Vortex DeFi

SMART CONTRACT AUDIT

ZOKYO.

Feb 25, 2021 | v. 2.0

PASS

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

TECHNICAL SUMMARY

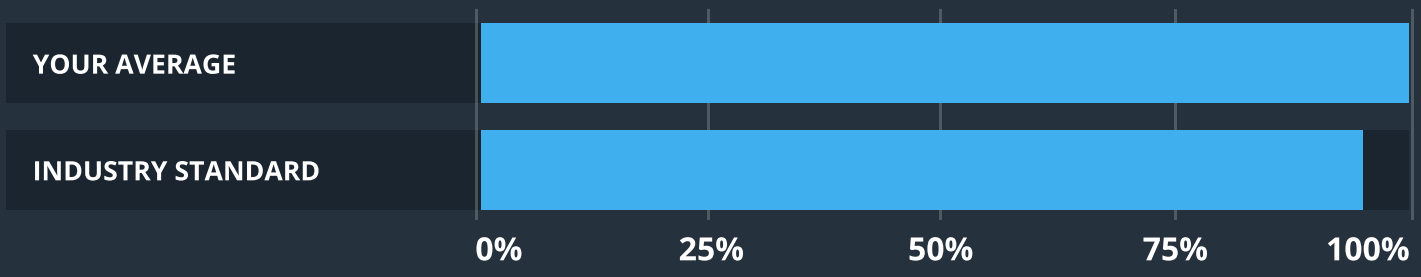
This document outlines the overall security of the Vortex smart contracts, evaluated by Zokyo's Blockchain Security team.

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

Contract Status

There were no critical issues found during the audit.

Testable Code



Testable code is 100% 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 secure contract that's able to withstand the Ethereum network's fast-paced and rapidly changing environment, we at Zokyo recommend that the Vortex 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 following github repository – https://github.com/DefiWizard/vortex-contracts.git.
Commit id – 07cdb3da5e52dac85ff00a0d9df5da95d5586a86.

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

- 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 Vortex smart contracts. To do so, the code is reviewed line-by-line by our smart contract developers, documenting any issues as they are discovered. 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

Zokyo's Security team would like to assess Vortex's smart contracts from two aspects: Technical and Economical.

Technical Summary. Based on the code analyzed, we can give a Good score to the codebase provided. The issues found and described below are stopping us from giving the Excellent score.

Economical Summary. Given the growing demand for review against possible economical exploits, we recommend every interested party to get acquainted with the findings below describing potential discrepancies, specific nature of how certain elements of the Vortex ecosystem.

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

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.

MANUAL REVIEW

HIGH | UNRESOLVED

Vortex contract deployer gets complete total supply.

Recommendation:
Auto distribute tokens during deployment of Token contract.

HIGH | UNRESOLVED

Vesting details are not defined in any public resources.

HIGH | UNRESOLVED

Rewards earned as part of staking tokens are not guaranteed to be transferred as they are not allocated by staking contract.

MEDIUM | RESOLVED

package.json, development dependency (lodash) added to dependencies.

MEDIUM | RESOLVED

package.json, dependency (openzeppelin/contracts) added to development dependencies.

LOW | UNRESOLVED

Smart contracts are not completely covered by NatSpec annotations.

LOW | RESOLVED

Variables used to calculate available percentage at PrivateDistribution contract can be defined as constant.

INFORMATIONAL | UNRESOLVED

Implement one method to set timestamp & initialized flag in PrivateDistribution contract same as in VortexTokenVesting contract.

