Tetragaze

Smart contract audit

MakiSwap

Tetragaze

May 21, 2021 | v 2.0

Pass

Zokyo Security Team has concluded that there are no issues that can have an impact on contract security. The contract is well written and is production-ready.

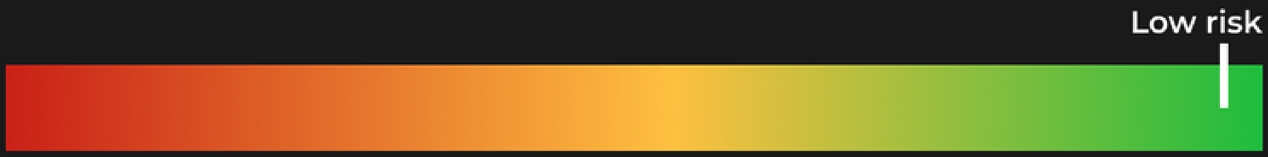
Score

93

Technical summary

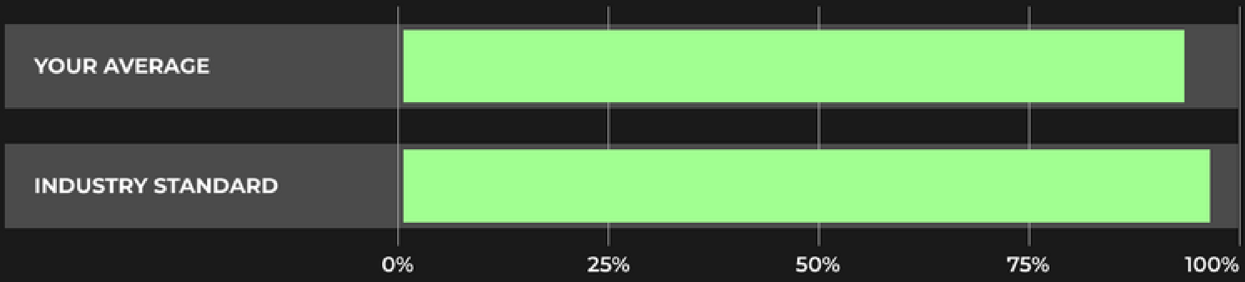
This document outlines the overall security of the Fuse Network smart contracts, evaluated by Zokyo's Blockchain Security team.
The scope of this audit was to analyze and document the Fuse Network smart contract codebase for quality, security, and correctness

Contract Status



There were no critical issues found during the audit.

Testable Code



Testable code is 94%, which is close to 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 Fuse team put in place a bug bounty program to encourage further and active analysis of the smart contract.

Table of content

Auditing Strategy and Techniques Applied

The source code for the Smart contract was obtained from the repository with the last commit on (date) and the previous commit on (date).

Requirements: FIP8 spec

During the review process, we made sure to carefully check that the token contract:

- Adheres to established Token standards and ensures effective implementation;
- Ensures that documentation and code comments accurately reflect the logic and behavior of the code;
- Ensures that token distribution is in line with calculations;
- Utilizes best practices to efficiently use gas, avoiding unnecessary waste;
- Employs methods that are safe from reentrance attacks and immune to the latest vulnerabilities;
- Follows best practices for code readability

Tetragaze team of expert pentesters and smart contract developers carefully evaluated the repository for security issues, code quality, and compliance with specifications and best practices. They conducted a line-by-line review, documenting any issues that were identified during the process. Special care was taken to ensure that the review was thorough and comprehensive.

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. There is only one low level finding which may have effect only in case of specific conditions when the list of validators will be too long. Contracts are well written and structured however, there still are some improvements to contract code marked with an informational rank. All the findings during manual audit have no impact on contract performance or security, so it is fully production-ready.

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.

● Low

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

Complete Analysis

pendingValidators and currentValidators storing structure can be optimized

Low

All the addresses are stored addressArrayStorage and when it is required to get/update an address contract loops overall array. It can cause a Loop Issue and exceed the block gas limit.

Recomendation:

Use mapping to mark address index in addressArrayStorage.

getBallotLimitPerValidator function structure can be optimized

Information

The function getBallotLimitPerValidator of VotingUtils contract lines 88-98 includes unneeded condition with limit value update.

