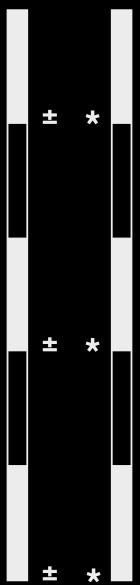
VERILOG

2023

SONORUS ERC20 & VESTER CONTRACT SECURITY REVIEW REPORT



Dec 7, 2023

Contents

1 | Introduction

Figure 1.1: Sonorus Report Cover

This report presents our engineering engagement with the Sonorus team on their token distribution contract.

Project Name	**Sonorus**
Repository Link	https://github.com/sonorus-network/sonorus-contract-audit
First Commit Hash	First: ab736d2;
Final Commit Hash	Final: 1c39b94;
Language	Solidity
Chain	BNB Smart Chain

2 | About Verilog Solutions

Founded by a group of cryptography researchers and smart contract engineers in North America, Verilog Solutions elevates the security standards for Web3 ecosystems by being a full-stack Web3 security firm covering smart contract security, consensus security, and operational security for Web3 projects.
Verilog Solutions team works closely with major ecosystems and Web3 projects and applies a quality above quantity approach with a continuous security model. Verilog Solutions onboards the best and most innovative projects and provides the best-in-class advisory services on security needs, including on-chain and off-chain components.

3 | Service Scope

3.1 | Service Stages

Our auditing service includes the following two stages:

- Smart Contract Auditing Service

3.1.1 | Smart Contract Auditing Service

The Verilog Solutions team analyzed the entire project using a detailed-oriented approach to capture the fundamental logic and suggested improvements to the existing code. Details can be found under Findings And Improvement Suggestions.

3.2 | Methodology

- **Code Assessment**
  - We evaluate the overall quality of the code and comments as well as the architecture of the repository.
  - We help the project dev team improve the overall quality of the repository by providing suggestions on refactorization to follow the best practices of Web3 software engineering.
- **Code Logic Analysis**
  - We dive into the data structures and algorithms in the repository and provide suggestions to improve the data structures and algorithms for the lower time and space complexities.
  - We analyze the hierarchy among multiple modules and the relations among the source code files in the repository and provide suggestions to improve the code architecture with better readability, reusability, and extensibility.

3.3 | Audit Scope

Our auditing for Sonorus covered the Solidity smart contracts under the folder 'contracts' in the repository (https://github.com/sonorus-network/sonorus-contract-audit) with commit hash **ab736d2**.

4 | Findings and Improvement Suggestions

Severity	Total	Acknowledged	Resolved
High	0	0	0
Medium	1	1	1
Low	2	2	2
Informational	1	1	1

4.1 | Medium

4.1.1 | Lack of checks in the `constructor()`

Severity	Medium
Source	contracts/distribution/SonorusVestingDistributor.sol#L32-L54;
Commit	40fc730;
Status	Resolved in commit 1afa480;

- **Description**
  The `constructor()` defines several variables that can't be changed in the future. Since the `SonorusVestingDistributor` contract will be deployed for every beneficiary, it is very important to perform checks that make sure that the setup is correct. We recommend performing the following checks:
  - Zero address check to the `beneficiary` address
  - `_initialVestRate` is less than 100
- **Exploit Scenario**
  N/A.
- **Recommendations**
  Add the respective checks to the `constructor()` of the `SonorusVestingDistributor` contract.
- **Results**
  Resolved in commit 1afa480.

  The checks were added.

4.2 | Low

4.2.1 | Lack of event emission for critical operations

Severity	Low
Source	contracts/distribution/SonorusVestingDistributor.sol#L88;
Commit	ab736d2;
Status	Resolved in commit 1afa480;

- **Description**
  Events are vital aids in monitoring contracts and detecting suspicious behavior. The `claim()` function performs state changes, therefore it should consider the emission of an event.
- **Exploit Scenario**
  N/A.
- **Recommendations**
  Please consider the emission of an event when the `claim()` function gets called.
- **Results**
  Resolved in commit 1afa480.

  The event was added.

4.2.2 | No function to change the `beneficiary` address

Severity	Low
Source	contracts/distribution/SonorusVestingDistributor.sol#L49;
Commit	40fc730;
Status	Resolved in commit 1afa480;

- **Description**
  The `beneficiary` address is defined in the `constructor()` and there's no function to change the address afterward. Other contracts like the Uniswap treasury vesting contract, consider this function.
- **Exploit Scenario**
  N/A.
- **Recommendations**
  Add a function to change the `beneficiary` address.
- **Results**
  Resolved in commit 1afa480.

  The suggestion was implemented.

4.3 | Informational

4.3.1 | Floating solidity pragma version

Severity	Informational
Source	Global;
Commit	ab736d2;
Status	Resolved in commit 1c39b94;

- **Description**
  Current smart contracts use version ^0.8.20 and compilers within versions ≥ 0.8.20 and <0.9.0 can be used to compile those contracts. Therefore, the contract may be deployed with a newer or latest compiler version which generally has higher risks of undiscovered bugs.

  It is a good practice to fix the solidity pragma version if the contract is not designed as a package or library that will be used by other projects or developers.

- **Exploit Scenario**
  N/A.

- **Recommendations**
  Use the fixed solidity pragma version.

- **Results**
  Resolved in commit 1e9b38a.

  The solidity version has been fixed to 0.8.20.

5 | Appendix

5.1 | Appendix I: Severity Categories

Severity	Description
High	Issues that are highly exploitable security vulnerabilities. It may cause direct loss of funds / permanent freezing of funds. All high severity issues should be resolved.
Medium	Issues that are only exploitable under some conditions or with some privileged access to the system. Users' yields/rewards/information is at risk. All medium severity issues should be resolved unless there is a clear reason not to.
Low	Issues that are low risk. Not fixing those issues will not result in the failure of the system. A fix on low severity issues is recommended but subject to the clients' decisions.
Information	Issues that pose no risk to the system and are related to the security best practices. Not fixing those issues will not result in the failure of the system. A fix on informational issues or adoption of those security best practices-related suggestions is recommended but subject to clients' decision.

5.2 | Appendix II: Status Categories

Severity	Description
Unresolved	The issue is not acknowledged and not resolved.
Partially Resolved	The issue has been partially resolved
Acknowledged	The Finding / Suggestion is acknowledged but not fixed / not implemented.
Resolved	The issue has been sufficiently resolved

6 | Disclaimer

Verilog Solutions receives compensation from one or more clients for performing the smart contract and auditing analysis contained in these reports. The report created is solely for Clients and published with their consent. As such, the scope of our audit is limited to a review of code, and only the code we note as being within the scope of our audit is detailed in this report. It is important to note that the Solidity code itself presents unique and unquantifiable risks since the Solidity language itself remains under current development and is subject to unknown risks and flaws. Our sole goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies. Thus, Verilog Solutions in no way claims any guarantee of security or functionality of the technology we agree to analyze.
In addition, Verilog Solutions reports do not provide any indication of the technology's proprietors, business, business model, or legal compliance. As such, reports do not provide investment advice and should not be used to make decisions about investment or involvement with any particular project. Verilog Solutions has the right to distribute the Report through other means, including via Verilog Solutions publications and other distributions. Verilog Solutions makes the reports available to parties other than the Clients (i.e., "third parties") – on its website in hopes that it can help the blockchain ecosystem develop technical best practices in this rapidly evolving area of innovation.