	VortexToken	VortexTokenVesting
Re-entrancy	Not affected	Not affected
Access Management Hierarchy	Not affected	Not affected
Arithmetic Over/Under Flows	Not affected	Not affected
Unexpected Ether	Not affected	Not affected
Delegatecall	Not affected	Not affected
Default Public Visibility	Not affected	Not affected
Hidden Malicious Code	Not affected	Not affected
Entropy Illusion (Lack of Randomness)	Not affected	Not affected
External Contract Referencing	Not affected	Not affected

Short Address/ Parameter Attack	Not affected	Not affected
Unchecked CALL Return Values	Not affected	Not affected
Race Conditions / Front Running	Not affected	Not affected
General Denial Of Service (DOS)	Not affected	Not affected
Uninitialized Storage Pointers	Not affected	Not affected
Floating Points and Precision	Not affected	Not affected
Tx.Origin Authentication	Affected	Not affected
Signatures Replay	Not affected	Not affected
Pool Asset Security (backdoors in the underlying ERC-20)	Not affected	Not affected

	VortexStaking	**PrivateDistribution**
Re-entrancy	Not affected	Not affected
Access Management Hierarchy	Not affected	Not affected
Arithmetic Over/Under Flows	Not affected	Not affected
Unexpected Ether	Not affected	Not affected
Delegatecall	Not affected	Not affected
Default Public Visibility	Not affected	Not affected
Hidden Malicious Code	Not affected	Not affected
Entropy Illusion (Lack of Randomness)	Not affected	Not affected

External Contract Referencing	Not affected	Not affected
Short Address/ Parameter Attack	Not affected	Not affected
Unchecked CALL Return Values	Not affected	Not affected
Race Conditions / Front Running	Not affected	Not affected
General Denial Of Service (DOS)	Not affected	Not affected
Uninitialized Storage Pointers	Not affected	Not affected
Floating Points and Precision	Not affected	Not affected
Tx.Origin Authentication	Affected	Affected
Signatures Replay	Not affected	Not affected
Pool Asset Security (backdoors in the underlying ERC-20)	Not affected	Not affected

CODE COVERAGE AND TEST RESULTS FOR ALL FILES

Tests written by Vortex team (original test coverage)

```
Unit tests
    PrivateDistribution
      ✓ should return the new initial timestamp once it's changed (87ms)
      1) should revert when non-owner tries to add investor
      ✓ should revert when non-whitelisted user tries to deposit or withdraw
      2) should add investor when owner tries to remove investor
      3) should revert when investor tries to deposit multiple times
      ✓ should be able recover token (92ms)
      4) should add multiple investor, withdraw VTX

Unit tests
    VortexStaking
      5) "before each" hook for "Two stakers with the same stakes wait 1 w"

Unit tests
    PrivateDistribution
      6) should be able withdraw allocations for rewards distribution
      - should revert for cliff allocations
      - should be able to with for cliff allocations after cliff passed

3 passing (14m)
2 pending
6 failing
```

FILE	% STMTS	% BRANCH	% FUNCS	% LINES	UNCOVERED LINES
contracts/	17.22	10.42	19.32	17.55	
BokkyPooBahsDateTimeLibrary.sol	1.64	1.72	2.56	1.65	... 389, 393, 394
ERC20Permit.sol	20.00	0.00	33.33	27.27	... 57, 59, 60, 67
IERC2612Permit.sol	100.00	100.00	100.00	100.00	
PrivateDistribution.sol	16.33	13.64	42.86	17.65	... 179, 181, 183
VortexStaking.sol	0.00	0.00	0.00	0.00	... 129, 133, 134
VortexToken.sol	28.57	0.00	25.00	25.00	... 44, 59, 60, 61
VortexTokenVesting.sol	53.52	27.50	61.54	56.06	... 211, 219, 220
All files	**17.22**	**89.53**	**19.32**	**17.55**	

Tests written by Zokyo Secured team

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

Tests were based on the functionality of the code, as well as a review of the Vortex contract to cover untested lines of code.

