HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: StrongBlock
Date: March 25th, 2022

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Document

Name	Smart Contract Code Review and Security Analysis Report for StrongBlock.
Approved By	Evgeniy Bezuglyi | SC Department Head at Hacken OU
Type of Contracts	Token; Swap
Platform	EVM
Language	Solidity
Methods	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
Website	https://strongblock.com/
Timeline	21.03.2022 - 23.03.2022
Changelog	24.03.2022 - Initial Review 25.03.2022 - Revising

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Table of contents

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Introduction

Hacken OÜ (Consultant) was contracted by StrongBlock (Customer) to conduct a Smart Contract Code Review and Security Analysis. This report presents the findings of the security assessment of the Customer's smart contracts.

Scope

The scope of the project is smart contracts in the repository:

Repository:
https://github.com/StrongBlock/StrongerToken
Commit:
7894e2a92dcfb3616d160f039488185c371bd214
JS tests: Yes
Contracts:
Stronger.sol
Technical Documentation: Yes (https://github.com/StrongBlock/StrongChain)

Repository:
https://github.com/StrongBlock/StrongSwap
Commit:
21271267c405d2ee6db02a92b1390861602cd2e7
JS tests: Yes
Contracts:
StrongSwap.sol
Technical Documentation: No

We have scanned this smart contract for commonly known and more specific vulnerabilities. Here are some of the commonly known vulnerabilities that are considered:

Category	Check Item
Code review	▪ Reentrancy ▪ Ownership Takeover ▪ Timestamp Dependence ▪ Gas Limit and Loops ▪ Transaction-Ordering Dependence ▪ Style guide violation ▪ EIP standards violation ▪ Unchecked external call ▪ Unchecked math ▪ Unsafe type inference ▪ Implicit visibility level ▪ Deployment Consistency ▪ Repository Consistency

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Functional review	▪ Business Logics Review ▪ Functionality Checks ▪ Access Control & Authorization ▪ Escrow manipulation ▪ Token Supply manipulation ▪ Assets integrity ▪ User Balances manipulation ▪ Data Consistency ▪ Kill-Switch Mechanism

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Executive Summary

Score measurements details can be found in the corresponding section of the methodology.

Documentation quality

The Customer provided Litepaper and tokenomics, but there are no technical requirements. There is no documentation for swap. Total Documentation Quality score is **4** out of **10**.

Code quality

Most of the code follows official language style guides. Unit tests were provided. Code quality score is **10** out of **10**.

Architecture quality

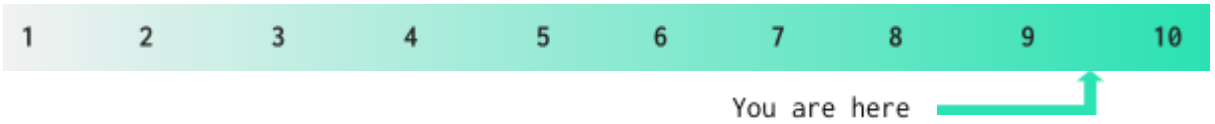
The architecture of the contract is clear. Architecture quality score is **10** out of **10**.

Security score

As a result of the audit, security engineers found no issues. The security score is **10** out of **10**.

Summary

According to the assessment, the Customer's smart contract has the following score: **9.4**

