



QuillAudits



Audit Report
April, 2021



Sappchat

Contents

| | |
|---|----|
| Scope of Audit | 01 |
| Techniques and Methods | 01 |
| Issue Categories | 02 |
| Introduction | 04 |
| Issues Found – Code Review/Manual Testing | 04 |
| Summary | 06 |
| Disclaimer | 07 |

Scope of Audit

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

Checked Vulnerabilities

- Re-entrancy
- Timestamp Dependence
- Gas Limit and Loops
- DoS with Block Gas Limit
- Transaction-Ordering Dependence
- Use of tx.origin
- Exception disorder
- Gasless send
- Balance equality
- Byte array
- Transfer forwards all gas
- ERC20 API violation
- Malicious libraries
- Compiler version not fixed
- Redundant fallback function
- Send instead of transfer
- Style guide violation
- Unchecked external call
- Unchecked math
- Unsafe type inference
- Implicit visibility level

Techniques and Methods

Throughout the audit of smart contracts care was taken to ensure:

- The overall quality of code.
- Use of best practices.
- Code documentation and comments match logic and expected behaviour.
- Token distribution and calculations are as per the intended behaviour mentioned in the whitepaper.
- Implementation of ERC-20 token standards.
- Efficient use of gas.
- Code is safe from re-entrancy and other vulnerabilities.

The following techniques, methods and tools were used to review all the smart contracts.

Structural Analysis

In this step we have analyzed the design patterns and structure of smart contracts. A thorough check was done to ensure the smart contract is structured in a way that will not result in future problems.

Static Analysis

Static Analysis of Smart Contracts was done to identify contract vulnerabilities. In this step a series of automated tools are used to test security of smart contracts.

Code Review / Manual Analysis

Manual Analysis or review of code was done to identify new vulnerability or verify the vulnerabilities found during the static analysis. Contracts were completely manually analyzed, their logic was checked and compared with the one described in the whitepaper. Besides, the results of automated analysis were manually verified.

Gas Consumption

In this step we have checked the behaviour of smart contracts in production. Checks were done to know how much gas gets consumed and possibilities of optimization of code to reduce gas consumption.

Tools and Platforms used for Audit

Remix IDE, Truffle, Truffle Team, Ganache, Solhint, Mythril, Slither, SmartCheck.

Issue Categories

Every issue in this report has been assigned with a severity level. There are four levels of severity and each of them has been explained below.

High severity issues

A high severity issue or vulnerability means that your smart contract can be exploited. Issues on this level are critical to the smart contract’s performance or functionality and we recommend these issues to be fixed before moving to a live environment.

Medium level severity issues

The issues marked as medium severity usually arise because of errors and deficiencies in the smart contract code. Issues on this level could potentially bring problems and they should still be fixed.

Low level severity issues

Low level severity issues can cause minor impact and or are just warnings that can remain unfixed for now. It would be better to fix these issues at some point in the future.

Informational

These are severity four issues which indicate an improvement request, a general question, a cosmetic or documentation error, or a request for information. There is low-to-no impact.

Number of issues per severity

| | High | Medium | Low | Informational |
|--------|------|--------|-----|---------------|
| Open | 0 | 0 | 0 | 2 |
| Closed | 0 | 0 | 0 | 0 |

Introduction

During the period of April 14th, 2021 to April 17th, 2021 - Quillhash Team performed a security audit for SappChat smart contracts. The code for the audit was taken from following the official BSCscan link:

<https://bscscan.com/address/0xa64ee256d51d52c31fec7eb5e372ac0c45a8a3f4#code>

Issues Found – Code Review / Manual Testing

High severity issues

Not Found

Medium severity issues

Not Found

Low level severity issues

Not Found

Informational

1. Coding Style Issues

Description:

Coding style issues influence code readability and in some cases may lead to bugs in future. Smart Contracts have naming convention, indentation and code layout issues. It's recommended to use the Solidity Style Guide to fix all the issues. Consider following the Solidity guidelines on formatting the code and commenting for all the files. It can improve the overall code quality and readability.

2. Order of layout

Description

The order of functions, as well as the rest of the code layout, does not follow the solidity style guide.

Layout contract elements in the following order:

- a. Pragma statements
- b. Import statements
- c. Interfaces
- d. Libraries
- e. Contracts

Inside each contract, library or interface, use the following order:

- a. Type declarations
- b. State variables
- c. Events
- d. Functions

Recommendation:

Please read the following documentation links to understand the correct order:-

<https://solidity.readthedocs.io/en/v0.5.3/style-guide.html#order-of-layout>

<https://docs.soliditylang.org/en/v0.6.12/style-guide.html#order-of-functions>

Closing Summary

Overall, smart contracts are very well written and adhere to guidelines. During the process of audit, No issues of high, medium as well as low severity were found, which might affect the intended behaviour of the contracts.

Moreover, no instances of Re-entrancy or Back-Door Entry were found in the contract.

Disclaimer

Quillhash audit is not a security warranty, investment advice, or an endorsement of the SappChat platform. This audit does not provide a security or correctness guarantee of the audited smart contracts. 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. Securing smart contracts is a multistep process. One audit cannot be considered enough. We recommend that the SappChat Team put in place a bug bounty program to encourage further analysis of the smart contract by other third parties.



Sappchat



QuillAudits



Canada, India, Singapore and United Kingdom



audits.quillhash.com



hello@quillhash.com