Contract: ERC20Permit
- ✓ should return correct initial nonce (95ms)
- ✓ should accepts owner signature and transferFrom work correctly (823ms)
- ✓ shouldn't reuse signature (540ms)
- ✓ should rejects other signature (262ms)
- ✓ should rejects expired permit (228ms)

Contract: PrivateDistribution

PrivateDistribution contract add investors phase
- ✓ should set initial Timestamp by an owner (240ms)
- ✓ shouldn't set initial Timestamp by not an owner (180ms)
- ✓ shouldn't set initial Timestamp which is initialized already (285ms)
- ✓ should recover token (379ms)
- ✓ should add investor by an owner (276ms)
- ✓ shouldn't add investor by not an owner (177ms)
- ✓ shouldn't add already added investor (362ms)
- ✓ shouldn't add investor with token allotment zero (203ms)
- ✓ shouldn't add investor with invalid address (197ms)
- ✓ shouldn't add investor with different arrays sizes (208ms)

PrivateDistribution contract withdraw phase
- ✓ should withdraw token on initial period (944ms)
- ✓ should release tokens to all investors (908ms)
- ✓ shouldn't withdraw token by not an investor (368ms)
- ✓ shouldn't withdraw token when timestamp is not initialized (458ms)
- ✓ shouldn't withdraw tokens if there is no withdrawable tokens (793ms)
- ✓ should withdraw tokens after 1 year is passed (1233ms)

Contract: VortexStaking contract
- ✓ should set reward distribution by an governance (120ms)

✓ should stake tokens (1145ms)
✓ shouldn't stake zero amounts of tokens (141ms)
✓ should unstake tokens (1257ms)
✓ shouldn't unstake zero amounts of tokens (156ms)
✓ should exit and get reward correctly (1996ms)
✓ should notify reward amount correctly (711ms)
✓ should get reward after staking period (2974ms)
✓ shouldn't notify reward amount by not a reward distribution (176ms)
✓ should recover excess token (486ms)

Contract: VortexTokenVesting

VortexTokenVesting contract creating phase

✓ should set initial Timestamp by an owner (491ms)
✓ shouldn't set initial Timestamp by not an owner (347ms)
✓ shouldn't set initial Timestamp which is initialized already (523ms)
✓ should return correct initial Timestamp (421ms)
✓ should recover excess token (591ms)

VortexTokenVesting contract withdraw phase

✓ should withdraw RESERVE token correctly (692ms)
✓ shouldn't withdraw RESERVE token before 1 year is passed (594ms)
✓ shouldn't withdraw tokens by not an owner (441ms)
✓ shouldn't withdraw tokens when time is not initialized (318ms)
✓ shouldn't withdraw TEAM_ADVISORS token before cliff period is passed (690ms)
✓ should withdraw TEAM_ADVISORS token after cliff period is passed (888ms)
✓ should withdraw TEAM_ADVISORS token after 1 year is passed (1489ms)
✓ should withdraw REWARDS token correctly (3058ms)

Contract: VortexToken contract

VortexToken contract basics

✓ should deploy with correct name (81ms)
✓ should deploy with correct symbol (178ms)
✓ should deploy with expected decimals (82ms)
✓ should deploy with expected total supply (172ms)

VortexToken contract behavior

✓ should return expected balance of tokens (173ms)
✓ should approve function work correctly (555ms)
✓ should transfer tokens with expected behavior (408ms)

VortexToken contract functionality

```
✓ should set governance correctly (303ms)
✓ shouldn't set governance by not a governance (400ms)
✓ should recover token by governance (700ms)
✓ shouldn`t recover token by not a  governance (628ms)
✓ shouldn`t recover token with  incorrect address of token (269ms)
✓ shouldn`t recover token with  incorrect amount (455ms)

57 passing (3m)
```

FILE	% STMTS	% BRANCH	% FUNCS	% LINES	UNCOVERED LINES
contracts/	100.00	89.53	100.00	100.00	
ERC20Permit.sol	100.00	100.00	100.00	100.00	
IERC2612Permit.sol	100.00	100.00	100.00	100.00	
PrivateDistribution.sol	100.00	95.45	100.00	100.00	
VortexStaking.sol	100.00	92.86	100.00	100.00	
VortexToken.sol	100.00	100.00	100.00	100.00	
VortexTokenVesting.sol	100.00	82.50	100.00	100.00	
All files	**100.00**	**89.53**	**100.00**	**100.00**	

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

ZOKYO.