{SecureContract.IO}

Solana Smart Contract Security Audit

Solana Token Test

securecontract@protonmail.com

Date of Engagement: October 9, 2021 - October 15, 2021

https://github.com/SecureContractIO/sample_reports

ATTENTION

THIS DOCUMENT MAY CONTAIN CONFIDENTIAL INFORMATION ABOUT ITS SYSTEMS AND INTELLECTUAL PROPERTY OF THE CUSTOMER AS WELL AS INFORMATION ABOUT POTENTIAL VULNERABILITIES AND METHODS OF THEIR EXPLOITATION. THE REPORT CONTAINING CONFIDENTIAL INFORMATION CAN BE USED INTERNALLY BY THE CUSTOMER OR IT CAN BE DISCLOSED PUBLICLY AFTER ALL VULNERABILITIES ARE FIXED - UPON DECISION OF CUSTOMER.

INTRODUCTION

SecureContract Team (Consultant) were contracted by ? (the client) to conduct a Smart Contracts Security Analysis. This report represents the findings of the security assessment of the customer‘s smart contracts and its code review conducted between 9 - 15 October 2021.

Project Scope

The scope of the project is NFT smart contracts. We have controlled the source code for commonly known and more specific vulnerabilities, below are those considered (the full list includes but is not limited to) under the scope of the Solana architecture.

- Reentrancy
- Timestamp Dependence
- Gas Limit and Loops
- DoS with (Unexpected) Throw
- DoS with Block Gas Limit

Project Scope

- Transaction-Ordering Dependence
- Byte array vulnerabilities
- Style guide violation
- Transfer forwards all gas
- Malicious libraries
- Compiler version not fixed
- Unchecked external call - Unchecked math
- Unsafe type inference

Project Scope

- Arithmetic errors

Executive Summary

According to the assessment, the customer's smart contract is "well secured".

Manual and localized checks are done. All issues were performed by our team, which included the analysis of code functionality, manual audit found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the audit overview section. In brief we obtained the following result:

- 0 Critical issues
- 0 High issues
- 0 Medium issues
- 0 Low issues
- 0 Very low issues

Severity Definitions

Risk Level	Description
Critical	straightforward to exploit and may lead to lost tokens.
High	difficult to exploit but can have significant impact
Medium	important to fix; however, they cannot lead to lost tokens.
Low	mostly cannot have a significant impact on execution.

SECURITY ANALYSIS OVERVIEW

Table: File : /state/mask.rs

Parameter	Status
LoC	420
Observation	Passed
Safety	Passed
Score	Passed
Conclusion	Secure

Table: File : /state/from_to_bytes.rs

Parameter	Status
LoC	530
Observation	Passed
Safety	Passed
Score	Passed
Conclusion	Secure

Table: File : /state/oracle_request.rs

Parameter	Status
LoC	75
Observation	Passed
Safety	Passed
Score	Passed
Conclusion	Secure

Table: File : /state/program_state.rs

Parameter	Status
LoC	99
Observation	Passed
Safety	Passed
Score	Passed
Conclusion	Secure

Table: File : entrypoint.rs

Parameter	Status
LoC	24
Observation	Passed
Safety	Passed
Score	Passed
Conclusion	Secure

Table: File : error.rs

Parameter	Status
LoC	343
Observation	Passed
Safety	Passed
Score	Passed
Conclusion	Secure

Table: File : processor.rs

Parameter	Status
LoC	654
Observation	Passed
Safety	Passed
Score	Passed
Conclusion	Secure

Table: File : instruction.rs

Parameter	Status
LoC	395
Observation	Passed
Safety	Passed
Score	Passed
Conclusion	Secure

Audit Findings

- Critical: No critical severity vulnerabilities were found.
- High: No high severity vulnerabilities were found.
- Medium: No medium severity vulnerabilities were found.
- Low: No low severity vulnerabilities were found.
- Very Low: No very low severity vulnerabilities were found.

Discussion

(*) Double check any hard-coded values before going to production.
(*) Consider to add more comments in the source files (e.g. chest.rs).

Disclaimers

The smart contracts given for audit have been analysed in accordance with the best industry practices at the date of this report, in relation to: cybersecurity vulnerabilities and issues in smart contract source code.
Because the total number of test cases are unlimited, the audit makes no statements or warranties on the security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bugfree status or any other statements of the contract. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only - we recommend proceeding with several independent audits and a public bug bounty program to ensure security of smart contracts.

Technical Disclaimer Smart contracts are deployed and executed on the blockchain. The platform, its programming language, and other software related to the smart contract can have their own vulnerabilities that can lead to hacks. Thus, the audit can't guarantee explicit security of the audited smart contracts.