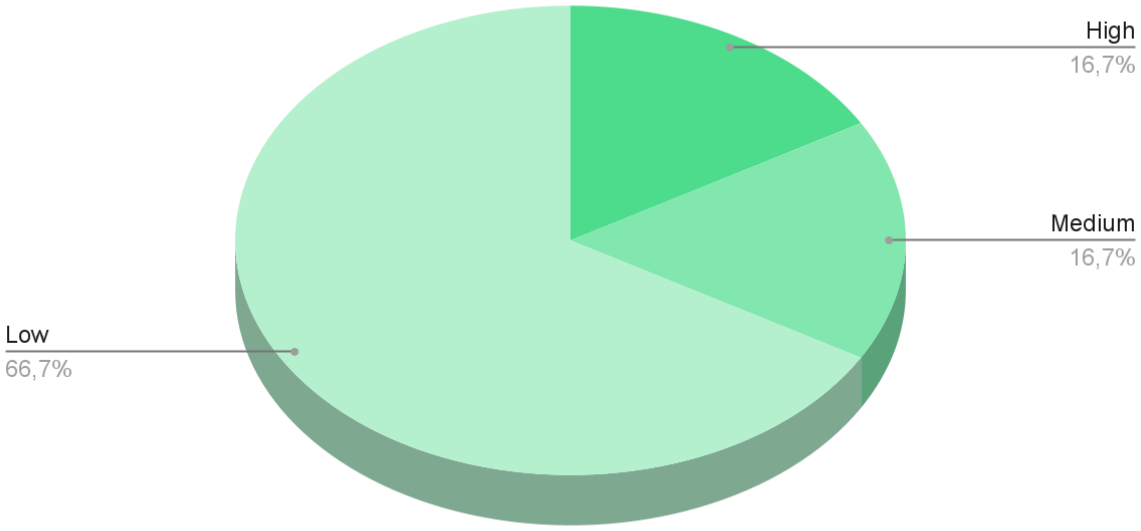


Notices

1. The owner can mint any amount of token.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Graph 1. The distribution of vulnerabilities after the first audit.



Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to assets loss or data manipulations.
High	High-level vulnerabilities are difficult to exploit; however, they also have a significant impact on smart contract execution, e.g., public access to crucial functions
Medium	Medium-level vulnerabilities are important to fix; however, they cannot lead to assets loss or data manipulations.
Low	Low-level vulnerabilities are mostly related to outdated, unused, etc. code snippets that cannot have a significant impact on execution

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Findings

Critical

No critical severity issues were found.

High

1. **Owner`s ability to change the conversion ratio anytime.**

   The owner can instantly change the conversion ratio without notifying users.

   Contracts: StrongSwap.sol

   Function: setConversionRatio

   Recommendation: make a two-step conversion ratio change, where the first step is a request for a change, and the second is its acceptance after a certain time or number of blocks or validate the amount of tokens that users expect to match the one they receive.

   Status: Fixed

Medium

1. **Zero token transferring allowed.**

   There is no check for zero amount to transfer, which can lead to excess gas consumption..

   Contracts: StrongSwap.sol

   Function: deposit, withdraw

   Recommendation: add check if the amount to transfer is greater than 0.

   Status: Fixed

Low

1. **The contract has a public function that is not used internally.**

   The mint function is not used internally and specified as public which uses more gas than external.

   Contracts: StrongSwap.sol

   Function: mint

   Recommendation: replace visibility to external.

   Status: Fixed

2. **Tokens state variables can be defined as immutable.**

   State variables _strongToken and _strongerToken are initialized in the constructor and are never changed. Using “immutable” with unchangeable variables saves gas.

Contracts: Stronger.sol

Function: -

Recommendation: use “immutable” modifier with _strongToken and _strongerToken.

Status: Mitigated. The Customer approved that modifier cannot be added.

3. **No messages in require conditions all over the code.**

   Contracts: StrongSwap.sol

   Function: init, swap, swapFor, deposit, withdraw, setStrongReceiver, setConversionRatio.

   Recommendation: it is better to add error messages in important places of the contract.

   Status: Fixed

4. **No caller verification in the initialization function.**

   There is no restriction that only the owner can call the initialization function.

   Contracts: StrongSwap.sol

   Function: init

   Recommendation: it is better to add only owner access to the initialization function.

   Status: Fixed

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Disclaimers

Hacken Disclaimer

The smart contracts given for audit have been analyzed by the best industry practices at the date of this report, with cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report (Source Code); the Source Code compilation, deployment, and functionality (performing the intended functions).

The audit makes no statements or warranties on the security of the code. It also cannot be considered a sufficient assessment regarding the utility and safety of the code, bug-free status, or any other contract statements. While we have done our best in conducting the analysis and producing this report, it is important to note that it should not rely on this report only – we recommend proceeding with several independent audits and a public bug bounty program to ensure the security of smart contracts.

Technical Disclaimer

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