Recomendation:

Add condition to check if MAX_LIMIT_OF_BALLOTS< validatorsCount in line 92 and return 1 instead of value update and spend gas on math operation in 93 line.

Outdated Compiler Version

Low

All the addresses are stored addressArrayStorage and when it is required to get/update an address contract loops overall array. It can cause a Loop Issue and exceed the block gas limit.

Recomendation:

Use mapping to mark address index in addressArrayStorage.

Functions order in contract file

Information

The function order is wrong according to the . code style

Functions should be grouped according to their visibility and ordered in the following way:

- 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.

Recomendation:

Use mapping to mark address index in addressArrayStorage.

Unused functions are present in contract code

Information

The contract ConsensusUtils contains the functions which are not used in the contract or contract which inherits ConsensusUtils:

- _setSnapshot (232-246 lines);
- _setNextSnapshotId (224-225 lines);
- _setLastSnapshotTakenAtBlock (216-218 lines);
- _getRandom (482-484 lines).

Recomendation:

Use mapping to mark address index in addressArrayStorage.

Unused functions are present in contract code

Information

The getSnapshotsPerCycle function is not needed as it is a copy of a public variable.

Recomendation:

Use mapping to mark address index in addressArrayStorage.

Code Coverage and Test Results for all files

Tests written by Fuse Network team

Code Coverage:

The resulting code coverage (i.e., the ratio of tests-to-code) is as follows:

FILE	% STMTS	% BRANCH	% FUNCS	% LINES	UNCOVERED LINES
contracts/	93.31	79.80	93.10	93.41	
BlockReward.sol	100.00	80.70	100.00	100.00	
Consensus.sol	100.00	83.33	100.00	100.00	
ConsensusUtils.sol	86.00	90.32	82.61	85.71	... 278, 479, 483
ConsensusUtils.sol	86.00	90.32	82.61	85.71	99, 103, 105
ConsensusUtils.sol	86.00	90.32	82.61	85.71	99, 103, 105
ConsensusUtils.sol	86.00	90.32	82.61	85.71	99, 103, 105
ConsensusUtils.sol	86.00	90.32	82.61	85.71	99, 103, 105
contracts/	93.31	79.80	93.10	93.41	
ConsensusUtils.sol	86.00	90.32	82.61	85.71	99, 103, 105
ConsensusUtils.sol	86.00	90.32	82.61	85.71	99, 103, 105
ConsensusUtils.sol	86.00	90.32	82.61	85.71	99, 103, 105
ConsensusUtils.sol	86.00	90.32	82.61	85.71	99, 103, 105
ConsensusUtils.sol	86.00	90.32	82.61	85.71	99, 103, 105
ConsensusUtils.sol	86.00	90.32	82.61	85.71	99, 103, 105
ConsensusUtils.sol	86.00	90.32	82.61	85.71	99, 103, 105

BlockReward.sol	100.00	80.70	100.00	100.00
contracts/	**93.31**	**79.80**	**93.10**	**93.41**

Test Results

```
Contract: BlockReward

  initialize
    ✓ default values (113644 gas)
  reward
    ✓ can only be called by system address  (113644 gas)
    ✓ should revert if input array contains more than one item  (113644 gas)
    ✓ can only be called by system address  (113644 gas)
    ✓ can only be called by system address  (113644 gas)
    ✓ can only be called by system address  (113644 gas)
    ✓ can only be called by system address  (113644 gas)
    ✓ can only be called by system address  (113644 gas)
    #getBlockRewardAmountPerValidator
      ✓ block reward with one validator  (113644 gas)
      ✓ block reward with one validator  (113644 gas)
      ✓ block reward with one validator  (113644 gas)
      ✓ block reward with one validator  (113644 gas)
      ✓ block reward with one validator  (113644 gas)
      ✓ block reward with one validator  (113644 gas)
      ✓ block reward with one validator  (113644 gas)
  emitRewardedOnCycle
    ✓ should be successful if shouldEmitRewardedOnCycle and consensus.isFinalized
      are true (113644 gas)
    ✓ can only be called by system address  (113644 gas)
    ✓ can only be called by system address  (113644 gas)
  upgrade to
    ✓ can only be called by system address  (113644 gas)
    ✓ can only be called by system address  (113644 gas)
    ✓ can only be called by system address  (113644 gas)
    ✓ can only be called by system address  (113644 gas)
```

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