the contract to compile or operate in a significant way.
- **Critical** - The issue affects the contract in such a way that funds may be lost, allocated incorrectly, or otherwise result in a significant loss.

Complete Analysis

Low severity, Unresolved: Lack of zero-check

- ERC20Handler.sol, line 39: constructor lacks a zero-check on “bridgeAddress” address
- GenericHandler.sol, line 71: constructor lacks a zero-check on “bridgeAddress” address
- ERC721Handler.sol, line 47: constructor lacks a zero-check on “bridgeAddress” address

Informational, Unresolved: Overpowered role

Factory.sol: The FeeTo and FeeToSetter addresses have unique permissions to change the setter and collect the fees. It is strongly recommended to setup the multisig contract and assign it these roles.

Informational, Unresolved: Different Solidity versions

Different pragma directives are used. Version used: 0.6.12, ^0.6.0, ^0.6.2

Informational,Unresolved: State variables that could be declared constant

State variables should be constant to save gas:

1. Bridge.sol, line 22: _chainID
2. Bridge.sol, line 25: _expiry

Informational,Unresolved: Variables that could be declared external

There are many public variables in the Bridge contracts that could be declared external to save gas.

We are grateful to have been given the opportunity to work with the ZERO Exchange 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 ZERO Exchange team put in place a bug bounty program to encourage further analysis of the smart contract by third parties.